

MSR-2 Temperature & Humidity Offsets

Simple Offsets for Temperature and Humidity:

Please add these to the bottom of your esphome yaml for your device then click save and install. Once you are finished, you will have two new boxes inside the home assistant esphome integration device page for your device where you can fill in an offset. Give them up to 1minute to take effect!

```
sensor:
  - id: !extend scd40
    humidity:
      name: "Humidity"
      id: "humidity"
      filters:
        - lambda: return x - id(scd40_humidity_offset).state;
    temperature:
      name: "Temperature"
      id: "temperature"
      filters:
        - lambda: return x - id(scd40_temperature_offset).state;

number:
  - platform: template
    name: SCD40 Humidity Offset
    id: scd40_humidity_offset
    restore_value: true
    initial_value: -18.86
    min_value: -70.0
    max_value: 70.0
    entity_category: "CONFIG"
    unit_of_measurement: "%"
    optimistic: true
    update_interval: never
    step: 0.1
```

```
mode: box
- platform: template
name: SCD40 Temperature Offset
id: scd40_temperature_offset
initial_value: 14.54
min_value: -70.0
max_value: 70.0
entity_category: "CONFIG"
unit_of_measurement: "°C"
optimistic: true
update_interval: never
step: 0.1
mode: box
```

DPS310 & SCD40 Sensors: Overcoming Temperature & Humidity Reading Challenges

The DPS310 and SCD40 sensors are known for their precision in measuring temperature and humidity. However, like all sensor systems, they can sometimes provide inaccurate readings due to various factors. In the case of the DPS310 and SCD40 sensors, one significant challenge arises from the heat produced by the ESP chip, which can alter the environment inside its enclosure, thereby skewing the readings.

Adjusting the DPS310 and SCD40 Temperature Offset

For users who want to fine-tune their sensors, the DPS310 and SCD40 Temperature Offset entities can be manually adjusted to match the conditions in their home. The offset values are subtracted from the raw temperature & humidity readings in the firmware to update the sensor readings in the home assistant entity. For example: `scd40_temperature` entity = raw `scd40` temperature reading - `scd40_offset`.

By default, these offsets are preset to values based on our NIST certified thermometer, it's important to note that these values are calibrated for our environment. They might not be accurate for all settings. Therefore, by using a reference thermometer, users can adjust the difference between the readings to get a more accurate representation.

Users will also notice the `dps310_temperature_calibrated`, `scd40_humidity_calibrated`, and `scd40_temperature_calibrated` entities. These values utilize the linear filter in the ESPHome firmware to adjust the readings based on our collected data. Again, due to environmental differences, these might not always be precise.

Modeling the Relationship Between Sensors

Another approach to getting accurate readings is to model the relationship between the ESP temperature and the other sensors compared against a reference temperature. This can be achieved by creating a template sensor in Home Assistant that employs a decision tree or our regression coefficients.

Here's an example that can be added to a configuration YAML:

```
sensor:
  - platform: template
    sensors:
      estimated_reference_temperature:
        friendly_name: "Estimated Temperature"
        unit_of_measurement: '°C'
        value_template: >
          {{
            0.3228 * states('sensor.apollo_msr_1_a79e24_esp_temperature') | float +
            0.8702 * states('sensor.apollo_msr_1_a79e24_scd40_temperature') | float +
            -0.1285 * states('sensor.apollo_msr_1_a79e24_dps310_temperature') | float +
            -0.0491 * states('sensor.apollo_msr_1_a79e24_scd40_humidity') | float +
            0.0851
          }}

      estimated_reference_humidity:
        friendly_name: "Estimated Humidity"
        unit_of_measurement: '%'
        value_template: >
          {{
            -1.2468 * states('sensor.apollo_msr_1_a79e24_esp_temperature') | float +
            -2.1959 * states('sensor.apollo_msr_1_a79e24_scd40_temperature') | float +
            2.9604 * states('sensor.apollo_msr_1_a79e24_dps310_temperature') | float +
            0.2380 * states('sensor.apollo_msr_1_a79e24_scd40_humidity') | float +
            1.8283
          }}
```

For users who want to gather their own data with a reference sensor and MSR-2 within their home, we recommend logging data to a CSV using the following YAML entries:

```
#Configuration.yaml
notify:
  - platform: file
    name: sensor_csv_log
    filename: /config/sensor_log.csv
    timestamp: False

#Automations.yaml
- id: log_sensor_data_to_csv
  alias: "Log Sensor Data to CSV"
  trigger:
    platform: time_pattern
    minutes: "/1" # Log data every minute to match reference sensor
  action:
    - service: notify.sensor_csv_log
      data_template:
        message: >
          {{ now().strftime('%Y-%m-%d %H:%M:%S') }} ,
          {{ states('sensor.apollo_msr_1_a79e24_esp_temperature') | default('NA') }} ,
          {{ states('sensor.apollo_msr_1_a79e24_scd40_temperature') | default('NA') }} ,
          {{ states('sensor.apollo_msr_1_a79e24_dps310_temperature') | default('NA') }} ,
          {{ states('sensor.apollo_msr_1_a79e24_scd40_humidity') | default('NA') }} ,
          {{ states('number.apollo_msr_1_a79e24_dps310_temperature_offset') | default('NA') }} ,
          {{ states('number.apollo_msr_1_a79e24_scd40_humidity_offset') | default('NA') }} ,
          {{ states('number.apollo_msr_1_a79e24_scd40_temperature_offset') | default('NA') }} ,
          {{ states('sensor.apollo_msr_1_a79e24_uptime') | default('NA') }}
```

The Interrelation of Temperature and Humidity

It's important to understand that temperature and humidity share an interdependent relationship. When the air temperature rises, its capacity to hold moisture increases, which can decrease relative humidity levels. Conversely, when the temperature falls, the air's capacity to hold moisture decreases, leading to increased humidity. This relationship plays a significant role in how sensors detect and interpret readings, making it even more crucial to ensure accuracy.

Advanced Accuracy with GPIO

For those seeking the highest accuracy, an advanced solution is available. The exposed mezzanine connector on the back of our Apollo boards can be utilized to connect a temperature/humidity sensor. This modification can dramatically improve both temperature and humidity readings,

providing data that's as accurate as possible.

Revision #6

Created 4 May 2024 03:05:28 by justin@apolloautomation.cloud

Updated 17 August 2024 14:09:44 by SmartHomeSellout