

# International Electricity Metering Protocol

## COSEM Client Stack

 <p><b>SANDS</b><sup>®</sup> Explore Excel Expand ISO 9001:2000 Certified Company</p>	<p><b>SANDS INSTRUMENTATION PVT. LTD.</b> <i>ISO 9001:2000 Certified Company</i> <b>H.O. :</b> MF-7, CIPET Hostel Road, Thiru. Vi. Ka. Industrial Estate, Ekkaduthangal, Chennai - 600 097. Tel : (044) 2225 3832 / 2225 3833 Fax : (044) 2225 3831 <b>E.mail :</b> sands@sandsindia.com Website : www.sandsindia.com</p>	
--	---	---

# Table of Contents

..... 1

Table of Contents.....2

Introduction.....3

Operation.....4

Implementation.....4

    Module I.....4

        UART\_Init.....4

        UART\_Tx.....4

        Interrupt Sub Routines.....4

    Module II.....5

        Send\_SNRM.....5

        Send\_AARQ.....5

        GET\_Request.....6

        SET\_Request.....7

        Action\_Request.....7

        DISC.....7

## Introduction

The *DLMS Client Stack* is an implementation of the *International Standard IEC-62056* defined *COSEM Protocol*. The client software is developed for a meter with *three layer HDLC Protocol* implemented in the *Direct HDLC Mode for Application Contexts Short Name and Logical Name Referencing*. The *Physical Interface* used is *RS-232*. The Stack is implemented in the *programming language C*.

DLMS is an *Application Layer Specification* and is independent of the lower layers and thus the mode of communication. The objective of DLMS is to provide an inter-operable environment for *Structured Modelling* and *Meter Data Exchange*. COSEM includes an evolution of DLMS providing a more metering specific view of the meter through the *COSEM Interface Objects* and *Interface Classes*. The current *DLMS Version of xDLMS* is 6.

*DLMS/COSEM* is an *Object Model* to view the functionality of the meter, as in it seen at its *communicating interface(s)*, a *Messaging Method* – based on *ASN.1 Encoding* – to communicate with the model & to turn the data to a series of bytes which a *Transportation Method* carries this information between the *Metering Equipment* and the *Data Collection System*.

## Operation

The project can be segmented into the following major parts:

- 1) The first part involves the *initialisation* and *configuration* of the *UART*. The *UART* should be configured for the same *Baud Rate* as that of the Server (*Meter* or *Data Terminal Unit*).
- 2) The second part includes the *DLMS Client Application*, where the queries to the server are designed. Like the Server Implementation incorporates a three Layer Protocol; a similar stack is to be developed at the Client side.

## Implementation

The implementation of the above mentioned model is now described in detail.

### Module I

The implementation of Module I – the Physical Layer – requires the following three functions:

#### UART\_Init

This function *initialises* the *UART Module* of the Meter Reading Instrument. The *UART* is configured for a baud rate known to the server prior and for *8 bits Word Length, No Parity* and *1 Stop Bit*. The client software should enable *interrupt* on *reception* of a *character*, which is stored in a buffer to be analysed by the DLMS Model.

#### UART\_Tx

This function takes in a character as input and *transmits* it via *RS-232* to the server.

#### Interrupt Sub Routines

In addition to the above two functions, the client software should have necessary *Interrupt Sub Routines* to store the data received from the server through *RS-232*.

## Module II

The DLMS Stack is developed from the following references:

- 1) IEC 62056-21 Direct Local Data Exchange
- 2) IEC 62056-46 Data Link Layer using HDLC Protocol
- 3) IEC 62056-53 COSEM Application Layer
- 4) IEC 62056-61 Object Identification System
- 5) IEC 62056-62 Interface Classes
- 6) IEC 13239:2000 Telecommunications and Information Exchange between Systems – HDLC Procedures

Following are the various requests sent from the client to the server:

### Send\_SNRM

The SNRM – *Set Normal Response Mode* – command is used to place the addressed secondary station in the *Normal Response Mode*. The function Send\_SNRM builds the SNRM frame and sends it to the server. The SNRM can be used to negotiate HDLC parameters such as the maximum data reception and transmission lengths etc, which are to be specified by the user. The server can be addressed with either one byte, two bytes or four bytes address. The server address needs to be included by the user.

Once the SNRM command is sent successfully, the server responds with the negotiated HDLC parameters. The parameters thus obtained are to be replaced in the program.

These parameters are as follows:

- 1) Maximum Information Field Length, transmit
- 2) Maximum Information Field Length, receive
- 3) Window size, transmit and
- 4) Window size, receive.

### Send\_AARQ

The AARQ – *Application Association Request* – is sent by the client to the server, on successful reception of response to the SNRM. As the name suggests, AARQ is used to establish an

association with the Server. The AARQ PDU (Protocol Data Unit) communicates the *Application Context* used and the *Authentication Value* (Secret Password to authenticate Client and/or Server) if any to the Server. An association is established when the server is compliant with the parameters sent by the client. In other cases, the association is dropped and the server goes into the Disconnected Mode.

This function takes as input the various configuration parameters to be sent to the server. Its output indicates whether or not the association was established.

On successful establishment of the association, the negotiated values replace the existing values and will be considered in communication henceforth.

The AARQ is used to negotiate the following parameters:

- 1) The Protocol Version,
- 2) Application Context Name,
- 3) Authentication Requirements (if authentication is used),
- 4) Association or xDLMS Information, which involves the following:
  - a. Dedicated Key,
  - b. Quality of Service,
  - c. Proposed DLMS Version,
  - d. Proposed DLMS Conformance Block and
  - e. Maximum Receive APDU Size.

## **GET\_Request**

This function is used to fetch data from the Server (Meter). The input to the function is the *OBIS Code* (or Short Name in case of SN Referencing) and the attribute of the object to be requested. The function then checks whether or not the client is entitled to the information requested, by analysing its *Access Rights*. On a favourable outcome, the function builds the request by including the OBIS Code, the Interface Class –obtained from the OBIS.h file – and the attribute and sends a request to the server.

On reception of a frame from the server, the function extracts the useful information and uses it accordingly.

## SET\_Request

This function is the complementary of the above stated function. Given the OBIS code of an object and the value to be modified at the server (meter) side, the function verifies whether it has the permission to update the particular object. If so, the function builds an *Application Frame*, obtaining the IC Number, Version etc from the file OBIS.h. Additionally, it takes as input the value to be set. The function will return the outcome of the request and in case of failure, the reason of failure will be indicated.

## Action\_Request

This function is used to carry out the methods of the COSEM objects. The function takes as input the OBIS Code of the Object and the method to be executed. The function will return the outcome of the request and in case of failure, the reason of failure will be indicated.

## DISC

The *DISC – Disconnect* – command is used to disconnect the logical link layer of the client from the server. On reception of a DISC command, the server responds with a *UA*, similar to the one sent on response to the SNRM. The server is said to be in *Normal Disconnected Mode* on reception of this command. *No data exchange* can be carried out in this state, except *Unnumbered Information* and *Mode Setting Command* such as the SNRM.