EFENTO

Sensor test dongle

2023-02-01

EFENTO

Table of Contents

Change log (Added or modified)

Chapter - 1 General information

USB dongle allows users to communicate with Efento sensors over Bluetooth. All sensors Bluetooth features are available using the dongle.

Chapter - 2 Using the dongle

- All the commands start with "sensor" (e.g. "sensor scan 28:2C:02:4F:FF:90")
- If you want to repeat the recent commands, press the arrow up key on the keyboard until you find the command you want to repeat
- If you want to use another command with the same sensor, can use "-" dongle will put in its place the recently used sensor's MAC address and PIN (e.g. "sensor scan - -")
- Sensor's MAC address needs to have ":" between each segment
- If you press TAB key while writing a command you will list all the commands, starting with the keyed in letters (e.g. typing "get*" and pressing TAB will list all the commands starting with "get*")
- If you want to list all the available commands along with the list of their parameters, type "sensor"
- Dongle's response is a YAML message, which always contains two fields: serial number and result

ET0-200 - Using the dongle in console

Dongle can be used with any operating system. Once connected to the computer connect to it over COM port.

- Windows: we recommend using PuTTY and open a serial connection with speed set to 115200
- MacOS: use "screen" command to connect to the dongle on the selected serial port

ET0-202 - Using the dongle with Python

- Make sure you have Python 3 and pip installed
- install **pyyaml** using command:

```
pip install PyYAML(Windows)
pip install pyyaml(Linux)
```

- install **pyserial** using command:

```
pip install pyserial (both Windows and Linux)
```

Chapter - 3 Available commands

All the variables in “<>” indicate the name of the variable that needs to be used in the command. For instance if the command description contains “<MAC_ADDRESS>“ it means that this part of the command has to be replaced to the sensor's mac address.

Note that some of the commands are only available for cellular sensors.

Commands listed below are available for sensors with software version 6.0 or higher.

ET0-118 - Scan

Command used to perform scanning for BLE device around the dongle. Return data from the BLE advertisement mode of a sensor with selected serial number

```
sensor scan <MAC_ADDRESS>
```

Optional parameters:

-t<TIMEOUT_VALUE> - defines the time of the scanning (e.g. 10 seconds)

-d - converts the measurement values to decimal (by default they are in HEX)

Examples

```
sensor scan 28:2C:02:4F:FF:90
sensor scan 28:2C:02:4F:FF:90 -t10
sensor scan 28:2C:02:4F:FF:90 -t10 -d
```

ET0-119 - Get software version

Returns sensor’s software version

```
sensor get_sw_version <MAC_ADDRESS>
```

ET0-120 - Enter Bootloader

Enters devices bootloader in order to perform software update

```
sensor  enter_bootloader <MAC_ADDRESS> <BOOTLOADER_PIN>
```

ET0-121 - Data Transfer

Returns measurements from sensor’s memory for the selected time period. Time period is defined by two epoch times.

```
sensor  data_transfer <START_TIMESTAMP> <END_TIMESTAMP>
```

Example

```
sensor data_transfer 28:2C:02:4F:FF:90 1608070309 1608073937
```

ET0-122 - Change measurement period

Sets new measurement’s period base and factor. Measurement period = base*factor (e.g base = 10s, factor = 6, measurement period will be set to 60s. The value of “base” is in seconds, the value of “factor” is a number. Changing the measurement period does not clear sensor’s memory as long as the optional “-r”parameter is not used

```
sensor change_period <MAC_ADDRESS> <PIN> <BASE> <FACTOR>
```

Optional parameters:

-r - removes all the measurements from sensor’s memory

Examples

```
sensor change_period 28:2C:02:4F:FF:90 1111 10 6
sensor change_period 28:2C:02:4F:FF:90 1111 10 6 -r
```

ET0-123 - Get measurement period

Returns the current measurement's period "base" and "factor" values. Measurement period = base*factor

```
sensor get_period <MAC_ADDRESS>
```

ET0-124 - Set sensor's time

Sets the time of the sensor's internal clock. Timestamp in epoch time

```
sensor set_time <MAC_ADDRESS> <PIN> <TIME_STAMP>
```

Example

```
sensor set_time 28:2C:02:4F:FF:90 1111 1608070309
```

ET0-125 - Get sensor's time

Returns the current time of the sensor's internal clock.

```
sensor get_time <MAC_ADDRESS> <PIN>
```

ET0-126 - Restore defaults

Restores the default (factory) settings of the sensor.

```
sensor restore_defaults <MAC_ADDRESS> <PIN>
```

ET0-127 - Get types of sensor’s channels

Returns the types of sensor’s channels (values measured by the sensor)

```
sensor get_channel_types <MAC_ADDRESS>
```

ET0-128 - Get sensor models

Returns the models of the sensors assigned to each slot

```
sensor get_channel_models <MAC_ADDRESS>
```

ET0-129 - Get factory descriptor

Returns the information about the hardware version and variant

```
sensor get_factory_descriptor <MAC_ADDRESS> <PIN>
```

ET0-130 - Get self test results

Returns the results of the self test performed on the device’s boot up

```
sensor get_selftest_data <MAC_ADDRESS> <PIN>
```

ET0-131 - Get battery status

Returns the battery status including the voltage and the battery reset date

```
sensor get_battery_status <MAC_ADDRESS> <PIN>
```

ET0-132 - Reset battery status

Resets the battery status and sets the battery reset day. Battery rest timestamp in epoch time

```
sensor reset_battery_status <MAC_ADDRESS> <PIN> <RESET_TIMESTAMP>
```

ET0-133 - Set sensor name

Sets sensor’s name. Name is used in Bluetooth advertisement frames

```
sensor set_name <MAC_ADDRESS> <PIN> <NAME>
```

ET0-134 - Get sensor name

Returns sensor’s name

```
sensor_get_name <MAC_ADDRESS> <PIN>
```

ET0-135 - Set sensor calibration date

Sets the date, when the sensor was calibrated (date1) and the date, when it should be calibrated again (date2). Dates are epoch times

```
sensor write_calibraiton_date <MAC_ADDRESS> <MANUFACTURER_KEY> <DATE1> <DATE2>
```

Example

```
sensor write_calibraiton_date 28:2C:02:4F:FF:90 111111 1608070309 1609070309
```

ET0-136 - Set encryption key

Sets encryption key used to encrypt the BLE communication

```
sensor set_encryption_key <MAC_ADDRESS> <ENCRYPTION_KEY>
```

ET0-137 - Get encryption counter

Returns the value of the encryption counter

```
sensor get_encryption_counter <MAC_ADDRESS>
```

ET0-138 - Set Bluetooth TX power

Sets the value of the Bluetooth TX power. Value in dBm. Maximum TX power value 4

```
sensor set_bluetooth_tx_power <MAC_ADDRESS> <PIN> <TX_POWER>
```

ET0-139 - Get Bluetooth TX power

Returns the value of Bluetooth TX power

```
sensor get_bluetooth_tx_power <MAC_ADDRESS> <PIN>
```

ET0-140 - Set Bluetooth turn off time

Sets time period, after which BLE interface will switch off. BLE will automatically turn off after this time period each time the device is turned on. Turn off time in seconds - e.g. if set to 600, BLE will turn off 600 seconds after the device boots up. To disable BLE turn off time (keep BLE interface always on), set the BLE turn off time to 0xFFFFFFFF.

```
sensor set_bluetooth_turn_off_time <MAC_ADDRESS> <PIN> <TURN_OFF_TIME>
```

ET0-141 - Get cellular status

Returns the status of the cellular communication (signal strength, registration status, etc.)

```
sensor get_cellular_network_status <MAC_ADDRES> <PIN>
```

ET0-142 - Get cellular module statistics

Returns the statistics of the cellular module (tx power, ecl, snr, pci, rsrq, etc.)

```
sensor get_cellular_module_stats <MAC_ADDRES> <PIN>
```

ET0-143 - Enter installation mode

Enters the installation mode

sensor enter_installation_mode <MAC_ADDRESS> <PIN>

ET0-144 - Exit installation mode

Exits the installation mode (if not used, sensor automatically exits the installation mode 5 minutes after it entered it)

sensor exit_installation_mode <MAC_ADDRESS> <PIN>

ET0-145 - Set APN

There are two commands used to configure APN on the device:

- For sensor's firmware >=6.11.0 use command "**set_apn**" (supports user name and password)
- For sensor's firmware <6.11.0 use command "**legacy_set_apn**" ** (does not support user name and password)

1) **set_apn (sensor's fw 6.11.0 or newer):**

Sets the APN, user name and password values.

sensor set_apn <MAC_ADDRESS> <PIN> ([--hex] <APN> [<USER_NAME>] [<PASSWORD>]) | (-a)

-a - used to set all values to “auto”
--hex - used to insert values in hex format (if not added, the default option is "string")
Possible valid configurations:

-
    - **All empty - sets APN value to "automatic":**

All values are empty (special character 0x7F inserted into APN, user name, password):

```
<MAC_ADDRESS> <PIN> (-a)
```
example:

sensor set_apn 28:2C:02:4F:FF:90 1111 -a

-
    - **Hex option:**

All values in hex (special character 0x7F is inserted into not used parameters):

```
<MAC_ADDRESS> <PIN> (—hex) <APN>
```

```
<MAC_ADDRESS> <PIN> (—hex) <APN> <USER_NAME>
```

```
<MAC_ADDRESS> <PIN> (—hex) <APN> <USER_NAME> <PASSWORD>
```

examples:

```
 sensor set_apn 28:2C:02:4F:FF:90 1111 --hex 746573742E61706E
```

```
sensor set_apn 28:2C:02:4F:FF:90 1111 --hex 746573742E61706E 75736572
```

```
sensor set_apn 28:2C:02:4F:FF:90 1111 --hex 746573742E61706E 75736572
717765727479
```

-
  - **String option:**

All values are strings (special character 0x7F inserted into not used parameters):

```
<MAC_ADDRESS> <PIN> <APN>
```

```
<MAC_ADDRESS> <PIN> <APN> <USER_NAME>
```

```
<MAC_ADDRESS> <PIN> <APN> <USER_NAME> <PASSWORD>
```

examples:

```
 sensor set_apn 28:2C:02:4F:FF:90 1111 test.apn
```

```
sensor set_apn 28:2C:02:4F:FF:90 1111 test.apn user
```

```
sensor set_apn 28:2C:02:4F:FF:90 1111 test.apn user qwerty
```

1) **legacy_get_apn (sensor's fw <6.11):**

Sets the APN value. To set APN value to “auto” use the optional “-a” parameter

```
sensor legacy_set_apn <MAC_ADDRESS> <PIN> ([--hex] <APN_VALUE>) | (-a)
```

Optional parameters:
-a - used to set APN value to “auto”
--hex used to insert value in hex format
Examples:
set APN value (string option) to “test.apn”

```
sensor legacy_set_apn 28:2C:02:4F:FF:90 1111 test.apn
```

set APN value (hex option) to “test.apn”

```
sensor legacy_set_apn 28:2C:02:4F:FF:90 1111 --hex 746573742E61706E
```

set APN value to “auto”

```
sensor legacy_set_apn 28:2C:02:4F:FF:90 1111 -a
```

ET0-146 - Get APN

There are two commands used to get the APN set on the device:

- For sensor's firmware >=6.11.0 use command "**get_apn**"
- For sensor's firmware <6.11.0 use command "**legacy_get_apn**"

1) **get_apn (sensor's fw 6.11.0 or newer):**

Gets the APN, user name and password values.

```
sensor get_apn <MAC_ADDRESS> <PIN>
```

Responses:
Example 1: APN, user name and password are filled:

```
sensor get_apn 28:2C:02:4F:FF:11 1111
apn: "test.apn"
userName: "user"
password: "qwerty"
```

Example 2: APN, user name are filled, password is configured as automatic:

```
sensor get_apn 28:2C:02:4F:FF:11 1111
apn: "test.apn"
userName: "user"
password: ""
```

Example 3: APN is filled, user name and password are configured as automatic:

```
sensor get_apn 28:2C:02:4F:FF:11 1111
apn: "test.apn"
userName: ""
```

password: ""

Example 4: APN, user name and password are configured as automatic:

```
sensor get_apn 28:2C:02:4F:FF:11 1111
apn: ""
userName: ""
password: ""
```

1) **legacy_get_apn (sensor's fw <6.11.0):**

Gets the APN value.

```
sensor legacy_get_apn <MAC_ADDRESS> <PIN>
```

ET0-147 - Set PLMN

Sets the value of PLMN. To set the value to “auto” use the optional “-a” parameter

```
sensor set_plmn <MAC_ADDRESS> <PIN> <PLMN>
```

Optional parametrers

-a - used to set PLMN value to "auto"

Examples

set PLMN value to 12345

```
sensor set_plmn 28:2C:02:4F:FF:90 1111 12345
```

set PLMN value to "auto"

```
sensor set_plmn 28:2C:02:4F:FF:90 1111 -a
```

ET0-148 - Get PLMN

Returns the value of PLMN

sensor get_plmn <MAC_ADDRESS> <PIN>

ET0-149 - Trigger the communication with server

Triggers the communication with server over cellular network

sensor trigger_communication_with_server <MAC_ADDRESS> <PIN>

ET0-150 - Set server configuration

Sets data and update servers configuration

Data server:

sensor set_server_configuraiton <MAC_ADDRESS> <PIN> 0 <SERVER_IP_ADDRESS> <PORT>

Update server:

sensor set_server_configuraiton <MAC_ADDRESS> <PIN> 1 <SERVER_IP_ADDRESS> <UDP_PORT> <COAP_PORT>

Examples

Set the data server

sensor set_server_configuraiton 28:2C:02:4F:FF:90 1111 0 127.0.0.1 5683

Set the update server

sensor set_server_configuraiton 28:2C:02:4F:FF:90 1111 1 127.0.0.1 6000 5683

ET0-151 - Get server configuration

Returns the configuration of data and update servers

Data server:

sensor get_server_configuraiton <MAC_ADDRESS> <PIN> 0

Update server:

sensor get_server_configuraiton <MAC_ADDRESS> <PIN> 1

ET0-152 - Set cloud token

Sets the cloud token value (can be used to disable cloud token, set cloud token to user specified value or set cellular module's IMEI as cloud token)

Cloud token value

sensor set_cloud_token <MAC_ADDRESS> <PIN> 1 <TOKEN_VALUE>

Set IMEI as cloud token

sensor set_cloud_token <MAC_ADDRESS> <PIN> 2 0

Do not set cloud token

sensor set_cloud_token <MAC_ADDRESS> <PIN> 255 0

ET0-153 - Get cloud token

Returns the value and type of the cloud token

sensor get_cloud_token <MAC_ADDRESS> <PIN>

ET0-154 - Set transmission interval

Sets sensor's transmission and ACK interval. To set ACK interval to always set its value to "0xFFFFFFFF" sensor set_transmission_config <MAC_ADDRESS> <PIN> <TRANSMISSION_INTERVAL> <ACK_INTERVAL>

ET0-155 - Get transmission interval

Returns sensor's transmission and ACK intervals

sensor get_transmission_config <MAC_ADDRESS> <PIN>

ET0-156 - Set transfer limit

Sets the transfer limit and transfer limit timer

sensor set_transfer_limit <MAC_ADDRESS> <PIN> <TRANSFER_LIMIT> <TRANSFER_LIMIT_TIMER>

ET0-157 - Get transfer limit

Returns the transfer limit and transfer limit timer

sensor get_transfer_limit <MAC_ADDRESS> <PIN>

ET0-158 - Set supervision period

Sets supervision period. Supervision period value should be in seconds. To disable set to "0xFFFFFFFF"

sensor set_supervision_period <MAC_ADDRESS> <PIN> <SUPERVISION_PERIOD>

ET0-159 - Get supervision period

Returns supervision period

sensor get_supervision_period <MAC_ADDRESS> <PIN>

ET0-162 - Set rule

Creates a new rule configuration / updates a rule with selected rule ID.

```
sensor set_rule <MAC_ADDRESS> <PIN> <RULE_ID> <CHANNEL_MASK> <CONDITION>
<ACTION> [<PARAM1> <PARAM2> <PARAM3> <PARAM4> <PARAM5>]
```

- RULE_ID - Id of the rule. Up to 12 rules can be defined. Range [0:11]
- CHANNEL_MASK - Channels to which the rule is assigned. One rule can be assigned to multiple channels as long as those are of the same type. Bit mask on bits [0:5]. E.g. To assign the rule for channel 1: "000001", to assign rule to channels 2 and 4: "001010"
- CONDITION - Condition to be checked by the device. If the condition is true, an action is triggered. Possible values:
    - 0 - Unspecified
    - 1 - Disabled
    - 2 - High Threshold
    - 3 - Low threshold
    - 4 - Differential threshold
    - 5 - Change of state (Binary sensors only)
    - 6 - Logic operator
    - 7 - On measurement
- ACTION - Action to be triggered. Currently the only possible action is to trigger the transmission. Other actions will be available in next SW releases. Possible values
    - 0 - Unspecified
    - 1 - Trigger the transmission
    - 2 - No action
    - 3 - Trigger the transmission with ACK
- PARAM1..5 - parameters of CONDITION. Available params depend on the type of condition

CONDITION: High Threshold

Available for continuous sensors only. If the measurement (or average from a few measurements) is over the threshold, an action is triggered.

- PARAM1 - Threshold value. Must match channel type
- PARAM2 - Hysteresis value. Must much channel type. Set to "0" to disable
- PARAM3 - Average calculation mode:
    - 1 - moving average (a1=(n1+n2+n3)/3, a2=(n2+n3+n4)/3, etc.)
    - 2 - window mode (a1=(n1+n2+n3)/3, a2=(n4+n5+n6)/3, etc.)
    - 3 - consecutive samples mode
- PARAM4 - Number of measurements for average value calculating. E.g PARAM4 equals 3, average value from three samples will be calculated and compared to the threshold value.
- PARAM5 - Type of measurement (as described in MeasurementType).

CONDITION: Low Threshold

Available for continuous sensors only. If the measurement (or average from a few measurements) is below the threshold, an action is triggered.

- PARAM1 - Threshold value. Must match channel type
- PARAM2 - Hysteresis value. Must much channel type. Set to "0" to disable
- PARAM3 - Average calculation mode:
    - 1 - moving average (a1=(n1+n2+n3)/3, a2=(n2+n3+n4)/3, etc.)
    - 2 - window mode (a1=(n1+n2+n3)/3, a2=(n4+n5+n6)/3, etc.)
    - 3 - consecutive samples mode
- PARAM4 - Number of measurements for average value calculating. E.g PARAM4 equals 3, average value from three samples will be calculated and compared to the threshold value.
- PARAM5 - Type of measurement (as described in MeasurementType).

CONDITION: Differential Threshold

Continuous sensors only. If the absolute value of the difference between the last value sent to the server and the measurement value (or average from a few measurements) is greater or equal to the value of the threshold set, an action is triggered.

- PARAM1 - Threshold value. Must match channel type
- PARAM2 - Average calculation mode:
    - 1 - moving average (a1=(n1+n2+n3)/3, a2=(n2+n3+n4)/3, etc.)
    - 2 - window mode (a1=(n1+n2+n3)/3, a2=(n4+n5+n6)/3, etc.)
    - 3 - consecutive samples mode
- PARAM3 - Number of measurements for average value calculating. E.g PARAM3 equals 3, average value from three samples will be calculated and compared to the threshold value.
- PARAM4 - Type of measurement (as described in MeasurementType).

CONDITION: Logic operator

Used for combining multiple rules into more complex conditions. If the logic condition specified by parameters (logic operator and selected rules) is met, an action is triggered.

- PARAM1 - Logic operator: EVERY selected rule must be active or AT LEAST ONE of the selected rules must be active
- PARAM2 - Rule selector. Bitmask of rule IDs that will be used while determining the result.
- PARAM3 - Rule negation. Bitmask of rule IDs that will be used while determining the result. State of those rules will be used negated.
- PARAM4 - Rule action delay [s]. Specifies time delay between the rule activation and the rule action being triggered.
- PARAM5 - Rule return delay [s]. Specifies time delay between the rule deactivation and the rule action being triggered

CONDITION: On measurement

The basic function is to trigger communication after measurement if at least 60s have passed since the last one. Transmission may occur every x measurement. Optionally dependency on the other rule can be configured, then, when all conditions are met, transmission is triggered.

- PARAM1 - Send every x measurement. This parameter specifies every which measurement transmission will be triggered if all other conditions are met. If parameter[0] equals 1, transmission will occur after every measurement.
- PARAM2 - Optional. Rule selector. Bitmask of rule IDs that will be used while determining the result.
- PARAM3 - Optional. Rule negation. Bitmask of rule IDs that will be used while determining the result. State of those rules will be used negated.

Examples

Set a rule, which will trigger the transmission, if the temperature measured on channel 1 is over 10 C. Rule will be assigned to Rule_ID = 0

sensor set_rule 28:2C:02:4F:FF:90 1111 0 000001 2 1 100 0 1 1 1
Set a rule, which will trigger the transmission, if the humidity measured on channel 2 or 3 is below 40%. Rule will be assigned to Rule_ID = 1

sensor set_rule 28:2C:02:4F:FF:90 1111 1 000110 3 1 40 0 1 1 2

ET0-163 - Get rule

Returns a rule configuration for the selected rule ID.

```
sensor get_rule <MAC_ADDRESS> <PIN> <RULE_ID>
```

ET0-180 - Collect memory statistics

Triggers a restart of collecting memory statistics.

```
sensor collect_memory_statistics <MAC_ADDRESS> <PIN>
```

ET0-181 - Get memory statistics

Returns a memory statistics.

```
sensor get_memory_statistics <MAC_ADDRESS> <PIN>
```

ET0-182 - Get runtime errors

Returns a table of runtime errors, and can clear it.

```
sensor get_runtime_errors <MAC_ADDRESS> <PIN>
```

Optional parameters:

-r - removes all errors from the table

-d - decode runtime errors to human readable information

Examples

```
sensor get_runtime_errors 28:2C:02:4F:FF:90 1111
sensor get_runtime_errors 28:2C:02:4F:FF:90 1111 -d
sensor get_runtime_errors 28:2C:02:4F:FF:90 1111 -r -d
```

ET0-184 - Set CoAP endpoints

Sets chosen CoAP endpoint (ASCII string).

Data endpoint:

sensor set_coap_endpoints <MAC_ADDRESS> <PIN> 0 <ENDPOINT>

Configuration endpoint:

sensor set_coap_endpoints <MAC_ADDRESS> <PIN> 1 <ENDPOINT>

Device info endpoint:

sensor set_coap_endpoints <MAC_ADDRESS> <PIN> 2 <ENDPOINT>

Time endpoint:

sensor set_coap_endpoints <MAC_ADDRESS> <PIN> 3 <ENDPOINT>

All occurrences of '#' symbol in <ENDPOINT> field will be parsed into '\x1B' (ESC) character.

ET0-185 - Get CoAP endpoints

Returns chosen CoAP endpoint.

Data endpoint:

sensor get_coap_endpoints <MAC_ADDRESS> <PIN> 0

Configuration endpoint:

sensor get_coap_endpoints <MAC_ADDRESS> <PIN> 1

Device info endpoint:

sensor get_coap_endpoints <MAC_ADDRESS> <PIN> 2

Time endpoint:

sensor get_coap_endpoints <MAC_ADDRESS> <PIN> 3

ET0-195 - Ping notification event

Command used to create a ping notification event. The result of the request will be transmitted on Notify characteristic.

```
sensor ping_notification_event <MAC_ADDRESS> <PING_NUMBER>
```

Example

```
sensor ping_notification_event 28:2C:02:4F:FF:90 1
```

ET0-207 - Set modem AT log state

Sets the modem AT log state:

```
sensor set_modem_at_log_state <MAC_ADDRESS> <PIN> <STATE>
```

Enable log:

```
sensor set_modem_at_log_state <MAC_ADDRESS> <PIN> 1
```

Disable log:

```
sensor set_modem_at_log_state <MAC_ADDRESS> <PIN> 0
```

ET0-208 - Get modem AT log state

Gets the modem AT log state:

```
sensor get_modem_at_log_state <MAC_ADDRESS> <PIN>
```

ET0-211 - Set DNS server

Sets the DNS server

```
sensor set_dns <MAC_ADDRESS> <PIN> <IP_V4>
```

Custom DNS server:

```
sensor set_dns <MAC_ADDRESS> <PIN> 8.8.8.8
```

Use network DNS server:

sensor set_dns <MAC_ADDRESS> <PIN> 255.255.255.255

ET0-212 - Get DNS server

Gets the DNS server:

sensor get_dns <MAC_ADDRESS> <PIN>

ET0-213 - Set DNS configuration

Sets the DNS configuration.

sensor set_dns_configuration <MAC_ADDRESS> <PIN> <TTL>

Optional parameters

-d - accept TTL from the DNS server
-cf - DNS request is only after communication failed

Examples

set the custom TTL to 86000 seconds:

sensor set_dns_configuration 28:2C:02:4F:FF:90 1111 86000

set the DNS TTL:

sensor set_dns_configuration 28:2C:02:4F:FF:90 1111 -d
set the DNS request on communication failed:

sensor set_dns_configuration 28:2C:02:4F:FF:90 1111 -cf

ET0-214 - Get DNS configuration

Returns the configuration of DNS:

```
sensor get_dns_configuration <MAC_ADDRESS> <PIN>
```

ET0-224 - Get rule calendar

Returns a calendar configuration for the selected calendar ID.

```
sensor get_rule_calendar <MAC_ADDRESS> <PIN> <CALENDAR_ID>
```

ET0-225 - Set rule calendar

Creates a new calendar configuration / updates an existing calendar with selected ID.

```
sensor set_rule_calendar <MAC_ADDRESS> <PIN> <CALENDAR_ID> <TYPE>
<RULES_MASK> [<PARAM1> <PARAM2> <PARAM3> <PARAM4>]
```

- CALENDAR_ID- Id of the calendar (up to 6 calendars possible). Range [0:5]
- TYPE- Calendar type:
  - 0 - Unspecified
  - 1 - Disabled
  - 2 - Week
- RULES_MASK- Bitmask of rules that should be driven by currently set calendar.
- PARAM1-4 - Parameters of TYPE. Available parameters depend on the type

TYPE: Week

- PARAM1 - Day mask. Bitmask of week days (sunday at index 0)
- PARAM2 - 'From time'. Timestamp in minutes from midnight (UTC) specifiyng the starting point from where a rule is enabled.
- PARAM3 - 'To time'. Timestamp in minutes from midnight (UTC) specifiyng the ending point where a rule is disabled.
- PARAM4 - Timezone. In 15 minutes offsets (see ES6-580)

Examples

Set a WEEK calendar with ID = 0, which will enable only rule with ID = 0. It will be enabled on Mondays from 1:00 AM (UTC+2) to 2:00 AM (UTC+2).

sensor set_rule_calendar 28:2C:02:4F:FF:FA 1111 0 1 1 2 60 120 8

ET0-230 - Get last update status

Returns a timestamp and a result of the last update

sensor get_last_update_status <MAC_ADDRESS> <PIN>

ET0-231 - Set slot output

Sets the output pin of the given slot.

```
sensor set_slot_output <MAC_ADDRESS> <PIN> <SLOT_NUMBER> <PIN_STATE>
```

- SLOT_NUMBER- Slot must be of IO Control type. Range [1:6]
- PIN_STATE:
    - 0 - Low
    - 1 - High

ET0-233 - Get NVM statistics

ET0-248 - Get time synchronization period

Returns time synchronization period in seconds.

```
sensor get_time_synchronization_period <MAC_ADDRESS> <PIN>
```

ET0-249 - Set time synchronization period

Sets the time synchronization period of the sensor's internal clock. `SYNC_PERIOD` time should be given in seconds and should be in range from 14 400 to 2 592 000 seconds.

```
sensor set_time_synchronization_period <MAC_ADDRESS> <PIN> <SYNC_PERIOD>
```

ET0-252 - Set Bluetooth connection parameters

Sets sensor’s low power Bluetooth connection's parameters.

sensor set_bluetooth_conn_params <address> <min_conn_interval> <max_conn_interval> <slave_latency> <conn_sup_timeout>

Default connection parameters and conversion details:

Parameter	Default raw value	Multiplier	Converted default value	Unit
min_conn_interval	7	1.25 [ms]	8.75	milliseconds

max_conn_interval	120	1.25 [ms]	150	milliseconds
slave_latency	0	1	0	connection events
conn_sup_timeout	40	10 [ms]	400	milliseconds

ET0-253 - Connect low power

Connects to the sensor with a low power type of Bluetooth connection. Once the connection is established, all the other commands supported by the dongle can be issued

```
sensor connect_low_power <MAC_ADDRESS>
```

To disconnect, issue command

```
sensor disconnect
```

See ET0-252 for details and information about this kind of connection's parameters.

ET0-258 - Send any command

Implements a possibility to send any command with parameters. Note, that the payload data has to be right-padded to the length of 16 bytes.

Usage scheme:

```
sensor send_any_command <address> <cmd_id> [[<pin>] <payload>]
```

Usage examples:

- get_public_key (0x23):

```
sensor send_any_command 28:2C:02:40:28:3C 0x23
```

- set_coap_endpoint (0xAC) - Set data endpoint to \x1Bs:

```
sensor send_any_command 28:2C:02:40:28:3C 0xAC 3925
001b730000000000000000000000000000
```

(!) In this usage case the endpoint indicator adds an additional 1 byte with the value of 0x00 to the beginning of the payload so its summed length is 17 bytes, not 16!

ET0-263 - Get time since event

Gets passed time since the given event.

Usage scheme:

```
sensor get_time_since_event <MAC_ADDRESS> <PIN> <EVENT>
```

- EVENT:
  - 0 - Time synchronization
  - 1 - Last communication
  - 2 - Last configuration

ET0-266 - Gateway hello

sensor gateway_hello <MAC_ADDRESS> <API_VERSION> <CONFIG_HASH>

ET0-267 - Gateway goodbye

sensor gateway_goodbye <MAC_ADDRESS> <STATUS>

ET0-268 - Socket attach

sensor socket_attach <MAC_ADDRESS> <SOCKET>

ET0-269 - Socket detach

sensor socket_detach <MAC_ADDRESS> <SOCKET>

ET0-270 - Socket send to

sensor socket_send_to <MAC_ADDRESS> <SOCKET> <RRC_RELEASE> <DATA>

ET0-271 - Fetch time

sensor fetch_time <MAC_ADDRESS>

ET0-272 - Set led configuration

Sets led configuration.

```
sensor set_led_configuration <MAC_ADDRESS> <PIN> <GREEN_PERIOD> <RED_PERIOD>
<COMM_ERR_OFF_TIME> <SENS_ERR_OFF_TIME> <LOW_PWR_OFF_TIME>
<NEW_MEAS_OFF_TIME> <TRANSMISSION_OFF_TIME> <SENS_OK_OFF_TIME> <BLINK_TIME>
```

- GREEN_PERIOD- period of flashing green led. Multiple of 5 seconds (from 1 to 120).
- RED_PERIOD- period of flashing red led. Multiple of 5 seconds (from 1 to 120).
- COMM_ERR_OFF_TIME - time in minutes after transition to normal state, after which red led won't blink if there is communication error. (from 0 to 240 or 255 for always on)
- SENS_ERR_OFF_TIME- time in minutes after transition to normal state, after which red led won't blink if there is sensor error. (from 0 to 240 or 255 for always on)
- LOW_PWR_OFF_TIME - time in minutes after transition to normal state, after which red led won't blink if battery is low. (from 0 to 240 or 255 for always on)
- NEW_MEAS_OFF_TIME - time in minutes after transition to normal state, after which green led won't blink on new measurement. (from 0 to 240 or 255 for always on)
- TRANSMISSION_OFF_TIME - time in minutes after transition to normal state, after which green led won't blink on transmission. (from 0 to 240 or 255 for always on)
- SENS_OK_OFF_TIME - time in minutes after transition to normal state, after which green led won't blink if sensor is ok. (from 0 to 240 or 255 for always on)
- BLINK_TIME - duration of single blink. Multiple of 5 ms (from 4 to 200).

Example

Set following configuration:

Period of flashing green led = 5s.

Period of flashing red led = 10s.

Flashing on communication error, sensor error, low power turned off.

Flashing on new measurement, transmission always turned on.

Flashing if sensor ok for 5 minutes from transition to normal mode.

Single blink duration = 100ms.

sensor set_led_configuration 28:2C:02:4F:FF:90 1111 1 2 0 0 0 255 255 5 20

ET0-273 - Get led configuration

Returns led configuration

```
sensor get_led_configuration <MAC_ADDRESS> <PIN>
```

ET0-279 - Set network troubleshooting

Sets network troubleshooting functionality.

```
sensor set_network_troubleshooting <MAC_ADDRESS> <PIN> <STATUS>
```

STATUS - Possible values:

- 1 - Disabled
- 2 - Enabled

ET0-280 - Get network troubleshooting

Returns network troubleshooting functionality status.

```
sensor get_network_troubleshooting <MAC_ADDRESS> <PIN>
```

Possible values at return:

- networkTroubleshooting: "Deactivated" - Network troubleshooting is disabled
- networkTroubleshooting: "Activated" - Network troubleshooting is enabled
- networkTroubleshooting: "Unknown" - Network troubleshooting status is unrecognized

ET0-283 - Set cellular configuration parameters

Sets cellular configuration parameters.

```
sensor set_cellular_configuration_parameters <MAC_ADDRESS> <PIN>
<MODEM_MODEL> [<PARAM1> <PARAM2> <PARAM3> <PARAM4> <PARAM5> <PARAM6>
<PARAM7> <PARAM8> <PARAM9> <PARAM10> <PARAM11>]
```

- MODEM_MODEL - Modem of cellular modem:
    - 1 - BC66
    - 2 - BC66NA
- PARAM1..11 - configuration parameters of cellular modem. The number of parameters can be from 1 to 11. The number and value of parameters depends on the modem type.

MODEM_MODEL: BC66 or BC66NA

- PARAM1 - EPCO:
    - 1 - Disable EPCO
    - 2 - Enable EPCO
- PARAM2 - COMBINEDATTACH:
    - 1 - Disable combined attach
    - 2 - Enable combined attach
- PARAM3 - UP:
    - 1 - Disable user plane function
    - 2 - Enable user plane function
- PARAM4 - AUTOPDN:
    - 1 - Disable PDN auto activation
    - 2 - Enable PDN auto activation
- PARAM5 - RIPIN:
    - 1 - Default output level is high for the RI pin
    - 2 - Default output level is low for the RI pin
- PARAM6 - INIT_LOCK_TIME:
    - Range: [2:31]. Configure the initial Sleep Lock duration. Unit: second with static offset 1. P5=2 -> 1 sec ... P5=31 -> 30 sec.
- PARAM7 - DSEVENT:
    - 1 - Disable the URC for the deep sleep event
    - 2 - Enable the URC for the deep sleep event.
- PARAM8 - AT_LOCK_TIME:
    - Range: [1:11]. Configure the Sleep Lock duration by AT commands. Unit: second with static offset 1. P7=1 -> 0 sec ... P7=11 -> 10 sec.

Examples

For BC66 modem model

```
sensor set_cellular_configuration_parameters 28:2C:02:4F:FF:90 1111 1 1 1 1 1 1 20 1 10
```

ET0-284 - Get cellular configuration parameters

Returns modem model and configuration parameters, the number of parameters can be from 1 to 11. The number and value of parameters depends on the modem type.

```
sensor get_cellular_configuration_parameters <MAC_ADDRESS> <PIN>
```

List of parameters for BC66 / BC66NA

- configurationParam1 - EPCO:
  - 1 - Disable EPCO
  - 2 - Enable EPCO
- configurationParam2 - COMBINEDATTACH:
  - 1 - Disable combined attach
  - 2 - Enable combined attach
- configurationParam3 - UP:
  - 1 - Disable user plane function
  - 2 - Enable user plane function
- configurationParam4 - AUTOPDN:
  - 1 - Disable PDN auto activation
  - 2 - Enable PDN auto activation
- configurationParam5 - RIPIN:
  - 1 - Default output level is high for the RI pin
  - 2 - Default output level is low for the RI pin
- configurationParam6 - INIT_LOCK_TIME:
  - Range: [2:31]. Configure the initial Sleep Lock duration. Unit: second with static offset 1. P5=2 -> 1 sec ... P5=31 -> 30 sec.
- configurationParam7 - DSEVENT:
  - 1 - Disable the URC for the deep sleep event
  - 2 - Enable the URC for the deep sleep event.
- configurationParam8 - AT_LOCK_TIME:
  - Range: [1:11]. Configure the Sleep Lock duration by AT commands. Unit: second with static offset 1. P7=1 -> 0 sec ... P7=11 -> 10 sec.

ET0-286 - Set modem log state

Sets the modem log state:

sensor set_modem_log_state <MAC_ADDRESS> <PIN> <STATE> [<INTERFACE_ID>]

- STATE - Modem logging state:
  - 0 - Disable request
  - 1 - Enable request
- INTERFACE_ID - Interface identification number (required only when STATE = Enable request):
  - 0 - USB Interface (BC66 and BC66NA)

sensor set_modem_log_state <MAC_ADDRESS> <PIN> 1 0
Disable log:

sensor set_modem_log_state <MAC_ADDRESS> <PIN> 0

ET0-287 - Get modem log state

Gets the modem log state:

sensor get_modem_log_state <MAC_ADDRESS> <PIN>
Returns:

- 0 - Disabled - Modem logging is disabled
- 1 - Enabled - Modem logging is enabled
- 2 - Incomplete - Modem logging state is unknown, the disable or enable request was not successful