CloudGate Probe

The information contained in this document is subject to change with or without notice. For this reason, always make sure you are using the latest version available at support.option.com. If you have any additional questions, please contact your point of contact.

OPTION - Geldenaaksebaan 329- 3001 Leuven - Belgium
T +32 16 317 411 – F +32 16 207 164 - www.option.com

1. Table of Contents

2. Revision history

Version	Description	Editor	Date
1.0	First release	J. Mariën	2024-04
1.1	- Added M-bus and Modbus limitations in the encode section - Modbus register numbering - Uplink & downlink relations	J. Mariën	2024-06
1.2	Added wiring/installation section	J.Mariën	2024-10

3. General overview

The CloudGate Probe is a compact DIN rail mountable metering device. This permits to monitor assets and energy in buildings remotely.

As a low cost and compact device, the CloudGate Probe can read and transmit data of a limited number of devices and meters to an IoT platform. The device can also change a limited number of settings of a HVAC device (e.g. control valve,…).

The CloudGate Probe has a limited CPU capability and is not capable of EDGE computing. It is a compact device that is intended to be used in limited use cases or as a slave device of a powerful gateway such as the CloudGate IoT Gateway.

CloudGate Probe is available in two variants: a Cellular (LTE-CATM, NB-IoT) version and a LoRaWAN version. Both devices can support many use cases, but the LoRaWAN version is intended to be used in buildings (i.e. smart building and building management platforms). The cellular variant is intended to be used in asset monitoring (i.e. remote pumps, generators, etc.).

4. Wiring/installation information

1. Power supply

- There exist 2 variants: AC and DC powered
- Indicated by the sticker on the device which variant you have.

2. RS485 interface

- Internal 120 ohm termination resistor – non configurable.
- Half-duplex

3. Digital output

- 0 = Digital out mode: OD
  The output pin behaves like a switch connected to ground. It can pull the line low (0V) but requires an external pull-up resistor to pull the line high. Suitable for interfacing with devices requiring open-drain signalling.

- 2 = Digital out mode: PP
  The output pin actively drives the signal high or low, sourcing or sinking current. This mode provides higher current capabilities and is used for directly driving devices like LEDs or relays.

5. Deployment

With every device, or batch of devices, a csv file will be provided with the keys needed to communicate with the device. If you did not receive such a file, please contact your Option point of contact.

Each device has a unique and fixed DevEUI and a fixed crypto-random AppKey. All devices have the same JoinEUI. The IEEE_OPTION_OUI is used as a prefix for the DevEUI and JoinEUI. During production the sensor generates the DevEUI and AppKey. These keys are not readable from any debug port.

6. Configuring the device using the CLI tool

Before usage, the CloudGate Probe needs to be configured. You can configure the device locally, using a rs485 dongle, or over the air through downlink commands.

Take note that initial configuration using OTA might not be possible since some basic connection parameters will be needed. For Lora this is e.g. the region.

In order to configure the device locally, you will need:

- A tool on your PC to talk Modbus. The easiest option is to use the tool provided by us.
- A rs485 dongle (Half Duplex) which can be connected to a PC. If you have a full duplex dongle, you can connect TX+ to RX+ and TX- to RX-, however this will cause an extra echo on the wire. The Option tool can handle this but you will need to explicitly set this option (-e 1)
- The CloudGate Probe to be in config mode. Since the CloudGate Probe doesn't have a dedicated config port, the Modbus interface is used. When the device is booted it will enter 'config mode' for 2 seconds. After sending a command using the tool, the device will stay in config mode for the next 10 minutes.

To configure the device using the tool provided by Option, follow these steps:

1. Connect the device to your PC using a RS485 dongle.
2. Ask the tool to export the config: ‘din2tool -d [DEVICE PORT] -x config.json’, where [DEVICE PORT] is the port to open e.g. 'COM6' or '/dev/ttyUSB1'
   You should see “waiting for device”.
   Note: if you are using a full duplex dongle wired as half duplex, you need to add '-e 1' to the command line.
3. Boot the CloudGate Probe.
   If all went well you should see "saved config to config.json".
4. Open the config.json file and change the desired settings.
5. Import the config again: ‘din2tool -d [DEVICE PORT] -i config.json’
6. Power or reset the device.
   Note: If you have a variant with the "last gasp" functionality you can reset the device using the reset button.

OPTION

7. API basics

4. General structure

- API Messages consists of TLV (Type Length Value) items
- Type & Length are 1 byte each
- Encoding/decoding of all data is little endian
- One message can contain multiple TLV items
- A TLV item will always be entirely present in a message (no segmentation). Because of this, a minimum lora packet size of 50 bytes is assumed.
- All configured measurements are sent with a frequency determined by the main interval.

5. Basic configuration flow

For more detailed information about the commands, see section 7: “How to encode downlink data”.

Normal configuration through downlink commands follows the following flow:

- set basic configuration:
  - Set basic configuration (downlink type = 7)
- set Modbus configuration:
  - Clear Modbus configuration (downlink type = 15)
  - Set Modbus interface (downlink type = 8)
  - Configure Modbus measurements (x times) (downlink type = 9)
    Limits:
    - Maximum 16 Modbus requests can be stored in the device config.
    - Maximum 32 data bytes can be stored per request.
- set M-bus configuration:
  - Clear m-bus configuration (downlink type = 14)
  - Configure m-bus measurements (x times) (downlink type = 13)
    Limits:
    - Maximum 4 M-bus loads
    - Maximum 8 frames will be read from a single meter
    - Maximum 6 M-bus requests can be stored in the device config.
    - Maximum 14 record filters can be set per request.
    - Maximum 20 user data can be reported per filtered record.
    - Maximum 50 filtered records can be reported (in total)
- save settings & config:
  - Save config (downlink type = 10)
  - Reboot (downlink type = 1)

note: In most cases you would like to know if your config command reached the device or not.
For confirmation, you can append your config command with a ping command. (downlink type = 16)

After the initial configuration, the configured measurements will be periodically collected and transmitted. Occasionally, you might want to trigger these configured measurements manually (without waiting for the period to elapse). For this you can use the collect measurements command. (downlink type = 2)

6. Set output pins

Prior to using the commands to set a specific output, make sure the pins are configured properly since these commands will only set the state. They will not change the pin configuration.

7. M-bus workflow

You can configure the CloudGate Probe to periodically send the full M-bus payload, but since most of time only a few fields are needed you can also apply some filters.
The configuration of the M-bus (filters) is a little less trivial so it deserves some attention. The first thing you should get your hands on is the raw M-bus frame(s) the meter is sending. If you already have this you can skip some of the following steps.

Use the "Request M-bus scan" command (downlink type = 11) to start a scan. For each meter detected you will receive an uplink (uplink type = 7), which holds the encoded M-bus address.

Finally when the scan is concluded you will receive another uplink (uplink type = 6) indicating the end off the scan which also contains how many meters have been found in total.

Once you have the (encoded) address, you can do a manual request to get the frame for that meter. (downlink type = 12)

Once you have the raw payload you have to decode this. You can do this by using https://dev-lab.github.io/tmbus/tmbus.htm. When using the "raw" view (do not use post-process), all records will be showed in order. Now you need to determine which fields you are interested in and note the record number.

Note: the above site shows record INDEX (starting at 0) while the CloudGate Probe filters are using the record NUMBER (starting at 1).

8. How to decode uplink data

Please note!Encoding and decoding of all data is little endian.

1. Analog input

- port: 1
- Data:
  - uint8_t Type = 0
  - uint8_t Length = 2
  - Value:
    - Uint16_t voltage (mV)
- Example:
  00 02 D2 04 [4 bytes]:
  - analog input voltage: 1234mV

2. Digital in counter

- port: 1
- Data:
  - uint8_t Type = 1
  - uint8_t Length = 4
  - Value:
    - Uint32_t counter
- Example:
  01 04 BC 02 00 00 [6 bytes]:
  - Digital input counter: 700

3. Modbus value

- port: 1
- Data:
  - uint8_t Type = 2
  - uint8_t Length
  - Value:
    - uint8_t id
    - uint24_t register number
    - uint8_t data[0]
      The length of this data is the TLV length - 4 (id and register number length)

- Example:
  02 08 01 81 1A 06 32 00 04 00 [10 bytes]:
  - id: 1
  - register number: 400001
  - raw data: 32 00 04 00

4. Power state

- port: 1
- Data:
    - uint8_t Type = 3
    - uint8_t Length = 1
    - Value:
        - uint8_t powered
          1 = powered
          0 = not powered
- Example:
  03 01 00 [3 bytes]:
    - The device is no longer powered

5. Shadow states

- port: 1
- Data:
    - uint8_t Type = 4
    - uint8_t Length
    - Value:
        - uint8_t digital out value: 0 or 1
        - uint16_t 10V analog out value in mV
        - uint16_t 20ma analog out value in µA

- Example:
  04 05 01 D2 04 10 27 [7 bytes]:
    - Digital out: 1
    - 10V analog out: 1234mV
    - 20mA analog out: 1000 µA

6. M-bus record

- port: 1
- Data:
  - uint8_t Type = 5
  - uint8_t Length
  - Value:
    - uint8_t short report id
    - uint8_t record number
    - uint8_t data[0]
- Example:
  05 06 01 01 27 04 85 02 [8 bytes]:
  - Report id = 1
  - Record number = 1
  - Data = 27 04 85 02

  05 05 01 02 27 01 00 [7 bytes]:
  - Report id = 1
  - Record number = 2
  - Data = 27 01 00

7. M-bus scan end

- port: 1
- Data:
  - uint8_t Type = 6
  - uint8_t Length = 3
  - Value:
    - uint16_t baud rate on which the scan was executed
    - uint8_t the total number of meters found
- Example:
  06 03 80 25 01 [5 bytes]:
  - Meters found: 1
  - Baud rate: 9600

8. M-bus scan meter found

- port: 1
- Data:
  - uint8_t Type = 7
  - uint8_t Length = 8
  - Value:
    - uint8_t address[8]
- Example:
  07 08 39 72 00 72 2D 2C 34 0D [10 bytes]

9. Fragment raw M-bus frame

m-bus frames may be too large to send in one lora packet, therefor they are fragmented.

- port: 1
- Data:
  - uint8_t Type = 8
  - uint8_t Length
  - Value:
    - uint8_t counter
    - uint8_t last + frame number:
      - last : the MSB of this byte.
        If this is 1, this is the last data frame
    - uint8_t last + fragment number:
      - last : the MSB of this byte.
        If this is 1, this is the last fragment
    - uint8_t data[0]
- Example:
  - **08 31 02 00 00 68 2A 2A 68 18 01 72 34 12 00 00 43 04 01 01 00 00 00 00 0C 13 27 04 85 02 0B 3B 27 01 00 4C 13 19 54 44 01 42 6C FF 0C 02 FD 17 00 00 1F** [51 bytes]:
    - Counter: 2
    - Last fragment: False
    - Fragment number: 0
    - Last frame: False
    - Frame number: 0
    - Data: 68 2A 2A 68 18 01 72 34 12 00 00 43 04 01 01 00 00 00 00 0C 13 27 04 85 02 0B 3B 27 01 00 4C 13 19 54 44 01 42 6C FF 0C 02 FD 17 00 00 1F
  - **08 05 02 00 81 58 16** [7 bytes]:
    - Counter: 2
    - Last fragment: True
    - Fragment number: 1
    - Last frame: False
    - Frame number: 0
    - Data: 58 16

- **08 31 02 81 00 68 29 29 68 08 01 72 34 12 00 00 43 04 01 01 00 00 00 00 0C 13 27 04 85 02 0B 3B 27 01 00 4C 13 19 54 44 01 42 6C FF 0C 02 FD 17 00 00 29** [51 bytes]:
  - Counter: 2
  - Last fragment: False
  - Fragment number: 0
  - Last frame: True
  - Frame number: 1
  - Data: 68 29 29 68 08 01 72 34 12 00 00 43 04 01 01 00 00 00 00 0C 13 27 04 85 02 0B 3B 27 01 00 4C 13 19 54 44 01 42 6C FF 0C 02 FD 17 00 00 29

- **08 04 02 81 81 16** [6 bytes]:
  - Counter: 2
  - Last fragment: True
  - Fragment number: 1
  - Last frame: True
  - Frame number: 1
  - Data: 16

10. Ping response

- port: 1
- Data:
    - uint8_t Type = 9
    - uint8_t Length = 2
    - Value
        - uint16_t ping id
          the ping ID given in the request
- Example:
  09 02 01 00 [4 bytes]:
    - Ping id: 1

11. Firmware version

- port: 1
- Data:
    - uint8_t Type = 10
    - uint8_t Length = 4
    - Value:
        - uint8_t major
        - uint8_t minor
        - uint16_t revision
- Example:
  0A 04 01 00 00 00 [6 bytes]:
    - Firmware version: 1.0.0

9. How to encode downlink data

Please note!Encoding and decoding of all data is little endian.

1. Reboot

- port: 2
- Data:
    - uint8_t type = 1
    - uint8_t length = 0
- Example:
  01 00 [2 bytes]:
  Send a reboot command to the device.

2. Collect measurements

Triggers an uplink of all configured measurements.

- port: 2
- Data:
    - uint8_t type = 2
    - uint8_t length = 0

- Example:
  02 00 [2 bytes]:
  Execute the configured measurements.

3. Blinky

This command makes the LED's blink and is used only for testing purposes.

- port: 2
- Data:
    - uint8_t type = 3
    - uint8_t length = 0

- Example:
  03 00 [2 bytes]:
  Make one of the LED's blink.

4. Set digital out

- port: 2
- Data:
    - uint8_t type = 4
    - uint8_t length = 1
    - uint8_t value:
      0 = Digital out mode: OD_HIGH
      1 = Digital out mode: OD_LOW
      2 = Digital out mode: PP_HIGH
      3 = Digital out mode: PP_LOW
- Example:
  04 01 01 [3 bytes]:
  Set the digital out to low (open drain)

5. Set analog out 10V pin

- port: 2
- Data:
    - uint8_t type = 5
    - uint8_t length = 2
    - uint16_t value (in mV)
- Example:
  05 02 D2 04 [4 bytes]:
  Set the analog out 10V pin to 1234mV

6. Set analog out 20mA pin

- port: 2
- Data:
    - uint8_t type = 6
    - uint8_t length = 2
    - uint16_t value (in µA)
- Example:
  06 02 D2 04 [4 bytes]:
  Set the analog out 20mA pin to 1234µA

7. Set basic configuration

- port: 2
- Data:
  - uint8_t type = 7
  - uint8_t length = 12
  - Value:
    - Uint16_t main interval (minutes)
    - Uint16_t analog out 10V (mV)
    - Uint16_t analog out 20mA (µA)
    - Uint16_t digital in debounce time (ms)
    - uint8_t analog in mode:
      - 0 = disabled
      - 1 = 10V
      - 2 = 3V
      - 3 = CL
    - uint8_t digital in mode:
      - 0 = DISABLED
      - 1 = S0_CNTR
      - 2 = GPIO_CNTR
    - uint8_t digital out mode & boot value:
      - 0 = OD_HIGH
      - 1 = OD_LOW
      - 2 = PP_HIGH
      - 3 = PP_LOW
    - uint8_t report output pins state in the periodic report:
      - 0 = false
      - 1 = true
- Example:
  07 0C 78 00 14 05 00 00 64 00 01 02 00 01 [14 bytes]:
  - Main interval: 120 minutes
  - Analog out 10V pin boot value: 1300mV
  - Analog out 20ma pin boot value: 0 µA
  - Digital in debounce time: 100ms
  - Analog in mode: 10V
  - Digital in mode: GPIO_CNTR
  - Digital out mode: OD_HIGH
  - Report output pins state: True

8. Modbus interface configuration

- port: 2
- Data:
  - uint8_t type = 8
  - uint8_t length = 9
  - Value:
    - uint32_t baud rate
    - uint16_t timeout (ms)
    - uint8_t mode
      - 0 = RTU
      - 1 = ascii
    - uint8_t parity
      - 0 = none
      - 1 = odd
      - 2 = even
    - uint8_t stop bits
      - 0 = 0.5
      - 1 = 1
      - 2 = 1.5
      - 3 = 2
- Example:
  08 09 00 4B 00 00 D0 07 00 00 01 [11 bytes]:
  - baud rate: 19200
  - timeout: 2000 ms
  - mode: RTU
  - parity: NONE
  - stop bits = 1

9. Modbus measurements configuration

- Device limitations:
  - Maximum 16 Modbus requests can be stored in the device config.
  - Maximum 32 data bytes can be stored per request.

- Modbus register numbering:
  Different type of registers (holding, coils, ...) are mapped to one range.

  notation : X00.000 with X being a prefix
  - Prefix coils: 0
  - Prefix discrete inputs: 1
  - Prefix input registers: 3
  - Prefix holding registers: 4

  Note: we use register numbers and not register addresses so the first register is 1 (and not 0)

- port: 2
- Data:
  - uint8_t type = 9
  - uint8_t Length = 6
  - Value:
    - uint8_t index
    - uint8_t Modbus slave id
    - uint24_t register start
    - uint8_t number of registers/coils to read
- Example:
  09 06 00 0D 81 1A 06 03 09 06 01 1B 05 00 00 0A [14 bytes]:
  2 requests/configs:
  - request 1:
    - index: 0 (store requests starting from index 0)
    - Modbus slave id: 13
    - register start number: 400001
    - number of (registers/coils) to read: 3

  - request 2:
    - index: 1 (second request in the configuration list)
    - Modbus slave id: 27
    - register start number: 5 (fifth coil)
    - number of (registers/coils) to read: 10

10. Save configuration

- port: 2
- Data:
  - uint8_t type = 10
  - uint8_t Length = 0
- Example:
  0A 00 [2 bytes]:
  save the configuration.

11. Request M-bus scan

- port: 2
- Data:
  - uint8_t type = 11
  - uint8_t Length = 2
  - Value:
    - uint16_t baud rate
- Example:
  0B 02 80 25 [4 bytes]:
  - baud rate: 9600

12. Request raw M-bus frames

- port: 2
- Data:
  - uint8_t type = 12
  - uint8_t Length = 3
  - Value:
    - uint16_t baud rate
    - uint8_t address[8]
- Example:
  0C 0A 80 25 34 12 00 00 43 04 01 01 [12 bytes]:
  - Baud rate: 9600
  - (unencoded) address: 34 12 00 00 43 04 01 01

13. M-bus measurements configuration

- Device limits:
  - Maximum 4 M-bus loads
  - Maximum 8 frames will be read from a single meter
  - Maximum 6 M-bus requests can be stored in the device config.
  - Maximum 14 record filters can be set per request.
  - Maximum 20 user data can be reported per filtered record.
  - Maximum 50 filtered records can be reported (in total)

- port: 2
- Data:
  - uint8_t type = 13
  - uint8_t length
  - Value:
    - uint8_t table index (starting from 0) where this request will be stored
    - uint8_t short report ID, will be used in the report uplink
    - uint8_t report mode
      - 0 = user data (filtered per record)
      - 1 = full raw M-bus telegrams
    - uint16_t M-bus baud rate
    - uint8_t slave address (encoded as per M-bus spec)
    - uint16_t filter[0] array of items of interest
- Example:
  0D 11 00 01 00 80 25 34 12 00 00 43 04 01 01 01 00 02 00 [19 bytes]:
  - Index: 0
  - Short report ID: 1
  - Report mode: user data
  - Baud rate: 9600
  - address: 34 12 00 00 43 04 01 01
  - filter: fields 1 and 2

14. Clear M-bus configurations

- port: 2
- Data:
  - uint8_t type = 14
  - uint8_t length = 0
- Example:
  0E 00 [2 bytes]

15. Clear Modbus configurations

- port: 2
- Data:
  - uint8_t type = 15
  - uint8_t length = 0
- Example:
  0F 00 [2 bytes]

16. Request ping

Triggers a ping response with the same ping_id.

- port: 2
- Data:
  - uint8_t type = 16
  - uint8_t length = 2
  - Value:
    - uint16_t ping id
- Example:
  10 02 01 00 [4 bytes]