NextPM User Guide

Last update : 2021/02/10

Author : A. DUMAS

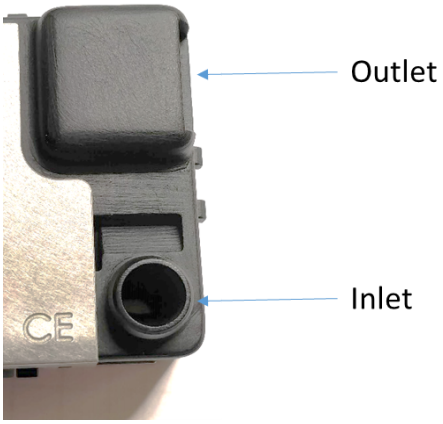
Version : 3.5

SUMMARY

1. NextPM integration advices

To be fully functional, the NextPM must be integrated in order to be in a vertical way with the inlet down and the outlet up.

You can set the NextPM sensor in another device thanks to the two holes in the back of the sensor. The holes are made to receive a self turret screw M2.5 and are 5 mm deep.



The sensor uses an active airflow to sample the particles. In order to keep his accuracy, we advise you to put the inlet directly to the air you want to monitor. If the integration is difficult, you can add a duct made with antistatic materials and with an internal diameter of 8 to 9 mm. The maximum length of the duct must be 100 cm.

Moreover, the inlet and the outlet need to be at a similar pressure and not obstructed.

The connector is a 6 PIN one. You can use an ADAM TECH 125CH-B-06 reference to connect with or an equivalent like a MOLEX 51021-0600 reference.

2. NextPM communication's protocols

2.1. Simplified serial port communication

2.1.1. Configuration

PIN connector assignment

1. GND
2. +5V
3. Tx (output)
4. Rx (input)
5. CS
6. GND

The serial communication is available thanks to PIN 3 and 4.

Serial port configuration

The serial port must be configured following these parameters :

- Speed : 115200 bauds,
- 8 bits,
- 1bit parity : even,
- 1 bit stop.

The NextPM reply to a request in more than 350 ms.

The NextPM must be power supplied with + 5 VDC.

The signal amplitude is +3.3V.

Warning 1 : to connect NextPM to a PC, prior, you must use a FTDI cable. One example of FTDI that could be used is : TTL-232R-3V3

Warning 2 : do not connect NextPM sensor directly on the RS232 port of a PC.

2.1.2. Send a command

The NextPM information's reading can be realized through the command frame: Address Command Checksum

To write information, use the command frame: Address Command DATA Checksum

The address is freezed at 0x81.

Below, the command possibilities:

Cmd_Id	Description	Example
0x11	Concentrations reading's averaged over 10 seconds and updated every 1 second	0x81 0x11 0x6E
0x12	Concentrations reading's averaged over 60 seconds and updated every 10 seconds	0x81 0x12 0x6D
0x13	Concentrations reading's averaged over 900 seconds and updated every 60 seconds	0x81 0x13 0x6C
0x14	Temperature and humidity readings	0x81 0x14 0x6B
0x15	Power on or sleep mode	0x81 0x15 0x6A (**note 1**)
0x16	Sensor state's readings	0x81 0x16 0x69
0x17	Firmware version readings	0x81 0x17 0x68
0x22	Modbus address read Modbus address *3* write (**note 2**)	0x81 0x22 0x00 0x5D 0x81 0x22 0x00 0x5A

Note 1: Each 0x15 command frame sent, the NextPM changes its functional state alternately. To know its state before sending the command frame, you can send a 0x16 command frame: if the 0 bit of the code state is 1, the NextPM is in sleep mode.

Note 2: The 0x22 command is used to read the Modbus Address (the third byte must be 0) or to write the new Modbus Address (the third byte is the address, 3 in this example).

The checksum is calculated in order that the sum of all the frame bytes is equal to a multiple of 256 (0x100).

Example :

0x81 + 0x16 + 0x69 = 0x100

0x81 + 0x21 + 0x55 + 0x09 = 0x100

Thus:

Checksum = 0x100 – MOD ((sum of the other bytes), 256).

2.1.3. NextPM responses

The NextPM reply to a command frame by a frame that always begins by its address (0x81) followed by the command frame asked and ending by a checksum.

The data sent are function of the command frame:

Command frame 0x11, 0x12 and 0x13

address	Cmd id	State (1 byte)	PM1 pcs/L (2 bytes)	PM2.5 pcs/L (2 bytes)	PM10 pcs/L (2 bytes)	PM1 µg/m3 (2 bytes)	PM2.5 µg/m3 (2 bytes)	PM10 µg/m3 (2 bytes)	Checksum
0x81	0x11	0x00	0x022B	0x06F4	0x06F4	0x0A82	0x1FC6	0x1FC6	0xF7
0x81	0x12	0x00	0x022B	0x06F4	0x06F4	0x0A82	0x1FC6	0x1FC6	0xF6
0x81	0x13	0x00	0x022B	0x06F4	0x06F4	0x0A82	0x1FC6	0x1FC6	0xF5
0x81	0x16	0x04							0x65

The particulate matter concentration in pcs/L and µg/m^3 are coded with 2 bytes (16 bits).

Example :

The response 0x81 0x12 0x00 0x32 0xE7 0x32 0xF5 0x32 0xF8 0x00 0x6A 0x00 0x72 0x00 0x85 0xA2

Signifies that the results are averaged over 1 minute. No error occurred during the measurement and the state code is 0.

In this example, the measured concentrations are:

	2 bytes data	Factor	Results
PM1 pcs/L	0x32E7	1	13031
PM 2.5 pcs/L	0x32F5	1	13045
PM10 pcs/L	0x32F8	1	13048
PM1 µg/m^3	0x006A	0.1	10.6
PM2.5 µg/m^3	0x0072	0.1	11.4
PM10 µg/m^3	0x0085	0.1	13.3

The concentrations are calculated from the 2 bytes read data and multiplied by a factor.

The state code must always be read, it highlights the functional state of the NextPM and allows to know the validity of the sent values.

The checksum can be checked, the sum of all the frame bytes is equal to a multiple of 256 (0x100), here:

0x81 + 0x12 + 0x00 + 0x32 + 0xE7 + 0x32 + 0xF5 + 0x32 + 0xF8 + 0x00 + 0x6A + 0x00 + 0x72 + 0x00 + 0x85 + 0xA2 = 0x600 is a multiple of 0x100, le checksum is OK.

State code

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Laser Error	Memory Error	Fan Error	T/RH Error	Heat Error	Not Ready	Degraded State	Sleep State

The bit 0 is set to 1 when the sensor is set to sleep state: the laser, the fan and the heat are switched off. The only command frame still possible is 0x16 (Read the NextPM State), the NextPM will respond to any other command frame as if it is the 0x16 command frame.

The bit 1 is set to 1 each time a minor error is detected, the sensor part in error is set to 1 in the state code, the NextPM can still send data but with less accuracy.

The minor errors are the following:

- Heat Error, the relative humidity stay above 60% during more than 10 minutes,
- T/RH Error, the sensor reading are out of specification,
- Fan Error, the fan speed is out of range but the fan is still working.
- Memory Error, the sensor can't access its memory, some internal smart functions will not be available.

The sensor will be set to Default State if the fan or the laser are broken. If the sensor is in the Default State, the degraded state flag is set to 0, the default part is indicated by its flag set to 1 and the bit 0 indicates that the sensor is in sleep state.

If the sensor replies by a 0x16 command frame, it means that the NextPM has no data to send neither because the sensor has just been switched on nor because the sensor is in the Default State or Sleep State.

Command frame 0x14

address	Command id	State (1 byte)	Temperature (2 bytes)	Humidity (2 bytes)	Checksum
0x81	0x14	0x00	0x0B40	0x13E7	0x26
0x81	0x16	0x76			0xF3

The temperature and relative humidity are sent with 2 bytes, you need to divide by 100 the obtained value in order to have the real value. Note that the temperature and relative humidity are not the environmental ones but the ones within the sensor, they could only be used for a debug diagnosis.

Example :

The NextPM replies 0x0B40 0x13E7, thus 2880 for a 28.80 °C temperature and 0x13E7 and 5095 for a 50.95% relative humidity.

If you want the real ambient Temperature and Relative Humidity, you need to deactivate the heating function first then apply a corrective coefficient to the data read with the NextPM sensor.

For the temperature in °C, you need to apply the following function:

$$y = 0.9754\ x_1 - 4.2488$$ (where x_1 represent the NextPM temperature raw data)

For the Relative Humidity in %, you need to apply the following function:

$$y = 1.1768\ x_2 - 4.727$$ (where x_2 represent the NextPM relative Humidity raw data)

If the sensor replies by a 0x16 command frame, it means that the NextPM has no data to send, neither because the NextPM has just been switched on, nor because the NextPM is in the T/RH Error State or Sleep State.

Command frame 0x15

address	Command id	State (1 byte)	Checksum
0x81	0x15	0x01	0x69
0x81	0x16	0x01	0x68

When the command frame 0x15 is sent to set the sensor in sleep mode, the NextPM replies with a state code with the flag SLEEP set to 1 and stops to work.

During the SLEEP mode, if the sensor receives a new 0x15 command frame, then the NextPM will be switched on and will send the first PM datas after 15 seconds.

During the SLEEP mode, the NextPM replies to all the other command frames by sending a 0x16 command frame with the state code.

If the NextPM is in the Default State, the 0x15 command frame allows you to try to switch on the NextPM.

Command frame 0x16

address	Command id	State (1 byte)	Checksum
0x81	0x16	0x33	0x36

The 0x16 command frame sends the State code. It's the default response when there is no data to transmit or when the asked command frame can't be done (for example, when the sensor is in SLEEP mode).

Command frame 0x17

address	Command id	State (1 byte)	Firmware Version (2 bytes)	Checksum
0x81	0x17	0x00	0x0034	0x34
0x81	0x16	0x01		0x68

The NextPM will send the firmware version. In the above example, the firmware version is 0x0034. If the NextPM is in the SLEEP mode, it will only reply its state code.

Command frame 0x21

address	Command id	State (1 byte)	Fan Speed (1 byte)	Checksum
0x81	0x21	0x00	0x42	0x1C
0x81	0x16	0x01		0x68

When a 0x21 command frame is sent, the sensor replies using the actual value.

The minimum fan speed is set to 30%. Below this value, the new speed value is not saved and the sensor will use its previous speed value. Note that if you send a NULL value, the NextPM will send you its last memory value.

If the command frame is sent when the sensor is in SLEEP mode, then the NextPM will reply with the 0x16 command frame to indicate that the command will not be used.

In the above example, the fan speed is set to 66% (0x42).

2.2. Modbus Protocol

2.2.1. Configuration

PIN connector assignment

1. GND
2. +5V
3. Tx (output)
4. Rx (input)
5. CS
6. GND

The serial communication is available thanks to PIN 3 and 4.

The communication is a Modbus RTU (binary) :

- Speed : 115200 bauds,
- 8 bits,
- 1bit de parity, even,
- 1 bit stop.

The NextPM responds to a request in more than 350 ms.

The NextPM is powered by + 5VDC.

The signals of the communication have an amplitude of + 3.3V.

The Modbus protocol complies with V1.1b specifications.

The frame for reading registers in the NextPM is in the following format :

A0 C1 R1 R2 N1 N2 V1 V2

Example :

01 03 00 01 00 01 D5 CA (in hexadecimal) reads the firmware version

- The A0 address of the NextPM module is 0x01.
- C1 is the read command: 0x03
- The address of the start register is coded on 2 bytes R1R2 (R1 most significant) 0x00 0x01.
- The number of registers is coded on 2 bytes N1N2 (N1 most significant) 0x00 0x01.
- The frame ends with two bytes of checksums V1V2 0xD5 0xCA:

Polynomial : 0xA001 // Polynomial = 2^15 + 2^13 + 2^0 = 0xA001
INIT CRC : 0xFFFF

Example of checksum calculation :

```
Private Function CRC16(buf() As Byte, lbuf As Integer) As Integer

        ' returns the MODBUS CRC of the lbuf first bytes of "buf" buffer (buf is a global array of bytes)

        Dim Crc As Integer
        Dim mask As Integer
        Dim i As Integer
        Dim j As Integer


        Crc = &HFFFF ' init CRC
        For i = 0 To lbuf - 1 Step 1 ' for each byte
        Crc = Crc Xor buf(i)

        For j = 0 To 7 Step 1 ' for each bit

        mask = 0
        If Crc / 2 <> Int(Crc / 2) Then mask = &HA001
        Crc = Int(Crc / 2) And &H7FFF: Crc = Crc Xor mask

            Next j
        Next i

        CRC16 = Crc

End Function
```

V1 = CRC16 AND 0xFF
V2 = INT (CRC16 / 256) AND 255

The NextPM must answer :

A1 C1 N1 R1 … R1 V1 V2
01 03 02 00 42 38 75 (in hexadecimal)

- A1 and the address of the NextPM : 0x01
- C1 is the callback of the read command: 0x03
- N1 is the number of octets transmitted, coded on a byte
- R1 … Rn corresponds to the data (bytes) transmitted, in the example, 0x00 0x42, firmware 4.2
- V1 and V2 is the checksum calculated as before.

Another example :

We want to read 10 16-bit registers from register 1 :

01 03 00 01 00 0A 94 0D

The NextPM must answer :

01 03 14 00 42 00 83 00 01 33 3C 00 00 D9 F0 00 00 01 43 00 04 00 00 29 5A

It sends 2 * 10 bytes (0x14).

Beware, depending on the configuration of the sensor, the registers described below are not always accessible.

2.2.2. Management Modbus functions

The following registers are only readable and accessible with a read or read/write multiple holding registers function command (0x03 or 0x17)

A register is coded on 16 bits.

List of decimal's registers :

Register	Name	description
1	FIRMWARE_VERSION	Embedded software version

The operating status or error code is displayed in register 19.

19	PM_STATUS	Error code or status of NextPM

PM_STATUS consists of 9 bits :

Bit 7 R	Bit 6 R	Bit 5 R	Bit 4 R	Bit 3 R	Bit 2 R	Bit 1 R	Bit 0 R/W
Laser error	Memory error	Fan error	T°C/Hum Error	Heater Error	Not Ready	Degraded Mode	Sleep Mode

Bit 0 is set to 1 when the sensor is in standby : the laser, the fan, the heating are then deactivated.
Bit 1 is set to 1 when a minor error is detected, the status of the erroneous element goes to 1 in the status code, the sensor can still send data.

Minor errors are the followings :

- Memory error, the memory is no longer accessible, there is no more correction of the aging of the sensor, the configuration parameters are those originally programmed at the factory during configuration
- Heating error, the humidity level stays above 60% for more than 10 minutes
- Temperature and humidity measurements error, their reading is not in the specifications
- Fan error, its speed is not in the specified operating range but the fan is not blocked

There is no degraded mode in the event of a laser fault or in the event of a fan breakdown. In both cases, the sensor switches to fault mode after 3 start attempts, bit 9 goes to 1. In fault mode, the

degraded mode flag is at 0, the faulty element is indicated by its flag at 1 and bit 0 indicates that the sensor is at a standstill.

2.2.3. Particulate matter data available

10 seconds average

50-51	GetTenSecondesAverageResult.Nb_1_0_L	average over 10s of particles number concentration / liter of size < 1µm
52-53	GetTenSecondesAverageResult.Nb_2_5_L	average over 10s of particles number concentration / liter of size < 2.5µm
54-55	GetTenSecondesAverageResult.Nb_10_L	average over 10s of particles number concentration / liter of size < 10µm
56-57	GetTenSecondesAverageResult.Mass_1_0_L	average over 10s of the mass concentration in µg / m3 of particles' size < 1µm
58-59	GetTenSecondesAverageResult.Mass_2_5_L	average over 10s of the mass concentration in µg / m3 of particles' size < 2.5µm
60-61	GetTenSecondesAverageResult.Mass_10_L	average over 10s of the mass concentration in µg / m3 of particles' size < 10µm

60 seconds average

62-63	GetOneMinuteAverageResult.Nb_1_0_L	average over 60s of particles number concentration / liter of size < 1µm
64-65	GetOneMinuteAverageResult.Nb_2_5_L	average over 60s of particles number concentration / liter of size < 2.5µm
66-67	GetOneMinuteAverageResult.Nb_10_L	average over 60s of particles number concentration / liter of size < 10µm
68-69	GetOneMinuteAverageResult.Mass_1_0_L	average over 60s of the mass concentration in µg / m3 of particles' size < 1µm
70-71	GetOneMinuteAverageResult.Mass_2_5_L	average over 60s of the mass concentration in µg / m3 of particles' size < 2.5µm
72-73	GetOneMinuteAverageResult.Mass_10_L	average over 60s of the mass concentration in µg / m3 of particles' size < 10µm

15 minutes average

74-75	GetFifteenMinuteAverageResult.Nb_1_0_L	average over 15 min of particles number concentration / liter of size < 1µm
76-77	GetFifteenMinuteAverageResult.Nb_2_5_L	average over 15 min of particles number concentration / liter of size < 2.5µm
78-79	GetFifteenMinuteAverageResult.Nb_10_L	average over 15 min of particles number concentration / liter of size < 10µm
80-81	GetFifteenMinuteAverageResult.Mass_1_0_L	average over 15 min of the mass concentration in µg / m3 of particles' size < 1µm
82-83	GetFifteenMinuteAverageResult.Mass_2_5_L	average over 15 min of the mass concentration in µg / m3 of particles' size < 2.5µm
84-85	GetFifteenMinuteAverageResult.Mass_10_L	average over 15 min of the mass concentration in µg / m3 of particles' size < 10µm

The averages are coded on 32 bits, thus on 2 registers. The actual value of the average is obtained by dividing by 1000 the value read in the two registers (see the following example).

Example :

Interrogation of all concentrations :

01 03 00 32 00 24 E4 1E

Sensor's reply :

01 03 **48 62 4F** 00 25 62 4F 00 25 62 4F 00 25 00 EC 00 00 00 EC 00 00 00 EC 00 00 6A 5D 00 13 99 6F 00 14 57 22 00 15 00 5E 00 00 01 82 00 00 03 A8 00 00 00 ED 00 17 CA FA 00 17 FE 29 00 17 00 A7 00 00 01 C8 00 00 02 69 00 00 77 09

Calculation of the 10s average of the particles number concentration / liter of size < 1µm :

This is the first 32-bit coded data :

- 0x624F (Less significant bit), 0x0025 (Most significant bit),
- Result : 0x0025624F,
- 2449999 in decimal,
- Thus, the final result of PM1 in pcs/L is 2449.999 for the average of 10s.

01 03 48 62 4F 00 25 62 4F 00 **25 62 4F** 00 25 00 EC 00 00 00 EC 00 00 00 EC 00 00 6A 5D 00 13 99 6F 00 14 57 22 00 15 00 5E 00 00 01 82 00 00 03 A8 00 00 00 ED 00 17 CA FA 00 17 FE 29 00 17 00 A7 00 00 01 C8 00 00 02 69 00 00 77 09

Calculation of the 10s average of the mass concentration of particles' size < 1µm :

This is the fourth 32-bit coded data :

- 0x00EC (Less significant bit), 0x0000 (Most significant bit),
- Result 0x000000EC,
- 236 in decimal,
- Thus the final result of PM1 in µg/m3 is 0.236 for the average of 10s.

2.2.4. Advanced Modbus Functions

The following registers are writable and accessible with a write or read/write multiple holding registers function command (0x10 or 0x17)

A register is coded on 16 bits.

88	ModBusAddress	NextPM with Modbus protocol

Currently, the NextPM address can be set in the range 1 to 15.

100	FanTarget	Cyclic ratio for fan rotation setpoint (expressed in %)

The setpoint can vary from 0.00 to 1.00 (where 0.00 is 0% and 1.00 is 100%), it is stored in the register multiplied by 10000
but we strongly recommend to not touch it.

101	HeaterPWM	Cyclic ratio for heating (expressed in %)

The duty cycle varies from 0.00 to 1.00 (where 0.00 is 0% and 1.00 is 100%), it is stored in the register multiplied by 10000.
It will strongly increase the current consumption when turned on.

2.2.5. Special Modbus Functions

The following register is only readable and accessible with a read or read/write multiple holding registers function command (0x03 or 0x17)

The register 102 returns the frequency of rotation of the fan if it is available.

102	FanSpeed	Frequency expressed in Hertz

The registers 106 and 107 allow you to know the humidity and the temperature, the values are stored in a register (of 16 bits), it is the real value multiplied by 100.

106	Humidity	Humidity in % multiplied by 100
107	Temperature	Temperature in °C multiplied by 100

Warning 3 : these data are only technical data.

If you want the real ambient Temperature and Relative Humidity, you need to deactivate the heating function first then apply a corrective coefficient to the data read with the NextPM sensor.

For the temperature in °C, you need to apply the following function :

$$y = 0.9754\ x_1 - 4.2488$$ (where x_1 represent the NextPM temperature raw data)

For the Relative Humidity in %, you need to apply the following function :

$$y = 1.1768\ x_2 - 4.727$$ (where x_2 represent the NextPM relative Humidity raw data)

3. Sample's test software

If you want to test the sensor before working on the integration, you could find two softwares on our website in order to communicate with the NextPM using a computer running with Windows 10.