# Project Raspberry PI, Domoticz, Mosquitto-MQTT, Node-Red, Arduino ESP32 with temperature sensors and relays

On my Raspberry PI 3B+ is installed Domoticz, Mosquitto as the MQTT broker, Node-Red and the Node-Red dashboard.

All messages between Domoticz, the Arduino and some processes programmed in Node-Red are, where needed, processed in Node-Red and then distributed by Mosquitto.

The Node-Red Dashboard is updated by the Arduino (messages via Node-Red), by Domoticz (messages via Node-Red), by manual intervention on Domoticz(sending messages via Node-Red), by manual intervention on the Node-Red Dashboard itself.

To the Arduino ESP32 are attached 0/1x DHT11 temperature+humidity sensor, 0/1/2x DS18B20 temperature sensor(s) and 0/1/2 relay(s).

The communication between the RPI and the Arduino is done via Wifi. On the Arduino a self written sketch is running which uses PubSubClient to handle the Wifi communication.

In the setup Domoticz is considered to be the master, therefore all inputs must be send to Domoticz with a layout based on the Domoticz defined structure. Every update of the Device Table in Domoticz is reported back by an output message and distributed by Mosquitto.

All output messages from the Arduino (1) are send to Domoticz by Mosquitto(1). Selected output messages from the Arduino are used to update the Node-Red dashboard (2).

Selected device table output messages from Domoticz (3) are send to the Arduino (4). These messages are reformated to reduce air-time. Also selected messages are used to update the Node-Red dashboard (5).

Some processes written in Node-Red will update the Node-Red dashboard (7) and will generated an output message to the Arduino ( 6) . These messages have a field IDX rather than idx. (The Arduino will report back its actions and will send output messages to Domoticz using Mosquitto(1)).

Some logic in Node-Red will update the Node-Red dashboard (9) and will generated a output message to Domoticz (8). (The processing of these messages by Domoticz is as described at (1)).

Some updates of the Node-Red dashboard are send to the Arduino (10).

Some updates of the Node-Red dashboard are send to Domoticz (11).

# More global details:

The Arduino informs Mosquitto to receive topic domoticz/output55 messages. By default Domoticz informs Mosquitto to receive topic domoticz/input messages.

Domoticz processes and stores incoming message. Domoticz always send after each update a complete device message to Mosquitto, topic domoticz/out. NR selects any by Arduino needed messages, strips and modifies the message to domoticz/output55, idx number from the device Heater in Domoticz and the status is on/off. This message is passes on to the Arduino .

Once evey x seconds the Arduino sends temperature and if present humidity information to Mosquitto, topic domoticz/in in 1 or more messages. Arduino's internal led is switched on and and the mentioned messages are followed by a Led status (on) message.

After a timeout (5 seconds) the Arduino switches the internal led off and send a Led status (off) message to Mosquitto.

Incoming messages to the Arduino originate from sending temperature messages (by the Arduino), sending a Led status message (by the Arduino), Generated messages from node-Red processes (for example the demo thermostate, see hereafter), manually changing switches in Domoticz, manually changing switches in Node-Red. The receipt of any incoming messages at the Arduino is indicated by a flash of the internal led.

The "demo"-thermostate part of NR sends (heater-airco) relays on/off messages to the Arduino, topic domoticz/output55, key IDX with the idx number from the device in Domoticz .

The Arduino switches the indicated relay (heater-airco) on/off. The Arduino reports this action to Domoticz by sending a message , topic domoticz/in, key idx, idx number from the device in Domoticz , and the status.

NR modifies this message and sends it to the dashbord to dim/hightlight the Heating/Airco indicator. Also the transmission indicator is highlighted as described before.

# Notes:

Please note that the processing of messages in Domoticz costs up to 7 seconds. As a consequence during this period the tab Switches in Domoticz, the graphs in Domoticz and the Dashboard in Node-Red are not in sync. To be more real-time are in Node-Red the Temperature message indicator and the process Led indicator switched off by Node-Red itself after 2 seconds.

Before running this system all devices and switches must be created in Domoticz. For a list see the "Dump domoticz.db for Node-Red", Domoticz-Switches.jpg, Domoticz-Setup.png, Domoticz-Devices.png of the by this project used devices. To ceate the wanted devices I used a dummy device. From the dump only fields ID, Name, Subtype are relevant. The dump was made with application DB Browser for SQLite. NEVER USE SQLITE ON A RUNNING DATABASE ON THE RPI, ALWAYS ON A COPY, SQLITE WILL ALWAYS DESTROY YOUR RUNNING DATABASE.

In Node-Red all values of compairs with an idx in a message is done using global variables. Therefore on new installations to install these values run once "setup idx values and switches" (after updating these values conform your domoticz database)

In Mosquitto the log is used for showing process status information of the functions.

On the Arduino ESP32 the serial communication interface is used for showing debug messages which also acts as process status information.

# Please refer to :

- the sketch for details of the processing at the Arduino Esp32.

- the flow and the JS statements in Node-Red for details of the processing.

- the dashboard in Node-Red for an impression on how the Node-Red flow presents itself and to exercize manual interventions.

- the tab switches in Domoticz for the manual interventions of the switch devices.

-- the logfile in Domoticz to see all incoming messages.

- the logfile of Node-Red for process information.

- the output of the serial io interface of the Arduino-Esp32 for process information.

Global overview on my implementation of the Node-Red interface.

The "Display" page contains:

2 Rules used to set the limit temperatures for Heat and Cool. Depending on the current temperature these limits will switch on and off the relays0 and 1 attached to the Esp2. The visual representation of the rules can be seen in the gauge controls at the Control page

With the Manual overwrite switch on, the status of the Heating and Airco switches is independant from testing the limit temperatures for Heat and Cool.

Temperature message switch. Whenever a temperature reading is received from the ESP32 this switch is set on for 2 seconds

Heating and Airco switches. If a received temperature reading is below the Heat limit of above the Cool limit the switch is turned on or off. If the Manual overwrite switch is on then the Heating and Airco switches can be manipulated (in Domoticz or Node-Red) and the status remains independant from the Heat and Cool limits.

To demonstrate the usability of linked light switches between Domoticz and Node-Red: Lights switch 59 and 60 are visible.

The Control page contains:

Gauge controls Heater and Airco of the 2 rulers Heat and Cool, see above.

Transmission Led. Any incoming message from the Arduino for Domoticz will lit the Led for 2 seconds. ( Temperature and Humidity readings will also lit the Temperature message switch, see above)

Temperature and Humidity readings from the DHT11 device are presented in the Temperature and Humidity gauges.

Temperature readings from the DS18B20 device(s) are presented in the Temperature-0 (and -1) gauges.

If a temperature or humidity device is not (anymore) present it may be graded out or frozen.

# Versions:

Version 1.2 15-06-2020 Node-Red Flow 1 Domoticz to Esp32

Version 2.3 ESP32\_MQTT\_DHT11\_V2.3 final.ino

Sketch, flow and documentation produced by hfmv 15-06-2010

This document is: Project Raspberry PI-Domoticz-Node-Red-Arduino.docx