Greenhouse Environmental Control — Descriptive Walkthrough

1) Function: FC_SensorScaling (ST)

What this block has

- **Inputs (raw):**
  TempRaw, HumRaw, LightRaw (analog words, 0…27648).
  Optional spans: TempMaxEU, HumMaxEU, LightMaxEU to define engineering-unit ranges.
- **Outputs (scaled):**
  TempValue (°C), HumValue (%RH), LightValue (Lux).

What it does when activated

- It **reads** the three raw analog values each scan.
- It **converts** each raw value to an engineering value by simple linear scaling:
  - TempValue = (TempRaw / 27648) * TempMaxEU (e.g., 0–50 °C)
  - HumValue = (HumRaw / 27648) * HumMaxEU (e.g., 0–100 %RH)
  - LightValue= (LightRaw/ 27648) * LightMaxEU (e.g., 0–1000 Lux)
- It **writes** the scaled results into the global DB (where OB1 connected it).

If conditions are… it does this

- If **sensors are normal** (within 0…27648): it outputs a proportional number in EU.
- If a **sensor reads very low/high** (close to 0/27648): the output approaches the span limits (near 0 or near the MaxEU).
- If you pass a **different MaxEU** (e.g., TempMaxEU = 80.0): the °C scale stretches accordingly — no other code changes needed.

Why it exists

- Keeps **all scaling math in one place**, reusable for any channel, and makes the rest of the program work with clean, meaningful units.

---

2) Function Block: FB_ClimateControl (ST)

What this block has

- **Inputs (from FC & DB):**
  TempValue, HumValue, LightValue (scaled);
  TempSet, HumSet, LightSet (setpoints);
  EnTempCtrl, EnHumCtrl, EnLightCtrl (feature enables);
  optional **min on/off times** for each actuator to avoid short cycling.
- **Outputs (to field):**
  Fan, Heater, Humidifier, GrowLight (BOOLs).

- **Static internals (remembered across scans):**
  Hysteresis values (TempHyst, HumHyst), request latches (Req*),
  TON timers for min on/off enforcement, and **latched states** (*StateLatched).
- **Temp (scratch):**
  One-scan booleans like WantFan, WantHeater, WantHumid, WantLight.

What it does when activated (sequence of operation)

1. **Reads current measurements** (TempValue, HumValue, LightValue) and **targets** (TempSet, HumSet, LightSet).
2. **Makes an intent ("Want…") decision** for each actuator:
   - **Temperature logic with hysteresis (mutually exclusive heat/cool):**
     - If TempValue > TempSet + TempHyst → **WantFan = TRUE**, **WantHeater = FALSE** (cooling).
     - If TempValue < TempSet - TempHyst → **WantFan = FALSE**, **WantHeater = TRUE** (heating).
     - If within TempSet ± TempHyst → **both FALSE** (deadband, no action).
   - **Humidity logic with hysteresis:**
     - If HumValue < HumSet - HumHyst → **WantHumid = TRUE**.
     - If HumValue ≥ HumSet → **WantHumid = FALSE**.
     - If in the gap (HumSet - HumHyst ≤ value < HumSet) → **hold previous request** to avoid chatter.
   - **Light logic (threshold):**
     - If LightValue < LightSet → **WantLight = TRUE**, else **FALSE**.
   - If a **control is disabled** (En*Ctrl = FALSE), that actuator's **Want** is forced **FALSE**.
3. **Stores the "Want" decisions** in static request latches (Req*) so they persist through deadbands.
4. **Enforces minimum ON/OFF times** per actuator using TON timers and state latches:
   - If a **request changes** (e.g., OFF→ON):
     - It checks the **opposite minimum time** (e.g., MinOff must be satisfied before allowing ON).
     - It starts the appropriate **TON gate** and **updates the latched state** when allowed.
   - If **no request change**, timers stay idle and the **latched state holds**.
5. **Drives outputs** by combining the **latched state** and the **current request**:
   Output := StateLatched AND Req.
   - This pattern ensures outputs only change when both the **intent** and the **timing rules** agree.

If conditions are… it does this

- If **TempValue jumps high** above TempSet + TempHyst: it **requests Fan ON**, **Heater OFF**; Fan will turn on immediately **or** after MinOff is satisfied (if configured).
- If **TempValue drops low** below TempSet - TempHyst: it **requests Heater ON**, **Fan OFF**; Heater will turn on immediately **or** after its MinOff is satisfied.
- If **humidity is low** (below HumSet - HumHyst): it **requests Humidifier ON**; it will turn OFF once humidity reaches HumSet and any MinOn requirement is met.
- If **light is low** (below LightSet): it **requests GrowLight ON**; OFF when light ≥ setpoint (and MinOn satisfied if used).
- If an **enable is OFF**: that actuator remains **OFF** regardless of measurements.
- If **MinOn/MinOff = 0s**: the actuator **follows the request immediately** (only hysteresis applies).

Why it exists

- Centralizes all **control decisions** with stability features (**hysteresis, min-times**) so hardware isn't abused by rapid cycling and the greenhouse environment remains steady.

---

3) Global Data Block: DB_Greenhouse

What this block has

- **Setpoints** you can tune at runtime: TempSet (°C), HumSet (%RH), LightSet (Lux).
- **Scaled measurements** (written by the FC): TempValue, HumValue, LightValue.
- **Enables** for each control loop: EnTempCtrl, EnHumCtrl, EnLightCtrl.
- **Min on/off time parameters** per actuator: MinOnSec_*, MinOffSec_*.
- **Actuator states** (written by the FB): Fan, Heater, Humidifier, GrowLight.

What it does when used

- Serves as the **single source of truth** for HMI and commissioning:
  - You **watch** live values here.
  - You **edit** setpoints/enables/timers here.
  - You **see** final output commands here.

If conditions are... it does this

- If you **change a setpoint** (e.g., TempSet from 25→27 °C), the FB on its next scan will use 27 °C and may **switch actions** accordingly.
- If you **disable** a loop (e.g., EnHumCtrl := FALSE), the FB **forces that output OFF** and **ignores humidity** until re-enabled.

Why it exists

- Keeps all operator-tunable and operator-visible values in one easy place, making HMI wiring and testing simple.

---

4) Main Program: OB1 (LAD, minimal)

What this block has

- **Network 1:** A **call** to FC_SensorScaling.
- **Network 2:** A **call** to FB_ClimateControl with its **Instance DB**.
- **Network 3: Coils/assignments** mapping DB outputs to physical outputs (Q0.x).

What it does when activated (every scan)

1. **Net 1 — Scale sensors:**
   Reads IW64/IW66/IW68 → calls FC_SensorScaling → **writes** TempValue/HumValue/LightValue into DB_Greenhouse.
2. **Net 2 — Decide actions:**
   Calls FB_ClimateControl using:
   - **Inputs from DB_Greenhouse** (values, setpoints, enables, min-times)
   - **Outputs back to DB_Greenhouse** (Fan/Heater/Humidifier/GrowLight)
3. **Net 3 — Drive hardware:**
   Assigns DB_Greenhouse.* outputs to Q0.0…Q0.3.

If conditions are… it does this

- If a **sensor changes**, Net 1 updates DB values; Net 2 **re-decides**; Net 3 **reflects** the new state on the outputs.
- If the **PLC scan repeats** (it does continuously), the chain **keeps updating** in this same order, ensuring a consistent read → decide → act loop.

Why it exists

- Keeps Ladder to the **bare minimum** (block calls + I/O mapping) while the **real logic lives in ST**.

End-to-End "What Happens When…"

A) Temperature suddenly rises above setpoint + hysteresis

1. **FC** scales TempRaw → higher TempValue.
2. **FB** sees TempValue > TempSet + TempHyst → **wants Fan ON**, **Heater OFF**.
3. If **MinOff for Fan** is satisfied (or 0 s), FB **latches Fan ON**.
4. **OB1** maps Fan = TRUE to **Q0.0 ON**.
5. As temp falls back into the deadband, FB **stops asking** for fan; if MinOn is met, it **turns Fan OFF**.

B) Temperature drops below setpoint – hysteresis

1. **FC** scales TempRaw → lower TempValue.
2. **FB** sees TempValue < TempSet − TempHyst → **wants Heater ON**, **Fan OFF**.
3. If **MinOff for Heater** is satisfied (or 0 s), FB **latches Heater ON**.
4. **OB1** maps Heater = TRUE to **Q0.1 ON**.

C) Humidity is too low

1. **FC** outputs low HumValue.
2. **FB** sees HumValue < HumSet − HumHyst → **wants Humidifier ON**.
3. After MinOff (if any), **Humidifier turns ON** at **Q0.2**.
4. When humidity reaches HumSet, FB requests **OFF**; after MinOn (if any), **Q0.2 turns OFF**.

D) Ambient light is insufficient

1. **FC** outputs low LightValue.
2. **FB** checks LightValue < LightSet → **wants GrowLight ON**.
3. After any timer gates, **Q0.3 turns ON** until ambient light ≥ setpoint.

E) A control loop is disabled

- If EnTempCtrl = FALSE, the FB **does not request Fan/Heater**, and they remain **OFF** regardless of temperature.

Practical Notes & Good Habits

- **Hysteresis** prevents rapid chatter around a setpoint; adjust TempHyst and HumHyst to match your equipment and greenhouse volume.
- **Min on/off times** protect hardware (heaters especially). Start with: Fan 5–10 s, Heater 10–30 s, Humidifier 8–15 s, Light 0–10 s (often 0).
- **Safety interlocks** (E-stop, overtemp thermostat, door switch) should be **hardwired** and also **checked in OB1** before energizing outputs.
- Keep all **runtime-tunable parameters** in DB_Greenhouse; this makes HMI design clean and commissioning fast.