**Drain valve Logic**

**Using macros**

* DRAIN\_VALVE\_WORKING\_TIMEOUT 🡪 This macro defines a timeout value in seconds.
* DRAIN\_VALVE\_NOT\_OPENING and DRAIN\_VALVE\_NOT\_CLOSING

🡪These macros are used to represent specific states or conditions related to the drain valve.

**Using variables**

* drainValveStatus

🡪 This variable represents the current status of the drain valve.

* drainValveControlState

🡪This variable represents the current control state of the drain valve .

* PrevdrainValveStatus

🡪This variable stores the previous state of the drain valve status.

* drainValveRunningTimeOut

🡪This variable is a timer that counts down when the drain valve is in operation.

* drainValveTimeout, drainValveOnTimeout, and drainValveOffTimeout

🡪 These variables are used to manage timeouts for specific actions related to the drain valve control.

**updateDrainValveSetting()**

🡪This function is used to update the drain valve settings with the provided on and off timeouts.

**drainValveControl()**

🡪 This function implements a state machine for controlling of drain valve.

**E\_DrainValveIdle**

* In this state, there are no specific actions or logic executed.

**E\_continuouslyOpen**

* **Purpose**

🡪This state is used during cooking operations when the previous step's configured temperature is higher than the current step's temperature. Its purpose is to cool down the system to the current temperature.

* **Actions**

🡪It checks if the drain valve is closed (IsDrainValveClosed()).

🡪If the drain valve is closed, it opens the drain valve (DRAIN\_VALVE\_OPEN()).

🡪It checks if a timer called drainValveRunningTimeOut is greater than 0 (initially set to 30 seconds).

🡪If the timer is still active (greater than 0), it decrements the timer.

🡪If the timer expires (reaches 0), it reports a diagnostic event to indicate that the drain valve couldn't open as expected in this state.

* **Switching**

**🡪** If the drain valve successfully opens and is determined to be open (IsDrainValveOpen()), it resets the drainValveRunningTimeOut back to its initial value (DRAIN\_VALVE\_WORKING\_TIMOUT) and in this state is switched to the E\_DrainValveIdle state.

**E\_openDrainValve**

* **Purpose**

🡪 This state is responsible for opening the drain valve.

* **Actions**

**🡪**It first checks if the drain valve is closed (IsDrainValveClosed()).

🡪If the drain valve is indeed closed, it issues a command to open the drain valve (DRAIN\_VALVE\_OPEN()).

🡪It then checks if a timer called drainValveRunningTimeOut is greater than 0 (initially set to 30 seconds).

🡪If the timer is still active (greater than 0), it decrements the timer.

🡪If the timer expires (reaches 0), it reports a diagnostic event to indicate that the drain valve couldn't open as expected in this state.

* **Switching**

🡪If the drain valve successfully opens and is determined to be open (IsDrainValveOpen()), it resets the drainValveRunningTimeOut back to its initial value (DRAIN\_VALVE\_WORKING\_TIMOUT).

🡪if there is ongoing cooking progress (IsCookingProgress()), in this state is switched to the E\_DrainValveOnTime state.

**E\_closeDrainValve**

* **Purpose**

🡪The purpose of this state is to close the drain valve.

* **Actions**

🡪It checks if the drain valve is open (IsDrainValveOpen()).

🡪If the drain valve is open, it issues a command to close the drain valve (DRAIN\_VALVE\_CLOSE()).

🡪It then checks if a timer called drainValveRunningTimeOut is greater than 0 (initially set to 30 seconds).

🡪If the timer is still active (greater than 0), it decrements the timer.

🡪If the timer expires (reaches 0), it reports a diagnostic event to indicate that the drain valve couldn't close as expected in this state.

* **Switching**

🡪If the drain valve successfully closes and is determined to be closed (IsDrainValveClosed()), it resets the drainValveRunningTimeOut back to its initial value (DRAIN\_VALVE\_WORKING\_TIMOUT).

🡪if there is ongoing cooking progress (IsCookingProgress()),in this state is switched to the E\_DrainValveOnTime state.

**E\_DrainValveOn**

* **Purpose**

**🡪** This condition is to control the time the drain valve should be left open.

* **Actions**

**🡪**It checks if the drain valve is open (IsDrainValveOpen()).

🡪If the drain valve is open, it proceeds to check a timer called drainValveTimeout to see if it's greater than 0.

🡪If drainValveTimeout is greater than 0, it decrements the timer.

🡪If drainValveTimeout reaches 0 (the timer expires), it performs the following actions:

🡪It loads the value of drainValveOffTimeout into drainValveTimeout.

🡪Finally, this state is switched to the E\_closeDrainValve state.

**E\_DrainValveOffTime**

* **Purpose**

🡪 This condition is to control the time the drain valve should be left closed.

* **Actions:**

🡪It checks if the drain valve is closed (IsDrainValveClosed()).

🡪If the drain valve is closed, it proceeds to check a timer called drainValveTimeout to see if it's greater than 0.

🡪If drainValveTimeout is greater than 0, it decrements the timer.

🡪If drainValveTimeout reaches 0 (the timer expires), it performs the following actions:

🡪It loads the value of drainValveOnTimeout into drainValveTimeout

🡪 Finally, this state is switched to the E\_openDrainValve state.

**E\_DeactivateDrainValveControl**

* **Purpose**

🡪The purpose of this state is to deactivate or turn off the drain valve control.

* **Actions**

🡪A command to close the drain valve (DRAIN\_VALVE\_CLOSE()). This command ensures that the drain valve is closed.

🡪After closing the drain valve, it sets the drainValveControlState to E\_DrainValveIdle, indicating that the drain valve control is now in an idle or inactive state.