| Attribute description | |
|---|---|
| **logical_name** | Identifies the “GSM Diagnostic” object instance. For logical names, see 6.2.23. |
| **operator** | Holds the name of the network operator e.g. “YourNetOp” |
| **status** | Indicates the registration status of the modem.<br>enum:<br>(0) not registered,<br>(1) registered, home network,<br>(2) not registered, but MT is currently searching a new operator to register to,<br>(3) registration denied,<br>(4) unknown,<br>(5) registered, roaming,<br>(6) ... (255) reserved |
| **cs_attachment** | Indicates the current circuit switched status.<br>enum:<br>(0) inactive,<br>(1) incoming call,<br>(2) active,<br>(3) ... (255) reserved |
| **ps_status** | The *ps_status* value field indicates the packet switched status of the modem.<br>enum:<br>(0) inactive,<br>(1) GPRS,<br>(2) EDGE,<br>(3) UMTS,<br>(4) HSDPA,<br>(5) LTE,<br>(6) CDMA,<br>(7) ... (255) reserved |
| **cell_info** | Represents the cell information:<br>cell_info_type::= structure<br>{<br>cell_ID: double-long-unsigned,<br>location_ID: long-unsigned,<br>signal_quality: unsigned,<br>ber: unsigned,<br>mcc: long-unsigned,<br>mnc: long-unsigned,<br>channel_number: double-long-unsigned<br>}<br>Where:<br>– cell_ID: Four-byte cell ID in hexadecimal format;<br>– location_ID: Two-byte location area code (LAC) in the case of GSM networks or Tracking Area Code (TAC) in the case of UMTS, CDMA or LTE networks in hexadecimal format (e.g. "00C3" equals 195 in decimal); |

| | |
|---|---|
| **cell_info** (continued) | – signal_quality: Represents the signal quality:<br>(0) –113 dBm or less,<br>(1) –111 dBm,<br>(2...30) –109...-53 dBm,<br>(31) –51 or greater,<br>(99) not known or not detectable;<br>– ber: Bit Error Rate (BER) measurement in percent:<br>(0…7) as RXQUAL_n values specified in ETSI GSM 05.08:1996, 8.2.4<br>(99) not known or not detectable.<br>– mcc: Mobile Country Code of the serving network, as defined in ITU-T E.212 (05.2008);<br>– mnc: Mobile Network Code of the serving network, as defined in ITU-T E.212 (05.2008);<br>– channel_number: Represents the absolute radio-frequency channel number (ARFCN or eaRFCN for LTE network). |
| **adjacent_cells** | array adjacent_cell_info<br><br>adjacent_cell_info::= structure<br>{<br>cell_ID: double-long-unsigned,<br>signal_quality: unsigned<br>}<br>Where:<br>– cell_ID: Four-byte cell ID in hexadecimal format;<br>– signal_quality: Represents the signal quality:<br>(0) –113 dBm or less,<br>(1) –111 dBm,<br>(2...30) –109...-53 dBm,<br>(31) –51 or greater,<br>(99) not known or not detectable. |
| **capture_time** | Holds the date and time when the data have been last captured.<br>*date-time* is formatted as specified in 4.6.1. |

### 5.6.9 LTE monitoring (class_id: 151, version: 0)

Instances of the ‘LTE monitoring” IC allow monitoring LTE modems by handling all data necessary data for this purpose.

| **LTE monitoring** | | **0...n** | **class_id = 151, version = 0** | | | |
|---|---|---|---|---|---|---|
| ***Attributes*** | | ***Data type*** | ***Min.*** | ***Max.*** | ***Def.*** | **Short name** |
| 1. logical_name | (static) | octet-string | | | | x |
| 2. lte_quality_of_service | (dyn.) | LTE_qos_type | | | | x + 0x08 |
| ***Specific methods*** | | ***m/o*** | | | | |

| **Attribute description** | |
|---|---|
| **logical_name** | Identifies the “LTE monitoring” object instance. See 6.2.23. |

| | |
|---|---|
| **lte_quality_of_service** | Represents the quality of service for the LTE network<br>LTE_qos_type::= structure<br>{<br>T3402: long-unsigned,<br>T3412: long-unsigned,<br>RSRQ: unsigned,<br>RSRP: unsigned,<br>qRxlevMin: integer<br>}<br>Where:<br>– T3402: timer in seconds, used on PLMN selection procedure and sent by the network to the modem. Refer to 3GPP TS 24.301 V13.4.0 (2016-01) for details;<br>– T3412: timer in seconds used to manage the periodic tracking area updating procedure and sent by the network to the modem. Refer to 3GPP TS 24.301 V13.4.0 (2016-01) for details;<br>– RSRQ: represents the signal quality as defined in 3GPP TS 24.301 V13.4.0 (2016-01):<br>(0) –19,5dB,<br>(1) –19 dB,<br>(2...31) –18,5…-3,5 dB,<br>(32) –3dB,<br>(99) Not known or not detectable;<br>– RSRP: represents the signal level as defined in 3GPP TS 24.301 V13.4.0 (2016-01):<br>(0) –140dBm,<br>(1) –139 dBm,<br>(2... 94) –138…-45 dBm,<br>(95) –44dBm,<br>(99) Not known or not detectable;<br>– qRxlevMin: specifies the minimum required Rx level in the cell in dBm as defined in 3GPP TS 24.301 V13.4.0 (2016-01). |

## 5.7 Interface classes for setting up data exchange via M-Bus

### 5.7.1 Overview

The M-Bus related interface classes specified in this subclause 5.7 are used in two different scenarios:

a) a DLMS/COSEM server hosted by a M-Bus master and exchanging dedicated M-Bus APDUs with M-Bus slaves;

b) a DLMS/COSEM client hosted by a M-Bus master and exchanging DLMS/COSEM APDUs with DLMS/COSEM servers hosted by M-Bus slaves;

In case a) instances of the following M-Bus interface classes are used to set up and manage the M-Bus media in the DLMS/COSEM server:

- M-Bus client (class_id = 72), see 5.7.3;
- M-Bus master port setup (class_id = 74), see 5.7.5;
- M-Bus diagnostic (class_id = 77, version = 0), see 5.7.7.

In case b) instances of the following M-Bus interface classes are used in the DLMS/COSEM server:

- DLMS/COSEM server M-Bus port setup (class_id = 76), see 5.7.6;
- M-Bus slave port setup (class_id = 25), see 5.7.2; and/or
- Wireless Mode Q channel (class_id = 73), see 5.7.4;
- M-Bus diagnostic (class_id = 77, version = 0), see 5.7.7.

### 5.7.2 M-Bus slave port setup (class_id = 25, version = 0)

NOTE 1 The name of this IC has been changed from “M-BUS port setup” to “M-Bus slave port setup”, to indicate that it serves to set up data exchange when a COSEM server communicates with a COSEM client using wired M-Bus.

This IC allows modelling and configuring communication channels according to EN 13757-2:2004. Several communication channels can be configured.

| **M-Bus slave port setup** | | **0...n** | **class_id = 25, version = 0** | | | |
|---|---|---|---|---|---|---|
| ***Attributes*** | | ***Data type*** | ***Min.*** | ***Max.*** | ***Def.*** | **Short name** |
| 1. logical_name | (static) | octet-string | | | | x |
| 2. default_baud | (static) | enum | 0 | 5 | 0 | x + 0x08 |
| 3. avail_baud | (static) | enum | 0 | 7 | | x + 0x10 |
| 4. addr_state | (static) | enum | | | | x + 0x18 |
| 5. bus_address | (static) | unsigned | | | | x + 0x20 |
| ***Specific methods*** | | ***m/o*** | | | | |

| **Attribute description** | |
|---|---|
| **logical_name** | Identifies the “M-Bus slave port setup” object instance. See 6.2.22. |
| **default_baud** | Defines the baud rate for the opening sequence.<br>enum: (0) 300 baud,<br>(3) 2 400 baud,<br>(5) 9 600 baud |
| **avail_baud** | Defines the baud rates that can be negotiated after startup.<br>enum: (0) 300 baud,<br>(1) 600 baud,<br>(2) 1 200 baud,<br>(3) 2 400 baud,<br>(4) 4 800 baud,<br>(5) 9 600 baud,<br>(6) 19 200 baud,<br>(7) 38 400 baud |
| **addr_state** | Defines whether or not the device has been assigned an address since last power up of the device.<br>enum:<br>(0) Not assigned an address yet,<br>(1) Assigned an address either by manual setting, or by automated method. |
| **bus_address** | The currently assigned address on the bus for the device.<br>NOTE 2 If no bus address is assigned, the value is 0. |

### 5.7.3 M-Bus client (class_id = 72, version = 1)

Instances of the “M-Bus client” allow setting up M-Bus slave devices using wired M-Bus and to exchange data with them. Each “M-Bus client” object controls one M-Bus slave device. For details on the M-Bus dedicated application layer, see EN 13757-3:2013.

NOTE 1 Version 1 of the “M-Bus client” IC is in line with EN 13757-3:2013.

The M-Bus client device may have one or more physical M-Bus interfaces, which can be configured using instances of the “M-Bus master port setup” IC, see 5.7.5.

An M-Bus slave device is identified with its Primary Address, Identification Number, Manufacturer ID etc. as defined in EN 13757-3:2013, Clause 5, Variable Data Send and Variable Data respond. These parameters are carried by the respective attributes of the M-Bus client IC.

Values to be captured from an M-Bus slave device are identified by the *capture_definition* attribute, containing a list of data identifiers (DIB, VIB) for the M-Bus slave device. The data are captured periodically or on an appropriate trigger. Each data element is stored in an M-Bus value object, of IC “Extended register”. M-Bus value objects may be captured in M-Bus “Profile generic” objects, eventually along with other, non M-Bus specific objects.

Using the methods of “M-Bus client” objects, M-Bus slave devices can be installed and de-installed.

It is also possible to send data to M-Bus slave devices and to perform operations like resetting alarms, synchronizing the clock, transferring an encryption key, etc.

Configuration field as defined in EN 13757-3:2013, 5.12 provides information about the encryption mode and number of encrypted bytes.

Encryption key status provides information if encryption key has been set, transferred to M-Bus slave device and is in use with M-Bus slave device.

| **M-Bus client** | | **0...n** | **class_id = 72, version = 1** | | | |
|---|---|---|---|---|---|---|
| ***Attributes*** | | ***Data type*** | ***Min.*** | ***Max.*** | ***Def.*** | ***Short name*** |
| 1. logical_name | (static) | octet-string | | | | x |
| 2. mbus_port_reference | (static) | octet-string | | | | x + 0x08 |
| 3. capture_definition | (static) | array | | | empty | x + 0x10 |
| 4. capture_period | (static) | double-long-unsigned | | | 0 | x + 0x18 |
| 5. primary_address | | unsigned | | | 0 | x + 0x20 |
| 6. identification_number | (dyn.) | double-long-unsigned | | | 0 | x + 0x28 |
| 7. manufacturer_id | (dyn.) | long-unsigned | | | 0 | x + 0x30 |
| 8. version | (dyn.) | unsigned | | | 0 | x + 0x38 |
| 9. device_type | (dyn.) | unsigned | | | 0 | x + 0x40 |
| 10. access_number | (dyn.) | unsigned | | | 0 | x + 0x48 |
| 11. status | (dyn.) | unsigned | | | 0 | x + 0x50 |
| 12. alarm | (dyn.) | unsigned | | | 0 | x + 0x58 |
| 13. configuration | (dyn.) | long-unsigned | | | 0 | x + 0x60 |
| 14. encryption_key_status | (dyn.) | enum | | | 0 | x + 0x68 |

| *Specific methods* | *m/o* | | |
|---|---|---|---|
| 1. slave_install (data) | o | | x + 0x70 |
| 2. slave_deinstall (data) | o | | x + 0x78 |
| 3. capture (data) | o | | x + 0x80 |
| 4. reset_alarm (data) | o | | x + 0x88 |
| 5. synchronize_clock (data) | o | | x + 0x90 |
| 6. data_send (data) | o | | x + 0x98 |
| 7. set_encryption_key (data) | o | | x + 0xA0 |
| 8. transfer_key (data) | o | | x + 0xA8 |

| **Attribute description** | |
|---|---|
| **logical_name** | Identifies the “M-Bus client” object instance. See 6.2.22. |
| **mbus_port_ reference** | Provides reference to an “M-Bus master port setup” object, used to configure an M-Bus port, each interface allowing to exchange data with one or more M-Bus slave devices. |
| **capture_ definition** | Provides the *capture_definition* for M-Bus slave devices.<br>NOTE 2 This attribute can be pre-configured or written as part of the installation procedure.<br>array capture_definition_element<br><br>capture_definition_element::= structure<br>{<br>data_information_block: octet-string,<br>value_information_block: octet-string<br>}<br>NOTE 3 The elements data_information_block and value_information_block correspond to Data Information Block (DIB) and Value Information Block (VIB) described in EN 13757-3:2013, 6.2 and Clause 7 respectively. |
| **capture_period** | >= 1: Automatic capturing assumed. Specifies the capture period in seconds.<br>0: No automatic capturing: capturing is triggered externally or capture events occur asynchronously. |
| **primary_ address** | Carries the primary address of the M-Bus slave device. The range is 0...250.<br>Each M-bus device is bound to a channel of the M-Bus master. However, there is no direct link between the primary address and the channel number.<br>NOTE 4 The specification of the B field of the OBIS codes limits the range to 1...64 within one logical device. See IEC 62056-6-1:2017, 5.2.<br>If the slave device is already configured and thus, its primary address is different from 0, then this value shall be written to the *primary_address* attribute. From this moment, the data exchange with the M-Bus slave device is possible.<br>Otherwise, the *slave_install* method shall be used; see below.<br>NOTE 5 The *primary_address* attribute cannot be used to store a desired primary address for an unconfigured slave device. If the *primary_address* attribute is set, this means that the M-Bus client can immediately operate with this primary address, which is not the case with an unconfigured slave device. |

| | |
|---|---|
| **identification_ number** | Carries the Identification Number element of the data header as specified in EN 13757-3:2013, 5.5.<br>This attribute, together with attributes 7, 8 and 9 are filled with the values found in the first message received after installation.<br>If in subsequent messages these values are not the same, the message is discarded. |
| **manufacturer_ id** | Carries the Manufacturer Identification element of the data header as specified in EN 13757-3:2013, 5.6. |
| **version** | Carries the Version element of the data header as specified in EN 13757-3:2013, 5.7. |
| **device_type** | Carries the Device type identification element of the data header as specified in EN 13757-3:2013, 5.8, Table 6. |
| **access_ number** | Carries the Access Number element of the data header as specified in EN 13757-3:2013, 5.9. |
| **status** | Carries the Status byte element of the data header as specified in EN 13757-3:2013, 5.10, Tables 7 and 8.<br>It is updated with every readout of the M-Bus slave device. |
| **alarm** | Carries the Alarm state specified in EN 13757-3:2013, Annex D.<br>It is updated with every readout of the M-Bus slave device. |
| **configuration** | Carries the Configuration field (previously: Signature field) as specified in EN 13757-3:2013, 5.12. It contains information about the encryption mode and the number of encrypted bytes.<br>It is updated with every readout of the M-Bus slave device. |
| **encryption_ key_status** | Provides information on the status of the encryption key. See also Annex B. enum:<br>(0) no encryption key,<br>(1) encryption_key set,<br>(2) encryption_key transferred,<br>(3) encryption_key set and transferred,<br>(4) encryption_key in use. |
| **Method description** | |
| **slave_install (data)** | Installs a slave device, which is yet unconfigured (its primary address is 0).<br>data::= unsigned<br>This method can be successfully invoked only if the current value of the *primary_address* attribute is 0. The following actions are performed:<br>– the M-Bus address 0 is checked for presence of a new device;<br>– if no uninstalled M-Bus slave is found, the method invocation fails;<br>– if the *slave_install* method is invoked with “0” as a parameter, then the primary address is assigned automatically. This is done by checking the *primary_address* attribute of all M-Bus client objects in the DLMS/COSEM device and then selecting the first unused number. The *primary_address* attribute is set to this address and it is then transferred to the M-Bus slave device;<br>– if the *slave_install* method is invoked with a primary address (other than 0) as a parameter, then the *primary_address* attribute is set to this value and it is then transferred to the M-Bus slave device.<br>NOTE 6 Unconfigured slave devices are configured with primary address as specified in EN 13757-3:2013, Annex E.5. |

| | |
|---|---|
| **slave_deinstall (data)** | De-installs the slave device. The main purpose of this service is to de-install the M-Bus slave device and to prepare the master for the installation of a new device. The following actions are performed:<br>– the M-Bus address is set to 0 in the M-Bus slave device;<br>– the encryption key transferred previously to the M-Bus slave device is destroyed; the default key is not affected;<br>– the *encryption_key_status* is set to (0): no encryption_key;<br>– the attribute *primary_address* is also set to 0.<br>NOTE 7 A new M-Bus slave can be installed only once the value of the *primary_address* attribute is 0.<br>data::= unsigned (0) |
| **capture (data)** | Captures values – as specified by the *capture_definition* attribute – from the M-Bus slave device.<br>data::= integer (0) |
| **reset_alarm (data)** | Resets alarm state of the M-Bus slave device.<br>data::= integer (0) |
| **synchronize_ clock (data)** | Synchronize the clock of the M-Bus slave device with that of the M-Bus client device.<br>data::= integer (0) |
| **data_send (data)** | Sends data to the M-Bus slave device.<br>data::= array data_definition_element<br>data_definition_element::= structure<br>{<br>data_information_block: octet-string,<br>value_information_block: octet-string,<br>data: CHOICE<br>{<br>-- simple data types<br>null-data [0],<br>bit-string [4],<br>double-long [5],<br>double-long-unsigned [6],<br>octet-string [9],<br>visible-string [10],<br>utf8-string [12],<br>bcd [13]<br>integer [15],<br>long [16],<br>unsigned [17],<br>long-unsigned [18],<br>long64 [20],<br>long64-unsigned [21],<br>float32 [23],<br>float64 [24]<br>}<br>} |

| | |
|---|---|
| **set_ encryption_ key (data)** | Sets the encryption key in the M-Bus client and enables encrypted communication with the M-Bus slave device.<br>data::= octet-string (encryption_key)<br>After the installation of the M-Bus slave, the M-Bus client holds an empty encryption key. With this, encryption of M-Bus frames is disabled.<br>Encryption can be disabled by invoking the *set_encryption_key* method with an octet-string of zero length.<br>Changing the encryption key requires two steps:<br>– first, the key is sent to the M-Bus slave, encrypted with the default key, using the *transfer_key* method;<br>– second, the key is set in the M-Bus master using the *set_encryption_key* method. |
| **transfer_key (data)** | Transfers an encryption key to the M-Bus slave device.<br>data::= octet-string (encrypted_key)<br>Before encrypted M-Bus frames can be used, an operational encryption key has to be sent to the M-Bus slave, by invoking the *transfer_key method*. The method invocation parameter is the operational encryption key encrypted with the default key of the M-Bus slave device. The M-Bus frame sent is not encrypted. After the execution, the encryption is enabled and all further telegrams are encrypted.<br>Each M-Bus slave device shall be delivered with a default encryption key.<br>A new encryption key can be set in the M-Bus client by invoking the *set_encryption_key* method with the new encryption key as a parameter.<br>With further invocations of the *transfer_key method*, new encryption keys can be sent to the M-Bus slave. The method invocation parameter is the new encryption key encrypted with the default key. The M-Bus frame is encrypted.<br>When an M-Bus slave is de-installed, the encryption key is destroyed but the default key is not affected. Encryption remains disabled until a new encryption key is transferred. |

### 5.7.4 Wireless Mode Q channel (class_id = 73, version = 1)

Instances of this IC define the operational parameters for communication using the mode Q interfaces. See also EN 13757-5:2015.

| **Wireless Mode Q channel** | | **0...n** | **class_id = 73, version = 1** | | | |
|---|---|---|---|---|---|---|
| ***Attributes*** | | ***Data type*** | ***Min.*** | ***Max.*** | ***Def.*** | **Short name** |
| 1. logical_name | (static) | octet-string | | | | x |
| 2. addr_state | (static) | enum | | | | x + 0x08 |
| 3. device_address | (static) | octet-string | | | | x + 0x10 |
| 4. address_mask | (static) | octet-string | | | | x + 0x18 |
| ***Specific methods*** | | ***m/o*** | | | | |

| Attribute description | |
|---|---|
| **logical_name** | Identifies the “Wireless Mode Q channel” object instance. See 6.2.22. |
| **addr_state** | Defines whether or not the device has been assigned an address since last power up of the device.<br>enum:<br>(0) not assigned an address yet,<br>(1) assigned an address either by manual setting or by automated method. |
| **device_address** | The currently assigned address of the device on the network. |
| **address_mask** | The group address the device will respond to when short form addressing is used. |

### 5.7.5 M-Bus master port setup (class_id = 74, version = 0)

Instances of this IC define the operational parameters for communication using the EN 13757-2 interfaces if the device acts as an M-bus master.

| **M-Bus master port setup** | | **0...n** | **class_id = 74, version = 0** | | | |
|---|---|---|---|---|---|---|
| ***Attributes*** | | ***Data type*** | ***Min.*** | ***Max.*** | ***Def.*** | **Short name** |
| 1. logical_name | (static) | octet-string | | | | x |
| 2. comm_speed | (static) | enum | 0 | 7 | 3 | x + 0x08 |
| ***Specific methods*** | | ***m/o*** | | | | |

| Attribute description | |
|---|---|
| **logical_name** | Identifies the “M-Bus master port setup” object instance. See 6.2.22. |
| **comm_speed** | The communication speed supported by the port<br>enum: (0) 300 baud,<br>(1) 600 baud,<br>(2) 1 200 baud,<br>(3) 2 400 baud,<br>(4) 4 800 baud,<br>(5) 9 600 baud,<br>(6) 19 200 baud,<br>(7) 38 400 baud |

### 5.7.6 DLMS/COSEM server M-Bus port setup (class_id = 76, version = 0)

Instances of the “DLMS/COSEM server M-Bus port setup” are used in DLMS/COSEM servers hosted by M-Bus slave devices, using the DLMS/COSEM wired or wireless M-Bus (wM-Bus) communication profile.