

Luis Vicente Jimeno Merino

---

**PROGRAMACIÓN DE CONTROLADOR  
PARA SISTEMA DE GENERACIÓN Y  
ALMACENAMIENTO TÉRMICO EN  
EDIFICIO**



Universidad de Valladolid



ESCUELA DE INGENIERÍAS  
INDUSTRIALES



Universidad de Valladolid



ESCUELA DE INGENIERÍAS  
INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERÍAS INDUSTRIALES

Grado en Ingeniería en Electrónica y Automática Industrial

**PROGRAMACIÓN DE CONTROLADOR PARA  
SISTEMA DE GENERACIÓN Y  
ALMACENAMIENTO TÉRMICO EN EDIFICIO**

**Autor: Luis Vicente Jimeno Merino**

**Tutores:**

**Baeyens Lázaro, Enrique**  
*Departamento de Ingeniería de  
Sistemas y Automática*

**Arnanz Gómez, Roberto**  
*FUNDACIÓN CARTIF*

**Valladolid, enero de 2022**



## Resumen

Este proyecto presenta la gestión energética de un sistema de generación renovable termoeléctrico. El sistema se compone de tres partes: la producción con tecnología solar híbrida, la transformación y almacenamiento de energía con sistemas de material de cambio de fase (MCF), un reactor termoquímico (RTQ) de amoníaco y bomba de calor, y la demanda de energía térmica. El control del sistema persigue el uso óptimo del recurso solar disponible a la vez que se cubren las necesidades térmicas del edificio, centrándose para ello en la gestión de las reacciones endotérmicas y exotérmicas del RTQ que permiten obtener tanto frío como calor, así como del resto de subsistemas. Este sistema de control se ha programado y validado en primer lugar en entorno Matlab/Simulink, y posteriormente se ha trasladado a un PLC usando para ello EC-GFX. Por último, se ha realizado un SCADA en ENVYSION, conectado el PLC mediante Modbus utilizando Python.

## Palabras Clave

Material Termoquímico, Material Cambio de Fase, Generación Renovable, Sistemas de Control, Modbus PLC.

## Abstract

This project presents the energy management of a thermoelectric renewable generation system. The system consists of three parts: production with hybrid solar technology, energy transformation and storage with phase change material systems (MCF), an ammonia thermochemical reactor (RTQ) and heat pump, and the demand for thermal energy. The control of the system pursues the optimal use of the available solar resource while meeting the thermal needs of the building, focusing for this on the management of the endothermic and exothermic reactions of the RTQ that allow obtaining both cold and heat, as well as the rest. of subsystems. This control system has been programmed and validated first in a Matlab / Simulink environment, and later it has been transferred to a PLC using EC-GFX. Finally, a SCADA has been carried out in ENVYSION, connected to the PLC through Modbus using Python.

## Keywords

Thermochemical Material (TCM), Phase-Change Material (PCM), Renewable Generation, Control Systems, Modbus PLC.



# Índice

|     |  |    |
|-----|--|----|
| 1   | Introducción y Objetivos .....   | 9  |
| 2   | Estudio teórico .....  | 13 |
| 2.1 | Introducción a los modos de operación del sistema .....                                  | 13 |
| 2.2 | Principales modos de operación térmicos .....  | 14 |
| 2.3 | Descripción general de condiciones de funcionamiento y especificaciones del sistema..... | 25 |
| 3   | Modelado del Sistema.....  | 37 |
| 3.1 | Variables de entrada, salida y globales .....  | 37 |
| 3.2 | Parámetros .....   | 43 |
| 4   | Diseño del control en Matlab/Simulink .....  | 47 |
| 4.1 | Interfaz .....   | 47 |
| 4.2 | Guía de Usuario .....  | 50 |
| 4.3 | Programación Controlador (StateFlow).....  | 52 |
| 4.4 | Otros archivos del modelo .....  | 58 |
| 4.5 | Validación del control.....  | 60 |
| 5   | Modelado en PLC.....   | 65 |
| 5.1 | Programación Controlador (EC-GFX) .....  | 65 |
| 5.2 | Interfaz (ENVYSION) .....  | 73 |
| 5.3 | Guía de Usuario .....  | 77 |
| 5.4 | Conexión Modbus.....   | 78 |
| 5.5 | Validación del modelo (Python).....  | 80 |
| 6   | Contexto.....  | 89 |
| 7   | Conclusión .....   | 91 |
| 8   | Bibliografía .....   | 93 |
| 9   | Anexos.....  | 98 |



## Índice de Figuras y Tablas

|   |    |
|---|----|
| Figura 1: Sistema Global .....  | 13 |
| Figura 2: Esquema del Sistema Térmico.....  | 14 |
| Figura 3: Esquema del Sistema térmico con componentes .....                       | 15 |
| Figura 4: Modos y Submodos de operación principales .....                         | 16 |
| Figura 5: Esquema de operación Carga RTQ .....                                    | 18 |
| Figura 6: Esquema de operación Descarga RTQ.....                                  | 20 |
| Figura 7: Esquema de operación Descarga RTQ en períodos muy fríos.....            | 22 |
| Figura 8: Esquema de operación Demanda cubierta por energía solar .....           | 23 |
| Figura 9: Esquema de operación Demanda cubierta por baterías .....                | 24 |
| Tabla 1: Temperaturas características de MCF Y RTQ.....                           | 26 |
| Tabla 2: Principales interacciones según periodo.....                             | 26 |
| Figura 10: Esquema de componentes del Sistema Térmico .....                       | 35 |
| Figura 11: Esquema de componentes y variables del Sistema Térmico .....           | 37 |
| Figura 12: Imagen del Modelado en Matlab/Simulink .....                           | 47 |
| Figura 13: Captura de código archivo cambio_paramtero_allow.....                  | 51 |
| Figura 14: Vista del controlador en StateFlow.....                                | 52 |
| Figura 15: Captura Diccionario .....  | 59 |
| Figura 16: Imagen valores iniciales del Modelo .....                              | 60 |
| Figura 17: Captura de la transición entre dos estados de simulación n°1 .....     | 61 |
| Figura 18: Capturas de la transición entre dos estados de la simulación n°7 ..... | 62 |
| Figura 19: Vista general de la ventana Entradas .....                             | 65 |
| Figura 20: Esquema de ejecución de ventanas según biestable.....                  | 66 |
| Figura 21: Vista general de la ventana máquina de estados .....                   | 68 |
| Figura 22: Vista general de la ventana Salidas .....                              | 72 |
| Figura 23: Vista de la ventana Sistema en ENVYSION.....                           | 74 |
| Figura 24: Vista de la ventana Sistema en ENVYSION con un estado concreto .....   | 75 |
| Figura 25: Gráfica Válvulas .....   | 76 |
| Figura 26: Gráfica Bombas.....  | 76 |
| Figura 27: Gráfica Ventilconvectores .....  | 77 |
| Figura 28: Esquema conexión Modbus.....   | 78 |
| Figura 29: Captura código Simulado .....  | 81 |
| Figura 30: Vista del estado OFF .....   | 81 |
| Figura 31: Vista del estado NOHEATdemandTCMCHARGING.....                          | 82 |
| Figura 32: Vista del estado Transition_HEATdemandTCMCHARGING .....                | 83 |
| Figura 33: Vista del estado HEATdemandTCMCHARGING .....                           | 84 |
| Figura 34: Vista del estado HEATdemandwhilePCMbeingheated .....                   | 85 |
| Figura 35: Vista del estado Transition_HEATdemandTCMDISCHARGING .....             | 86 |
| Figura 36: Vista del estado HEATdemandTCMDISCHARGING .....                        | 86 |



## Glosario de Abreviaturas

| Nombre                          | Abreviatura en español | Abreviatura en inglés |
|---------------------------------|------------------------|-----------------------|
| Agua Caliente Sanitaria         | ACS                    | DHW                   |
| Bomba de calor                  | BC                     | HP                    |
| Coeficiente de rendimiento      | CDR                    | COP                   |
| Controlador Lógico Programable  | PLC                    | PLC                   |
| Estado de Carga                 | EDC                    | SOC                   |
| Flujo de transferencia de calor | FTC                    | HTF                   |
| Material Cambio de Fase         | MCF                    | PCM                   |
| Panel Térmico Fotovoltaico      | PTF                    | PVT                   |
| Reactor Termoquímico            | RTQ                    | TCM                   |



# 1 Introducción y Objetivos

En el presente documento se presenta el desarrollo del control de un sistema de generación renovable termoeléctrico, que se encarga de la gestión de la energía solar y donde se combinan dos elementos innovadores: un reactor termoquímico de amoníaco (RTQ) y una serie de baterías térmicas de material de cambio de fase (MCF), que se usan para almacenar la energía. Además, el sistema incluye tecnología híbrida solar (termoeléctrica) y una bomba de calor. El objetivo principal del control es la provisión de energía calorífica a un edificio a través de la gestión del calor producido en los módulos solares termofotovoltaicos, y su transformación y almacenamiento mediante el RTQ, la bomba de calor y las baterías de MCF, tanto en invierno como en verano. De esta forma, se pretende satisfacer la demanda de calor y frío de un edificio durante todo el año, adaptando la operación del sistema a los distintos rangos de temperaturas tanto externa (temperatura fuera del sistema) como internas (flujos dentro del sistema).

Cabe destacar que se realiza un trabajo previo de estudio del diseño del sistema que se va a controlar. Este diseño ha sido realizado en el marco del proyecto europeo “Minimal Size Thermal and Electrical Energy Storage System for In-Situ Residential Installation - MiniStor” (Grant Agreement 869821) del cual Fundación CARTIF es socio. Este diseño se utiliza como base para las especificaciones del controlador, y se muestra en este proyecto para facilitar la comprensión del control desarrollado. El desarrollo del controlador contenido en este TFG se ha realizado en colaboración con investigadores de la División de Energías de Fundación CARTIF, y sirve como ejemplo para la validación de estrategias de programación previo al desarrollo del controlador real de dicho sistema de generación de energía. Es importante señalar que las figuras que representan el código de programación están en inglés por esta razón.

Este proyecto se centra tanto en la programación software del control como la implementación práctica en un dispositivo PLC Eclipse modelo S-1000.

Para el desarrollo de ambos se han usado varios entornos de programación. Uno es Matlab/Simulink junto con el programa StateFlow (que es una librería de funciones de MATLAB) utilizado para la validación del diseño del software de control, y el otro es EC-GFX, programa para la implementación en el PLC indicado. Además, para el diseño gráfico del SCADA se ha usado el programa ENVYISION.

Por otro lado, también ha usado el lenguaje de programación Python para validar el funcionamiento del control utilizando baterías de datos de

entradas/salidas y probar además la conexión Modbus, que es la que utiliza el PLC para comunicarse con su entorno en la aplicación real.

Las versiones de los programas y sistemas usados son:

- MATLAB Version: 9.11.0.1769968 (R2021b)
- StateFlowVersion: Version 10.5 (R2021b)
- EC-GFX Program: Eclipse Firmware Version1.17.21259.755
- ENVYSION: Eclipse Firmware Version1.17.21259.755
- Python: Version 3.8.8
- OperatingSystem: Microsoft Windows 10 Pro Version 10.0 (Build 19042)

En cuanto a los objetivos este proyecto sirve para dar validar el proceso de diseño de un controlador para un sistema de obtención, almacenamiento y uso de energía como el descrito.

El objetivo principal es demostrar la viabilidad de este sistema de control de cara a su uso práctico. El objetivo del estudio teórico previo consiste en mostrar la eficiencia del mismo no solo por el uso de baterías térmicas (MCF) y un reactor termoquímico (RTQ), sino por la combinación de ambos, algo que no se había contemplado con anterioridad dando un toque innovador al proyecto.

El trabajo se ha estructurado de la siguiente manera. La introducción da una ligera pincelada del trabajo a desarrollar, cómo se ha realizado y de los recursos que se han utilizado. El desarrollo del trabajo se ha estructurado en 4 puntos. El primero es el estudio teórico del sistema que ofrece una descripción detallada del sistema diseñado y cómo funcionan sus modos de operación con las condiciones de funcionamiento de los mismos. En segundo lugar, se muestra el modelado del sistema que especifica los elementos del sistema, incluye una descripción de los mismos y presenta el sistema diseñado de forma esquemática. Los dos últimos puntos indican los dos pasos para la programación del sistema: una es a través de MATLAB/Simulink y la otra la implementación práctica en el PLC.

Finalmente se encuentran el contexto y las conclusiones, además de la bibliografía que incluye las referencias del estudio teórico y la búsqueda bibliográfica de TFG previos.

Respecto a la búsqueda bibliográfica se ha usado la base de datos de la Universidad de Valladolid. En esa búsqueda se han introducido las palabras clave en una búsqueda exhaustiva de artículos relacionados con el tema de este trabajo y se puede decir, que no se ha encontrado nada sobre el estudio teórico y diseño del sistema, puesto que la combinación de los elementos MCF

y RTQ de la que se habla antes es algo innovador. Se encuentran en cambio artículos y libros que hablan sobre ambos pero por separado, es el caso de *“Thermal energy storage for low and medium temperature applications using phase change materials”* (Cunha Jose, (2016)) sobre los MCF y *“Reduction kinetics for large spherical 2:1 iron–manganese oxide redox materials for thermochemical energy storage”* (Hamidi (2019)) sobre los RTQ. Todos están disponibles en la base de datos de la Uva y se muestran en la bibliografía.

Si se busca sobre la programación del modelo de control en MATLAB/Simulink y EC-GFX sí es cierto que se encuentran numerosos ejemplos de programación, ya que es un control basado en máquina de estados, muy usado en dinámica de sistemas. Se tiene por ejemplo el libro *“Sistemas de control automático”* (C. Kuo (1996)), y se adjuntan en la bibliografía un par de referencias.

También se han encontrado artículos sobre la conexión Modbus entre Python y el SCADA que es algo que se ha realizado en este trabajo, un ejemplo es *“Accurate modeling of Modbus/TCP for intrusion detection in SCADA systems”* (Goldenberg (2013)).



## 2 Estudio teórico

En este apartado se incluye una pequeña introducción de la estructura que conforma el sistema, para a continuación enumerar sus principales modos de operación, y finalmente se describen sus condiciones de funcionamiento y especificaciones.

Para los modos de operación principales, primero se incluye una descripción básica de los pasos involucrados, y luego se presenta una descripción más detallada de los procesos subsiguientes.

### 2.1 Introducción a los modos de operación del sistema

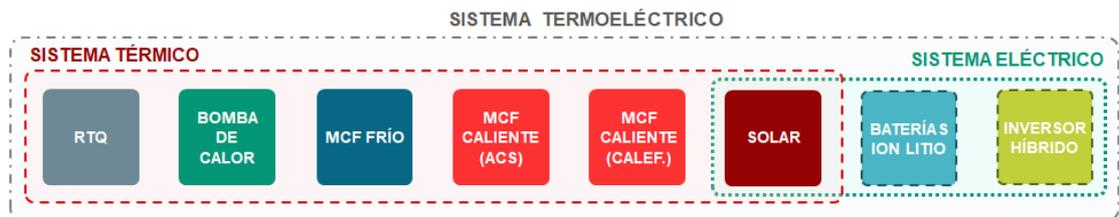


Figura 1: Sistema Global

Desde una perspectiva operativa, el sistema que se contempla en este trabajo está constituido por ocho subsistemas, tanto térmicos como eléctricos: RTQ (que incluye también el ciclo de amoníaco), bomba de calor, MCF caliente para calefacción, MCF caliente para ACS, MCF frío, sistema solar, inversor híbrido inteligente y baterías de iones de litio. Estos subsistemas están interconectados para proporcionar un funcionamiento dinámico del sistema.

A pesar de que tanto los sistemas térmicos como eléctricos operan de manera integrada (es decir, interactúan entre sí dependiendo de las condiciones dadas), ambos sistemas tienen sistemas de control independientes. Por tanto, la interacción de los sistemas eléctricos y térmicos es inherente a la naturaleza del subsistema solar: los módulos solares producen tanto electricidad, como calor.

Mientras la electricidad se gestiona en el inversor híbrido inteligente, que decide si es necesario almacenar energía eléctrica en las baterías de iones de litio o se puede consumir, ya sea en el edificio o en el sistema, el calor se envía al circuito térmico (que incluyen el almacenamiento de MCF, el RTQ y la bomba de calor), donde se requiere electricidad para el funcionamiento de los

diferentes dispositivos. Sin embargo, desde una perspectiva de control, tanto los sistemas térmicos como eléctricos pueden operar correctamente de manera aislada. El sistema eléctrico no requiere ninguna entrada operativa o de control del sistema térmico porque el control principal lo realiza el inversor híbrido inteligente, y el sistema térmico requiere electricidad, pero desde una perspectiva operativa, la fuente de esa electricidad no es relevante. Esta separación permite una configuración más robusta del sistema, así como una definición más sencilla de los modos de funcionamiento.

Este trabajo se centra únicamente en la parte térmica, por tanto, se contempla como sistema a controlar la parte térmica del sistema global descrito anteriormente.

## 2.2 Principales modos de operación térmicos

Los principales subsistemas del sistema térmico, así como sus principales interacciones térmicas posibles, se muestran en la Figura 2, aunque estas interacciones no ocurren al mismo tiempo. Consumos eléctricos asociados y dispositivos auxiliares adicionales para que todos los subsistemas interactúen entre sí (p. Ej. bombas, válvulas, ventiloconvectores), no se incluyen en esta representación simplificada. Este diagrama se utiliza como base para describir los conceptos básicos de los principales modos de funcionamiento en las secciones siguientes.

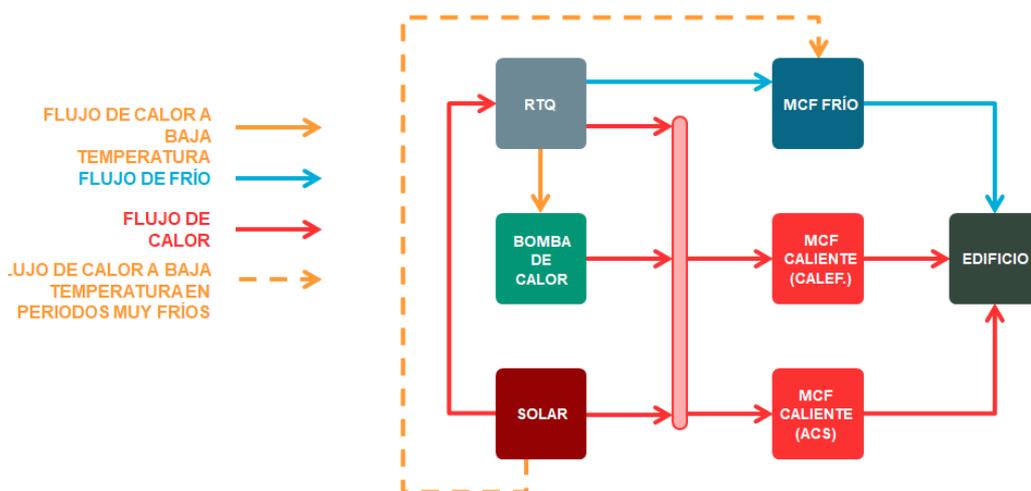


Figura 2: Esquema del Sistema Térmico

La Figura 3 representa el diseño del sistema térmico y contiene los componentes principales desde una perspectiva funcional, tanto dentro de los límites de cada subsistema, como entre los subsistemas, para conectarlos. Este diagrama se establece como base para mostrar una representación detallada de los principales modos de funcionamiento. Incluye la posición de las válvulas y los flujos térmicos activos, mostrando su dirección y temperatura de forma cualitativa, siguiendo el código de colores mostrado en la figura anterior.

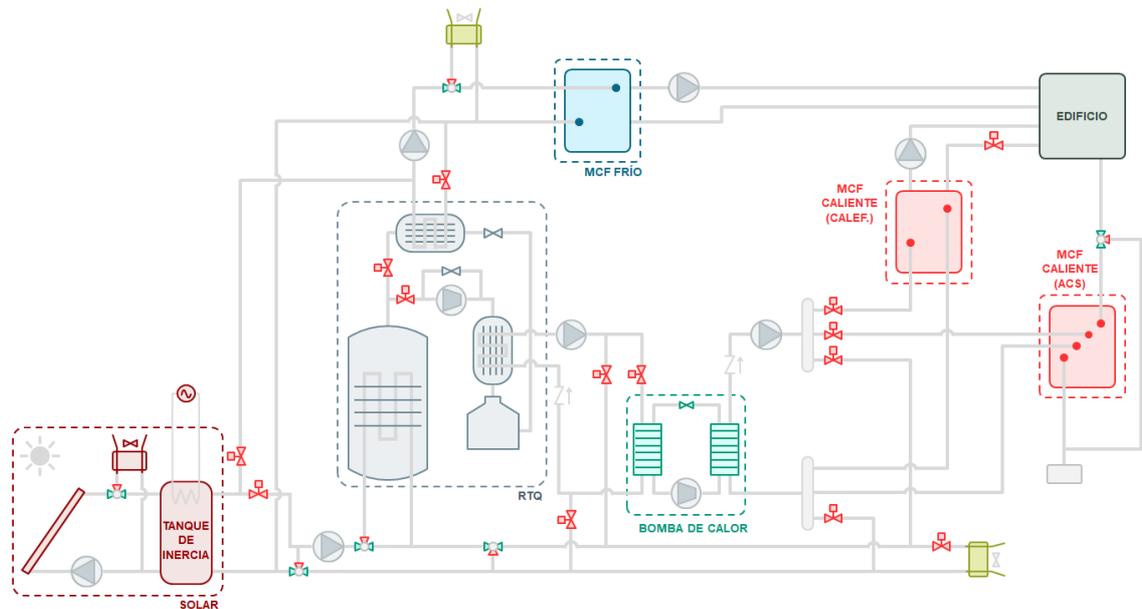


Figura 3: Esquema del Sistema térmico con componentes

A pesar de la complejidad del sistema térmico, todos los modos de operación se pueden clasificar en cuatro grupos principales, los cuales se resumen en la Figura 4. Como se puede observar, dentro del modo de operación “Descarga RTQ”, existen tres variaciones diferentes dependiendo de la temporada y las condiciones ambientales, pero todas comparten el mismo concepto como base.

Aparte de estos cuatro modos principales, puede producirse una gran cantidad de combinaciones de estos modos de funcionamiento o submodos, pero todos ellos podrían simplificarse en uno o más de estos modos principales. Por ejemplo, dentro del modo de descarga RTQ, la demanda térmica podría ser solo frío y ACS (mientras que no se necesita calefacción), demanda de calefacción y ACS pero no frío, o cualquiera de estos tipos de demandas por separado (es decir ACS, frío o calefacción), y un conjunto diferente de unidades MCF recibiría energía en consecuencia. Sin embargo, todas estas combinaciones se han considerado submodos del modo de descarga RTQ por razones de simplicidad.

Además, también pueden ocurrir algunas variantes que representan condiciones operativas específicas (por ejemplo, rechazo de calor usando un ventilador porque las unidades MCF están completamente cargadas, rechazo de calor por razones de seguridad). Todos estos modos de funcionamiento están sujetos a restricciones técnicas y de control. Se explica con más detalle en posteriores apartados.



Figura 4: Modos y Submodos de operación principales

A continuación, se resumen conceptualmente los modos y submodos principales:

- **Carga RTQ:** la energía producida en el sistema solar se introduce en el reactor RTQ para cargarlo; es decir, desencadenar la reacción que da como resultado la liberación de amoníaco, que se almacena en el depósito. Este proceso también produce calor en el condensador de amoníaco.
  - **Carga RTQ en invierno:** el calor producido en el condensador de amoníaco se introduce en el sistema de BC a través del evaporador de BC para aumentar la temperatura del flujo de energía, de modo que el flujo de salida en el condensador de BC cumpla las condiciones requeridas por las baterías de calor para almacenar su energía en las unidades MCF.
  - **Carga RTQ en verano:** el calor producido en el condensador de amoníaco se expulsa directamente al ambiente por defecto.
  - **Transición al modo de carga RTQ:** primera fase del modo de carga, que hace circular el agua contenida en el sistema hidráulico hasta que la temperatura del flujo de agua en la

entrada de las baterías térmicas alcanza la temperatura requerida para ser almacenada. Se espera que este modo ocurra después de que finalice la fase de descarga, o cuando la fase de carga se reanude después de haber estado en pausa.

- **Carga RTQ sin demanda:** el calor producido en el condensador de amoniaco no se puede almacenar y se expulsa directamente al ambiente. Este submodo es equivalente a la carga RTQ en verano, pero puede ocurrir en cualquier momento del año.
- **Descarga RTQ:** el amoniaco se libera del depósito y se recombina en el reactor RTQ. Como resultado, se produce frío y calor, en el evaporador de amoniaco y en el reactor RTQ, respectivamente.
  - **Descarga del RTQ en verano:** el frío se almacena en la batería fría (MCF frío), mientras que el calor liberado en el reactor RTQ se expulsa al ambiente mediante el ven.
  - **Descarga del RTQ en invierno:** el frío se expulsa al ambiente mediante el ventiloincador (circuito frío) y el calor producido en el reactor RTQ se almacena en las unidades MCF.
  - **Descarga del RTQ durante períodos muy fríos:** fase preliminar requerida antes de la fase de descarga del RTQ cuando la temperatura exterior es inferior a 5°C. El MCF frío se calienta hasta que se alcanza una cierta temperatura conectándolo con el tanque de inercia, para que la presión de trabajo sea lo suficientemente alta (compatible con la temperatura requerida para la descarga de calor de la unidad RTQ).
- **Transición** a modo descarga RTQ: primera fase del modo descarga, que hace circular el agua contenida en el sistema hidráulico hasta que la temperatura del flujo de agua en la entrada de la salida del reactor RTQ esté por encima de la requerida en las unidades MCF calientes, la temperatura del agua a la salida del evaporador de amoniaco está por debajo de la requerida en la unidad MCF fría, o ambas condiciones ocurren al mismo tiempo. Se espera que este modo ocurra después de que finalice la fase de carga, o cuando la fase de descarga se reanude después de haber estado en pausa.
- **Suministro de calor directo de la energía solar:** en los casos en que la temperatura del circuito solar sea lo suficientemente alta, la energía se

puede almacenar directamente en los MCF calientes. Es poco probable que este modo ocurra durante el invierno.

- **Demanda cubierta únicamente por MCF:** las demandas de calor (calefacción y / o ACS) y refrigeración del edificio están cubiertas únicamente por la energía almacenada en las baterías térmicas.

### 2.2.1 Modo de Operación: Carga RTQ

En este modo de funcionamiento, el recurso solar se utiliza para proporcionar el calor necesario para cargar el RTQ y, como resultado de este proceso, se proporciona calor adicional a las unidades de MCF calientes, que se pueden utilizar en el edificio. La producción de frío no ocurre en este modo.

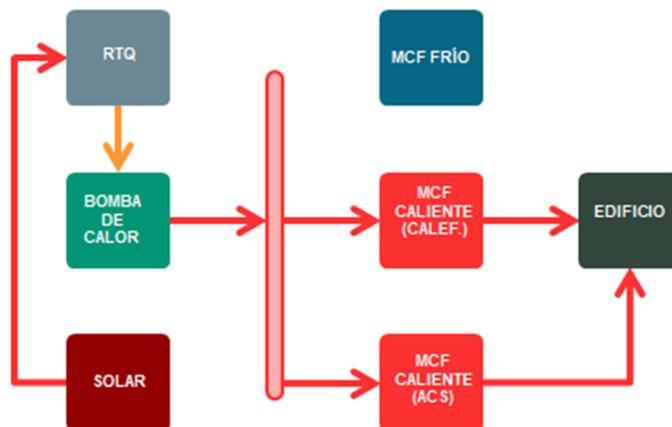


Figura 5: Esquema de operación Carga RTQ

Pasos principales:

1. El calor producido en el campo solar se utiliza en primer lugar para realizar el proceso de carga del sistema RTQ.
2. Como resultado, se produce un flujo de calor a baja temperatura y se envía a la bomba de calor.
3. En la bomba de calor, se aumenta la temperatura y el calor resultante se introduce en las unidades MCF calientes.
4. Finalmente, el calor se libera de las unidades MCF y se envía al edificio siempre que sea necesario.

Pasos detallados (Ver Figura 1 Anexo):

- 1.1 Primero, durante los períodos de suficiente insolación solar, la temperatura del FTC que fluye a través del campo solar aumenta.
- 1.2 El calor de los módulos solares fluye hacia el tanque de inercia, donde se almacena hasta que se alcanza un umbral de temperatura específico.
- 1.3 Una vez alcanzada esta temperatura mínima, el calor del sistema solar se suministra al RTQ durante el proceso de carga, mediante un circuito hidráulico que pasa por el reactor e intercambia calor con él.
- 2.1 El calor del reactor se utiliza para la reacción de descomposición, lo que hace que el amoníaco gaseoso salga del reactor.
- 2.2 El amoníaco se impulsa a través de un compresor, lo que también provoca un aumento de su temperatura
- 2.3 En el condensador RTQ, este calor del amoníaco se transfiere a un flujo diferente que está conectado al evaporador de la bomba de calor.
- 2.4 Después de este intercambio de calor, el amoníaco líquido se almacena en un depósito.
- 3.1 El calor recuperado del flujo de amoníaco, se conecta al evaporador de la bomba de calor, que aumenta su temperatura y lo libera a una línea hidráulica conectada al condensador de la bomba de calor.
- 3.2 Desde el condensador de la bomba de calor, el calor se envía a las unidades MCF calientes, donde se transfiere y almacena en el material MCF.
- 4.1 El calor que se almacena en las unidades MCF calientes se transfiere a los circuitos de calefacción y ACS del edificio para cubrir los requisitos térmicos.

### **2.2.2 Modo de Operación: Descarga RTQ**

Para este modo de funcionamiento, la condición previa es que el RTQ esté completamente cargado. Durante este proceso, el proceso RTQ se revierte, por

lo tanto, el amoníaco se reintroduce en el reactor. Como consecuencia de este proceso, se producen tanto frío como calor, que pueden almacenarse en las unidades MCF y posteriormente utilizarse en el edificio.

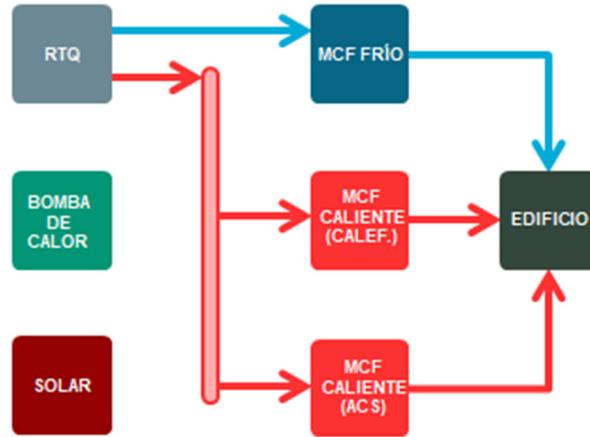


Figura 6: Esquema de operación Desarga RTQ

Pasos principales:

1. Durante la primera etapa del ciclo de descarga del RTQ, se produce frío en el evaporador del RTQ, que se envía a la unidad MCF fría durante el período de verano o períodos muy fríos en invierno, o se rechaza al ambiente a través de un serpentín débil durante períodos suaves en invierno.
2. Como segunda etapa, la reacción exotérmica en el reactor RTQ produce calor, que puede introducirse en las unidades MCF calientes en invierno o rechazarse al ambiente mediante un ventiloconvector en verano (no representado explícitamente en el diagrama básico anterior).
3. Tanto el frío como el calor se liberan de las unidades MCF cuando es necesario (frío en verano, solo), y se utilizan en el edificio para cubrir la demanda correspondiente.

Pasos detallados (Ver Figura 2 Anexo):

- 1.1 Una vez que se completa el ciclo de carga del RTQ (es decir, la reacción de descomposición finaliza), el amoníaco puede salir del

depósito y, posteriormente, su temperatura disminuye como resultado del proceso de expansión.

- 1.2 En el evaporador, el frío del amoníaco se intercambia con el circuito hidráulico que lo conduce al MCF frío o a un ventiloinvector, y luego el amoníaco entra al reactor.
- 1.3 El flujo que sale del evaporador pasa a la unidad MCF fría durante el verano o los períodos de invierno muy frío, donde se almacena el frío. En otros períodos de invierno, es decir, cuando la temperatura ambiente externa no es muy baja, el flujo del evaporador RTQ pasa a través de la bobina del ventilador exterior y el frío se rechaza al ambiente.
- 2.1 En el reactor RTQ, el amoníaco interactúa con las sales de cloruro de calcio, lo que da como resultado una reacción exotérmica.
- 2.2 El calor producido durante la reacción se transfiere a un flujo de paso, que envía este calor a los MCF calientes durante el invierno, donde se almacena. En verano, el calor liberado del reactor RTQ se expulsa al ambiente.
- 3.1 Como último paso, el frío y el calor de las baterías del MCF se envían al edificio para cubrir la demanda térmica de los sitios de demostración, es decir, refrigeración, calefacción y ACS. No obstante, cabe señalar que el escenario exacto no refleja ninguna situación realista. No se espera que la demanda de frío y calor ocurra durante el mismo ciclo de descarga, sino solo las demandas de calefacción y ACS, las demandas de frío y ACS, o cualquiera de las demandas por separado.

### **2.2.3 Descarga RTQ durante periodos inferiores a 5°C**

A diferencia de la mayor parte del año (es decir, los períodos de verano e invierno suave), para los cuales no se requieren pasos adicionales y solo pequeñas variaciones difieren según las condiciones dadas, durante los períodos muy fríos, se requiere un paso adicional antes del proceso descrito anteriormente (paso 0). Este paso permite asegurar las temperaturas y presiones requeridas para operar el reactor. Aparte de este procedimiento anterior, el resto de pasos son los mismos que los explicados en el apartado anterior.

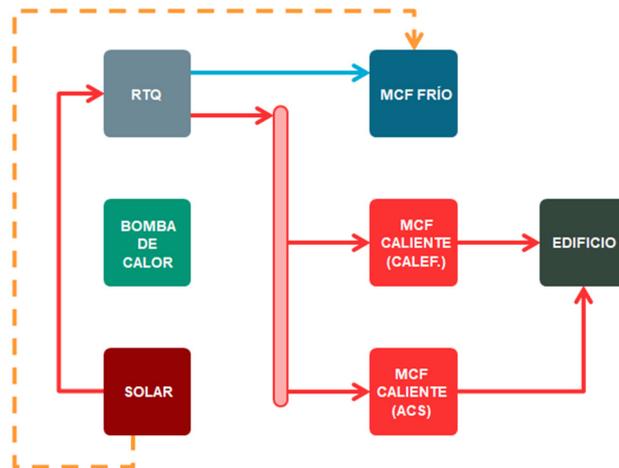


Figura 7: Esquema de operación Descarga RTQ en períodos muy fríos

Paso adicional:

0. En períodos muy fríos, el calor de baja temperatura del subsistema solar se envía al MCF frío para mantener una temperatura mínima dentro de la unidad.

Procesos principales dentro del paso adicional (Ver Figura 3 Anexo):

- 0.1 Durante los momentos en los que la radiación solar es baja, el calor ganado por los módulos solares es limitado, pero aún puede contribuir a aumentar la temperatura del FTC que fluye a través del circuito solar. Alternativamente, en los casos en los que la radiación solar es demasiado baja para alcanzar una temperatura mínima, se puede usar una resistencia eléctrica como calentador de respaldo durante un período de tiempo limitado.
- 0.2 El calor del campo solar o del calentador eléctrico aumenta la temperatura general del FTC almacenado dentro del tanque de inercia.
- 0.3 Un circuito hidráulico conectado a la unidad MCF fría envía calor a ella, de modo que siempre se mantenga una temperatura mínima, por lo tanto, la presión en el evaporador RTQ se puede aumentar durante el proceso de descarga, y la temperatura requerida en el reactor RTQ permanece dentro de límites apropiados para cargar los MCF calientes.

## 2.2.4 Modo de Operación: Demanda cubierta por energía solar

Este modo de funcionamiento consiste principalmente en el uso directo de energía de los subsistemas solares para el proceso de carga de MCF calientes. Dados los requisitos de temperatura en las unidades MCF y el recurso solar limitado durante los períodos invernales, este modo de operación ocurre, salvo situaciones extraordinarias, durante el período estival. En consecuencia, el calor se utiliza principalmente en el MCF caliente para ACS, y es poco probable que se utilice en el MCF caliente para calefacción.

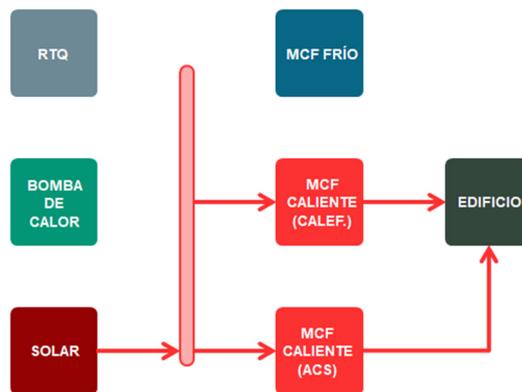


Figura 8: Esquema de operación Demanda cubierta por energía solar

Pasos principales:

1. En los casos en que el subsistema RTQ está completamente cargado y hay suficiente energía solar disponible a la temperatura requerida, el calor del campo solar puede enviarse a los MCF calientes.
2. El calor almacenado en las unidades MCF calientes se envía al edificio cuando es necesario, para cubrir las necesidades térmicas.

Pasos detallados (Ver Figura 4 Anexo):

- 1.1 Siguiendo el mismo procedimiento que el correspondiente al modo de funcionamiento “Carga RTQ”, se aumenta en primer lugar la temperatura del FtC que fluye a través del campo solar, durante los períodos de suficiente insolación solar.

- 1.2 A continuación, el calor de los módulos solares fluye hacia el tanque de inercia, donde se almacena hasta que se alcanza un umbral de temperatura específico.
- 1.3 El calor del tanque de inercia se conduce a las unidades MCF calientes (una o ambas) donde se almacena.
- 2.1 El calor de las baterías del MCF se envía al edificio para cubrir la demanda de calor de los sitios de demostración, es decir, la demanda de ACS y calefacción.

### 2.2.5 Modo de Operación: Demanda cubierta por las baterías solo

En los casos en que no se disponga de subsistemas tanto solar como RTQ, se pueden utilizar unidades de almacenamiento térmico MCF para cubrir la demanda del edificio, utilizando la energía que previamente se ha introducido en ellos.

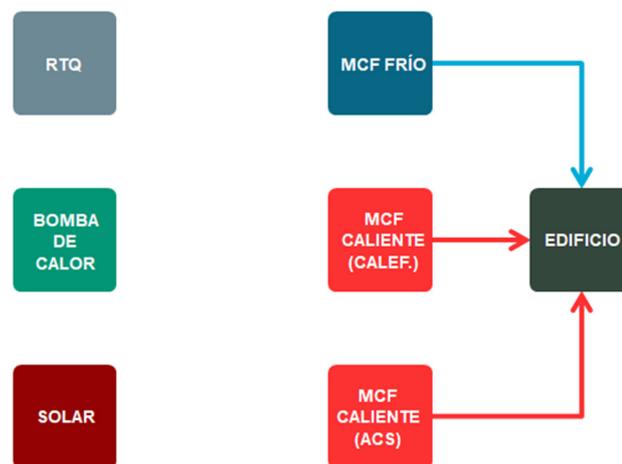


Figura 9: Esquema de operación Demanda cubierta por baterías

Paso principal:

1. El calor y el frío almacenados en las unidades MCF se pueden enviar al edificio para cubrir las necesidades térmicas. El resto de subsistemas no participan en este modo de operación.

Pasos detallados (Ver Figura 5 Anexo):

- 1.1 Las unidades MCF constituyen dispositivos de almacenamiento térmico que pueden utilizarse para cubrir la demanda en los edificios. En los casos en que la energía en el subsistema solar no esté disponible y el RTQ no esté completamente cargado o se prefiera no iniciar el proceso de descarga, las unidades MCF pueden liberar el calor y el frío almacenados intercambiando la energía disponible para el consumo térmico del circuito del edificio.

## 2.3 Descripción general de condiciones de funcionamiento y especificaciones del sistema

### 2.3.1 Factores que dan forma a las condiciones operativas del sistema

Este sistema se caracteriza por unas condiciones de funcionamiento bastante estables en cada modo de funcionamiento, que sin embargo varían entre los períodos de verano e invierno. Esto se atribuye a que estas condiciones vienen impuestas por un lado por las especificaciones de los subsistemas de almacenamiento y por otro lado por las condiciones ambientales, ya que existe un intercambio de calor frecuente entre el sistema y el medio ambiente. La Tabla 1 resume las temperaturas características de los subsistemas de almacenamiento incorporados.

| Subsistema de almacenamiento    | Temperatura característica | Descripción   |
|---------------------------------|----------------------------|---|
| RTQ                             | 60°C                       | Temperatura de entrada necesaria de FtC para lograr la densidad de almacenamiento de energía deseada de 200 kWh / m <sup>3</sup>              |
| MCFCALOR<br>(Calefacción y ACS) | 58°C                       | Temperatura de cambio de fase del MCF preferido. Alternativamente, se puede utilizar un MCF diferente con un punto de fusión más bajo (48 °C) |
| MCF FRIO                        | 11°C                       | Temperatura de fusión del MCF preferido. Alternativamente, se puede utilizar un MCF diferente con un punto de fusión más bajo (5,5°C).        |

Tabla 1: Temperaturas características de MCF Y RTQ

Las temperaturas de cambio de fase de los MCF imponen condiciones previas a la temperatura de entrada de FtC. Por lo tanto, para cargar, es decir, fundir, el MCF caliente se requiere una temperatura de entrada de 63 °C, mientras que para cargar, es decir, solidificar, el MCF frío, el valor máximo correspondiente es de aproximadamente 8 °C. La Tabla 2 describe las principales interacciones entre los diversos subsistemas y el ambiente. Como puede verse fácilmente, estos intercambios de calor solo se refieren al subsistema de almacenamiento RTQ, sin embargo, no se limitan a uno solo de sus componentes, sino que involucran a varios de ellos. En general, se desea un funcionamiento sin obstáculos del sistema para condiciones ambientales de 35°C en verano y -10°C en invierno.

| Subsistema | Componente del Subsistema | Periodo normal de interacción |
|------------|---------------------------|-------------------------------|
| RTQ        | reactor RTQ               | Verano                        |
| RTQ        | condensador RTQ           | Verano                        |
| RTQ        | evaporador RTQ            | Invierno                      |

Tabla 2: Principales interacciones según periodo

## 2.3.2 Condiciones de funcionamiento en los modos de funcionamiento principales definidos

### 2.3.2.1 Condiciones de funcionamiento en modo de carga RTQ

Durante el modo de carga del RTQ en invierno, las condiciones de funcionamiento del sistema están determinadas por las temperaturas características del RTQ y el MCF. En primer lugar, el suministro de calor al RTQ debe estar en el rango de 60°C, si se quiere lograr una densidad de almacenamiento de energía de 200 kWh/m<sup>3</sup>. Esto a su vez significa que la temperatura y la presión de equilibrio del reactor son de aproximadamente 55°C y 2 bar respectivamente (Ver Figura 6 Anexo). Por otro lado, la temperatura de cambio de fase de los MCF calientes de 58°C significa que el disipador de calor del BC, que actúa como el “transformador de energía” entre los subsistemas RTQ y los MCF calientes, debe ser de aproximadamente 63°C. Las condiciones intermedias de condensación de amoníaco (28°C, aprox. 11 bar), se definen por dos factores:

- i. Las características de los compresores de amoníaco disponibles comercialmente, es decir, su relación de compresión, que normalmente

no es superior a 6 bar y, por tanto, para una presión de aspiración de 2 bar determinan una presión de condensación máxima de 12 bar

- ii. Las características de las bombas de calor disponibles, ya que las máquinas que son capaces de proporcionar las condiciones de disipador de calor requeridas por los MCF requieren temperaturas de la fuente de calor en el rango de 20-30°C. Para lograr valores altos de CDR de BC, deben evitarse temperaturas de condensación muy bajas.

En el modo de carga RTQ en verano, la temperatura característica de los MCF calientes no impone ninguna restricción en las condiciones de funcionamiento del sistema, ya que se prevé que estos recipientes se carguen con el suministro directo del modo de funcionamiento solar. Esto también significa que el funcionamiento de BC para la actualización térmica no es necesario. Sin embargo, las restricciones son impuestas por las condiciones exteriores y más particularmente por la necesidad de rechazar el calor de condensación del RTQ al ambiente (Ver Figura 7 Anexo). Considerando una temperatura ambiente de 35°C, la condensación de amoníaco debe realizarse a una temperatura de alrededor de 40°C y una presión correspondiente de 16 bar. Así, las condiciones de equilibrio dentro del reactor deben ser de al menos 3 bar y 63-64°C, que a su vez requieren un aporte de calor solar de aproximadamente 68°C. Dado que en verano la radiación solar presenta sus valores anuales más altos, un prerrequisito de un suministro de calor de 70°C al RTQ es razonable, lo que reduciría aún más la relación de compresión y el consumo eléctrico del compresor de amoníaco. Los MCF de frío y ACS, si se cargan, pueden descargar cubriendo las necesidades energéticas correspondientes del edificio.

### **2.3.2.2 Condiciones de funcionamiento en modo de descarga RTQ**

En el modo de descarga del RTQ de invierno, el RTQ debe generar calor a una temperatura adecuada para realizar la carga de los MCF calientes. Así, la temperatura de cambio de fase de este último es el único límite de las condiciones de funcionamiento del sistema, traducido en una temperatura mínima de equilibrio dentro del reactor de 63-65°C (Ver Figura 8 Anexo). La presión correspondiente es de aproximadamente 4.5-5 bar, mientras que la presión de evaporación debe ser un poco más alto que este valor para permitir un flujo de amoníaco gaseoso hacia el reactor. Sin embargo, la presión de evaporación también está relacionada con la interacción entre el evaporador y el medio ambiente del RTQ, ya que este último es la fuente de calor del proceso de evaporación del amoníaco. Para una presión de evaporación de 4,5 bar, la temperatura correspondiente es de unos 2°C, lo que da como resultado una temperatura ambiente mínima de 5°C, mientras que estos valores son incluso

superiores (4 y 7°C respectivamente) para una presión de evaporación de 5 bar. Por lo tanto, ambas condiciones límite solo pueden satisfacerse si la temperatura ambiente es superior a 5°C. En un caso diferente, la descarga del RTQ no podría producir calor a la temperatura adecuada para cargar los MCF calientes y, en consecuencia, se desperdiciaría la energía almacenada.

Para superar este problema, considerando también que la necesidad de utilizar la energía almacenada y confinar los picos de demanda de calor coincidiría razonablemente con temperaturas ambiente bajas, se prevé el submodo de descarga RTQ durante períodos muy fríos. Como se describe en la Figura 6, este submodo implica la carga del MCF frío utilizando calor solar a baja temperatura. Teniendo en cuenta la temperatura de cambio de fase relevante de 11°C, el suministro de calor del campo solar debe ser superior a 16°C. Ésta es una condición previa que puede cumplirse con relativa facilidad incluso en períodos de mucho frío. Como el evaporador del RTQ utiliza el calor solar almacenado en el MCF frío como fuente de calor de evaporación, la temperatura del MCF fría característica sustituye a la temperatura ambiente como el segundo límite de las condiciones de funcionamiento del sistema como se muestra (Ver Figura 9 Anexo). Esto se traduce en un calor suministrado al evaporador a una temperatura media de aproximadamente 7-8°C, lo que permite la generación de calor del RTQ a una temperatura lo suficientemente alta para cargar los MCF calientes. La única limitación en este caso es la capacidad del MCF frío, que puede no ser lo suficientemente alta para proporcionar la energía necesaria para una descarga completa del RTQ.

De manera similar, para la descarga de RTQ durante períodos muy fríos, en verano el evaporador de RTQ está conectado al recipiente de MCF frío. Sin embargo, en este caso el propósito principal de esta interacción es producir el frío necesario para cargar el subsistema MCF frío. Una temperatura de evaporación de 0°C, considerada suficiente para la implementación de este intercambio de calor, corresponde a condiciones de equilibrio del reactor de aproximadamente 4 bar y 58°C (Ver Figura 10 Anexo). Dado que este último valor no es lo suficientemente alto para permitir la carga de los MCF calientes, el calor generado durante este modo de funcionamiento se rechaza al ambiente. Esta interacción entre el medio ambiente y el reactor RTQ es el otro límite de la operación del sistema, lo que significa que el proceso de evaporación se puede realizar a temperaturas tan bajas como -10°C, sin obstaculizar la liberación de calor del RTQ al ambiente.

### **2.3.2.3 Condiciones de funcionamiento en modos demanda cubierta por solar y cubierta por baterías**

Como se puede observar en la Figura 8 y la Figura 4 de Anexos, en el suministro directo desde el modo de operación solar, los únicos subsistemas térmicos involucrados son el campo solar junto con los MCF calientes. Por lo tanto, la temperatura característica de este último domina las condiciones de funcionamiento del sistema, como se observa (Ver Figura 11 Anexo). Es interesante el hecho de que un suministro de calor de campo solar realizado a una temperatura de 70°C, según lo determinado en el modo de carga de verano RTQ, puede cumplir con los requisitos de carga de los MCF calientes también.

Finalmente, el caso en el que la demanda está cubierta solo por los MCF no impone ninguna restricción a las condiciones de funcionamiento de los componentes del sistema térmico en reposo. Los recipientes MCF intercambian calor o frío con el sistema de calefacción y refrigeración del edificio, respectivamente, y solo el caudal máximo y el umbral de presión de trabajo de los recipientes limitan la velocidad de esta transferencia de energía. Obviamente, esto último ocurre a una temperatura definida por las temperaturas características del MCF (máx. 50-55 °C para los MCF calientes y mín. 14-18°C para los fríos)

### **2.3.3 Especificaciones del sistema principal**

Del análisis anterior se puede deducir que no se espera que el reactor RTQ y el evaporador experimenten presiones superiores a 5-6 bar en condiciones normales de funcionamiento. Por el contrario, en el tramo comprendido entre la descarga del compresor y el depósito de amoníaco líquido se consideran presiones que van desde 12 hasta 16 bar. En consecuencia, las especificaciones del compresor RTQ, el condensador y el tanque de amoníaco, junto con las de las válvulas y tuberías de conexión, deben garantizar un funcionamiento sin fugas bajo presiones tan altas. Esto es de suma importancia, especialmente considerando la toxicidad del amoníaco como se describe en el Contexto de este proyecto.

En cuanto a las temperaturas desarrolladas en el subsistema RTQ, sus valores normales no superarían los 65/75°C dependiendo de la reacción que se produzca. La temperatura máxima de condensación estaría en el rango de 28-30°C, mientras que la de evaporación podría ser tan baja como -15 °C. Sin embargo, debido a la alta temperatura del amoníaco gaseoso que sale del reactor, los valores correspondientes en la descarga del compresor serían tan altos como 230°C. Este último valor está muy por encima de la temperatura

máxima de funcionamiento de los compresores de amoníaco disponibles comercialmente, por lo que es necesario que el vapor de amoníaco se enfríe hasta alrededor de  $-15^{\circ}\text{C}$  en la descarga del compresor. Esto se realiza mediante un sistema de inyección de amoníaco líquido, potencialmente en combinación con un enfriador de gas. El funcionamiento suave y sin problemas de estos sistemas es importante para evitar daños en el compresor, lo que a su vez dificultaría la implementación del modo de carga RTQ.

Las especificaciones de BC tal como están configuradas por el modo de operación de carga del RTQ son de invierno, ya que no se prevé que la operación de este subsistema ocurra en otros modos del sistema. Como se mencionó anteriormente, BC debe combinar un disipador de calor de  $63^{\circ}\text{C}$  con una fuente de calor de evaporación en el rango de  $22\text{-}26^{\circ}\text{C}$ . El CDR de BC bajo estas condiciones de operación debe ser lo más alto posible para minimizar el consumo eléctrico del sistema y así lograr una alta eficiencia general. Igualar las capacidades de los subsistemas BC y RTQ también es importante para garantizar un funcionamiento adecuado y eficiente del sistema. Por el contrario, los MCF (fríos y calientes por igual) son los límites del sistema térmico hacia el edificio y, por lo tanto, sus especificaciones se utilizan para estimar las de otros componentes internos del sistema. De manera similar, las especificaciones de los MCF con respecto a su presión de trabajo y el caudal permitido deben tenerse en cuenta al diseñar la conexión del sistema térmico con el sistema de calefacción y refrigeración del edificio. Finalmente, el requisito de RTQ para el suministro de calor a una temperatura específica es una restricción importante que debe tenerse en cuenta al definir el diseño del campo solar y elegir los modelos de PFT y colectores solares. Sin embargo, es necesario incluir equipos (disipadores de aire, válvulas de alivio de presión, etc.) para mantener las temperaturas y presiones del calor desarrollado en el campo solar dentro de los umbrales especificados para evitar daños en los paneles o los colectores.

#### **2.3.4 Restricciones y condiciones operativas por subsistema**

Debido a la complejidad del sistema, se definieron cuestiones sobre el funcionamiento de cada sistema de forma individual, y el sistema térmico en su conjunto. A pesar de que alguna información ya se ha explicado, a continuación, se incluyen las principales conclusiones que afectan al sistema de control:

### **2.3.4.1 Sistema de captación solar**

El tanque de inercia incluye una resistencia eléctrica en el nivel superior, que se utiliza como calentador de respaldo en los casos en que la temperatura en el tanque sea alta, pero no lo suficiente para activar el proceso de carga del RTQ (ciclo de amoníaco). A pesar de que este calentador consume electricidad (y por lo tanto utiliza energía de la red o los paneles fotovoltaicos), se espera un uso óptimo para aprovechar el calor de "baja temperatura" en los casos en que, de lo contrario, este calor se desperdiciaría en forma de pérdidas térmicas.

Como enfoque inicial, se propone programar la resistencia eléctrica para que funcione de la siguiente manera:

- 1) Todos los días, en un período de tiempo específico, se lee la temperatura dentro del tanque de inercia de 14:00 a 15:00.
- 2) En caso de que esta temperatura sea superior a 50°C, pero inferior a 60°C, se activa la resistencia eléctrica.
- 3) El funcionamiento de la resistencia eléctrica se limita a una hora como máximo. Por lo tanto, en los casos en que las condiciones activen el calentador de respaldo, se apaga después de una hora. Por lo tanto, tomando como ejemplo, si la temperatura en el tanque de inercia alcanza los 50°C, por ejemplo, a las 14:34 (siempre dentro del período de tiempo especificado), se apaga a las 15:34.

El sistema solar también incluye un disipador de aire (ventiloconvector) por razones de seguridad, que se activa siempre que se alcance la temperatura máxima aceptable dentro del PFT o los colectores de placa plana, CPP (es decir, 85 °C o 145 °C, respectivamente).

### **2.3.4.2 Sistema RTQ**

El reactor RTQ requiere un flujo de entrada a temperaturas iguales o superiores a 55°C durante la fase de carga. Dado que se considera una histéresis, la temperatura mínima para conectar el circuito solar con el reactor RTQ es de 60°C, pero el sistema de control no cierra la conexión hasta que la temperatura de flujo sea de 55°C.

Los procesos de carga y descarga se pueden pausar en cualquier momento (por ejemplo, si el recurso solar es demasiado bajo) y se pueden reanudar más tarde, pero el proceso de carga debe completarse por completo antes de pasar al proceso de descarga, y viceversa. El cambio del proceso de carga al proceso

de descarga (y viceversa) puede ocurrir inmediatamente después de que se completa uno de los procesos, sin restricciones de tiempo ni pérdida de capacidad de almacenamiento.

El proceso de descarga dentro del RTQ (ciclo de amoníaco) es activado por el sistema de control del RTQ siempre que haya una llamada externa para calentar o enfriar, tan pronto como el RTQ esté completamente cargado. Luego, el reactor comienza a absorber amoníaco del evaporador y comienza la reacción exotérmica. Al mismo tiempo, la bomba de circulación conectada a la entrada del reactor RTQ se activa para enfriar el reactor y el agua fluye a través del circuito de agua del RTQ.

Durante la primera fase del proceso de descarga de RTQ, el calor recuperado puede tener una temperatura más baja que la del interior de las unidades de MCF calientes. Por lo tanto, el agua no ingresa a las unidades de intercambio de calor MCF, sino que se recircula hasta que la temperatura sea lo suficientemente alta. Como se explicó anteriormente, esto se ha considerado como un submodo de funcionamiento del sistema térmico.

Como se explicó antes el proceso de descarga se vería afectado por temperaturas ambiente por debajo de 5 °C: la temperatura y la presión en el evaporador de amoníaco serían más bajas y esta presión se impondría al reactor RTQ. En consecuencia, el reactor funcionaría a una temperatura más baja y la temperatura de descarga podría no ser lo suficientemente alta para usarse en los MCF calientes (es decir, 63 °C). Para evitar esto, se ha propuesto un nuevo submodo en el que el MCF frío se calienta hasta que se alcance una cierta temperatura, es decir, al menos 5 °C por encima de la temperatura de fusión del MCF, y estar conectado al evaporador de amoníaco durante estos períodos, de modo que la presión de trabajo sea lo suficientemente alta (compatible con la temperatura requerida para la descarga de calor de la unidad RTQ). Según las especificaciones, la temperatura de fusión del MCF frío es de 5 °C, por lo que se debe alcanzar una temperatura mínima de alrededor de 10 °C. Para asegurar esto, la activación del proceso de fusión del MCF frío puede ocurrir durante períodos en los que la temperatura dentro del tanque de inercia está entre 10 y 25 °C.

Cuando está cargado, el sistema de control interno del RTQ (ciclo de amoníaco) emite una señal que indica que el sistema está completamente cargado. Si continúa enviando calor al reactor, se considera un modo de falla.

Si el RTQ se carga a una temperatura más alta, también funciona a una temperatura más alta. En los casos en que la temperatura ambiente es alta (por ejemplo, 35 °C), la relación de compresión se puede aumentar para alcanzar también una temperatura más alta en el condensador RTQ (por

ejemplo, 40-45 °C). Tan pronto como se respete una relación de compresión de 6 para el compresor de amoníaco, la variación de la presión de condensación mediante un control dinámico de la relación de compresión puede mitigar el efecto de las altas temperaturas ambientales. Si la relación de compresión requerida es demasiado alta, el proceso de carga del RTQ solo se pausa (no opera).

La presión en el reactor RTQ es impulsada por su temperatura, por lo tanto, una temperatura más alta en el circuito solar significa una presión más alta en el reactor RTQ. Por ejemplo, en invierno, el calor fluido portador del tanque de inercia está a una temperatura mínima de 55-60 °C cuando ingrese al reactor RTQ, lo que corresponde a una presión en el reactor de alrededor de 2 bar. En verano, con temperaturas más altas del circuito solar, el reactor RTQ funciona a alrededor de 3 bar. En consecuencia, se podría alcanzar una presión de alrededor de 18 bar después del compresor, en el condensador.

La relación de compresión variable implica un sistema de control dinámico dentro del sistema RTQ, que debe tenerse en cuenta durante la fase de diseño de este sistema. En particular, siempre que la temperatura requerida en el condensador sea demasiado alta, el sistema de control debe enviar una señal que indique que se necesita una relación de compresión superior a la relación máxima aceptable, por lo que debe dejar de funcionar. El sistema de control maestro necesita recibir una señal para conocer esta situación, de modo que se integre en el funcionamiento general del sistema.

#### **2.3.4.3 Baterías de calor y frío (unidades MCF)**

Las unidades MCF calientes (baterías de calor) pueden proporcionar calor a la temperatura requerida sin demoras tan pronto como no se agoten por completo. Además, según el fabricante, el MCF caliente seleccionado no ha mostrado pérdida de rendimiento después de funcionar durante el equivalente a 50 años de uso doméstico normal (40.000 ciclos). No se espera que el MCF frío se degrade.

La temperatura de funcionamiento de las unidades MCF calientes es de 58 °C, por lo que la temperatura mínima que se debe proporcionar a las unidades MCF calientes (y comenzar a fundirlas) es de 63 °C, mientras que la temperatura máxima es de 80 °C.

#### **2.3.4.4 Bomba de calor**

La bomba de calor no puede funcionar en modo de refrigeración según la disposición del sistema, sino que solo puede proporcionar calor utilizando

como entrada el calor a baja temperatura liberado en el condensador de amoniaco.

#### **2.3.4.5 Integración del sistema y el sistema existente en el edificio**

Está previsto instalar una válvula termostática mecánica, que no requiere ningún control una vez puesto en marcha, en el circuito de ACS. Esta válvula asegura que la temperatura del ACS no sea demasiado alta conectando el flujo de ACS con la entrada principal de agua.

#### **2.3.5 Operación general**

El funcionamiento del sistema térmico prioriza siempre el uso del recurso solar disponible en cada momento.

Durante el proceso de carga de las unidades MCF calientes, se da prioridad al MCF de calefacción. Por lo tanto, el MCF de calefacción se carga completamente antes de que el MCF de ACS comience a cargarse.

El sistema térmico solo genera frío durante el proceso de descarga de la unidad RTQ. En consecuencia, en caso de que exista demanda de llamada, se prioriza el uso de frío del MCF, seguido de la descarga del RTQ (una vez que haya una llamada de frío).

Durante el verano, el calor producido por el sistema RTQ siempre es rechazado al ambiente. La razón principal de esto es producir frío a la temperatura más baja posible. En caso de que se decidiera utilizar el calor del proceso de descarga del RTQ para cargar los MCF calientes, se requeriría una temperatura más alta en el reactor RTQ, lo que conduciría a una temperatura más alta del frío generado en el evaporador que podría no ser suficiente para solidificar el MCF frío. Además, durante el verano, la unidad MCF de calefacción no se carga.

En caso de que la unidad MCF de ACS requiera ser calentada durante el verano, el sistema solar es el sistema responsable de proporcionar ese calor (no el proceso de descarga). En consecuencia, el proceso de descarga del RTQ y la carga del MCF caliente para ACS (con calor del circuito solar) nunca ocurran al mismo tiempo.

En invierno, la energía solar no se utiliza para cargar directamente los MCF calientes, a menos que el RTQ esté completamente cargado y todavía se tenga suficiente calor disponible en el circuito solar. En general, este escenario es improbable, pero se ha considerado en la definición del sistema de control.

Además, durante el invierno, la unidad MCF fría no funciona como almacenamiento en frío, sino que se utiliza como un "disipador de calor" para compensar temperaturas ambiente demasiado bajas y alcanzar una temperatura mínima en el evaporador (consulte la sección RTQ).

Las consignas de EDC de las distintas baterías térmicas, por encima de las cuales se envía una llamada de calefacción o refrigeración, se han definido de la siguiente manera:

- MCF frío: agotamiento del 50% / 90% para activar la llamada al frío.
- ACS MCF: 10% EDC (agotamiento del 90%) para activar la llamada de calefacción.
- MCF de calefacción: 50% EDC (50% de agotamiento) para activar la llamada de calefacción.

Como resultado de este análisis, se definieron las principales entradas, salidas y parámetros, los cuales se incluyen en la Figura 11. Asimismo, se definieron los principales límites de operación y condiciones a ser consideradas por el sistema de control global, que también están integrados en la Figura 10. Lo que se presentan en las siguientes figuras son los elementos que son imprescindibles para llegar a un sistema de control plenamente funcional, según esta valoración teórica. Cabe destacar también que la definición teórica de la consigna del EDC no es implementada en la práctica de este trabajo.

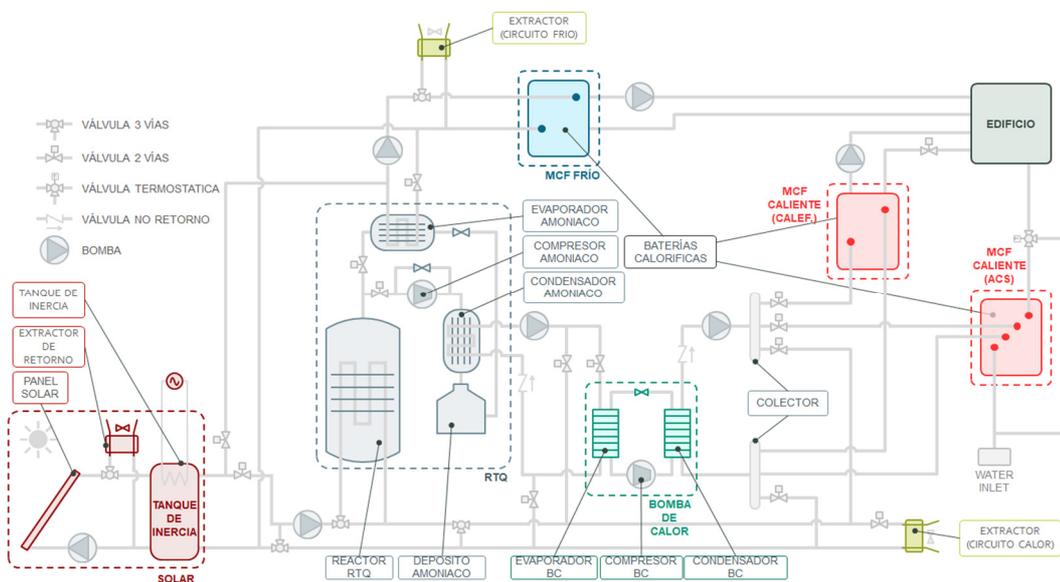


Figura 10: Esquema de componentes del Sistema Térmico



### 3 Modelado del Sistema

Para modelar el sistema en los entornos de programación se tiene que crear y definir una serie de variables y parámetros que se muestran en la Figura 11. En este apartado se define sus nombres, a qué bus de entrada o salida pertenecen y para qué se han definido. Importante señalar que el nombre de las variables está en inglés ya que es el idioma original en el que se ha definido el diseño del sistema. Para facilitar la comprensión en la tabla de abreviaturas se incluyen las abreviaturas también en inglés. También se incluyen las variables y parámetros de las baterías, aunque no se implemente en este control, y se podrían usar para un control más completo.

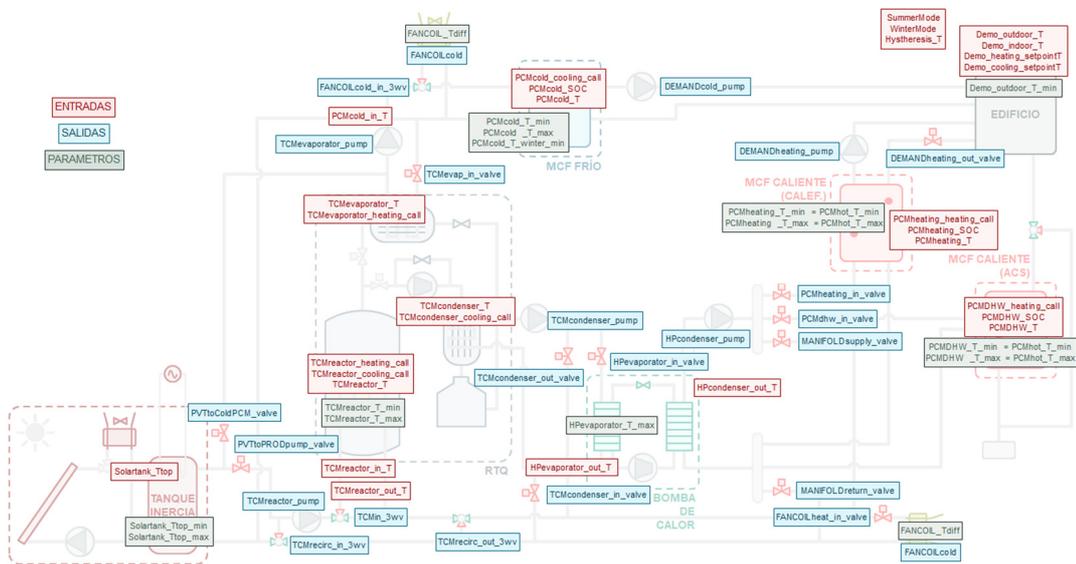


Figura 11: Esquema de componentes y variables del Sistema Térmico

#### 3.1 Variables de entrada, salida y globales

##### 3.1.1 Variables de Entrada:

- **Entradas\_Solar:**
  - *Solartank\_Ttop* (°C): Es una variable tipo *double* que indica la temperatura presente en el tanque de inercia.
- **Entradas\_Demo:**
  - *Demo\_outdoor\_T* (°C): Es una variable tipo *double* que indica la temperatura ambiente fuera del edificio.

- *Demo\_indoor\_T* (°C): Es una variable tipo *double* que indica la temperatura ambiente dentro del edificio.
- *SummerMode*: Es una variable tipo *booleana* que indica que el sistema está en periodo verano.
- *WinterMode*: Es una variable tipo *booleana* que indica que el sistema está en periodo invierno.
- *Demo\_heating\_setpointT* (°C): Es una variable tipo *double* que indica la temperatura fija dentro del edificio durante el invierno dada por el usuario.
- *Demo\_cooling\_setpointT* (°C): Es una variable tipo *double* que indica la temperatura fija dentro del edificio durante el verano dada por el usuario.
- *Demo\_heating\_demand*: Es una variable tipo *booleana* que indica la existencia de demanda de calor por parte del edificio
- *Demo\_cooling\_demand*: Es una variable tipo *booleana* que indica la existencia de demanda de frio por parte del edificio.
- **Entradas\_RTQ:**
  - *TCMcondenser\_T* (°C): Es una variable tipo *double* que indica la temperatura dentro del condensador del ciclo de amoniaco
  - *TCMreactor\_heating\_call*: Es una variable tipo *booleana* que indica que se requiere calor por parte del RTQ para iniciar el ciclo de carga.
  - *TCMreactor\_cooling\_call*: Es una variable tipo *booleana* que indica que se requiere frio por parte del RTQ para iniciar el ciclo de descarga.
  - *TCMreactor\_in\_T* (°C): Es una variable tipo *double* que indica la temperatura que entra en el reactor.
  - *TCMreactor\_out\_T* (°C): Es una variable tipo *double* que indica la temperatura que sale del reactor.

- *TCMevaporator\_T* (°C): Es una variable tipo *double* que indica la temperatura dentro del evaporador.
- **Entradas\_calorMCF:**
  - *PCMheating\_heating\_call*: Es una variable tipo *booleana* que indica al RTQ que la batería de calor calefacción requiere calor para cargarse porque esta baja.
  - *PCMheating\_SOC* (%): Es una variable tipo *double* que estima el porcentaje de EDC de la batería calor calefacción.
  - *PCMheating\_T* (°C): Es una variable tipo *double* que indica la temperatura en la batería calor calefacción.
- **Entradas\_frioMCF:**
  - *PCMcold\_in\_T* (°C): Es una variable tipo *double* que indica la temperatura que entra a la batería frío.
  - *PCMcold\_cooling\_call*: Es una variable tipo *booleana* que indica al RTQ que la batería frío requiere frío para cargarse porque esta baja.
  - *PCMcold\_SOC* (%): Es una variable tipo *double* que estima el porcentaje de EDC de la batería frío.
  - *PCMcold\_T* (°C): Es una variable tipo *double* que indica la temperatura en la batería frío.
- **Entradas\_ACSMCF:**
  - *PCMDHW\_heating\_call*: Es una variable tipo *booleana* que indica al RTQ que la batería ACS requiere calor para cargarse porque esta baja.
  - *PCMDHW\_SOC* (%): Es una variable tipo *double* que estima el porcentaje de EDC de la batería ACS.
  - *PCMDHW\_T* (°C): Es una variable tipo *double* que indica la temperatura en la batería ACS.
- **Entradas\_BC:**
  - *HPcondenser\_out\_T* (°C): Es una variable tipo *double* que indica la temperatura que sale del condensador de la bomba de calor.

- *HPevaporator\_out\_T* (°C): Es una variable tipo *double* que indica la temperatura que sale del evaporador de la bomba de calor.

### 3.1.2 Variables de Salida:

- **Válvulas:**
  - *PVTtoPRODpump\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre el tanque de inercia y el resto del circuito de calor.
  - *PVTtoColdPCM\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre el tanque de inercia y la batería de frío.
  - *TCMrecirc\_in\_3wv*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre la sección de entrada del RTQ y la recirculación al tanque de inercia.
  - *TCMin\_3wv*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión para que entre agua al reactor o no.
  - *TCMrecirc\_out\_3wv*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre la sección de salida del RTQ y la recirculación al tanque de inercia.
  - *TCMcondenser\_in\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre la entrada del condensador de amoniaco y la salida del ventiloconvectorde calor en caso de que se necesite expulsar el calor.
  - *TCMcondenser\_out\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre la salida del condensador de amoniaco y la salida del ventiloconvectorde calor en caso de que se necesite expulsar el calor.
  - *HPevaporator\_in\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión

entre la salida del condensador de amoniaco y la entrada del evaporador del BC.

- *FANCOILheat\_in\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre la entrada del ventilconvectorde calor y el resto del sistema.
- *PCMheating\_in\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre la entrada de la batería calefacción y el resto del sistema
- *PCMDHW\_in\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre la entrada de la batería ACS y el resto del sistema
- *MANIFOLDsupply\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre las baterías de calor con el condensador de BC o con el sistema RTQ o Solar.
- *MANIFOLDreturn\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre el circuito de recirculación de las baterías de calor con el condensador de BC o con el sistema RTQ o solar.
- *TCMevaporator\_in\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre la entrada del evaporador de amoniaco con el ventilconvectorde frio o la batería frio.
- *FANCOILcold\_in\_3wv*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre la entrada del ventilconvectorcon el resto del sistema.
- *DEMANDheating\_out\_valve*: Es una variable tipo *booleana* que indica la posición de la válvula responsable de permitir la conexión entre la batería de calor heat y el edificio.
- **Bombas:**
  - *TCMreactor\_pump*: Es una variable tipo *booleana* que indica el estado de la bomba que mueve agua entre el circuito solar y las baterías de calor o el RTQ.

- *TCMcondenser\_pump*: Es una variable tipo *booleana* que indica el estado de la bomba que mueve agua entre el condensador de amoníaco y el BC o el ventiloinvectoror de calor.
- *TCMevaporator\_pump*: Es una variable tipo *booleana* que indica el estado de la bomba que mueve agua entre el circuito solar y la batería de frío o el evaporador de amoníaco o el ventiloinvectoror de frío.
- *HPcondenser\_pump*: Es una variable tipo *booleana* que indica el estado de la bomba que mueve agua entre el BC y las válvulas *MANIFOLDreturn* o *MANIFOLDsupply*.
- *DEMANDcold\_pump*: Es una variable tipo *booleana* que indica el estado de la bomba que mueve agua entre la batería de frío y el edificio.
- *DEMANDheating\_pump*: Es una variable tipo *booleana* que indica el estado de la bomba que mueve agua entre la batería de calefacción y el edificio.
- **Ventiloinvectorores:**
  - *FANCOILheat*: Es una variable tipo *booleana* que indica que se puede extraer el calor al ambiente.
  - *FANCOILcold*: Es una variable tipo *booleana* que indica que se puede extraer el frío al ambiente.
- **Etiquetas:**
  - *TCMSTATE*: Es una variable tipo *string* que indica el estado del sistema RTQ.
  - *CHARGING\_PCM*: Es una variable tipo *string* que indica que la batería está cargando.
  - *DEMAND\_OF\_PCM*: Es una variable tipo *string* que indica que se demanda del edificio.
  - *STATE\_TANK*: Es una variable tipo *string* que indica si el tanque está caliente o frío.

- VS: Es una variable tipo string que indica el estado del sistema térmico.

### 3.1.3 Variables Globales:

- *PVT\_enabledbyTCM*: Es una variable tipo *booleana* que indica que las placas solares pueden dar calor al reactor RTQ.
- *HP\_enabledbyPCM*: Es una variable tipo *booleana* que indica que el BC puede dar calor a las MCF de calor.
- *PCM\_heat\_REQUIRED*: Es una variable tipo *booleana* que indica que alguna de las baterías de calor necesita calor porque están bajas.
- *PCM\_cold\_REQUIRED*: Es una variable tipo *booleana* que indica que la batería de frío necesitan frío porque está baja.
- *TCM\_event\_discharging*: Es una variable tipo *booleana* que indica que el RTQ está siendo descargado.
- *TCM\_event\_charging*: Es una variable tipo *booleana* que indica que el RTQ está siendo cargado.
- *Heat\_avaible*: Es una variable tipo *booleana* que indica que hay calor suficiente en el sistema para cargar las MCF.
- *PCMcold\_enabledbyTCM*: Es una variable tipo *booleana* que indica que el RTQ puede dar frío a la batería frío.
- *PCMhot\_T*: Es una variable tipo *double* que indica la temperatura de la batería MCF que se está cargando, ya que hay dos cuando se solicita una u otra este valor coge la temperatura de esa batería.

## 3.2 Parámetros

- *Allow\_SummerHeatDemand\_withHPandTCM*: Es una variable tipo *booleana* que indica solo si la demanda en verano se puede cubrir por el RTQ y el BC durante la fase de carga del RTQ.

- *Allow\_SummerHeatDemand\_withTCM*: Es una variable tipo *booleana* que indica solo si la demanda en verano se puede cubrir por el RTQ durante la fase de descarga del RTQ.
- *Demo\_outdoor\_T\_min* (°C): Es una variable tipo *double* que indica la temperatura mínima que tiene que hacer fuera para proceder con el ciclo de descarga y expulsar el frío al ambiente.
- *FANCOIL\_Tdiff* (°C): Es una variable tipo *double* que indica la diferencia de temperatura mínima requerida por los ventiloconvectores para poder actuar.
- *HPevaporator\_Tmax* (°C): Es una variable tipo *double* que indica la temperatura máxima permitida para que el flujo entre en el evaporador del BC.
- *Hystheresis\_T* (°C): Es una variable tipo *double* que indica la diferencia mínima entre temperaturas.
- *PCMcold\_SOC\_highlimit* (%): Es la máxima carga permitida en la *MCFfrío* para operar.
- *PCMcold\_SOC\_lowlimit* (%): Es la mínima carga permitida en la *MCFfrío* para operar.
- *PCMcold\_T\_max* (°C): Es una variable tipo *double* que indica la temperatura máxima en la *MCFfrío*.
- *PCMcold\_T\_min* (°C): Es una variable tipo *double* que indica la temperatura mínima en la *MCFfrío*.
- *PCMcold\_T\_winter\_min* (°C): Es una variable tipo *double* que indica la temperatura mínima en la *MCFfrío* en invierno.
- *PCMDHW\_SOC\_highlimit* (%): Es la máxima carga permitida en la *MCFACS* para operar.
- *PCMDHW\_SOC\_lowlimit* (%): Es la mínima carga permitida en la *MCFACS* para operar.
- *PCMheating\_SOC\_highlimit* (%): Es la máxima carga permitida en la *MCFcalor* para operar.

- *PCMheating\_SOC\_lowlimit* (%): Es la mínima carga permitida en la MCFcalor para operar.
- *PCMhot\_T\_max* (°C): Es una variable tipo *double* que indica la temperatura máxima en las MCF de calor.
- *PCMhot\_T\_min* (°C): Es una variable tipo *double* que indica la temperatura mínima en las MCF de calor.
- *TCMreactor\_T\_max* (°C): Es una variable tipo *double* que indica la temperatura máxima en el reactor.
- *TCMreactor\_T\_min* (°C): Es una variable tipo *double* que indica la temperatura mínima en el reactor.
- *Threshold* (°C): Es la diferencia mínima permitida entre la temperatura del condensador del RTQ y la del evaporador del BC para que salte el error del BC.



## 4 Diseño del control en Matlab/Simulink

El entorno de programación Matlab/Simulink se ha utilizado para realizar un primer diseño del controlador y validar que cumple las especificaciones de los diferentes modos de control. El controlador se ha programado utilizando la herramienta Stateflow y siguiendo una filosofía de programación basada en máquina de estados.

En los distintos apartados se explica cómo está diseñada la interfaz, cómo se ha programado el control en StateFlow y cómo se ha validado el modelo mediante distintas pruebas.

### 4.1 Interfaz

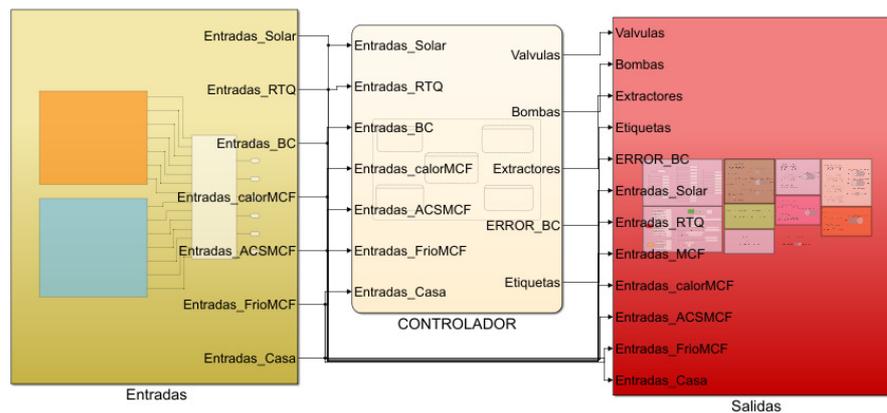


Figura 12: Imagen del Modelado en Matlab/Simulink

En la Figura 12 se ve que el sistema se distribuye en 3 bloques:

- **Entradas:** corresponden a todas las variables necesarias para la implementación de control. Se componen de temperaturas, llamadas del sistema y porcentajes de carga de las baterías (EDC).
- **Controlador:** en este bloque es donde se desarrolla la programación de estados de StateFlow e implementa el control mediante el diseño transición-estado.
- **Salidas:** corresponden a la decisión que toma el control sobre el sistema termoquímico. Se componen de señales *lógicas* que indican qué válvulas, bombas o ventiloconvectores de calor encender o apagar para el funcionamiento del sistema. Se parte de ellas hay señales que indican

el estado del sistema y una señal de error de funcionamiento que se explica más adelante.

#### 4.1.1 Entradas

Las entradas se dividen a su vez en 2 sub-bloques (Ver Figura 12 Anexos):

- **Entradas mediante Simulink:** Son las entradas que se introducen desde la consola durante la ejecución del modelo.
- **Entradas mediante Excel:** Son las entradas que se introducen desde la hoja llamada “pruebas” de Excel para simular una secuencia del sistema y probar su funcionamiento. A diferencia de las entradas de Simulink no se puede modificar cuando el modelo está ejecutándose.

Otro elemento de estas entradas es el *bloque Selector* que se encarga de elegir unas entradas u otras según elija el usuario y que se ve después.

##### 4.1.1.1 Entradas mediante Excel

Para introducir los valores mediante Excel se usa el bloque *FromSpreadsheet* cuyo bloque está a la izquierda de la Figura 13 de los Anexos y su inicialización a la derecha. Para inicializar el bloque primero se selecciona el “*File name*” que va a leer, en este caso la hoja de Excel “pruebas”. Después en el apartado “*Sheetname*” la hoja de datos de ese libro que se quiere leer, en nuestro caso la “*Hoja1*”. Por último, se indica que rango de valores se quieren leer y cómo se leen de manera secuencial según avance el tiempo de simulación. Además de ese bloque se tienen varios bloques llamados *Bus Creator* cuya función es unificar una serie de entradas en un único bus de datos con el fin de organizar las entradas y reducir cables innecesarios. Para ver como se inicializa este bus se incluye como ejemplo uno de ellos *Entradas\_RTQ* que se ve en la Figura 14 de Anexos. Se ve que en este bloque se introducen todas las entradas relacionadas con el RTQ y sale un único bus. Para crear este bus hay que seleccionar el número de entradas y el nombre ordenado de cada señal según se leen en el controlador porque si se hace en otro orden se intercambian las entradas y el sistema sería erróneo. Para introducirles solo hace falta poner el nombre en cada cable y dar al botón “*Add*”. Respecto al tipo de salida del bus se pone en modo auto puesto que hay salidas tanto *doubles* como *booleanas*.

#### 4.1.1.2 Entradas mediante Simulink

Para meter las entradas durante la ejecución de *Simulink* se usan *Sliders* para la temperatura y *Switchs* para las llamadas. Estos bloques tienen que estar asociados a un bloque de entrada donde cambia el valor, en este caso se han usado bloques *Constant* y una vez más se han agrupado en buses de datos (Ver Figura 15 y 16 Anexos). Otra cosa que se aprecia es que hay dos *Switch Case* que controlan cada uno dos entradas. Esto se debe a que hay entradas que no pueden coexistir en un mismo estado o que el propio sistema no puede recibir. Es por ejemplo el caso de modo invierno y verano que se explica en el apartado Modelado del Sistema, y como se puede intuir cuando el sistema está en un modo es inverosímil que este en el otro. Por eso se usan esos *Switch Case* que en función del estado del *Switch* ponga a 1 un modo y a 0 el otro.

En cuanto a la consola de la que se ha hablado antes, es desde donde el usuario introduce las entradas durante la simulación a fin de obtener el resultado del control en función de estas variaciones. Como se ve está estructurado en sub-consolas que representan cada uno de los buses que entran en el controlador. Se ve que algunas entradas están descomentadas puesto que en esta primera versión no se usan. La posición de las sub-consolas está puesta en concordancia con la Figura 10 de cara a que sea más intuitivo a la hora de encontrar la entrada a modificar. Por último, cabe destacar un *Switch* posicionado en la parte inferior de la consola. Este botón se encarga de seleccionar el tipo de entrada (Excel o Simulink). Aunque esté aquí su entrada está asociada en el bloque Selector que se explica a continuación. La ubicación de esta consola puede estar junto con las consolas de salida para hacer más cómodo el manejo.

#### 4.1.1.3 Selector

Se compone de siete bloques *Switch Case* que indican qué entradas de qué bloque seleccionar. Esa decisión depende del valor del bloque *Constant* que está asociado al *Switch tal y como se indicó* anteriormente (Ver Figura 17 Anexos). Por último, cabe destacar que se vuelve a usar un *Bus Creator* en *Entradas\_Solar* aunque está compuesta por una sola entrada. De este modo todas las entradas están definidas como buses de forma homogénea

### 4.1.2 Salidas

En este bloque hay dos consolas que muestran entradas introducidas (a fin de comprobar las salidas junto con las entradas ver Figura 18 y 22 Anexos),

mensajes de error, estado de las válvulas, bombas y ventiloconvectores del sistema y estado del control. Dentro de la consola de salidas tanto el mensaje de error como los estados del control son meramente informativos, siendo lo importante para nuestra validación los estados de las válvulas, bombas y ventiloconvectores cuyo valor se puede ver de dos formas: mediante bloque Display o por un archivo de datos (en formato Matlab).

#### **4.1.2.1 Salidas mediante Display en Simulink**

Las salidas mediante el bloque *Display* son las que se muestran de forma directa en el Interfaz (Ver Figura 22 Anexos). Su función es observar el funcionamiento del control en tiempo real ya que el *Display* varía a cada paso de simulación con cada variación de alguna de las entradas.

#### **4.1.2.2 Salidas mediante archivo MATLAB**

Las salidas mediante archivo están condensadas en 3 bloques rojo, verde y amarillo (Ver 22 Anexos) y como ya se dijo es solo para los estados de válvulas bombas y ventiloconvectores ya que son los elementos que finalmente actúan en el sistema y cuyos valores interesa recoger a lo largo del tiempo a fin de validarlos. Para ver como se ha hecho se explica el bloque de las válvulas (Ver Figura 19 Anexos).

En este bloque se ve como el bus de datos se utiliza como entrada del bloque *to File* llamado "*valves.mat*". Los otros dos bloques son para poder visualizar los valores en el *Display* pasando el tipo de *Non-Virtual Bus* a *Vector*.

También se observa que el archivo crea una estructura formada por cada una de las 16 válvulas y que dentro de ellas guarda su estado (apagado 0 o encendido 1) a lo largo de cada paso de la simulación, que es 1 segundo en este caso (Ver Figura 20 Anexos).

## **4.2 Guía de Usuario**

En este apartado se explica cómo debe el usuario actuar a la hora de simular.

Lo primero que debe saber es que lo simula de manera indefinida. No hay un tiempo de simulación establecido y es el usuario quien decide cuando la simulación termina a fin de que pueda interactuar con el sistema probando todos los modos y situaciones posibles para verificar el funcionamiento de cara

a la implementación física en el PLC. La segunda cosa importante es que el paso de simulación establecido es un segundo y que el tiempo que transcurre en la simulación es el mismo que en la realidad. Es decir, que un segundo de simulación es un segundo real de funcionamiento del sistema (Ver Figura 21 Anexos). Se configura así para que el usuario pueda ver los cambios en la salida en tiempo real.

A la hora de interactuar con el modelo el usuario se debe situar en el bloque de Salidas puesto que desde allí puede interactuar con la consola en tiempo real si introduce los datos desde Simulink y además visualizarlos o solo visualizar en el caso de que las entradas sean los casos predefinidos de Excel (en este modo la consola no funciona, puede mover los *Sliders* y *Switchs* pero no se modifican las entradas). (Ver Figura 16 Anexos),

Otra cosa importante que debe saber el usuario, es que debido a que los parámetros *Allow\_SummerHeatDemand\_withHPandTCM* y *Allow\_SummerHeatDemand\_withTCM* varían, es posible que para pasar de un estado a otro a fin de verificar el control necesite cambiar su valor. Como Simulink no permite el cambio de un parámetro del diccionario durante la simulación lo hace mediante un Script de Matlab llamado "cambio\_parametro\_allow.m" ya que si no habría que parar la simulación y volver a ejecutar algo engorroso.

```
cambio_parametro_allow.m x +
1 %Nombre del modelo de Simulink
2 NombreModelo='LocalController_Complete_14_10';
3
4 %Nombre de la base de datos
5 NombreBD='LocalController_v00_1.sidd';
6
7 %Se abre el acceso a la BD
8 myDictionaryObj = Simulink.data.dictionary.open(NombreBD);
9 dDataSectObj = getSection(myDictionaryObj,'Design Data');
10
11 %Se selecciona el parámetro si es allow withHTCM se modifica esto 1
12 param = getEntry(dDataSectObj,'Allow_SummerHeatDemand_withHPandTCM');
13 paramvalue=getValue(param);
14
15 %Se modifica el valor del parámetro si es de 0 a 1 poner 1 en param value
16 paramvalue.Value=0; 2
17 setValue(param,paramvalue);
18 %Se guardan los cambios en la BD
19 saveChanges(myDictionaryObj);
20
21 %Se actualiza el modelo de Simulink para que lea el nuevo valor
22 %Sólo se actualiza si está en ejecución, ya que en cualquier otro caso se
23 %actualizará automáticamente al ejecutarlo
24 estado=get_param(NombreModelo,'SimulationStatus');
25 if strcmp(estado,'running') || strcmp(estado,'paused')
26     set_param(NombreModelo,'SimulationCommand','Update');
27 end
28
29 %Cerrar diccionario
30 close(myDictionaryObj);
```

Figura 13: Captura de código archivo cambio\_parametro\_allow

En ese Script para modificar el valor del parámetro, el usuario solo necesita saber dos cosas: la primera es que en el recuadro 1 introduce el parámetro

Allow que desea modificar de los dos y que segundo en el recuadro 2 especifica ese valor si 0 o 1 ya que es una variable *lógica*. La función modifica el valor independientemente del que hubiera antes, y no lo escala como los *doublés*.

Si el usuario quiere saber qué significa un parámetro o modificarlo tiene que acudir al Diccionario, explicado en el apartado “Otros archivos” más adelante.

### 4.3 Programación Controlador (StateFlow)

El siguiente apartado explica la programación que se ha seguido en el último bloque del modelo encargado de implementar el control. Se ha diseñado a partir de StateFlow que sigue el paradigma transición-estado.

Para comenzar se va a visualizar una figura de cómo se muestra el bloque de control dentro del subsistema de Simulink. A partir de ahora se habla de estados y subestados para referirnos a las distintas partes del control.

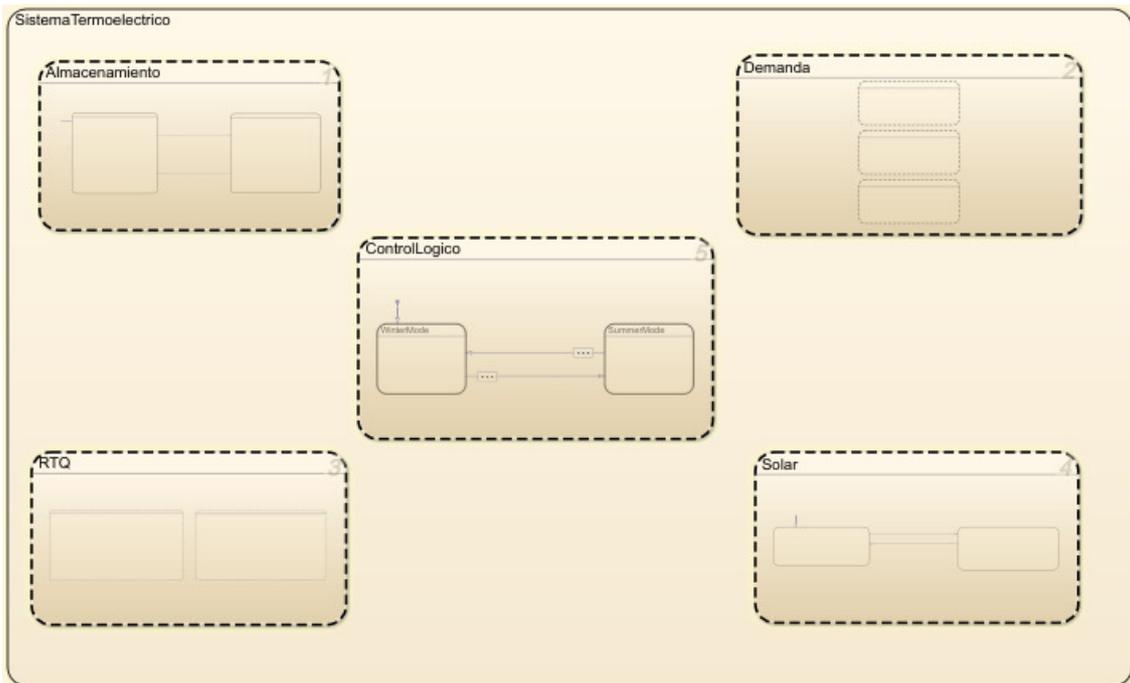


Figura 14: Vista del controlador en StateFlow

En la Figura 4 se ve el estado principal del controlador que engloba las distintas partes. No hay transiciones puesto que ejecuta los subestados en paralelo. Esto significa que en cada paso de simulación se ejecutan todos ellos, aunque lo hace siguiendo un orden indicado con un número en la parte superior derecha de cada subestado. A continuación, se explica la función de cada subestado

según prioridad y después se muestra la programación y diseño de cada uno de los mismos. Cabe destacar que el orden de ejecución está establecido de manera lógica ya que no todos, pero algunos de ellos parten del resultado del subestado anterior.

#### 4.3.1 Almacenamiento (Storage)

El primer subestado arriba a la izquierda de la Figura 4 que se ejecuta es Almacenamiento. Su función es satisfacer la demanda por parte de las baterías al sistema. Como se sabe hay tres tipos de baterías, pero solo la batería ACS se usa en los dos modos dentro de Almacenamiento, ver Figura 23 Anexos. Existen subestados que según el modo invierno o verano indican qué dos baterías están en activo y por tanto a qué dos pueden el RTQ o la energía solar proveer calor.

Dentro del subestado “*StorageWinter*” ya se tiene la programación base con estados y transiciones. En este estado solo se puede satisfacer la demanda de la batería ACS y la de calefacción puesto que como se ve en invierno es inverosímil solicitar frío. Por eso, aunque la batería de frío lo solicite no se va a satisfacer, esté en el estado en el que esté.

Se empieza partiendo del estado sin demanda ya que no es lógico que la haya en un arranque. Para las transiciones depende del estado de las entradas *Inputs\_HeatingPCM.PCMheating\_heating\_call* y *Inputs\_DHWPCM.PCMDHW\_heating\_call* que indican esa demanda de la batería.

En estos estados se definen las variables locales *GlobalStates.PCM\_heat\_REQUIRED* y *GlobalStates.PCM\_cold\_REQUIRED* que se explicaron previamente y que se ponen a *True* en el caso de la primera si hay demanda por parte de las baterías ACS y calefacción, y en la segunda si hay demanda por parte de la batería de frío. En este caso *GlobalStates.PCM\_cold\_REQUIRED* siempre es *false* ya que el sistema no contempla pedir frío en invierno. Las variables *Valves.PCMheating\_in\_valve* y *Valves.PCMDHW\_in\_valve* se ponen a *True* según se demande de la batería calefacción o ACS respectivamente y haya calor Solar o del RTQ para sufragarlo. En cualquier otro caso es *False*. La variable *GlobalStates.PCMhot\_T* recibe el valor de la temperatura de la demanda que se va a satisfacer. En caso de que se demanden ambas (ACS y calefacción), como se ve en Figura 24 Anexos en el estado inferior, predomina la demanda de calefacción sobre ACS. Esto se ha decidido debido a que se tiene que tomar una referencia de la temperatura de la batería a cargar y predomina calefacción puesto que ACS no tiene restricción según modo invierno o verano y generalmente su temperatura es inferior.

Por último se genera la señal *string\_signals.CHARGING\_PCM* como salida para indicar al usuario que el sistema funciona correctamente. Su valor depende del estado y puede tomar los valores “HEATING”, “DHW” Y “NONE”. (Ver Figura 25 Anexos)

Dentro del subestado “StorageSummer” ya se tiene la programación base con estados y transiciones. En este estado solo se puede satisfacer la demanda de la batería ACS y frío puesto que como se ve en verano es inverosímil solicitar calefacción. Por eso, aunque la batería de calefacción solicite calor no se va a satisfacer, este en el estado en el que este.

Una vez más se empieza partiendo del estado sin demanda ya que no es lógico que la haya en un arranque. Para las transiciones depende del estado de las entradas *Inputs\_ColdPCM.PCMcold\_cooling\_call* e *Inputs\_DHWPCM.PCMDHW\_heating\_call* que indican esa demanda de la batería.

En estos estados también se definen las variables locales *GlobalStates.PCM\_heat\_REQUIRED* y *GlobalStates.PCM\_cold\_REQUIRED*, en este caso *GlobalStates.PCM\_heat\_REQUIRED* solo puede ser puesta a *True* por la batería ACS. En este caso la variable *Valves.PCMheating\_in\_valve* siempre es *False* puesto que nunca se demanda (se pone por robustez) y *Valves.PCMDHW\_in\_valve* se pone a *True* según se demande de la batería ACS y haya calor Solar o del RTQ para sufragarlo, en cualquier otro caso es *False*. La variable *GlobalStates.PCMhot\_T* recibe el valor de la temperatura de la demanda que se va a satisfacer En este caso esta variable no tendría sentido puesto que siempre es ACS pero puesto que es única para todo el modelo se pone en este modo de verano también. En este caso sí se pueden cargar ambas baterías a la vez ya que se requieren ambos calores.

Por último se genera la señal *string\_signals.CHARGING\_PCM* como salida para indicar al usuario que el sistema de funciona correctamente. Su valor depende del estado y puede tomar los valores “COLD”, “DHW”, “DHW Y COLD” Y “NONE”. (Ver Figura 25 Anexos)

### 4.3.2 Demanda (Demand)

Este segundo subestado representado en la parte superior derecha de la Figura 4 representa la demanda de calor o frío por parte del edificio a las baterías, se divide en 3 subestado: *HeatingCircuit*, *DHWCircuit* y *CoolingCircuit* (Ver 26 Anexos) debido a que hay 3 baterías. El segundo de ellos está vacío puesto que la demanda de ACS como se explica siempre hay que satisfacerla. Se incluye un subestado asociado para facilitar la comprensión de la estructura de

programación. Una cosa a tener en cuenta es que el usuario puede solicitar esa demanda de frío o calor en cualquier época del año. Aunque sea lógico que el usuario solicite calor en invierno y viceversa no se limita esa elección, aunque sí se limita la solicitud de ambas ya que eso no tendría sentido.

Si se pasa al primer circuito de demanda, el de la batería calor calefacción, se ve que las variables de salida *Pumps.DEMANDheating\_pump* y *Valves.DEMAND heating\_out\_valve* se ponen a *True* si se solicita esa demanda y a *False* en caso contrario (Ver Figura 27 Anexos). Como antes también se muestra una señal de salida llamada *string\_signals.DEMAND\_OF\_PCM* para que el usuario sepa que lo que está solicitando se cumple.

Si se pasa al último circuito de demanda, el del batería frío, se ve que las variables de salida *Pumps.DEMANDcold\_pump* se pone a *True* si se solicita esa demanda y además sino se ha solicitado la demanda de la batería calor calefacción (ya que como se dijo no se pueden demandar a la vez) y a *False* en caso contrario (Ver Figura 28 Anexos). Como antes también se muestra una señal de salida llamada *string\_signals.DEMAND\_OF\_PCM* para que el usuario sepa que lo que está solicitando se cumple.

### 4.3.3 Maquina Termoquímica (RTQ)

Este tercer subestado representado en la Figura 4 en la parte inferior izquierda, representa el RTQ dentro de él está el RTQ y el otro subestado es sobre un error en la bomba de calor que se ha incluido ahí porque es necesario que se ejecute antes del control lógico. (Ver Figura 29 Anexos)

#### 4.3.3.1 Error bomba calor

Este subestado gestiona el error de la bomba de calor situada tras el RTQ. Este error es indicativo y se produce cuando durante un lapso de tiempo (en este caso 5 segundos) la entrada *Inputs\_TCM.TCMcondenser\_T* supera a la entrada *Inputs\_HP.HPEvaporator\_out\_T*. Es un error puesto que una bomba de calor tiene que expulsar más calor del que recibe y y por si deja de cumplirse se apaga. (Ver Figura 30 Anexos)

#### 4.3.3.2 RTQ

Este bloque se encarga de programar el paso de los distintos estados del RTQ que se explican en el estudio teórico y donde se ve que ese cambio depende

de que las entradas *Inputs\_TCM.TCMreactor\_heating\_call* y *Inputs\_TCM.TCMreactor\_cooling\_call* estén activas o no y de que la entrada *Inputs\_TCM.TCMreactor\_in\_T* es decir la temperatura que entre en el reactor supere o no la máxima del mismo. , También depende del estado en el que esté el subestado control lógico lo cual es indicado por las variables locales *GlobalStates.TCM\_event\_charging* y *GlobalStates.TCM\_event\_discharging*.

Para indicar el estado del RTQ al control lógico se usa la variable global *TCMstate* (del enumeration *EnumTCMState* que se explica después) y una señal de salida llamada *string\_signals.TCMSTATE* para que el usuario sepa que el estado es cargado, descargado.... (Ver Figura 31 Anexos)

#### 4.3.4 Placas Solares (Solar)

Este cuarto subestado representado en la parte inferior derecha de la Figura 4 representa el estado de las placas solares. Aunque ya se dijo que no se iba a modelar ese sistema se necesita saber cuándo está activo (es decir caliente) y cuando no (es decir frío). Eso se sabe si como se ve la temperatura del tanque de inercia solar, representada como la entrada *Inputs\_Solar.Solartank\_Ttop* está por encima de la mínima para que el RTQ considere que está caliente y pueda usar ese calor. Si se cumple se pone la variable local *PVTstate* (del enumeration *EnumSolar State* que se explica después) al estado caliente (*EnumSolarState.Solarhot*) sino al frío (*EnumSolarState.Solarcold*) .(Ver Figura 32 Anexos)

Como antes también se muestra una señal de salida llamada *string\_signals.STATE\_TANK* para que el usuario sepa que el estado calor y frío de las placas funciona.

#### 4.3.5 Control lógico (Logical Control)

Este último subestado representado muestra el control lógico, es decir, la decisión del estado global del sistema y la correspondiente salida. En el estudio teórico se explicaron estos modos y en este caso se explica cómo actúa el control dentro de cada modo. (Ver Figura 33 Anexos)

#### 4.3.5.1 Modo Invierno (WinterMode)

En el modo invierno se ven los nueve estados que lo componen ocho modos y submodos, que ya se explicaron, y el otro es el estado *OFF* desde donde arranca el sistema. (Ver Figura 34 Anexos)

La Figura 34 de Anexos es una vista de la disposición de los estados en el modo invierno y sus distintas conexiones que se razonó en la explicación de modos del estudio teórico. A continuación se explican los estados más detallados con una descripción de lo que se ha programado en su interior. (Ver Figura 35 Anexos)

En todos los estados están las 13 válvulas que regulan el funcionamiento del sistema. Las otras 3 que faltan son las que cubren la demanda de las baterías de calor. También se tienen 4 bombas para regular el sistema (las otras 2 que faltan cubren la demanda de las baterías al edificio) y 2 ventilosconvectores para regular la temperatura del sistema. Todas estas variables se modifican a todo o nada para permitir o no el paso del flujo por los conductos del sistema de tal manera que se reproduzca cada uno de los modos que se representan en las figuras del estudio teórico de cada estado.

En el caso de la figura superior izquierda en el estado *OFF* todos los actuadores están desconectados.

Otra cosa que se programa en cada estado es la solicitud de carga o descarga al RTQ de tal manera que cada estado controla el funcionamiento del mismo.

En los estados de carga pone el evento global de carga a 1 y en los de descarga pone el de descarga a 1, nunca pueden coexistir los dos eventos activos puesto que el RTQ carga o descarga, pero no realiza ambas.

Cabe destacar que al cubrir la demanda con solar ambos eventos están a False puesto que el RTQ no realiza ninguna de las dos operaciones.

Por último en los estados *Transition\_HEATDemandCoveredbyTCMDISCHARGING* y *HEATDemandCoveredbyTCMDISCHARGING*, se ve que el valor del ventilosconvectorde frío y su válvula asociada varían según la temperatura del exterior. Esto es debido a que en invierno no se puede expulsar calor al exterior que difiera más de lo que valga el parámetro *FANCOIL\_Tdiff*.

#### 4.3.5.2 Modo Verano (SummerMode)

En el modo verano se ven los ocho estados que lo componen siete modos y submodos, que ya se explicaron, y el otro es el estado *OFF* desde donde arranca el sistema. (Ver Figura 36 Anexos)

La imagen anterior es una vista de la disposición de los estados en el modo verano y sus distintas conexiones que se razonó en la explicación de modos del estudio teórico continuación se incluyen los ocho estados más detallados de los cuales se explica qué se ha programado en su interior. (Ver Figura 37 Anexos)

Como en el modo invierno, en todos los estados se incluye el estado de las 13 válvulas, las 4 bombas y los 2 ventilosconectores. Todas estas variables se modifican a todo o nada para permitir o no el paso del flujo por los conductos del sistema de tal manera que se reproduzca cada uno de los modos que se representan en las imágenes del estudio teórico de cada estado.

Otra cosa que se programa en cada estado es la solicitud de carga o descarga al RTQ de tal manera que cada estado controla el funcionamiento del mismo. A diferencia del modo invierno en este no se tiene el estado que calienta la batería de frío y en la descarga el sistema se centra en satisfacer frío y no calor.

Por último en los estados *Transition\_COLDemandCoveredbyTCMDISCHARGING* y *COLDDemandCoveredbyTCMDISCHARGING* se ve que el valor del ventilosconector caliente varía según la temperatura del exterior en el estado "COLD". Otra cosa que se puede dar es que el evaporador del RTQ esté más caliente que la batería de frío. Si eso sucede los ventilosconectores expulsan el calor y no se puede satisfacer la demanda de frío.

## 4.4 Otros archivos del modelo

### 4.4.1 Archivos Enumeration

Los archivos de enumeración se usan para enumerar los distintos estados de una variable sea entrada, local o salida. Se usan para que el sistema enumere cada estado de esa variable con un número y hacer más fácil y organizado el código y el sistema. En este modelo se utilizan 3 que son:

- *EnumSolarState*: Para determinar en qué estado están las placas solares. (Ver Figura 38 Anexos)
- *EnumStringSysTCM*: Para determinar el estado del sistema. (Ver Figura 39 Anexos)

- *EnumTCMState*: Para determinar el estado del RTQ. (Ver Figura 40 Anexos)

Como se puede ver se almacenan un número de valores que debería tomar una única variable en función del estado en el que este.

#### 4.4.2 Diccionario

| Name                              | Scope     | Port | Resolve Signal           | DataType                  |
|-----------------------------------|-----------|------|--------------------------|---------------------------|
| Valves                            | Output    | 1    | <input type="checkbox"/> | Bus: Valves               |
| Pumps                             | Output    | 2    | <input type="checkbox"/> | Bus: Pumps                |
| Inputs_Solar                      | Input     | 1    | <input type="checkbox"/> | Bus: Inputs_Solar         |
| GlobalStates                      | Local     |      | <input type="checkbox"/> | Bus: GlobalStates         |
| TCMreactor_T_min                  | Parameter |      |                          | double                    |
| TCMstate                          | Local     |      | <input type="checkbox"/> | Enum: EnumTCMState        |
| Inputs_TCM                        | Input     | 2    | <input type="checkbox"/> | Bus: Inputs_TCM           |
| TCMreactor_T_max                  | Parameter |      |                          | double                    |
| HPevaporator_T_max                | Parameter |      |                          | double                    |
| FANCOIL_Tdiff                     | Parameter |      |                          | double                    |
| PCMcold_T_min                     | Parameter |      |                          | double                    |
| PCMcold_T_winter_min              | Parameter |      |                          | double                    |
| PCMhot_T_min                      | Parameter |      |                          | double                    |
| Inputs_HP                         | Input     | 3    | <input type="checkbox"/> | Bus: Inputs_HP            |
| Inputs_HeatingPCM                 | Input     | 4    | <input type="checkbox"/> | Bus: Inputs_HeatingPCM    |
| Inputs_DHWPCM                     | Input     | 5    | <input type="checkbox"/> | Bus: Inputs_DHWPCM        |
| Inputs_Demo                       | Input     | 6    | <input type="checkbox"/> | Bus: Inputs_Demo          |
| Inputs_ColdPCM                    | Input     | 7    | <input type="checkbox"/> | Bus: Inputs_ColdPCM       |
| Fancoils                          | Output    | 3    | <input type="checkbox"/> | Bus: Fancoils             |
| Allow_SummerHeatDemand_withHPa... | Parameter |      |                          | boolean                   |
| Allow_SummerHeatDemand_withTCM    | Parameter |      |                          | boolean                   |
| Demo_outdoor_T_min                | Parameter |      |                          | double                    |
| Hystheresis_T                     | Parameter |      |                          | Inherit: Same as Simulink |
| PVTstate                          | Local     |      | <input type="checkbox"/> | Enum: EnumSolarState      |
| ERROR_HP                          | Output    | 4    | <input type="checkbox"/> | boolean                   |
| string_signals                    | Output    | 5    | <input type="checkbox"/> | Bus: string_signals       |
| threshold                         | Parameter |      |                          | double                    |

Figura 15: Captura Diccionario

Este archivo se llama “LocalController\_v00\_1.sidd” y se accede a él directamente desde el modelo de Simulink a través de la opción de menú *Modeling>Model Explorer>External Data>LocalController\_v00\_1>Design Data*. En el apartado “Modelado del Sistema” se detalla el significado de cada dato, su valor, el tipo de dato y su unidad. Engloba todos los datos sea parámetro o variable de entrada o de salida.

## 4.5 Validación del control

En este apartado se valida el control programado en MATLAB/Simulink mediante el cambio de las variables de entrada y la comprobación de las salidas generadas. Los parámetros son iguales en todos los casos salvo para los parámetros *Allow\_SummerHeatDemand\_withHPandTCMyAllow\_SummerHeatDemand\_withTCM* que también se varían en modo verano para poder pasar por todos los estados. Estos parámetros se modifican mediante la ejecución de un script .m como se explicó anteriormente. Las pruebas son sucesivas, de tal manera que se parte de un estado inicial que se muestra en la siguiente imagen y se van cambiando las variables de entrada y comprobando cómo el sistema llega al estado esperado.

Cabe destacar que hay ocasiones en la que el sistema pasa por un estado intermedio antes de llegar al esperado. Otra cosa que hay que decir con respecto a este modelo es que al ejecutarse de manera gradual con pasos de tiempo las variables *doublé* también se modifican paso a paso cosa que no sucede con las *booleanas*. Por ejemplo si en el instante  $t=0$   $TCMcondenser\_T=40$  y en  $t=10$   $TCMcondenser\_T=65$  aumentando 2.5 en cada paso de simulación. Mientras que si  $TCMreactor\_heating\_call=e$  0 en el instante  $t=0$  y en  $t=10$   $TCMreactor\_heating\_call=1$ , cambia su valor a 1 en el instante  $t=1$ . El estado inicial del sistema es apagado. Las variables en amarillo no se usan en el modelo pero sí son parte del sistema teórico y por eso se ponen.

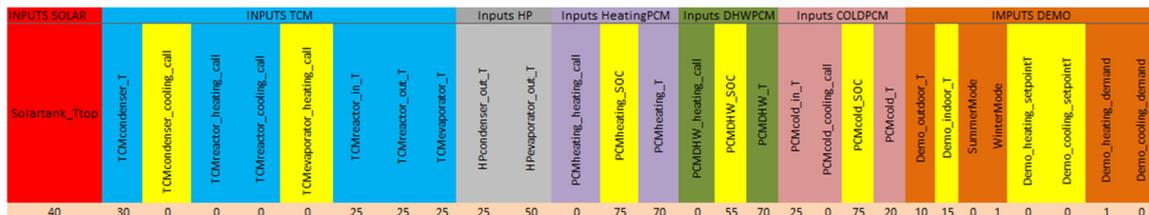


Figura 16: Imagen valores iniciales del Modelo

### 4.5.1 Pruebas sobre el control lógico

En la tabla 1 de los Anexos se muestra todas las pruebas hechas para validar el modelo. Como se puede ver se corresponde con el archivo de entradas de MATLAB/Simulink y se hace de manera continua, es decir que solo se ejecuta una vez para realizar todas las pruebas. Se simula tanto en modo invierno como en modo verano.

Para demostrar que este sistema funciona se muestra el proceso de dos simulaciones la 1 y la 7.

En este caso se ve en la Figura 17 que la simulación 1 pasa de sus valores iniciales a que suba la temperatura del tanque solar híbrido y haga calor por lo que se cumple la c1. La subida de la temperatura del condensador y la ausencia de demanda además de estar en el estado inicial de descargado hacen que el sistema pase a *NOHEATDemandTCMCHARGING* ya que se cumple la c23.

| NUMERO | MODOS | ENTRADA/S MODIFICADA/S   | ESTADO ESPERADO                    |
|--------|-------|--|------------------------------------|
| 1      | INV   | Solartank_Ttop:40->65<br>TCMcondenser_T:30->90<br>TCMreactor_heating_call:0->1 | <del>NOHEATDemandTCMCHARGING</del> |

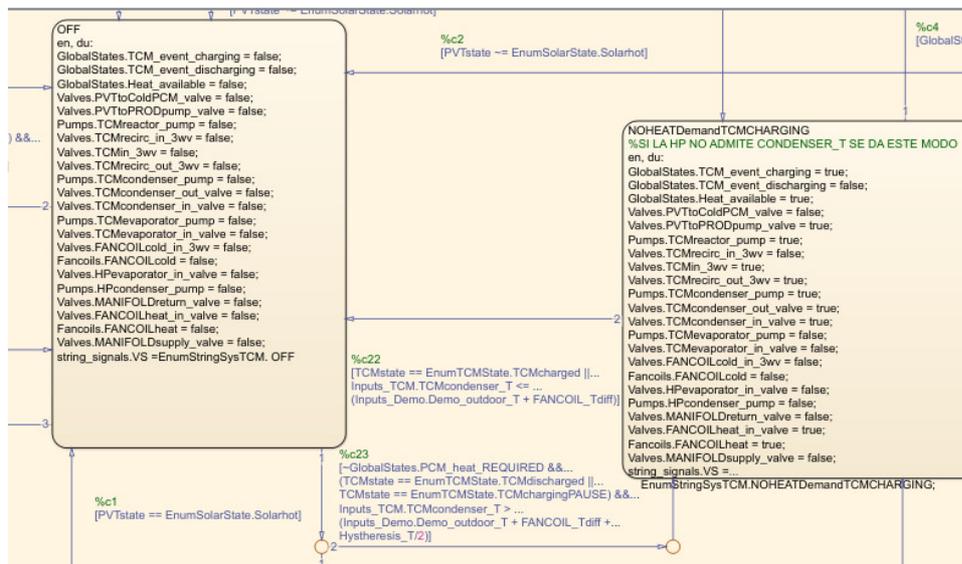


Figura 17: Captura de la transición entre dos estados de simulación nº1

En la simulación 7 se está en el estado *HEATDemandCoveredbyTCMCHARGING* no se parte del inicial lo que sucede es que al dejar de demandar el RTQ la descarga de frío el sistema pasa de estado descargando a descargado como se ve en la Figura 18 a la izquierda y por tanto se cumple la c20 como se ve en la de la derecha pasando al estado *OFF*.

| NUMERO | MODO | ENTRADA/S<br>MODIFICADA<br>/s                     | ESTADO<br>ESPERADO |
|--------|------|---|--------------------|
| 7      | INV  | Solartank_Ttop:65->50<br>Demo_cooling_demand:1->0 | OFF                |

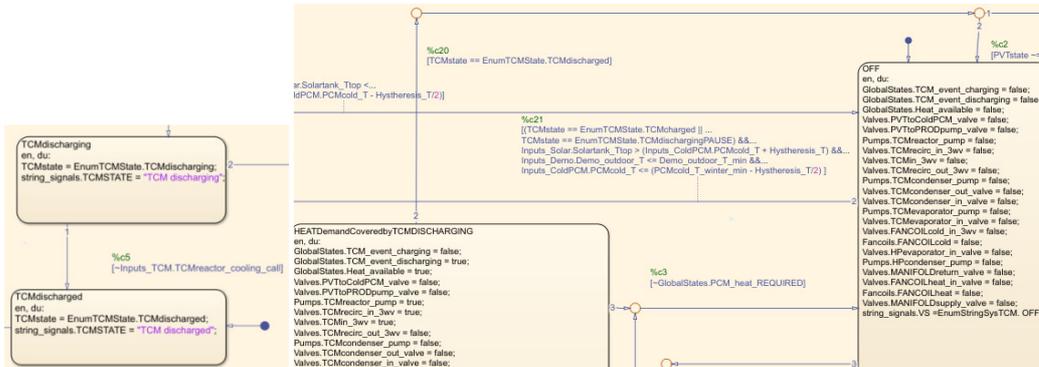


Figura 18: Capturas de la transición entre dos estados de la simulación n<sup>o</sup>7







El segundo elemento es una pequeña interfaz (Ver Figura 47 Anexos) que intenta replicar la de MATLAB/Simulink con las variables de entrada y los estados de los bloques como salida. Las salidas reales son las señales de apertura o cierre de válvulas, bombas y ventiloconvectores, pero como son combinación de los estados es más útil mostrar estos ya que son más informativos. Las entradas se dividen en 8 grupos de variables, es decir 8 buses, que entran el sistema como se hizo en Simulink.

Las salidas se componen de los mismos 5 sub-sistemas que se programaron en Simulink.

El último elemento es un bloque central donde se programan cada una de las condiciones para cambiar de estado y además se actualiza el estado anterior al actual. Se asimilaría a la acción de carga de un biestable. Este bloque tiene como elementos de E/S:

- Variables de entrada a las que se le añaden los parámetros de entrada (variables fijas) y se hace en los bloques que se ven (Ver Figura 48 Anexos).
- Estados internos que no son salidas como tal, sino que se usan para el funcionamiento entre subsistemas es el caso de *Heat\_avaible*. Por ejemplo, *Heat\_avaible* sirve para que el subsistema Control Lógico comunique a Almacenamiento si hay calor disponible para almacenar. Esa comunicación sería un estado interno.
- Este bloque central de entradas es un condicional por lo tanto necesita ser habilitado. Lo que se ha hecho es programar un biestable que lo habilite y deshabilite alternativamente cada segundo. Se ha hecho porque se necesita que primero entren las entradas y se lean los estados y después se genere la salida y se escriban los nuevos estados porque si no los estados se escriben y se leen a la vez haciendo que el sistema no funcione. (Ver Figura 49 Anexos).

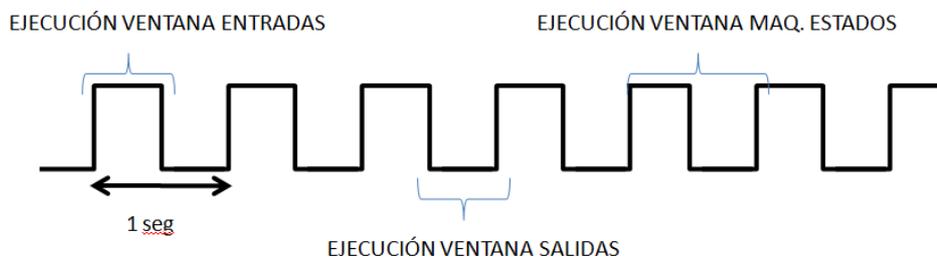


Figura 20: Esquema de ejecución de ventanas según biestable

- Referencias para comunicar la salida de esta ventana con las entradas de la ventana máquina de estado, donde se da la condición y el estado actualizado de cada subsistema. Más adelante se explica cómo se generan las condiciones.(Ver Figura 50 Anexos)

En las condiciones de cambio de estado además se actualiza y lee los estados de la ejecución anterior (Ver Figura 51 Anexos).

Dentro de este bloque se tienen los cinco sub-sistemas en los que se divide el sistema como en Simulink y uno más para el cambio de modo. El motivo de la división es organizativo ya que la complejidad y gran cantidad de variables hace que para identificar una condición o una transición sea necesaria esa organización. En la Figura 58 de Anexos se muestra una vista general del sistema que hay dentro de ese bloque. Además de los subsistemas se ven las entradas y salidas de cada uno, hay variables como *WinterMode* o *SummerMode* que se usan en dos Subsistemas.

Una vez vista la estructura de las entradas se pasa a explicar la programación a bajo nivel (puertas lógicas). En el primer sub-bloque Modo se tiene la programación de las condiciones de los modos invierno y verano donde se ve que la condición de que sea verano (COND1) se activa solo si se da verano y no invierno y la de invierno se activa en cualquiera de los otros casos. Aunque no debería pasar por seguridad se incluyen todos los casos para que en caso de error este en algún modo (Ver Figura 52 Anexos)

Si se entra en las entradas Solar se ven las dos condiciones del sub-bloque en este caso dependientes de temperaturas donde también se usan los bloques mayor y menor y el bloque suma. (Ver Figura 53 Anexos)

El tercer sub-bloque es Demanda donde las cuatro condiciones son combinación de dos entradas lógicas. (Ver Figura 54 Anexos)

El cuarto sub-bloque Almacenamiento ya aumenta la complejidad puesto que tiene un mayor número de condiciones. Como depende del modo en que se esté se ha subdividido en otros dos sub-bloques. Dentro del bloque de invierno se ve cómo en el bloque Demanda hay cuatro condiciones que dependen de la combinación de dos entradas. Es importante señalar que además de las ocho condiciones se obtienen dos temperaturas ya que se necesitan para la programación del estado interno *PCM\_hot* (Ver Figura 55 Anexos)

Metiéndonos en el Sub-bloque RTQ se tienen las seis condiciones de RTQ cuya programación depende tanto de temperaturas como de señales y además de estados internos. La alarma de la BC incluye la programación de dos condiciones: una de activación y la contraria de desactivación. En esta primera

es necesario un temporizador (*Timer*) para verificar que se ha dado el error correctamente y no ha sido una falsa alarma. (Ver Figura 56 Anexos)

Por último, el sub-bloque control el más complejo de todos. También divide sus condiciones según el modo, y en total incluye 55 condiciones que dependen tanto de temperaturas como de señales, además de incorporar varios temporizadores. La complejidad del código de la figura se debe a esas 55 condiciones compuestas por combinaciones de las entradas. (Ver Figura 57 Anexos)

### 5.1.2 Máquina de Estados

La máquina de estados es donde se programan las transiciones de un estado a otro. Se divide en entradas que son las condiciones y los estados y las salidas que son los nuevos estados generados.

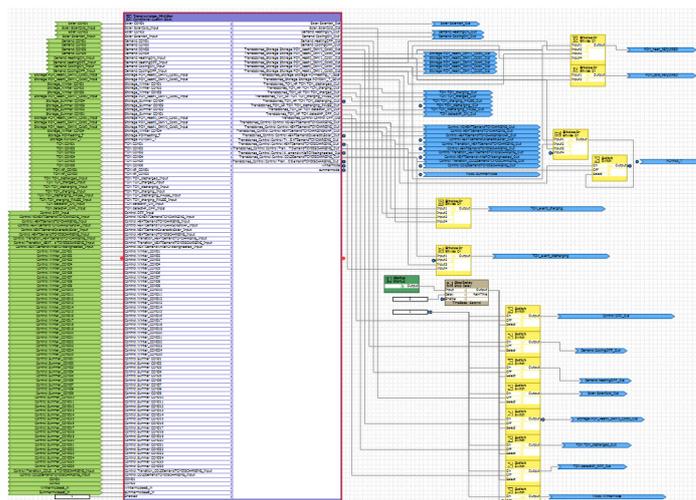


Figura 21: Vista general de la ventana máquina de estados

Como se puede observar en la salida se tiene un bloque *Startup* que reinicia el programa cada vez que se hace una ejecución. Esto se hace por dos razones. La primera es que al ejecutar todos los estados se inician a *False* y se necesitan poner a *True* los estados iniciales como son *OFF*, *Solarcold...* para que el sistema pueda comenzar su control. La segunda es la posibilidad de reinicio. Cuando se está probando una serie de datos y sale algo mal o se quiere que vuelva al estado inicial, se ejecuta y se reinicia para volver a comenzar de nuevo. Esta segunda razón es más de cara a la simulación ya que en el uso real el sistema no falla. El bloque *Delay* que se ve tras el bloque *Startup* es solo para mantener el reseteo unos segundos para asegurar que se ejecuta. (Ver Figura 59 Anexos)

Dentro de este bloque central de máquina de estados están los 6 sub-bloques que hay en el apartado entradas (Ver Figura 58 Anexos) pero en ellos se programan las transiciones no las condiciones. Esta distribución es una copia de la programada en StateFlow (Ver Figura 4).

Todos los bloques tienen una estructura general: tienen transiciones que activan cada estado y transiciones que les desactivan. Los bloques Modo, Solar, Demanda y Almacenamiento no pasan de esa estructura, pero los bloques RTQ y Control además de esas condiciones tienen prioridades. Algo que en StateFlow estaba definido aquí necesita ser programado de tal manera que siga una estructura general que es que se cumple esta transición sino se ha cumplido la transición de prioridad mayor.

Viendo los cinco bloques a bajo nivel se tiene:

- No tienen prioridades en el bloque
  - Solar se ve que es una programación muy sencilla puesto que solo se tienen dos estados y dos condiciones por lo que cada estado solo tiene una condición de activación y la otra de desactivación y no tiene prioridades puesto que no hay más estados a los que acudir. (Ver Figura 60 Anexos)
  - En el bloque Demanda algo más complejo se tienen cuatro estados y cuatro condiciones pese a ello tampoco hay prioridades puesto que los bloques interactúan dos a dos por lo tanto no hay prioridades posibles de ahí viene esa separación en dos bloques que se ve. (Ver Figura 61 Anexos)
  - En el bloque Almacenamiento hay un mayor aumento de la complejidad este sub-bloque se divide en otros dos sub-bloques que según el selector el tercer sub-bloque de la figura muestra unas salidas u otras si se está en invierno o en verano (para eso se quiere el sexto sub-bloque de entradas para poder hacer esta operación y simplificar salidas). Dentro de cada sub-bloque hay 4 estados y 8 transiciones anidadas (es decir agrupadas cuya única función es simplificar). La razón de que aquí no haya prioridades si en realidad un estado puede ir a los otros 3 es que la condición que se tiene que dar para transicionar depende en todos los casos de las 2 mismas entradas, por lo tanto, no se pueden dar varias condiciones a la vez solo una combinación de ellas cada vez. Por ejemplo, en invierno sí se le mete la entrada de que se requiere ACS y frío sólo va a saltar esa transición no puede saltar que sólo se requiera ACS. En RTQ o Control Lógico

donde la condición de cada transición parte de los valores de distintas entradas. (Ver Figura 62 Anexos).

Otra cosa importante de este sub-bloque es que tanto el sub-bloque “*SummerStorage*” como el sub-bloque “*WinterStorage*” generan 2 mismos estados de salida así que es necesario programar un selector que filtre las que han de salir según el modo en el que se está y que los estados de salida que no se repiten (es decir los otros 4) se pongan a 0 sino se está en el modo que les corresponde. Se puede ver como se ha hecho, usando los bloques *Switch*. (Ver Figura 63 Anexos)

- Si tienen prioridades en el bloque
  - En RTQ como en todos se programan las transiciones. Si se cumple la condición para pasar de estado y se está en el estado anterior se pasar al siguiente. Por ejemplo, si se está en estado cargando y el RTQ ha terminado el proceso de carga pasa a cargado poniendo ese estado a 1. Esto sería un ejemplo de transición de activación. Por otro lado, en ese mismo ejemplo al cumplirse esa condición y estar en el estado anterior ese estado se desactiva poniendo a 0. Eso sería una transición de desactivación. Es decir que en una misma transición para el estado nuevo es un tipo de transición y para el viejo es lo contrario. (Ver Figura 64 Anexos)

Si se va a las prioridades en este bloque se ve que sólo se produce en dos ocasiones al salir de cargando y al salir de descargando. El motivo de la mayor prioridad está justificado en que si se dan las condiciones de que el sistema se ha cargado y a la vez de carga en pausa, se vaya a cargado puesto que algo que está ya cargado no se puede volver a cargar y por tanto no se puede pausar. El mismo razonamiento es válido para la descarga. En cuanto a la programación de la transición se hace que si se han dado ambas condiciones la segunda transición sólo pueda activar el estado correspondiente si la primera no se ha activado. (Ver Figura 65 Anexos)

Dentro del bloque RTQ se añade una alarma para la BC que se explica en la programación de StateFlow. Esta alarma es independiente del otro bloque y solo se compone de dos estados y dos condiciones. Como elemento nuevo se tiene un *Timer* que también se puede ver. (Ver Figura 66 Anexos)

- En Control sucede lo mismo que en RTQ pero de un modo mucho más complejo. La cantidad de estados (9 para el modo Invierno y 8 para el modo Verano) y el gran número de condiciones (28 Invierno y 27 en Verano) hace que su comprensión sea un poco menos intuitiva. (Ver Figura 67 Anexos)

En cuanto a la forma de programarlo es como el sub-bloque Almacenamiento en cuanto a que se divide en 2 bloques según el modo y tiene un selector (ver Figura 68 Anexos).

Si se pasa a las prioridades, la complejidad por el número de transiciones hace que también aumente la complejidad de la prioridad ya que hay estados que tienen 6 transiciones y por lo tanto 6 prioridades. El problema en cuestión es que se prioriza que se pueda cumplir la 6 si no se cumple la 5 por ejemplo, pero si esa prioridad 5 tiene varias transiciones anidadas o las tiene la transición de prioridad 6 ahí hay que programar que si se cumple cualquiera de las transiciones anidadas no se pudiera cumplir la 6. Eso aumenta la complejidad como se ve en la imagen al añadir el *OR*, tanto si las transiciones anidadas son de la prioridad 5 como de la 6. Otro problema es que la prioridad 6 no solo se activa sino lo hace la 5 y también habría que incluir que no se cumplan la 4, 3, 2 y 1.

A parte de eso las prioridades no revisten mayor complejidad. Hay estados que no tienen prioridades como es el caso de *HEATDemandCoveredbySolar* y *HEATDemandCoveredbyTCMCHARGING*. El sistema térmico va a tender siempre a ir esos dos estados anteriores por eso no tienen otro estado prioritario que vaya antes que ellos.

Para tener una idea de cómo se ha programado a bajo nivel se va a mostrar y explicar el ejemplo de programación del estado *HEATDemandTCMCHARGING* ya que es uno de los más simples. Primeramente, se tienen las condiciones de desactivación que se agrupan en un *AND* ya que se tienen que cumplir todas para que no se salga del estado con que suceda una el estado se desactiva. Después se tienen las condiciones de activación en este caso solo hay una. Por último, las prioridades como este caso no tienen no son necesarias programarlas. (Ver Figura 68 Anexos)

Otra característica que se ve en la figura anterior es un *Switch*. Al final ese *Switch* se usa para reiniciar el estado si se cambia de modo ya que cuando se pasa de modo el sistema tiene que apagarse y comenzar en el nuevo. Esto solo sucede aquí. El estado del RTQ por ejemplo no se puede reiniciar puesto que el sistema no puede recircular el amoníaco esa funcionalidad corresponde al

sistema interno del RTQ. En la siguiente figura se ve como se ha programado: se ha usado un bloque *Falling* para que cuando el modo se ponga a 0 reinicie y el estado *OFF* se ponga a 1. Para mayor seguridad se hace que esa condición se mantenga dos segundos con un *Delay*. (Ver Figura 69 Anexos)

### 5.1.3 Salidas

En cuanto a las salidas también se dividen en un bloque: como entradas las salidas de la máquina de estados y como salidas los estados de todos los sub-bloques. Dentro del bloque están los 5 sub-bloques de Simulink, pero aquí consiste en actualizar la salida generada en la máquina de estados para actualizar los valores (descarga del biestable) de válvulas, bombas y ventiloconvectores.

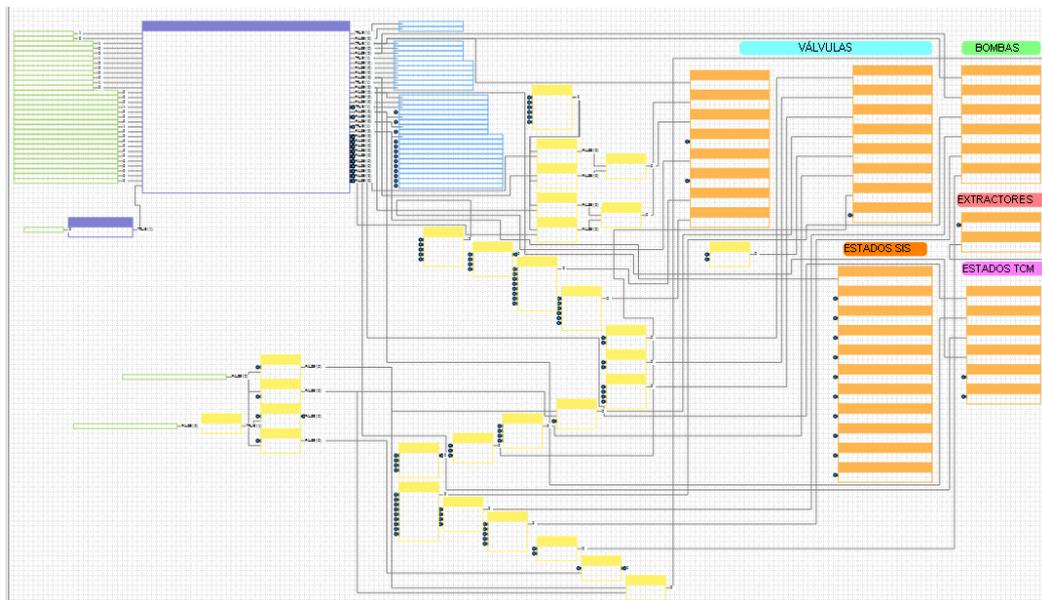


Figura 22: Vista general de la ventana Salidas

Fuera de ese bloque se tienen las salidas que se acaban de mencionar. En las de las válvulas la mayoría de los valores que se generan proceden de los estados del bloque de control lógico, sólo 2 dependen del bloque *Demanda* y otras 2 del bloque *Almacenamiento*. Además, estas de Almacenamiento aparte de depender de ese bloque depende de la variable interna *Heat\_available* que se genera en Control. Luego hay una válvula llamada *fancoilcold* que depende de una condición que sólo es posible que se dé sí se ha dado un estado por eso se encuentran en la salida y no en la máquina de estados, donde se programan todas, porque es una condición que está dentro de un estado. El resto de las condiciones están en las transiciones. (Ver Figura 70 Anexos)

En cuanto a las bombas también hay una que depende del bloque *Demanda*. Las otras 5 dependen exclusivamente de los estados del bloque *Control*.

Finalmente, los dos ventiloconvectores dependen también de las condiciones que se hablaron en las válvulas.

Todas estas variables se meten en *Internal Variable* (con el fin de visualizarlas en el SCADA) también se meten los estados de Control y RTQ para visualizarlos allí como se explica en el apartado Interfaz. (Ver Figura 71 Anexos)

Para que se tenga cierta idea de cómo se han programado las salidas de los estados se muestra una. (Ver Figura 72 Anexos).

Como se puede ver se introduce el nuevo valor en una *Internal Variable* para guardarlo y se muestra con el fin de que vuelva a la entrada empezando un nuevo ciclo.

Para las salidas de las válvulas, bombas y ventiloconvectores se explica cómo se ha programado la válvula *HPevaporator\_in* ya que es la más simple. Para que se active esta válvula se tiene que activar alguno de los dos estados ya sea *HEATdemandCoveredbyTCMCHARGING* o *Transition\_HEATdemandCoveredbyTCMCHARGING* (Ver Figura 73 Anexos).

## 5.2 Interfaz (ENVYSION)

Como parte del proyecto se ha programado un SCADA para visualizar de forma sencilla el estado de las entradas y salidas del PLC, facilitando la supervisión de la batería de pruebas que se introducen desde un programa en Python. En este apartado se explican las partes que componen el SCADA y se incluye una pequeña guía de uso de cara a que el usuario controle la simulación desde un programa en Python.

Todo el sistema se visualiza desde ENVYSION que es plataforma online propia de supervisión y desarrollo del PLC (ENVYSION y EC-GFX son programas que parten de la misma interfaz para programar el PLC). La comunicación entre el PLC y el programa en Python se lleva a cabo mediante el protocolo Modbus.

Centrándonos en las distintas partes del ENVYSION se tiene:

## 5.2.1 Sistema

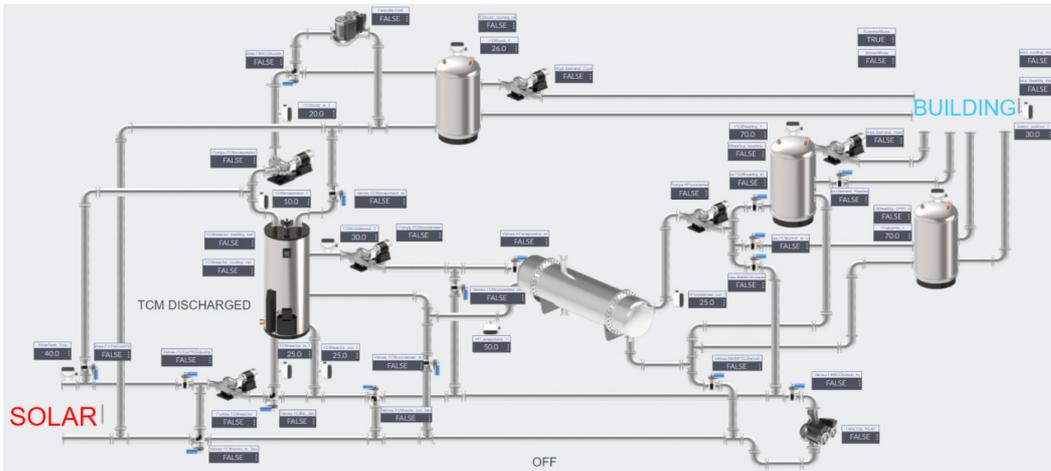


Figura 23: Vista de la ventana Sistema en ENVYSION

Es donde se muestra el control y la supervisión del sistema. Como se ve en la Figura 23 se ha diseñado los conductos reales las válvulas (de tipo todo o nada), las bombas (tipo todo o nada, aunque en la realidad es modular tipo PWM), los ventiloconvectores (tipo todo o nada), los sensores de temperatura, las baterías, el RTQ y la BC. Cabe destacar que para estos tres últimos componentes se ha usado la imagen más representativa disponible en las librerías de ENVYSION ya que son componentes muy específicos que no se hallan en ella. Cada uno de estos componentes tiene asociada un indicador que es la que recibe y muestra el dato Modbus correspondiente a cada objeto. Cuando el sistema está en funcionamiento el diseño muestra el flujo de agua coloreando el conducto por el que discurre y además ese color depende de la temperatura del sensor más cercano al conducto que tiene flujo, de cara a que sea un diseño más realista y mucho más visual.

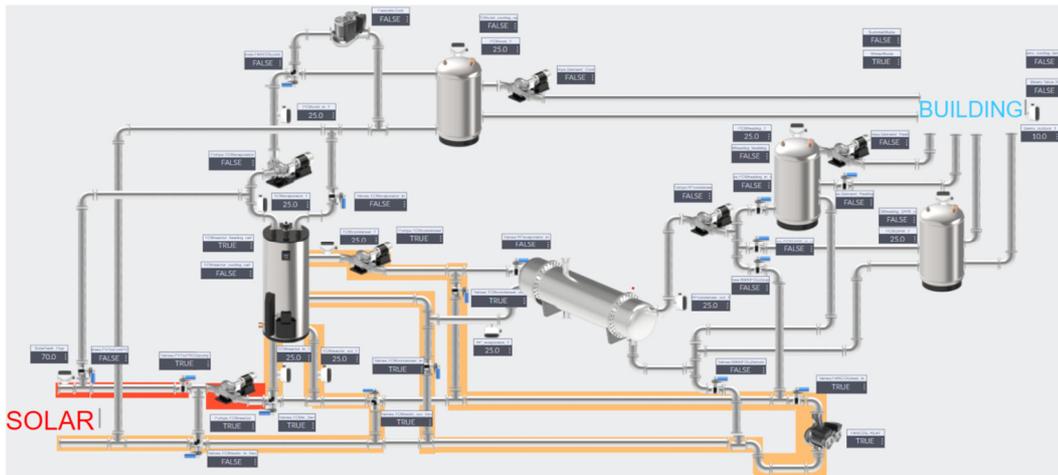


Figura 24: Vista de la ventana Sistema en ENVYISION con un estado concreto

En la Figura 24 también se ve que el sistema recibe los conductos de entrada del sistema solar y le devuelve al sistema edificio, sin entrar en esos sistemas. En esta figura se puede ver un ejemplo del estado *NOHEATDemandTCMCHARGING*.

## 5.2.2 Controles

La ventana *controles* muestra unos controles que permiten modificar las entradas del sistema de manera manual y no adquiriendo los datos a través del servidor. Su uso es el de validar una serie de entradas específicas introducidas por el usuario, pero en la práctica no se usa. Es el servidor de donde provengan las entradas al sistema. Para que funcionen correctamente si se quieren usar se ha de pausar el envío de datos de Python. (Ver Figura 41 Anexos)

## 5.2.3 Gráficas

En esta ventana se tienen todas las gráficas que sirven para validar los datos a lo largo del tiempo ya que el SCADA funciona en tiempo real. También se tiene el CSV donde se guardan estos valores, pero no se ve de una manera tan clara como aquí.

En la siguiente figura se muestran las gráficas de unas de las válvulas donde se ve cómo oscila entre 0 y 1. Unas están más tiempo abiertas en cambio otras no se abren o se abren muy pocas veces.

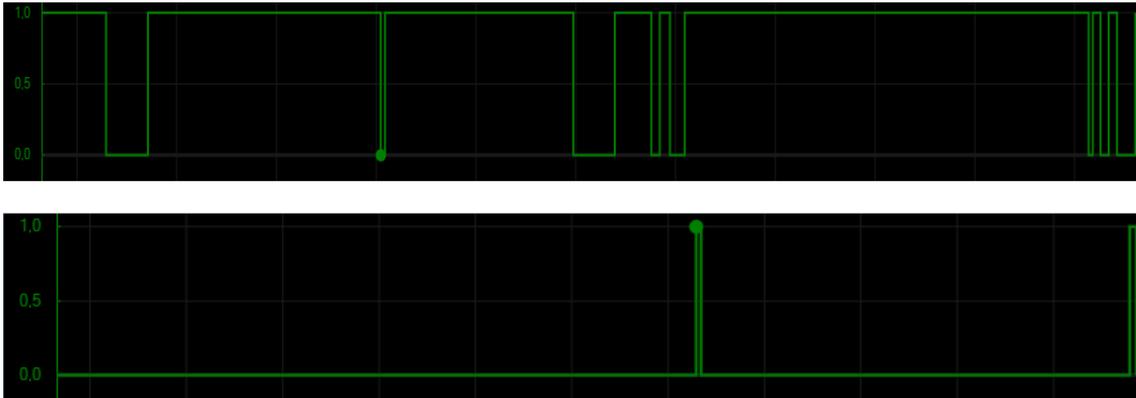


Figura 25: Gráfica Válvulas

Estas gráficas recogen los datos de las marcas que guarda EC-GFX. Es posible configurar cómo se capturan si cambian de valor (cov) y eso se comprueba cada 1 segundo como se indica. (Ver Figura 42 Anexos)

En la Figura 26 se muestra la gráfica de las bombas en la que sucede lo mismo: su encendido y apagado es mayor en unas que otras debido a que es más común el uso de algunas como por ejemplo *TCM\_reactor* (Figura 26 superior) frente a *DemandCooling* (Figura 26 inferior) ya que esta se encarga de recircular el agua que entra del sistema y la otra solo se usa si hay demanda de la batería frío.

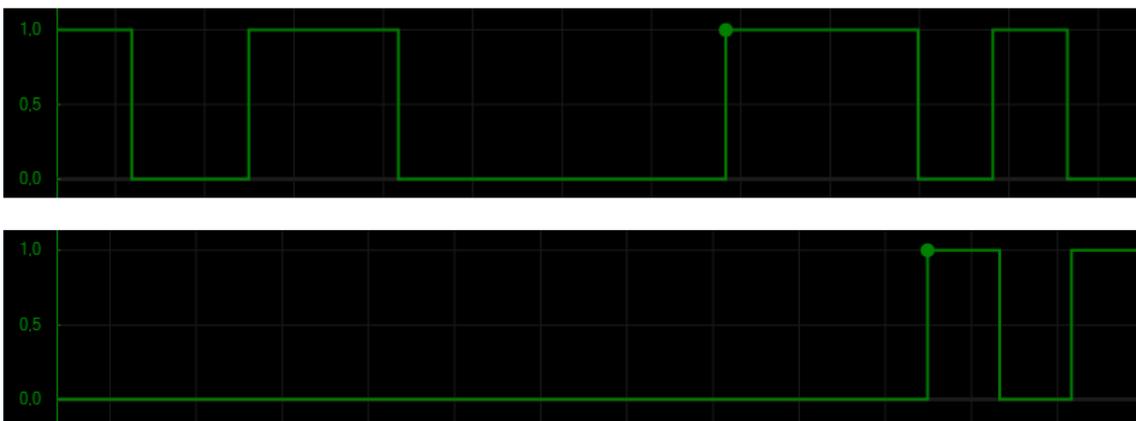


Figura 26: Gráfica Bombas

En la Figura 27 se muestra la gráfica de los ventilosconvectores los cuales se ven muy pocas veces abiertos ya que el sistema lo último que quiere es perder calor, aunque a veces lo haga por razones de seguridad. De hecho, durante la simulación de una batería de casos el ventilosconvectorde frío ni se ha abierto como se ve en la figura superior.



Figura 27: Gráfica Ventiloinvectores

### 5.3 Guía de Usuario

ENVYISION dispone de una pestaña (que se muestra en la siguiente Figura 43 de Anexos) en la que si se selecciona “Edit” se pueden mover todos los objetos y editarlos, por lo que como el sistema ya está configurado el usuario debe tenerlo siempre en la opción “Preview” que permite visualizarlo e interactuar. No es posible simular este SCADA y que el usuario no pueda modificarlo.

En la ventana *controles* el usuario puede introducir los valores que quiera siempre que el servidor de datos esté pausado, como se dijo antes. Esos controles se dividen en botones de encendido y apagado los cuales al pulsar se activa o desactiva la variable *lógica* asociada. El otro tipo son las casillas numéricas donde se pueden introducir valores enteros de signo positivo o negativo. (Ver Figura 44 Anexos)

Respecto a la pestaña *graficas* se ven la siguiente ventana (Ver Figura 45 Anexos) que se puede seleccionar y deseleccionar la variable que se quiera visualizar. Las gráficas tienen otras opciones como el tiempo en el que se obtienen los valores, el intervalo entre los mismos y por último que valor quieres el último, la media...También hay otra opción que es exportarlo a un CSV útil de cara a recoger datos de simulaciones.

Otra cosa que se tiene es la vista del SCADA que ya se explicó y que varía si se introducen valores con los controles que se han mencionado o con las baterías de prueba desde un programa en Python.

Para ejecutar adecuadamente la batería de datos de prueba primero hay que ejecutar EC-GFX para que la función *Startup* reinicie el sistema, así se sabe que entran desde el estado inicial. Después desde Python se ejecuta primero el

servidor y luego el cliente para que escriba los datos del CSV que va leyendo ya automáticamente el PLC. Para dejar de simular si se quiere ver un caso concreto solo habría que pausar el cliente de escritura del programa en Python.

Por último, hay que explicar cómo se introducen en la Interfaz las variables de EC-GFX. Para ello se accede a la ventana configuración y en ella se puede ver la carpeta “ENVYSION Data Tree” donde se asignan todas las variables de datos de EC-GFX que se quiere que aparezcan en ENVYSION. Como se puede ver se han dividido en subcarpetas para una mayor organización (Ver Figura 46 Anexos).

## 5.4 Conexión Modbus

Como se puede ver en la ventana entradas, se halla un bloque llamado *ModbusDevice*. Este bloque nos permite realizar una conexión tipo Modbus entre el PLC y el sistema real. Para probar esta conexión en este proyecto se ha configurado de tal manera que lea desde un servidor (ya que el PLC es el cliente) una serie de datos de temperatura y señales de llamada a través de unos registros de lectura.(Ver Figura 80 Anexos).

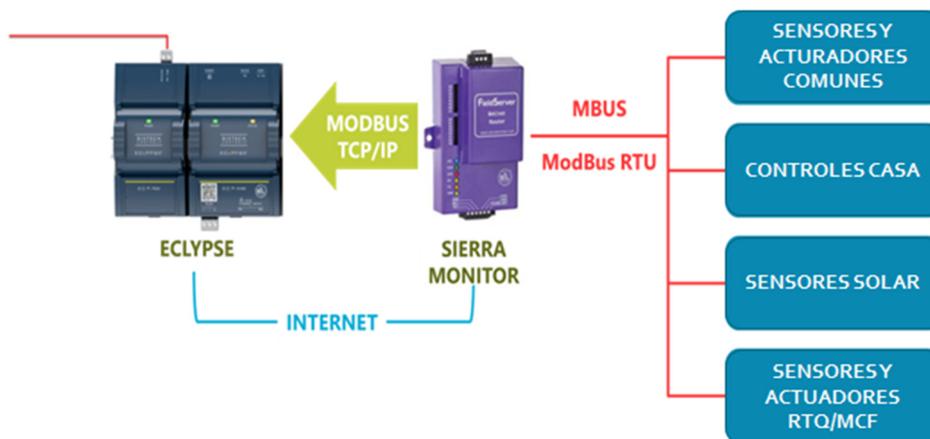


Figura 28: Esquema conexión Modbus

Para explicar la configuración se puede consultar la Figura 74 Anexos donde se muestra cómo hay que especificar la IP del servidor donde lee o escribe, el puerto y también la dirección Modbus.

Otro punto que es necesario especificar es el número máximo de registros, que quiere decir el número de datos que se van a mandar dentro de cada registro. En este caso se tienen veintiún registros de entrada, pero por cada registro se envía un dato, y esto se ejecuta a cada paso de simulación.

Una vez configurado el *ModbusDevice* se tienen que ir añadiendo registros que como se ha dicho son las entradas y salidas, en total 21 registros los cuales se configuran según los siguientes pasos:

- Asignar un valor de registro, se especifica el tipo decimal
- Decir que tipo de registro es, son tipo analógico ya que las *booleanas* se mandan como 0 y 1 y tipo  *Holding*  ya que se quiere que se mantenga el valor hasta que se reescriba después se debe decir sí el cliente lee en el registro o lo escribe y sí tiene la unidad de medida en este caso solo las temperaturas tienen °C, también se especifica un valor por defecto que se muestra en caso de que no se mande nada (solo en el primer instante sucede eso).
- Especificar cada cuanto se lee se escribe el registro en nuestro caso es cada segundo. (Ver Figura 75 Anexos)

Para la comprobación de esta conexión y con el fin de validar el modelo se usa Python. Se programa un servidor desde el que lee y escribe el PLC y un cliente que escribe las entradas que recibe el PLC. En la vida real estas entradas las escribirían los sensores y controles de cada elemento individual, mientras que aquí son escritas por Python y definidas desde un archivo CSV. Para esta programación se ha usado la librería *PymodbusTCP*.

En el servidor lo que se hace es crear el servidor especificando la IP del servidor (que sería la del ordenador) y el puerto que se quiere usar, en este caso el 502 con la dirección 1 como se especifica.

Se abre el servidor con el comando “*PyModbusTCP.server*” de la librería y se deja abierto con “*start()*” ya que el sistema está siempre en funcionamiento. (Ver Figura 76 Anexos)

Luego se tiene un cliente en Python que escribe de CSV los datos además de otras aplicaciones que se explican a continuación.

En el cliente de escritura lo que se hace es leer de un CSV la cadena de valores que se quieren escribir. Para ello se importa la librería CSV y se leen el número de filas del archivo. El cliente se crea con el comando *PyModbus.client* y se abre. Después se llama al CSV del que se leen las filas que tiene para saber cuántas veces se va a repetir el envío. (Ver Figura 77 Anexos)

Dentro se programa un bucle *FOR* que envíe todos los valores de cada fila en cada iteración. Para ello lee los valores de la fila y se escribe como registro. Una vez hecho esto se espera un tiempo determinado entre envíos, lo cual permite

la librería *Time*. Después de esto se lee la siguiente fila hasta finalizar todas. (Ver Figura 78 Anexos)

Además los valores que son *double*, como son todas las temperaturas, reciben un tratamiento especial antes de ser enviados. Primeramente para que sea más realista se ha escalado la diferencia de temperatura entre el valor de una fila y otra en pasos (10 por defecto) de tal manera que el primero se envíe de manera normal pero el valor de la segunda fila se escale con el valor anterior. Por ejemplo si el primer valor es 30 y el de la fila siguiente 50, lo que se hace es que esa diferencia que sería 20 se divida entre el número de pasos que son diez por tanto en vez de enviar un valor que pase de treinta a cincuenta se mandan diez valores incrementan de dos en dos (en este caso, sería 30, 32, 34...50). Para ello también se envían el resto de valores sean booleanos o temperaturas que no varíen- Esto es redundante puesto que se manda el mismo dato varias veces, pero es lo más fácil para no perder esa estructura de cadena. (Ver Figura 79 Anexos)

Otra característica que se ha realizado con los valores es asignarles un decimal. Para ello, se multiplican por 10 ya que los registros sólo envían valores enteros, y de esta manera se leen los valores y se calculan multiplicados por diez y en el PLC se dividen entre diez para obtener el decimal. (Ver figura 80 Anexos).

Como se puede ver en Figura 79 de Anexos para programar este cálculo escalar del que se ha hablado antes se usó un *IF-ELSE* y un *FOR*, que restando el dato actual a enviar menos el anterior y dividiendo entre el nº de pasos, obtenga el incremento a sumar cada vez y lo haga.

Después se multiplica por 10 para hacer el dato un entero ya que los registros solo envían enteros.

Además, se espera un tiempo el cual es proporcional al número de pasos como se puede ver en la (Figura 77 de Anexos) por ejemplo el tiempo que se espera entre la fila uno y dos es 10 segundos (TIME) pero el tiempo en la fila dos entre envíos es de 10 segundos / nº de pasos es decir en este caso sería cada 1 segundo cuando se envía realmente una cadena (newtime).

## 5.5 Validación del modelo (Python)

Lo que se hace para demostrar el correcto funcionamiento del control en vez de poner toda la batería de casos que son los mismos que en Simulink (tabla 1 en Anexos) y demostrar la veracidad con tres simulaciones como se hizo en StateFlow, es mostrar un caso, es decir, mostrar todas las simulaciones que conforman un ciclo completo de funcionamiento del sistema. Para ello se

explica lo que va sucediendo en cada captura de ENVYSION. Inicialmente se muestra una imagen del CSV con los valores introducidos.

| Solartank_Top | TCMcondenser_T | TCMreactor_heating_call | TCMreactor_cooling_call | TCMreactor_in_T | TCMreactor_out_T | TCMEvaporator_T | HPcondenser_out_T | HPevaporator_out_T | PCMheating_heating_call | PCMheating_T | PCVIDHW_heating_call | PCMDHW_T | PCMcold_in_T | PCMcold_cooling_call | PCMcold_T | Demo_outdoor_T | SummerMode | WinterMode | Demo_heating_demand | Demo_cooling_demand |
|---------------|----------------|-------------------------|-------------------------|-----------------|------------------|-----------------|-------------------|--------------------|-------------------------|--------------|----------------------|----------|--------------|----------------------|-----------|----------------|------------|------------|---------------------|---------------------|
| 40            | 30             | 0                       | 0                       | 25              | 25               | 25              | 25                | 50                 | 0                       | 70           | 0                    | 71       | 25           | 0                    | 20        | 10             | 0          | 1          | 1                   | 0                   |
| 65            | 30             | 1                       | 0                       | 25              | 25               | 25              | 25                | 50                 | 0                       | 70           | 0                    | 71       | 25           | 0                    | 20        | 10             | 0          | 1          | 1                   | 0                   |
| 65            | 30             | 1                       | 0                       | 25              | 25               | 25              | 25                | 50                 | 1                       | 72           | 0                    | 71       | 25           | 0                    | 20        | 10             | 0          | 1          | 0                   | 1                   |
| 65            | 30             | 1                       | 0                       | 83              | 80               | 25              | 79                | 50                 | 1                       | 72           | 0                    | 73       | 25           | 0                    | 20        | 10             | 0          | 1          | 0                   | 0                   |
| 65            | 30             | 1                       | 0                       | 83              | 80               | 25              | 79                | 50                 | 1                       | 72           | 0                    | 73       | 25           | 0                    | 20        | 10             | 0          | 1          | 0                   | 0                   |
| 65            | 30             | 0                       | 1                       | 25              | 25               | 25              | 79                | 50                 | 1                       | 72           | 0                    | 73       | 25           | 0                    | 2         | 0              | 0          | 1          | 1                   | 1                   |
| 65            | 30             | 0                       | 1                       | 25              | 25               | 25              | 79                | 50                 | 1                       | 72           | 0                    | 73       | 25           | 0                    | 20        | 10             | 0          | 1          | 1                   | 1                   |
| 65            | 30             | 0                       | 1                       | 70              | 70               | 25              | 79                | 50                 | 1                       | 72           | 0                    | 73       | 25           | 0                    | 20        | 10             | 0          | 1          | 1                   | 0                   |
| 65            | 30             | 0                       | 1                       | 70              | 70               | 25              | 79                | 50                 | 1                       | 72           | 0                    | 73       | 25           | 0                    | 20        | 10             | 0          | 1          | 1                   | 0                   |
| 65            | 30             | 0                       | 1                       | 70              | 70               | 25              | 79                | 50                 | 1                       | 72           | 0                    | 73       | 25           | 0                    | 20        | 10             | 0          | 1          | 1                   | 0                   |
| 40            | 30             | 0                       | 0                       | 25              | 25               | 25              | 25                | 50                 | 0                       | 70           | 0                    | 70       | 20           | 0                    | 20        | 30             | 1          | 0          | 0                   | 1                   |

Figura 29: Captura código Simulado

Este caso se simula en periodo invierno como se ve en la figura anterior, se compone de 10 simulaciones con un paso de 10 y un tiempo de 10 segundos por simulación con lo cual se reproduce en 100 segundos. Cabe destacar que esto es de cara a mostrarlo ya que en la realidad este caso podría darse en un lapso de muchas horas incluso un día entero o más. Comenzando con la primera simulación se ve que la temperatura del tanque de inercia solar es 40°C por lo tanto se considera frío y el sistema permanece apagado como se ve en la Figura 27. Si no hay calor el sistema no puede arrancar, permaneciendo descargado el RTQ.

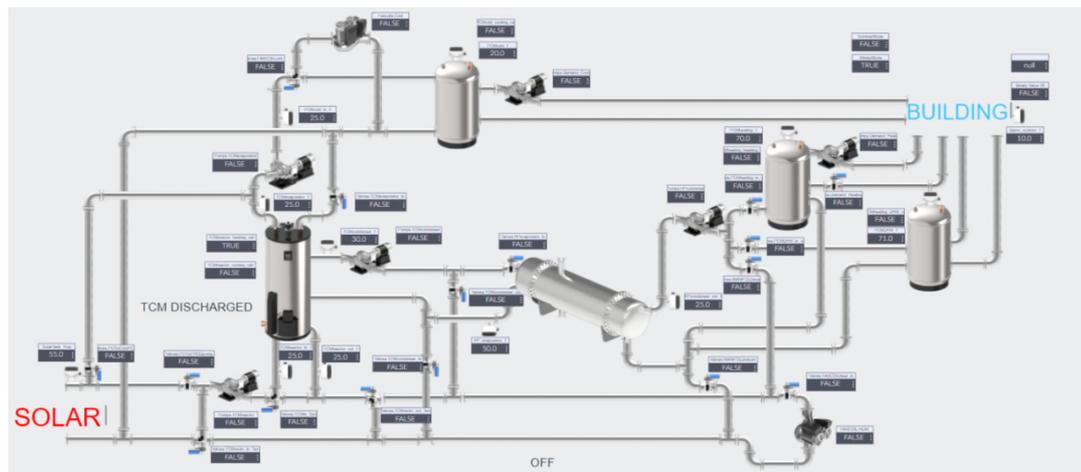


Figura 30: Vista del estado OFF

En la segunda simulación se aprecian 2 cambios:

- Primero la subida de la temperatura del tanque a una temperatura de 65°C, que el sistema ahora si considera calor y por lo tanto arranca. Una vez arrancado comprueba el estado del RTQy su señal. Como la señal esta a 1 significa que se puede cargar y como esta descargado el sistema accede a cargar el RTQ.
- Después el sistema comprueba si va a cargar el RTQ con demanda o sin ella, se ve en la simulación que las señales de demanda de las baterías están todas a 0, por lo tanto no hay demanda.

Estos cambios le hacen acceder al estado *NOHEATdemandTCMCHARGING*, que se muestra en la Figura 31 donde se comprueba que lo comentado es correcto.

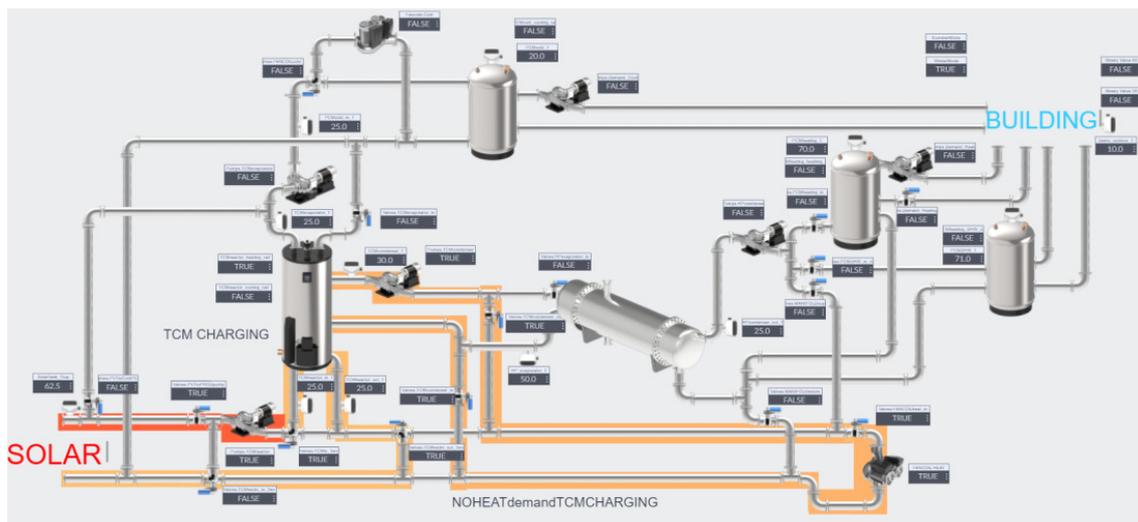


Figura 31: Vista del estado *NOHEATdemandTCMCHARGING*

En la tercera simulación el único cambio es que una de las baterías ahora sí demanda calor por lo tanto el sistema puede pasar a los estados de carga donde sí haya una demanda. Que pase a *Transition\_HEATdemandTCMCHARGING* y no a *HEATdemandTCMCHARGING* o *HEATdemandTCMCHARGINGnoHP* se debe a que:

- La primera alternativa no puede suceder porque la temperatura de salida de la BC en ese momento siempre es inferior a la que tiene la batería, y por tanto cargarla en ese caso es ilógico, ya que se enfriaría. Además aunque fuera mayor, cosa muy improbable, siempre pasa por

el estado transición de cara a una apertura y cierre de válvulas controlada.

- La segunda alternativa se podría dar pero la temperatura del condensador del RTQ tendría que ser 40°C o mayor y superar a la de la batería cosa que solo puede suceder en un instante del arranque y de manera excepcional por lo tanto es un modo de difícil acceso.

Una vez explicado el motivo de porque ha ese estado, se ve en la siguiente figura, que la explicación es correcta y la temperatura de salida de la BC son 25°C y la de la batería 70°C, por lo tanto tiene que transicionar hasta que se caliente por encima de esta para cambiar de estado.

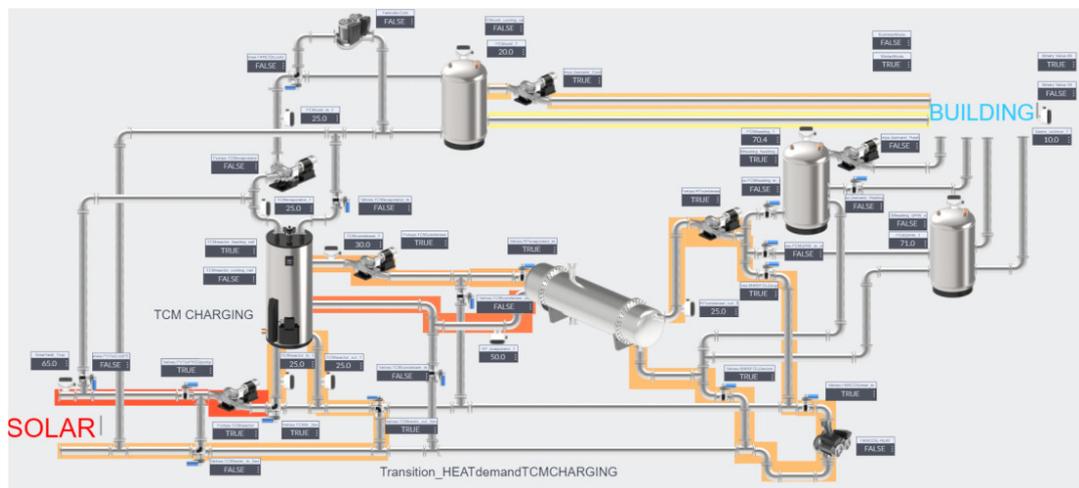


Figura 32: Vista del estado Transition\_HEATdemandTCMCHARGING

En las dos posteriores simulaciones la temperatura de salida de la BC aumenta hasta superar la de la batería de calefacción. En este caso no hay más opciones posibles puesto que *Transition\_HEATdemandTCMCHARGING* como su nombre dice es una transición a *HEATdemandTCMCHARGING*. Además hay que explicar porqué se han usado dos simulaciones. La razón es que esa condición de temperatura se debe cumplir durante al menos 5 segundos, puesto que en la primera simulación no lo consigue, se simula la misma línea dos veces para asegurar que esa condición de 5 segundos se cumpla.



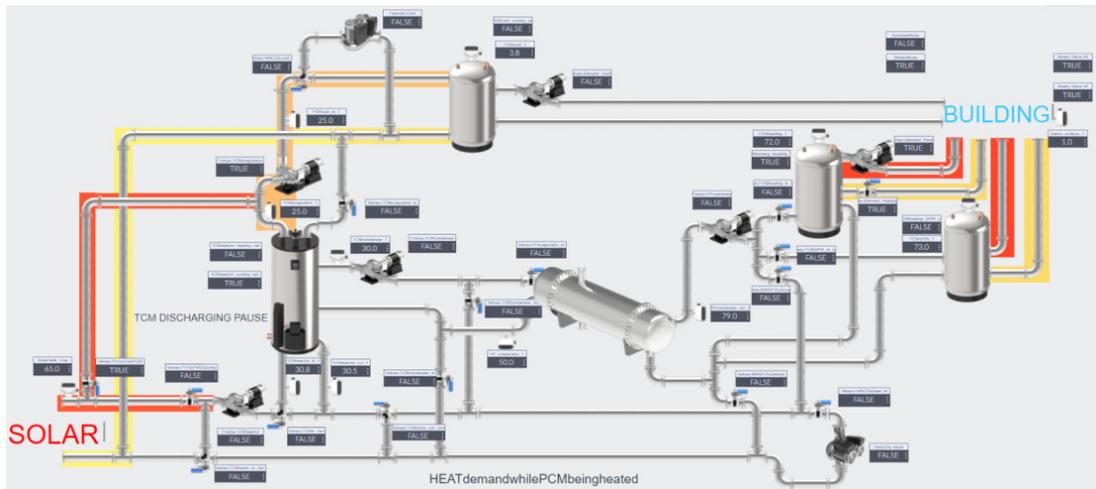


Figura 34: Vista del estado HEATdemandwhilePCMbeingheated

Después de haberse calentado la batería por encima del límite, se da el único estado posible con esos valores de la simulación 7 que es *Transition\_HEATdemandTCMDISCHARGING*.

Que no pase a estados como *HEATdemandTCMDISCHARGING*, *HEATdemandCoverbySolar*, *HEATdemandwhilePCMbeingheated* u *OFF* se debe a que:

- Al primero no puede ir porque tiene que estar al menos 5 segundos en la transición no puede acceder directamente.
- Al segundo no puede ir porque sigue haciendo demasiado frío para que el sol cumpla los requisitos necesarios para satisfacer la demanda de calor.
- Al tercero no puede ir porque la batería está por encima del mínimo después de calentarse.
- Al último tampoco porque sigue habiendo calor en el tanque y se siguen necesitando cargar la batería de calefacción, lo cual se puede hacer en la descarga.

Por tanto en este caso se inicia la transición a la descarga donde se recircula el calor del RTQ hasta que supera al calor de la batería demandada en este caso la de calefacción. En cuanto al frío sí se puede cargar la batería de frío

aunque no se demande. Esto se hace sí la temperatura del evaporador del RTQ es inferior a la existente en la batería de frío. Si esto no sucede se expulsa al exterior (si es posible).

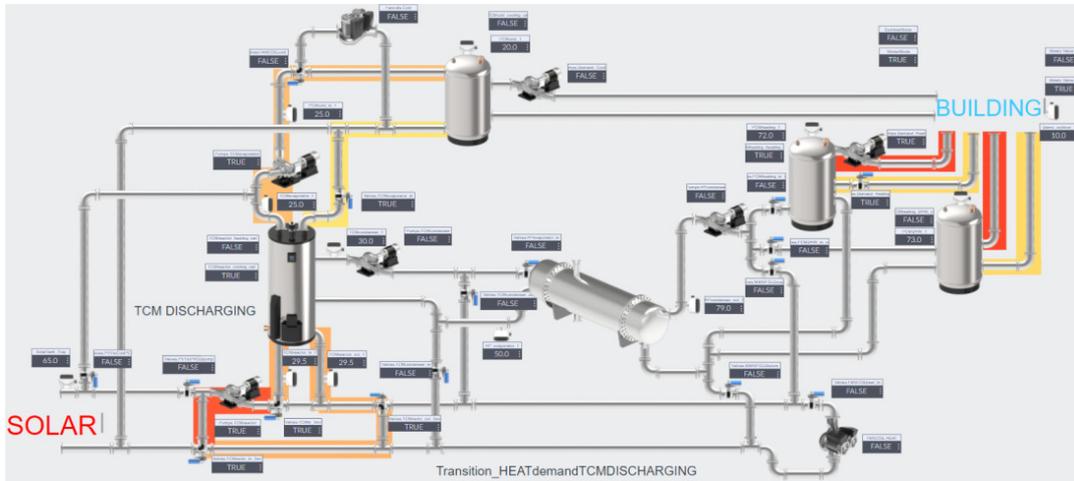


Figura 35: Vista del estado Transition\_HEATdemandTCMDISCHARGING

Después se tienen las simulaciones 8 y 9 donde se ve que se va calentando la recirculación del reactor hasta que llega un punto que supera la temperatura de la batería dándose la condición óptima para poder cargarla. La razón de dos simulaciones iguales es la misma que en carga: se necesita que esa condición se cumpla pasado un tiempo, para asegurar que la temperatura sea estable y no un pico o un error del sensor.

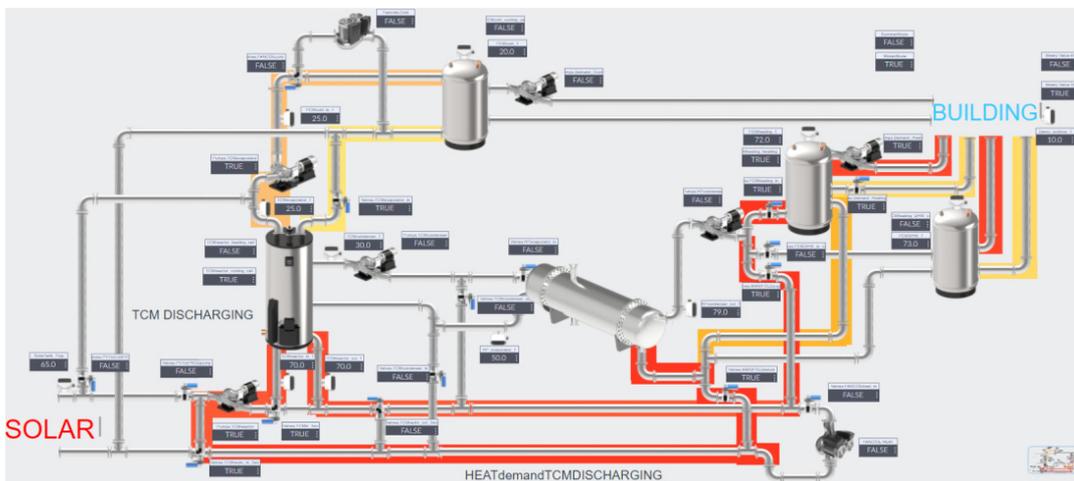


Figura 36: Vista del estado HEATdemandTCMDISCHARGING

Una vez llegado se descarga como ocurre en la simulación 10. Si no hay calor en el tanque el sistema se apaga, sino comenzaría un nuevo ciclo. En este caso baja a 40°C lo que se considera frío por tanto se apaga.





## 6 Contexto

Es necesario hablar de la eficiencia del sistema ya que al interactuar con el entorno tiene algún impacto en el mismo aunque sea leve. Este sistema expulsa tanto calor como frío al ambiente lo cual a primera vista podría ser perjudicial para el entorno pero no lo es porque el sistema contempla una condición de seguridad adicional y es que solo se expulsa si la diferencia no radica en más de 5°C lo cual hace que el impacto sea muy leve otra posible causa de impacto sería el hecho de usar un compuesto químico como es el amoníaco que puede causar daños graves pero eso no sucede puesto que la seguridad tanto del propio reactor como del sistemas impiden que salga al exterior ni en forma líquida ni gaseosa.

En cuanto a las repercusiones económicas este sistema sería muy ventajoso no solo porque nos permite obtener energía termo-solar gracias a las placas sino porque el almacenamiento en forma de calor sin ninguna transformación para su almacenamiento, permite un ahorro de costes y unas bajas pérdidas de energía nada reseñables. Además este control también nos permite proporcionar tanto frío como calor durante todo el año otra ventaja que nos permite una mejora sustancial.

En mi opinión las energías renovables tienen muchas más posibilidades de ser una energía alternativa cien por cien a las fósiles, pero se debe ahondar más en la investigación de nuevos sistemas o materiales para mejorar el almacenamiento, transformación, transporte y distribución. Este es un sistema que lo intenta conseguir con la energía solar, pero combinando otros sistemas seguro que se puede maximizar mucho el rendimiento de este tipo de energías limpias que son minoritarias en la actualidad.



## 7 Conclusión

El objetivo principal de este trabajo era lograr una implementación viable de un sistema de control que pudiera gestionar el funcionamiento del sistema termoeléctrico.

Primero se ha realizado un estudio teórico del sistema a controlar ya que su diseño y forma de funcionamiento evidentemente es fundamental para realizar nuestro control.

Después se ha demostrado de dos formas, diseñando ese control mediante un diagrama de estados usando StateFlow de Matlab/Simulink y posteriormente mediante la programación de un PLC físico usando el entorno EC-GFX.

Para verificar aún más esa implementación se han replicado las conexiones del sistema de control mediante la simulación de un envío de datos cliente-servidor en utilizando un programa en Python.

Finalmente, de cara a poder mostrar toda esta programación y demostrar que funciona se ha creado un SCADA para facilitar la comprensión del código de control.

Como posible continuación de este trabajo estaría la programación a alto nivel con el objetivo de optimizar el control de cara a su uso en la vida real.

Otro punto a tratar sería la conexión de los sensores con una plataforma que recoja los datos en tiempo real y los envíe con la conexión Modbus desarrollada, y ya finalmente su verificación en un sistema térmico real después de su completa validación en los entornos simulados.



## 8 Bibliografía

Pereira da Cunha Jose, Eames Philip (2016). Thermal energy storage for low and medium temperature applications using phase change materials

Hamidi Marziyeh, Bayon Alicia, Wheeler Vincent M, Kreider Peter, Wallace Mark A., Tsuzuki Takuya, Catchpole Kylie, Weimer Alan W. (2019). Reduction kinetics for large spherical 2:1 iron–manganese oxide redox materials for thermochemical energy storage

Goldenberg Niv, Wool Avishai (2013). Accurate modeling of Modbus/TCP for intrusion detection in SCADA systems

Benjamín C. Kuo, Guillermo Aranda Pérez, Francisco Rodríguez Ramírez (1996). Sistemas de control automático

Cabeza, L.F. (2019). Latent Thermal Energy Storage. In A. Frazzica & L. F. Cabeza (Eds), Recent Advancements in Materials and Systems for Thermal Energy Storage. Green energy and Technology: Springer, Cham. doi: [https://doi.org/10.1007/978-3-319-96640-3\\_2](https://doi.org/10.1007/978-3-319-96640-3_2)

Cabeza, L. F., Castell, A., Barreneche, C., de Gracia, A. & Fernández, A. I. (2011). Materials used as PCM in thermal energy storage in buildings: A review. Renewable and Sustainable Energy Reviews, 15(3), 1675-1695. doi:<https://doi.org/10.1016/j.rser.2010.11.018>

Cuypers, R., Maraz, N., Eversdijk, J., Finck, C., Henquet, E., Oversloot, H., de Geus, A. (2012). Development of a Seasonal Thermochemical Storage System. Energy Procedia, 30, 207-214. doi:<https://doi.org/10.1016/j.egypro.2012.11.025>

Finck, C., Spijker, H., Jong, A.-J., Henquet, E., Oversloot, H., & Cuypers, R. (2013). Design of a modular 3 kWh thermochemical heat storage system for space heating application. Paper presented IC-SES 2, Sustainable Energy Storage in Buildings Conference, Dublin, Ireland.

Milone, C., Kato, Y., & Mastronardo, E. (2019). Thermal Energy Storage with Chemical Reactions. In A. Frazzica & L. F. Cabeza (Eds.), Recent Advancements in Materials and Systems for Thermal Energy Storage. Green Energy and Technology: Springer, Cham. doi:[https://doi.org/10.1007/978-3-319-96640-3\\_3](https://doi.org/10.1007/978-3-319-96640-3_3)

Sögütoglu, L. C., Donkers, P. A. J., Fischer, H. R., Huinink, H. P., & Adan, O. C. G. (2018). In-depth investigation of thermochemical performance in a heat

battery: Cyclic analysis of K<sub>2</sub>CO<sub>3</sub>, MgCl<sub>2</sub> and Na<sub>2</sub>S. *Applied Energy*, 215, 159-173. doi:<https://doi.org/10.1016/j.apenergy.2018.01.083>

Sarbu, I. & Sebarchievici, C. (2017). Chapter 6 - Heat Distribution Systems in Buildings. In I. Sarbu & C. Sebarchievici (Eds.), *Solar Heating and Cooling Systems* (pp. 207-239): Academic Press.

Tyagi, V. V., Kaushik, S. C., & Tyagi, S. K. (2012). Advancement in solar photovoltaic/thermal (PV/T) hybrid collector technology. *Renewable and Sustainable Energy Reviews*, 16(3), 1383-1398. doi:<https://doi.org/10.1016/j.rser.2011.12.013>



