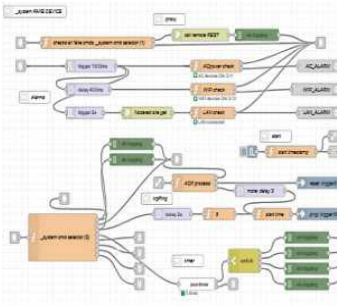
# *tuyaDAEMON device _system*



## Model: system 2.0

Tuya: `custom` Tuya Mod: `SW only device`

*tuyaDAEMON utilities and control*

This is a fake software device, to extend and improve the capabilities of tuyaDAEMON
areas: security and alarms, performance, timers, network

Power supply:
Broadcast:
Capabilities: **SET,GET,SCHEMA**

Reference:
Infos: *https://github.com/msillano/tuyaDAEMON/wiki/custom-device-_system*
Sellers:

## *User defined Data Points:*

| | | | | | |
|---|---|---|---|---|---|
| **_ACpower** | | RO | boolean | true\|false | |
| | *Signals when all AC devices are down. From tuya-smart-device-node. See also node AC_ALARM. PUSHed at any change.* | | | | |
| **_ACunconnected** | | RO | | a list like: ['dev_name1', 'dev_name2'...] | |
| | *List of all AC devices unconnected. From tuya-smart-device-node. PUSHed at any change.* | | | | |
| **_benchmark** | | RW | | a task:{ device, property (, value (,timeout)) } | |
| | *defines the task used in benckmarks. defaults: _system._zeroTask, timeout: 20000 ms* | | | | |
| **_benchmark_end** | | RW | | any | |
| | *To terminate a running benchmark trigger, usually auto, no log* | | | | |
| **_benchmark_step** | | TRG | | any | |
| | *called via share by the task on benchmark internal use, no log* | | | | |
| **_DBase** | | RO | boolean | true\|false | |
| | *Signals when a DB is down. In CORE, from mysql nodes. See also node DB_ALARM. PUSHed at any change.* | | | | |
| **_doBenchmark** | | RW | | any | |
| | *trigger* | | | | |
| **_doTrigger** | | WO | int | trigger value | |
| | *Help property to send a TRIGGER to tuya. note: in code is wired the 'tuya_bridge' countdown 'dp'.* | | | | |
| **_LANnet** | | RO | boolean | true\|false | |
| | *Signals when the WEB access is down. See also node LAN_ALARM. PUSHed at any change.* | | | | |
| **_laststart** | | RO | string | '2021-03-24 16:06:33' | |
| | *PUSHed some seconds after the restart. See also node start_DAEMON* | | | | |
| **_name** | | RO | string | any | |

| | | | | | |
|---|---|---|---|---|---|
| | *GET: returns the name of actual instance of tuyaDAEMON same as global.remotemap.itself* | | | | |
| **_proxy** | | RW | | std_msg: { (remote:'name',) (device:'name\|pseudoDP\|id' (, property:'name\|dp' (, value:'any'))))} | |
| | *SET: uses REST to connect a remote tuyaDAEMON. GET: retuns last response from 'remote', in global.alldevices.<system>.<proxy>* | | | | |
| **_timerList** | | RO | | [{id:<any>, datetime: N, dateTimeStr: 'xxx', alarmPayload: {<share>}},..] | |
| | *GET a list of all active timers. see https://flows.nodered.org/node/node-red-contrib-jsontimer for more info* | | | | |
| **_timerOFF** | | WO | string | the timer 'id' | |
| | *To delete a timer. The 'id' is mandatory. see https://flows.nodered.org/node/node-red-contrib-jsontimer for more info* | | | | |
| **_timerON** | | WO | | {('id': '<any>',) 'timeout\|datetime\|time': <atime>, 'alarmPayload': { 'share': [... ] } } | |
| | *Set a new timer and the 'share' actions associated. see https://flows.nodered.org/node/node-red-contrib-jsontimer and https://github.com/msillano/tuyaDAEMON/wiki/tuyaDAEMOM-global.alldevices#share-actions* | | | | |
| **_toDebug** | | WO | | <string> or {[<string>,<object>,...]} | |
| | *Sends the value to debug pad.* | | | | |
| **_toFastIN** | | WO | | std_msg: { (remote:'name',) (device:'name\|id' (, property:'name\|dp' (, value:'any'))))} | |
| | *To inject a msg to 'fast_cmds' node. Format: see core.'fast_cmds' node.description* | | | | |
| **_toLogging** | | WO | | like std. answer: 'payload':{ 'deviceId': <id\|gateway-id>, 'data': { ('t': <timestamp>,) ('cid': <cid>,) 'dps': { <dp>:<value>}}}; | |
| | *To process a response: echo on Pad, DB and global.tuyastatus updated. Format: see core.'logging' node.description* | | | | |
| **_toLowIN** | | WO | | {'toDev':<deviceID>, 'payload':{<for the tya_smart_device>}}. | |
| | *To inject a msg to a real device, using the 'low_level_IN' node To test new features (in devices or connection node). see core.'low_level_IN' node.description* | | | | |
| **_toShare** | | WO | | { 'share': [ { 'test': [ '<test1>',..], 'action': [{ 'device': 'name\|id', 'property': 'name\|id\|@exp', 'value': 'any\|@exp' },... ] },... ] } | |
| | *To inject a msg to 'share IN' node. see 'share': https://github.com/msillano/tuyaDAEMON/wiki/tuyaDAEMOM-global.alldevices#share-actions* | | | | |
| **_toStdCmd** | | WO | | std_msg: { remote:'name', device:'name\|id', property:'name\|dp', value:'any'} | |
| | *To inject a msg to 'std_cmd' node. Format: see core.'std_cmd' node.description* | | | | |
| **_toWarn** | | WO | string | any | |
| | *Show in locale the value using node.warn().* | | | | |
| **_trgPing** | | RW | | GET: object like: { count: 5, avg: 300, min: 220, max: 420} SET: any | |
| | *Time for a tuya-trigger roundtrip. GET: returns last values (if any) from tuyastatus. SET: starts a new update.* | | | | |

| | | | | | |
|---|---|---|---|---|---|
| **_tuyastatus** | | WO | | like a std_msg: { ('device':<name>, ( 'property':<name>, ('value':<any>))))} | |
| | *Alternative access to global.tuyastatus (e.g. from remote). Performs: LIST, if value:undef, SCHEMA, if value.property:undef, GET, if .value.value: undef, else SET* | | | | |
| **_WiFinet** | | RO | | true\|false | |
| | *Signals when all WIFI devices are down. From tuya-smart-device-node. See also node WiFi_ALARM. PUSHed at any change.* | | | | |
| **_WiFiunconnected** | | RO | | a list like: ['dev_name1', 'dev_name2'...] | |
| | *List of all WiFi devices unconnected. From tuya-smart-device-node. PUSHed at any change.* | | | | |
| **_zeroLog** | | RW | | any | |
| | *Task for benchmark with log. Like zeroTask, but this updates debug pad, DB and tuyastatus.* | | | | |
| **_zeroTask** | | RW | | any | |
| | *Task for benchmark without log. triggers the fastest fake task: it does nothing.* | | | | |