STD.NEMA C12.21-ENGL 1999 6470247 0520627 365

AMERICAN NATIONAL STANDARD

American National Standard

ANSI C12.21-1999

American National Standard

Protocol Specification for Telephone Modem Communication

NATIONAL ELECTRICAL MANUFACTURERS ASSOCIATION
1300 NORTH 17TH STREET, SUITE 1847, ROSSLYN, VA 22209
(703) 841-3200 FAX: (703) 841-3300

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

ANSI C12.21-1999

AMERICAN NATIONAL STANDARD

Protocol Specification for Telephone Modem Communication

Accredited Standard Committee on Electricity Metering, C12

Accredited by

American National Standards Institute

Approved October 14, 1999

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS

Not for Resale

American National Standards Institute, Inc.

American National Standard

Approval of an American National Standard requires verification by ANSI that the requirements for due process, consensus, and other criteria for approval have been met by the standards developer.

Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that a concerted effort be made toward their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether he has approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not conforming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give an interpretation of any American National Standard. Moreover, no person shall have the right or authority to issue an interpretation of an American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

CAUTION NOTICE: This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken periodically to reaffirm, revise, or withdraw this standard. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.

Published by

National Electrical Manufacturers Association
1300 N. 17th Street, Rosslyn, Virginia 22209

© Copyright 1999 by the National Electrical Manufacturers Association. All rights including translation into other languages, reserved under the Universal Copyright Convention, the Berne Convention for the Protection of Literary and Artistic Works, and the International and Pan American Copyright Conventions.

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without prior written permission of the publisher.

Printed in the United States of America

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

TABLE OF CONTENTS

1 Scope 1
2 References 1
3 Definitions 1
3.1 PSEM 1
3.2 Table 1
3.3 Document syntax 1
4 Protocol details 2
4.1 Order of transmission 2
4.2 LAYER 7 - Application layer 3
4.2.1 Data Structure 3
4.2.2 Language - Protocol Specification for Electric Metering ("PSEM") 3
4.2.2.1 Request Codes 3
4.2.2.2 Response Codes 4
4.2.2.3 Identification Service 4
4.2.2.4 Read Service 5
4.2.2.5 Write Service 5
4.2.2.6 Logon Service 5
4.2.2.7 Security Service 5
4.2.2.8 Logoff Service 5
4.2.2.9 Authenticate service 5
4.2.2.10 Negotiate Service 6
4.2.2.11 Wait Service 6
4.2.2.12 Terminate Service 6
4.2.2.13 Timing Setup Service 6
4.2.2.14 Disconnect Service 7
4.3 LAYER 6 - Presentation layer 7
4.4 LAYER 5 - Session layer 7
4.5 LAYER 4 - Transport layer 8
4.6 LAYER 3 - Network layer 8
4.7 LAYER 2 - Data link layer 8
4.7.1 Basic data information 8
4.7.1.1 Fixed Settings 8
4.7.1.2 Variable Settings 8
4.7.2 Packet definition 9
4.7.3 CRC selection 9
4.7.4 Acknowledgment 10
4.7.5 Retry attempts 10
4.7.6 Timeouts 10
4.7.6.1 Channel Traffic Time-out 10
4.7.6.2 Inter-Character Time-out 10
4.7.6.3 Response Time-out 10
4.7.7 Turn around delay 11
4.7.8 Collision 11
4.7.9 Duplicate packets 11
4.8 LAYER 1 - Physical layer 11
5 Compliance 11
ANNEX A - Protocol syntax 12
ANNEX B - Communication example (Layer 7 & Layer 2) 15
ANNEX C - Packet transmission example 17
ANNEX D - Service sequence state control 19
ANNEX E - Modifications and extensions to C12.19- 1997 21
E.1 RDATE Type 21
E.2 TABLE 03 ED_MODE STATUS TABLE 23
E.3 TABLE 07 - Procedure initiate table 26

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

E4 - DECADE 90: Telephone Control Tables 27
E.5 History & Event Log Codes 41
E.6 Default sets for decade tables 42
E.7 Indicies for partial table access 43
ANNEX F - CRC examples 46
ANNEX G - Error handling 48
ANNEX H - Data encryption standard 50
ANNEX I - I Command operational description 58

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

Foreword (This foreword is not part of ANSI C12.21)

This American National Standard establishes acceptable performance criteria for electricity meters. Accuracy class designations, current class designations, voltage and frequency ratings, test current values, service connection arrangements, pertinent dimensions, form designations, and environmental tests are covered.

Suggestions for improvement to this standard are welcome. They should be sent to:

National Electrical Manufacturers Association
Vice President of Engineering
1300 North 17th Street
Suite 1847
Rosslyn, VA 22209

At the time this standard was completed, Subcommittee 17 of ANSI Committee C12, which developed and revised this standard, had the following membership:

Wesley Ray, Chair

Michael Anderson
Ed Beroset
William J. Buckley
Stuart M. Garland
Larry Kotewa
Avygdor Moise
Jack Pazdon
Terry Penn
Marc Purc
Wesley Ray
William F. Rush
Chris M. Schafer
Richard Tucker
Michel Veillette
Ted York

At the time this standard was completed, the American National Standards Committee C12 had the following membership:

Nile Oldham, Chair **Khaled Masri, Secretary**

American Public Power Association	Cruz R. Gomez
Measurement Canada	Avygdor Moise (Alternate) Vuoung Nguyen
Edison Electric Institute and Electric Light and Power Group	Brad Johnson James Mining (Alternate) John McEvoy Lauren Pananen Tim Morgan Tim Vahlstrom Wesley Ray (Alternate)

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

National Electrical Manufacturers Association	Ed Beroset (Alternate) Francis A. Marta Jack Pazdon (Alternate) Michael Anderson Ronald Shaw (Alternate) Scott Weikel (Alternate) Tom C. Drew Warren Germer
Institute of Electrical and Electronics Engineers	Herman M. Millican Richard Tucker William J. Buckley
National Institute of Standards and Technology	Nile Oldham
National Association Public Utility Commission	James Ruehl
Rural Utility Service	Stephen Jones
Underwriters Laboratories, Inc.	Ron Breschini

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS

Not for Resale

1 Scope

This document details the criteria required for communications between an electric power metering device and a utility host via a modem connected to the switched telephone network. The utility host could be a laptop or portable computer, a master station system, an electric power metering device, or some other electronic communications device.

This standard does not specify the implementation requirements of the telephone switched network to the modem, nor does it include definitions for the establishment of the communication channel.

This document provides details for an implementation of the OSI 7-layer model.

The protocol specified in this document was designed to transport data in table format. The table definitions are referenced in Section 2.0, References, and Annex E of this document.

This document specifies the differences between ANSI Std C12.18-1996, *Protocol Specification for ANSI Type 2 Optical Port* and ANSI Std C12.19-1997, *Utility Industry End Device Data Tables* and those features and services required to describe a protocol specification for Telephone Modem Communications.

2 References

ANSI Std C12.18-1996:	Protocol Specification for ANSI Type 2 Optical Port
ISO 7498/1:	OSI Reference Model
ISO 3309-1993(E):	Information technology – Telecommunications and information exchange between systems – High-level data link control (HDLC) procedures – Frame Structure, Annex A Explanatory Notes On Implementation of the Frame Checking Sequence
ANSI Std C12.19-1997:	Utility Industry End Device Data Tables
ANSI Std X3.92-1981:	Data Encryption Algorithm

3 Definitions

For the purposes of this document, the following definitions are made for terms and syntax used throughout this document.

3.1 PSEM

Protocol Specification for Electricity Metering as referenced in C12.18-1996.

3.2 Table

Functionally related data elements, grouped together into a single data structure for transport as defined by ANSI standard C12.19-1997.

3.3 Document syntax

Document syntax is identical to that of C12.18-1996.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

4 Protocol details

Following the guidelines established by the OSI seven-layer model, the protocol described in this document provides three functions. These are:

- The modification of the communication channel
- The transport of information to and from the metering device
- The orderly closure of the communication channel

Three layers are used to provide these communication capabilities. These are the Physical, Data Link, and Application layers.

This standard does not specify the implementation requirements of the telephone switched network to the modem, nor does it include definitions for the establishment of the communication channel. The communication channel is considered available once the telephone connection has been successfully established and modems have synchronized. The PSEM Identification Service is required to establish the protocol version and revision in use. Certain communication parameters may be modified through the use of the PSEM Negotiate and Timing Setup services in the application layer. These services are optional and if not implemented in the metering device or not used during actual communications, the communication channel parameters will remain at the default settings specified by this document. Device implementers are strongly encouraged to implement these optional application services.

Once the data link is established, the application layer provides:

- Logon, Security, and Logoff Services for session activation, access control, and deactivation
- Read and write services for issuing data transmission requests
- Terminate Service for returning to the base state
- Disconnect Service for shutdown of the communication channel
- A response structure that provides information regarding the success or failure of the service requests

An example of a typical communications session would consist of the following services with appropriate responses, in the order listed: Identification; Negotiate; Logon; Security; Read (zero or more); Write (zero or more); Logoff; and Disconnect. Note that this brief example does not detail the packet structure or other aspects of the protocol. For a more detailed example of a typical communications session reference Annex B, Communication Example.

4.1 Order of transmission

Order of transmission is identical to that specified in C12.18-1996 and C12.19-1997.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

4.2 LAYER 7 — Application layer

The application layer provides a minimal set of services and data structures required to support end devices for purposes of configuration, programming, and information retrieval.

Layer 7 shall be defined as the ANSI Std C12.18-1996 Protocol Specification for ANSI Type 2 Optical Port, Layer 7, except for the specific items mentioned in the following subsections of 4.2.

4.2.1 Data structure

This protocol shall transport table structures. The table specifications this standard was designed to transport are referenced in Clause 2 and Annex E of this document.

4.2.2 Language — Protocol Specification for Electric Metering ("PSEM")

The language PSEM has been designed to provide an interface between a utility host and a metering device over a point-to-point communication medium. The PSEM language consists of 12 services. Each service consists of a request and a response. Each of these requests and responses is described in the following service sections.

<requests>	::=	<ident> |	{ Identification request }
		<read> |	{ Read request }
		<write> |	{ Write request }
		<logon> |	{ Logon request }
		<security> |	{ Security request }
		<logoff> |	{ * Logoff request }
		<negotiate> |	{ Negotiate request }
		<wait> |	{ Wait request }
		<terminate> |	{ * Terminate request }
		<timing_setup> |	{ ** Timing setup request }
		<disconnect> |	{ ** Disconnect request }
		<authenticate>	{ ** Authenticate request }
<responses>	::=	<ident_r> |	{ * Identification response }
		<read_r > |	{ Read response }
		<write_r > |	{ Write response }
		<logon_r > |	{ Logon response }
		<security_r> |	{ Security response }
		<logoff_r > |	{ * Logoff response }
		<negotiate_r > |	{ Negotiate response }
		<wait_r > |	{ Wait response }
		<terminate_r > |	{ * Terminate response }
		<timing_setup_r> |	{ ** Timing setup response }
		<disconnect_r> |	{ ** Disconnect response }
		<authenticate_r>	{ ** Authenticate response }

Note: * Definition or content revised from C12.18-1996.
** New in C12.21.

4.2.2.1 Request codes

PSEM requests always include a one byte request code. Code numbers are assigned as follows:

00_H-$1F_H$	Codes shall not be used to avoid confusion with response codes
20_H-$7F_H$	Codes are available for use within ANSI C12 protocols
80_H-FF_H	Codes shall be reserved for protocol extensions

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

4.2.2.2 Response codes

Response codes are identical to those specified in C12.18-1996.

4.2.2.3 Identification service

This service shall be the first service issued once the physical connection is established or after a terminate service. The service returns a code identifying the reference standard, the version and revision of the reference standard implemented, and an optional feature list.

Request:

<ident> ::= 20_H

Response:

The responses <err>, <bsy>, and <isss> indicate a problem with the received service request. The response <ok> indicates the identification service request was accepted and the standard, version, revision, and optional feature list are included in the response.

<ident_r>	::=	<err> | <bsy> | <isss> | <ok> <std> <ver> <rev> <feature>* <end_of_list>	
<std>	::=	<byte>	{Code identifying reference standard. The codes are defined as follows: 00_H = ANSI C12.18 01_H = For use by Industry Canada 02_H = ANSI C12.21 03_H-FF_H = Reserved }
<ver>	::=	<byte>	{Binary representation of the referenced standard version number to the left of the decimal point. This value shall be 1.}
<rev>	::=	<byte>	{Binary representation of the referenced standard version number to the right of the decimal point. This value shall be 0.}
<feature>	::=	<auth_ser> | <auth_ser_ticket>	{Features available}
<end_of_list>	::=	00_H	{End of list indicator.}
<auth_ser>	::=	01_H <auth_type> <auth_alg_id>	{If present in the feature list, authentication is supported by the end device.}
<auth_ser_ticket>	::=	02_H <auth_type> <auth_alg_id> <ticket_length> <ticket>	
<auth_type>	::=	<byte>	{Authentication type identifier. Bit 0: SESSION_LEVEL_AUTHENTICATION If true "Authentication Service" is supported. Bits 1..7 are reserved, must be zero.}
<auth_alg_id>	::=	<byte>	{The authentication algorithm used. 00_H = ANSI Std X3.92-1981: Data Encryption Algorithm}
<ticket_length>	::=	<byte>	{Length of <ticket>, in bytes.}
<ticket>	::= <byte>+		{Value use by the authentication algorithm.}

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS Not for Resale

4.2.2.4 Read service

The read service is identical to that in C12.18-1996.

4.2.2.5 Write service

The write service is identical to that in C12.18-1996.

4.2.2.6 Logon service

The logon service is identical to that in C12.18-1996.

4.2.2.7 Security service

The security service is identical to that in C12.18-1996.

4.2.2.8 Logoff service

The logoff service provides for an orderly shutdown of the session established by the logon service.

Request:

Following a logoff service request the communication channel will return to the ID State, retaining the parameters previously established.

<logoff> ::= 52_H

Response:

The responses <err>, <bsy>, and <isss> indicate a problem with the received service request.

The response <ok> indicates the acceptance of the logoff service and the cessation of the session established by the logon service. Prior to further data transfers with the metering device, the logon service must be reissued.

<logoff_r> ::= <err> |<bsy> |<isss> |<ok>

4.2.2.9 Authenticate service

The authenticate service is used when a higher level of security is desired. The intent of this service is to provide a two way authentication with playback rejection at the session level. The contents of the <auth_request> and <auth_response> fields are a function of the authentication algorithm used. This algorithm is returned by the identification service response.

Request:

<authenticate>	::= 53_H <auth_req_length> <auth_request>	
<auth_req_length>	::= <byte>	{<auth_req_length> number of bytes of the <auth_request> field.}
<auth_request>	::= <byte>$^+$	{Information used to authenticate the initiator of this service.}

Response:

The responses <err>, <bsy>, and <isss> indicate a problem with the received service request.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS Not for Resale

The response <isc> indicates the authentication failure of the requester.

The response <ok> indicates the authentication service was successfully completed and the access permission associated with the <auth_request> field was granted.

<authenticate_r>	::=	<err> | <bsy> | <isc> | <isss> | <ok> <auth_res_length> <auth_response>	
<auth_res_length>	::=	<byte>	{<auth_res_length> number of bytes of the <auth_response> field.}
<auth_response>	::=	<byte>+	{Information used to authenticate the recipient of this service.}

4.2.2.10 Negotiate service

The negotiate service is identical to that in C12.18-1996. However, the <baud_rate> field of the request shall be ignored because in this standard the data rate is established by the modem.

4.2.2.11 Wait service

The wait service is identical to that in C12.18-1996.

4.2.2.12 Terminate service

The terminate service provides for immediate transfer to the base state. All parameters return to their default values.

Request:

<terminate>	::=	21_H

Response:

The response <err> indicates a problem with the received service request and the channel retains its negotiated settings.

The response <ok> indicates the service request was accepted and the channel will return to default settings as stated in Section 4.7.1.2, Variable Settings, upon receipt of a positive acknowledgment.

<terminate_r>	::=	<err> | <ok>

4.2.2.13 Timing setup service

The timing setup service provides the mechanism for reconfiguring time-outs, delays, and retry attempts from the default values specified in this document.

This service is initiated by the utility host. It is optional and, if not used, the communications channel operates with the default parameters established by this document.

Request:

<timing_setup>	::=	71_H <traffic> <inter_char> <resp_to> <nbr_retries>

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

<traffic>	::=	<byte>	{ Channel traffic time-out in seconds }
<inter_char>	::=	<byte>	{ Inter-character time-out in seconds }
<resp_to>	::=	<byte>	{ Response time-out in seconds }
<nbr_retries>	::=	<byte>	{ Maximum number of retry attempts }

Response:

The responses <err>, <sns>, <bsy>, and <isss> indicate a problem with the received service request and the timer parameters will maintain their current settings.

The response <ok> indicates the service request was accepted and the new settings now apply. The data following the <ok> indicates the setting that will apply upon positive acknowledgment.

<timing_setup_r> ::= <err> |
<sns> |
<bsy> |
<isss> |
<ok> <traffic> <inter_char> <resp_to>
<nbr_retries>

4.2.2.14 Disconnect service

This service is used for immediate disconnection of the communication channel.

Request:

This service should be used for reasons such as excessive errors, security issues, internal error conditions, end of session, or other reasons as set by the device manufacturer.

<disconnect> ::= 22_H

Response:

The response <err> indicates a problem with the received service request and the channel remains open.

The response <ok> indicates the service request was accepted and the channel will be disconnected.

<disconnect_r> ::= <err> |
<ok>

4.3 LAYER 6 — Presentation layer

NULL LAYER

The end device will not provide queuing capabilities for service requests.

4.4 LAYER 5 — Session layer

NULL LAYER

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

Communications with the end device over the communications path will be connection-oriented and consist of a single session. The session is defined to begin with the acceptance of the logon service and terminates with the acceptance of either the logoff, terminate, or disconnect service.

4.5 LAYER 4 — Transport layer

NULL LAYER

4.6 LAYER 3 — Network layer

NULL LAYER

4.7 LAYER 2 — Data link layer

Services of upper layers are transported in one or more packets. Each packet is variable in length but cannot exceed a maximum packet size. The maximum packet size has a default value when the communication channel is opened. The maximum packet size can be changed through the use of the negotiate service.

For each packet received, a positive or negative acknowledgment is sent by the target. This acknowledgment consists of a single byte transmitted outside of the packet structure. If the requester does not receive an acknowledgment before the Response Timeout, or a negative acknowledgment is received, the same packet is re-transmitted up to the current maximum number of negotiated retry attempts. After the final retry attempt, the requester should assume termination has occurred.

4.7.1 Basic data information

Communication channel settings are specified below.

4.7.1.1 Fixed settings

DATA FORMAT	8 data bits, 1 start bit, 1 stop bit, no parity
DATA TYPE	Asynchronous, serial bit (start - stop), half duplex operation, bi-directional communications
DATA POLARITY	Start bit, space, logical 0 Stop bit, mark, logical 1, quiescent state
DATA RATE	The transmitting speed is determined when connection is established.
TURN-AROUND DELAY	As required by physical layer

4.7.1.2 Variable settings

Default settings apply at the initiation of communication. Most settings may be changed through the negotiate service and the timing service. Channel settings will return to defaults as a result of the terminate service or channel traffic time-out.

Setting	Default Value	Changed by
NUMBER OF PACKETS	1	Negotiate
PACKET SIZE	64 bytes	Negotiate
CHANNEL TRAFFIC TIMEOUT	30 seconds	Timing Setup
INTER-CHARACTER TIMEOUT	1 second	Timing Setup
RESPONSE TIMEOUT	4 seconds	Timing Setup
RETRY ATTEMPTS	3	Timing Setup

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

4.7.2 Packet definition

<packet>	::=	<stp> <identity> <ctrl> <seq_nbr> <length> <data> <crc>	
<stp>	::=	EE_H	{ Start of packet character. }
<identity>	::=	<byte>	{ End device (meter, etc.) identity. It identifies the end device in both the request and response packets. In requests, this byte may be set to 00_H as a universal identity. All devices shall respond to this identifier value as a minimum. This value (00_H)should not be used in a multidrop environment. The individual end device identity must be in the range 01_H to FE_H and can be specified by PSEM identify field in Global Parameters Table (Table 92). In requests, this byte may be set to FF_H to target the calling party during an inbound call. Only the end device that initiated the inbound call shall process packets with identity byte equal to FF_H. The value FF_H may be used in both single drop and multidrop environments. The end device shall use its own identity byte in the response for all requests. Including 00_H and FF_H }
<ctrl>	::=	<byte>	{ Control field. Bit 7. If true (1_H) then this packet is part of a multiple packet transmission. Bit 6. If true (1_H) then this packet is the first packet of a multiple packet transmission. Bit 5. Represents a toggle bit to reject duplicate packets. This bit is toggled for each new packet sent. Retransmitted packets keep the same state as the original packet sent. Bits 0 to 4, Reserved.}
<seq_nbr>	::=	<byte>	{Number that is decremented by one for each new packet sent. The first packet in a multiple packet transmission shall have a <seq_nbr> equal to the total number of packets minus one. A value of zero in this field indicates that this packet is the last packet of a multiple packet transmission. If only one packet is in a transmission, this field shall be set to zero.}
<length>	::=	<word16>	{ Number of bytes of <data> in packet. }
<data>	::=	<byte>$^+$	{ <length> number of bytes of actual packet data. This value is limited by the maximum packet size minus the packet overhead of 8 bytes. This value can never exceed 8183. }
<crc>	::=	<word16>	{ HDLC implementation of the polynomial $X^{16} + X^{12} + X^5 + 1$}

4.7.3 CRC selection

The CRC selected for implementation is the polynomial $X^{16} + X^{12} + X^5 + 1$. The method for calculation and insertion is the HDLC standard per ISO 3309-1993(E) Annex A "EXPLANATORY NOTES ON IMPLEMENTATION OF THE FRAME CHECKING SEQUENCE."

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

In the PSEM frame, there is no opening flag as referenced in ISO 3309-1993(E) Annex A. The PSEM start of packet character EE is included in the CRC calculation. The result of the CRC calculation shall be transmitted least significant byte first per the ISO 3309 Annex. See Annex F for examples of computation and coding.

4.7.4 Acknowledgment

A positive or negative acknowledgment is used by the communicating devices to indicate either acceptance or rejection of a packet.

An <ack> is issued by the receiving device to the transmitting device for each packet received that meets the constraints established by this section.

<ack> ::= 06_H

A <nak> is issued by the receiving device to the transmitting device for each packet received that does not meet the constraints established by this section. Examples of problems with received packets indicated by a <nak> response include CRC errors, packet structure errors, incorrect bit patterns, and missing characters.

<nak> ::= 15_H

4.7.5 Retry attempts

The same packet shall be retransmitted if a <nak> is received, an invalid character is received (neither an ACK nor a NAK), or the response time-out occurs.

After the failure of the final retry attempt, the following actions will occur:

If the utility host is the requestor, then the host returns to the base state but may or may not terminate the call.

If the end device is the requestor, the end device will terminate the call.

4.7.6 Timeouts

4.7.6.1 Channel traffic time-out

The end device will terminate communications after waiting some period of time for a valid packet or acknowledgment. This period of time, is defined as the channel traffic time-out.

4.7.6.2 Inter-character time-out

The recipient of the packet must handle the case when the end of a packet is lost. Inter-character time-out is defined as the minimum amount of time the recipient shall wait between characters within a packet when receiving data over the communication channel. The recipient shall wait at least this amount of time before it ceases to wait for the remainder of the packet and sends a <nak>.

4.7.6.3 Response time-out

The sender of the packet must handle the case when the acknowledgment or negative acknowledgment is lost. Response time-out is defined as the minimum amount of time the sender shall wait after a packet is sent to receive an acknowledgment over the communication channel. If this amount of time elapses, the sender will send the packet again.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS Not for Resale

4.7.7 Turn around delay

The turn around delay is the minimum delay the recipient must wait after receiving a packet and before sending a positive or negative acknowledgement.

4.7.8 Collision

If the end device detects a collision (utility host and end device are transmitting at the same time), the end device shall cease transmission and wait for the transmission from the utility host.

4.7.9 Duplicate packets

A duplicate packet is defined as a packet whose identity, toggle bit, and valid CRC are identical to those of the immediate previous packet. If a duplicate packet is received by the target device, the target should disregard the duplicate packet and return an <ack>. At the start of session, the toggle bit in the first packet may assume either state.

4.8 LAYER 1 — Physical layer

The physical layer is not defined in this document.

5 Compliance

A device is considered to be in compliance with this document if all of the following conditions are met:

a) The device shall accept and act upon all service requests defined in Section 4.2, Layer 7 – Application Layer. If the service is not supported the end device shall respond with an <sns>. The minimum services supported shall include Identification, Logon and at least one form of Read. Any service mentioned in this document, if implemented, must comply with the C12.21 service description.

b) The device is compliant with the parent standard ANSI C12.19-1997.

c) Layer two must be adhered to in its entirety.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

ANNEX A — Protocol syntax

Informative

These are additions to ANSI C12.18, Annex A.

A.1 LAYER 7 SYNTAX

<authenticate> ::= 53_H <auth_req_length> <auth_request>

<authenticate_r> ::= <err> |
<bsy> |
<isc> |
<isss> |
<ok> <auth_res_length> <auth_response>

<auth_alg_id> ::= <byte>

<auth_req_length> ::= <byte>

<auth_request> ::= <byte>*

<auth_res_length> ::= <byte>

<auth_response> ::= <byte>*

<auth_ser> ::= 01_H <auth_type> <auth_alg_id>

<auth_ser_ticket> ::= 02_H <auth_type> <auth_alg_id> <ticket_length> <ticket>

<auth_type> ::= <byte>

<disconnect> ::= 22_H

<disconnect_r> ::= <err> |
<ok>

<end_of_list> ::= 00_H

<feature> ::= <auth_ser> |
<auth_ser_ticket>

<ident> ::= 20_H

<ident_r> ::= <err> | <bsy> | <isss> |
<ok> <std> <ver> <rev> <feature>* <end_of_list>

<inter_char> ::= <byte>

<logoff> ::= 52_H

<logoff_r> ::= <err> |<bsy> |<isss> |<ok>

<nbr_retries> ::= <byte>

<requests> ::= <ident> |
<read> |
<write> |
<logon> |
<security> |
<logoff> |

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

<negotiate> |
<wait> |
<terminate> |
<timing_setup> |
<disconnect> |
<authenticate>

<responses> ::= <ident_r> |
<read_r > |
<write_r > |
<logon_r > |
<security_r> |
<logoff_r > |
<negotiate_r > |
<wait_r > |
<terminate_r > |
<timing_setup_r> |
<disconnect_r> |
<authenticate_r>

<resp_to> ::= <byte>

<rev> ::= <byte>

<std> ::= <byte>

<terminate> ::= 21_H

<terminate_r> ::= <err> |
<ok>

<ticket_length> ::= <byte>

<ticket> ::= <byte>$^+$

<timing_setup> ::= 71_H <traffic> <inter_char> <resp_to>
<nbr_retries>

<timing_setup_r> ::= <err> |
<sns> |
<bsy> |
<isss> |
<ok> <traffic> <inter_char> <resp_to>
<nbr_retries>

<traffic> ::= <byte>

<ver> ::= <byte>

A.2 LAYER 2 SYNTAX

<ack> ::= 06_H

<crc> ::= <word16>

<ctrl> ::= <byte>

<data> ::= <byte>$^+$

<identity> ::= <byte>

<length> ::= <word16>

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

<nak> ::= 15_H

<packet> ::= <stp> <identity> <ctrl> <seq_nbr> <length> <data> <crc>

<seq_nbr> ::= <byte>

<stp> ::= EE_H

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS

Not for Resale

ANNEX B — Communication example (Layer 7 & Layer 2)

Informative

Figures B.1 and B.2 show an example of a communications session between a utility host and an end device.

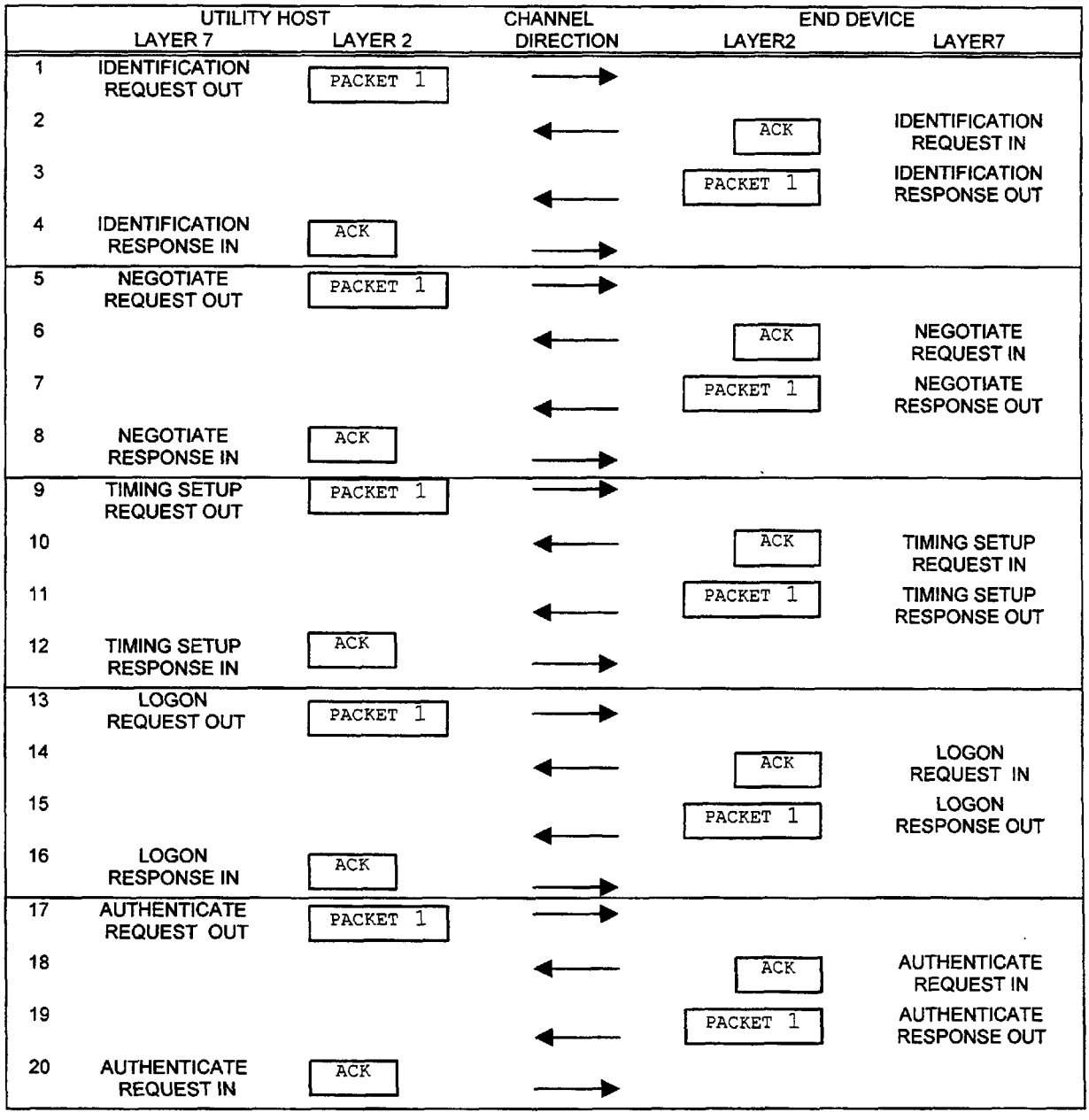


Figure B.1 – Communication example

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

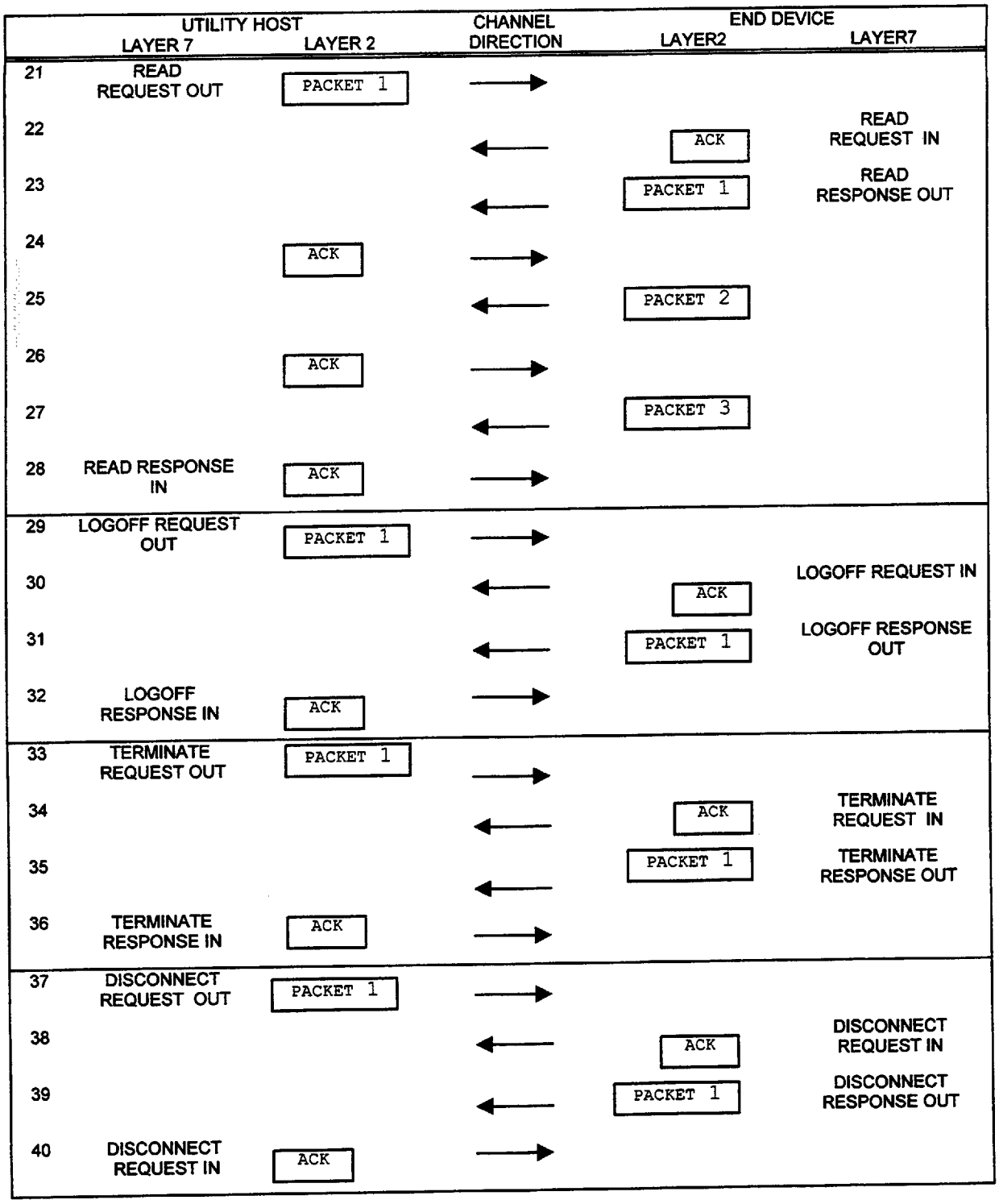


Figure B.2 – Communication example *continued*

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

ANNEX C — Packet transmission example

Informative

Figure C.1 shows the actual packet data being transmitted in Figures B.1 and B.2 in Annex B. Numbers 1 – 40 refer to the numbers in Figures B.1 and B.2. All values are specified in hexadecimal format. The following arbitrary information was used.

The data returned from the partial Table read in step 21 does not represent C12.19 Table 1.

<std>	::=	02
<ver>	::=	01
<rev>	::=	00
<ticket>	::=	30 36 31 37 34 30 33 30
<packet_size>	::=	0040 (64 bytes)
<nbr_packet>	::=	04 (4 packets)
<baud_rate>	::=	06 (9600 bits per second)
<traffic>	::=	1E (30 sec)
<inter_char>	::=	04 (4 sec)
<resp_to>	::=	04 (4 sec)
<nbr_retries>	::=	03 (3 times)
<user_id>	::=	0000
<username>	::=	41 42 43 44 45 46 47 48 49 4A ("ABCDEFGHIJ")
<key_id>	::=	00
<ini_auth_vector>	::=	DF A9 10 4C 37 BC 1E 26
<tar_auth_vector>	::=	CC C8 09 95 63 9E B3 2C
<table_id>	::=	0001 (Table 1)
<offset>	::=	000010 (16 bytes)
<count>	::=	0096 (150 bytes)
<data>	::=	01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

```
1) → EE 00 00 00 00 01 20 13 10
2) ← 06
3) ← EE 00 00 00 00 11 00 02 01 00 02 01 00 08 30 36 31 37 34 30 33 30 00
     C5 6A
4) → 06
5) → EE 00 20 00 00 04 60 00 40 04 FD BF
6) ← 06
7) ← EE 00 20 00 00 05 00 00 40 04 06 03 1C
8) → 06
9) → EE 00 00 00 00 05 71 1E 04 04 03 D4 AA
10) ← 06
11) ← EE 00 00 00 00 05 00 1E 04 04 03 63 B4
12) → 06
13) → EE 00 20 00 00 0D 50 00 00 41 42 43 44 45 46 47 48 49 4A EE 54
14) ← 06
15) ← EE 00 20 00 00 01 00 80 51
16) → 06
17) → EE 00 00 00 00 0B 53 09 00 DF A9 10 4C 37 BC 1E 26 AC BC
18) ← 06
19) ← EE 00 00 00 00 0B 00 09 00 CC C8 09 95 63 9E B3 2C 93 7C
20) → 06
21) → EE 00 20 00 00 08 3F 00 01 00 00 10 00 96 2B 50
22) ← 06
23) ← EE 00 E0 02 00 38 00 00 96 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
     10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27
     28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 09 F0
24) ← 06
25) ←EE 00 80 01 00 38 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47
     48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F
     60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 43 39
26) → 06
27) ← EE 00 A0 00 00 2A 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F
     80 81 82 83 84 85 86 87 88 98 A8 B8 C8 D8 E8 F0 90 91 92 93 94 95 96 27
     1C 9E
28) → 06
29) → EE 00 00 00 00 01 52 86 40
30) ← 06
31) ← EE 00 00 00 00 01 00 11 31
32) → 06
33) → EE 00 20 00 00 01 21 0B 61
34) ← 06
35) ← EE 00 20 00 00 01 00 80 51
36) → 06
37) → EE 00 00 00 00 01 22 01 33
38) ← 06
39) ← EE 00 00 00 00 01 00 11 31
40) → 06
```

Figure C.1 – Packet transmission example

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

ANNEX D — Service sequence state control

Informative

PSEM communications is defined as a series of "Service Sequence States." The use of each service may be restricted to one or more states. Specific services may cause transitions between states. The transition is implemented upon positive acknowledgement of the service. The recognized states include:

Base State:	This is the state at which communication begins. At this point the default data transmission parameters apply.
ID State:	Once the metering device has been identified, this is the state that is entered.
Session State:	When a successful logon has been completed, this is the state achieved.

The relationship between PSEM services and service sequence states is:

Identification Service requests are accepted at the base state only. Acceptance of an identification service request transitions communications to the ID state. This service cannot originate from the metering device.

Wait service requests are accepted in the ID and session states. Acceptance of these requests does NOT result in any sequence state changes. This service can originate from either end of the communication channel.

Negotiate and Timing Setup service requests are accepted in the ID state only. Acceptance of these requests does NOT result in any sequence state changes. Negotiated services are NOT implemented until after acceptance. These services cannot originate from the metering device.

Logon service requests are accepted in the ID state only. Acceptance of a logon service request transitions communications to the session state. This service cannot originate from the metering device.

Security and Authenticate service requests are accepted in the session state only. Acceptance of these requests do NOT result in any sequence state changes. These services cannot originate from the metering device.

Read and write service requests are accepted in the session state only. Acceptance of these requests do NOT result in any sequence state changes. These services can originate from either end of the communication channel.

Logoff service requests are accepted in the session state only. Acceptance of a logoff service request transitions communications to the ID state. This service can originate from either end of the communication channel.

Terminate service requests are accepted in all states. Acceptance of a terminate service request transitions communications to the base state. This service can originate from either end of the communication channel.

Disconnect service requests are accepted in all states. Acceptance of a disconnect service request disconnects the communication channel. This service can originate from either end of the communication channel.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

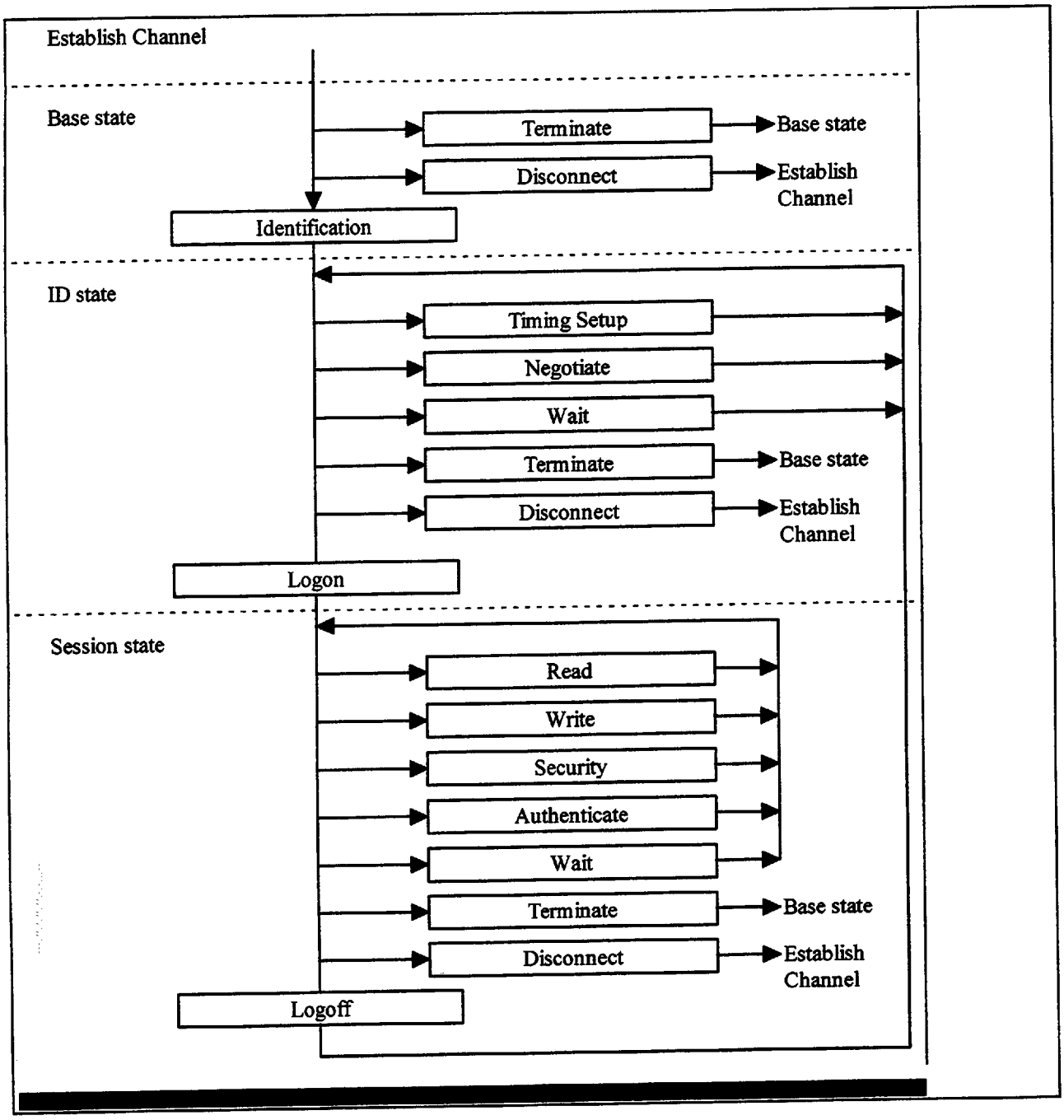


Figure D.1 – Communication state diagram

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

ANNEX E — Modifications and extensions to C12.19- 1997

Normative

This Annex describes :

E1- modification to RDATE Type
E2- modifications to Table 03, ED_MODE STATUS Table
E3- modifications to Table 07, PROCEDURE INITIATE Table
E4- Decade 90, Telephone Control Tables which are not currently in C12.19- 1997
E5- History and Event Log Codes which extend those already defined in ANNEX B "HISTORY & EVENT LOG CODES" of ANSI C12.19-1997
E6- Values assigned to the decade and decade +1 table values based upon the value of the TABLE 00 variable DEFAULT_SET_USED
E7- Values for use with the partial table access method identified as index-count. For purposes of this standard, the indices for "PARTIAL TABLE ACCESS" defined in C12.19-1997 are considered informative only.

E.1 RDATE type

This type defines a recurrent date. This recurrence can be yearly, monthly, weekly, based on a constant cycle of days or minutes.

```
TYPE RDATE = BIT FIELD OF UINT16
    MONTH                                   : UINT(0..3);
    CASE MONTH OF
        0:          PERIOD_IN_MINS          : UINT(4..15);
        1..13 :  OFFSET                 : UINT(4..7);
                    WEEKDAY             : UINT(8..10);
                    DAY                 : UINT(11..15);
        14 :        FILLER1             : FILL(4..7);
                    WEEKDAY             : UINT(8..10);
                    FILLER2             : FILL(11..15);
        15 :        PERIOD_IN_DAYS      : UINT(4..9);
                    DELTA               : UINT(10..15);
    END;
END;
```

Identifier	Value	Definition
MONTH	0	Action is repeated each **PERIOD_IN_MINS**. The starting date and time shall be supplied separately.
	1..12	Month of year
	13	Action is repeated monthly
	14	Action is repeated weekly
	15	Action is repeated each **PERIOD_IN_DAYS** referenced by an **ANCHOR_DATE** plus **DELTA**. The **ANCHOR_DATE** shall be supplied separately.
OFFSET	0	No offset
	1	Advance to **WEEKDAY** before date entered.
	2	Postpone to the first **WEEKDAY** on or after date entered.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

	3	Postpone to the second **WEEKDAY** on or after date entered.
	4	Postpone to the third **WEEKDAY** on or after date entered.
	5	Postpone to the fourth **WEEKDAY** on or after date entered.
	6	Postpone to the last **WEEKDAY** of the **MONTH** on or after date entered.
	7	Observe on date entered as well as day following date entered.
	8	Postpone to Monday if Sunday.
	9	Advance to Friday if Sunday.
	10	Postpone to Monday if Saturday.
	11	Advance to Friday if Saturday.
	12	Postpone to Monday if Sunday or Saturday.
	13	Advance to Friday if Sunday or Saturday.
	14	Postpone to Monday if Sunday, advance to Friday if Saturday.
	15	Do not observe date entered. Observe on day following date entered.
WEEKDAY	0..6	Sunday to Saturday.
	7	Unassigned
PERIOD_IN_ MINS	0..4095	0 to 4095 minutes (or 68 hours).
PERIOD_IN_DAYS	0..63	0 to 63 days.
DELTA	0..63	0 to 63 days.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

E.2 TABLE 03 ED_MODE STATUS TABLE

Table 03 Data Description

NOTE—This table definition is consistent with ANSI C12.19-1997 except for the addition of the TAMPER_DETECT_FLAG and the REVERSE_ROTATION_FLAG.

ED_MODE_STATUS_TBL (Table 03) provides the current operating mode and status of various conditions in the end device. It allows for both standard and manufacturer specific status definitions.

```
TYPE ED_MODE_BFLD = BIT FIELD OF UINT8
    METERING_FLAG                   : BOOL(0);
    TEST_MODE_FLAG                  : BOOL(1);
    METER_SHOP_MODE_FLAG            : BOOL(2);
    FILLER                          : FILL(3..7);
END;

TYPE ED_STD_STATUS1_BFLD = BIT FIELD OF UINT16
    UNPROGRAMMED_FLAG               : BOOL(0);
    CONFIGURATION_ERROR_FLAG        : BOOL(1);
    SELF_CHK_ERROR_FLAG             : BOOL(2);
    RAM_FAILURE_FLAG                : BOOL(3);
    ROM_FAILURE_FLAG                : BOOL(4);
    NONVOL_MEM_FAILURE_FLAG         : BOOL(5);
    CLOCK_ERROR_FLAG                : BOOL(6);
    MEASUREMENT_ERROR_FLAG          : BOOL(7);
    LOW_BATTERY_FLAG                : BOOL(8);
    LOW_LOSS_POTENTIAL_FLAG         : BOOL(9);
    DEMAND_OVERLOAD_FLAG            : BOOL(10);
    POWER_FAILURE_FLAG              : BOOL(11);
    TAMPER_DETECT_FLAG              : BOOL(12);
    REVERSE_ROTATION_FLAG           : BOOL(13);
    FILLER                          : FILL(14..15);
END;

TYPE ED_STD_STATUS2_BFLD = BIT FIELD OF UINT8
    FILLER                          : FILL(0..7);
END;
TYPE ED_MFG_STATUS_RCD = PACKED RECORD
    ED_MFG_STATUS                   : SET(GEN_CONFIG_TBL.DIM_MFG_STATUS_USED);
END;


TYPE ED_MODE_STATUS_RCD = PACKED RECORD
    ED_MODE                         : ED_MODE_BFLD;
    ED_STD_STATUS1                  : ED_STD_STATUS1_BFLD;
    ED_STD_STATUS2                  : ED_STD_STATUS2_BFLD;
    ED_MFG_STATUS                   : ED_MFG_STATUS_RCD;
END;

TABLE ED_MODE_STATUS_TBL = ED_MODE_STATUS_RCD;
```


Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

Identifier	Value	Definition
ED_MODE_BFLD		
METERING_FLAG	FALSE	End Device is not in Metering mode. It may or may not be accumulating measured or input quantities.
	TRUE	End Device is in Metering mode and is accumulating measured or input quantities.
TEST_MODE_FLAG	FALSE	End Device is not in Test Mode.
	TRUE	End Device is in Test Mode.
METER_SHOP_MODE_FLAG	FALSE	End device is not in Meter Shop Mode.
	TRUE	End device is in Meter Shop Mode.
ED_STD_STATUS1_BFLD		
UNPROGRAMMED_FLAG	FALSE	End Device is programmed.
	TRUE	End Device is not programmed or it is in a factory default state.
CONFIGURATION_ERROR_FLAG		
	FALSE	End Device did not detect a configuration error.
	TRUE	End Device did detect a configuration error.
SELF_CHK_ERROR_FLAG	FALSE	End Device did not detect a self check error.
	TRUE	End Device did detect a self check error.
RAM_FAILURE_FLAG	FALSE	End Device did not detect a RAM memory failure.
	TRUE	End Device did detect a RAM memory failure.
ROM_FAILURE_FLAG	FALSE	End Device did not detect a ROM memory failure.
	TRUE	End Device did detect a ROM memory failure.
NONVOL_MEM_FAILURE_FLAG	FALSE	End Device did not detect a nonvolatile memory failure.
	TRUE	End Device did detect a nonvolatile memory failure.
CLOCK_ERROR_FLAG	FALSE	End Device did not detect a clock error.
	TRUE	End Device did detect a clock error.
MEASUREMENT_ERROR_FLAG	FALSE	End Device did not detect a measurement element error.
	TRUE	End Device did detect a measurement element error.
LOW_BATTERY_FLAG	FALSE	End Device did not detect a low battery error.
	TRUE	End Device did detect a low battery error.
LOW_LOSS_POTENTIAL_FLAG		
	FALSE	End Device did not detect a potential that is below a predetermined value.
	TRUE	End Device did detect a device potential that is below a predetermined value.
DEMAND_OVERLOAD_FLAG	FALSE	End Device did not detect a demand threshold overload.
	TRUE	End Device did detect a demand threshold overload.
POWER_FAILURE_FLAG	FALSE	End Device did not detect a power failure.
	TRUE	End Device did detect a power failure.
TAMPER_DETECT_FLAG	FALSE	End Device did not detect tamper activity.
	TRUE	End Device did detect tamper activity.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

REVERSE_ROTATION_FLAG	FALSE	End Device did not detect reverse rotation.
	TRUE	End Device did detect reverse rotation.
ED_STD_STATUS2_BFLD		Standard status code bit field 2 is a place holder for future expansion.
ED_MFG_STATUS_RCD		Set containing the manufacturer specific status flags.
ED_MODE_STATUS_RCD		
ED_MODE		See **ED_MODE_BFLD**.
ED_STD_STATUS1		See **ED_STD_STATUS1_BFLD**.
ED_STD_STATUS2		See **ED_STD_STATUS2_BFLD**.
ED_MFG_STATUS		See **ED_MFG_STATUS_RCD**.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

E.3 TABLE 07 — Procedure initiate table

NOTE—This Annex describes an additional standard procedure that is not defined in ANSI C12.19-1997, Table 07.

PROCEDURE 20 Initiate an Immediate Call

This procedure causes the end device to immediately initiate a call with the phone number specified. Call windows are ignored.

TBL_PROC_NBR	20	Procedure number.
PARM_RCD		Defined below.
RESP_DATA_RCD		Not used.

Parameters

```
TYPE PARM_RCD = PACKED RECORD
    PHONE_NUMBER_INDEX      : UINT8;
END;
```

Identifier	Value	Definition
PARM_RCD		
PHONE_NUMBER_INDEX	0..6	Selection of which phone number touse for the call originating procedure. This value is an index to the PHONE_NUMBERS array in table ORIGINATE_PARAMETERS_TBL (Table 93).

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

E.4 DECADE 90—Telephone Control Tables

This decade contains tables associated with the use of a telephone modem.

TABLE 90 Dimension Telephone Table

Table 90 Data description

DIM_TELEPHONE_TBL (Table 90) specifies the maximum dimensional values for this decade.

```
TYPE TELEPHONE_FLAGS_BFLD = BIT FIELD OF UINT8
    ANSWER_FLAG                 : BOOL(0);
    S_ANCHOR_DATE_FLAG                  : BOOL(1);
    OFFHOOK_DETECT_FLAG                 : BOOL(2);
    BIT_RATE                    : UINT(3..4);
    ID_IN_PURPOSE                       : BOOL(5);
    NO_LOCKOUT_PARM             : BOOL(6);
    FILLER                      : FILL(7..7);
END;

TYPE TELEPHONE_RCD = PACKED RECORD
    TELEPHONE_FLAGS             : TELEPHONE_FLAGS_BFLD;
    NBR_ORIGINATE_WINDOWS       : UINT8;
    NBR_SETUP_STRINGS           : UINT8;
    SETUP_STRING_LENGTH                 : UINT8;
    PREFIX_LENGTH                       : UINT8;
    NBR_ORIGINATE_NUMBERS       : UINT8;
    PHONE_NUMBER_LENGTH                 : UINT8;
    NBR_RECURRING_DATES                 : UINT8;
    NBR_NON_RECURRRING_DATES            : UINT8;
    NBR_EVENTS                  : UINT8;
    NBR_WEEKLY_SCHEDULES        : UINT8;
    NBR_ANSWER_WINDOWS                  : UINT8;
    NBR_CALLER_IDS              : UINT8;
    CALLER_ID_LENGTH            : UINT8;

END;

TABLE DIM_TELEPHONE_TBL = TELEPHONE_RCD;
```

Identifier	Value	Definition
TELEPHONE_FLAGS_BFLD		
ANSWER_FLAG	FALSE	End device is not capable of answering a phone call.
	TRUE	End device is capable of answering a phone call.
S_ANCHOR_DATE_FLAG	FALSE	End device is not capable of accepting a separate telephone schedule anchor date for the Period/Delta **RDATE** type (recurring date).
	TRUE	End device is capable of accepting a separate telephone schedule anchor date for the Period/Delta **RDATE** type (recurring date).
OFFHOOK_DETECT_FLAG	FALSE	End device is not capable of detecting off hook conditions.
	TRUE	End device is capable of detecting off hook conditions.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

BIT_RATE	0	End device is not capable of modifying its bit rate.
	1	End device is capable of modifying globally its originating and answering bit rate.
	2	End device is capable of modifying separately its originating and answering bit rate.
	3	Reserved.
ID_IN_PURPOSE	**FALSE**	End device is not capable of including an ID field in **CALL_PURPOSE_TBL**.
	TRUE	End device is capable of including an ID field in **CALL_PURPOSE_TBL (Table 96)**.
NO_LOCKOUT_PARM	**FALSE**	End device is capable of supporting lockout parameters in table **ANSWER_PARAMETERS_TBL** (Table 95).
	TRUE	End device is not capable of supporting lockout parameters in table **ANSWER_PARAMETERS_TBL** (Table 95).

TELEPHONE_RCD

TELEPHONE_FLAGS		See definition above.
NBR_ORIGINATE_WINDOWS	0..255	Maximum number of originate time window descriptions supported by the end device.
NBR_SETUP_STRINGS	0..255	Maximum number of setup strings supported by the end device.
SETUP_STRING_LENGTH	0..255	Maximum length (in bytes) of setup strings supported by the end device.
PREFIX_LENGTH	0..255	Maximum length (in bytes) of phone number prefix supported by the end device.
NBR_ORIGINATE_NUMBERS	0..255	Maximum number of originate phone numbers supported by the end device.
PHONE_NUMBER_LENGTH	0..255	Maximum length (in bytes) of phone numbers supported by the end device.
NBR_RECURRING_DATES	0..255	Maximum number of recurring dates supported to create the origination schedule.
NBR_NON_RECURRRING_DATES		0..255 Maximum number of non-recurring dates supported to create the origination schedule.
NBR_EVENTS	0..255	Maximum number of events supported to create the origination schedule.
NBR_WEEKLY_SCHEDULES	0..255	Maximum number of weekly schedules supported to create the origination schedule.
NBR_ANSWER_WINDOWS	0..255	Maximum number of answer time window descriptions supported by the end device.
NBR_CALLER_IDS	0..255	Maximum number of Caller ID strings supported by the end device.
CALLER_ID_LENGTH	0..255	Maximum length (in bytes) of Caller ID strings supported by the end device.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

TABLE 91 Actual Telephone Table

Table 91 Data description

ACT_TELEPHONE_TBL (Table 91) contains actual dimension values for this decade.

TABLE ACT_TELEPHONE_TBL = DIM_TELEPHONE_TBL.TELEPHONE_RCD;

Identifier	Value	Definition
TELEPHONE_FLAGS_BFLD		
ANSWER_FLAG	FALSE	End device shall not answer a phone call.
	TRUE	End device may answer a phone call.
S_ANCHOR_DATE_FLAG	FALSE	End device does not accept a separate telephone schedule anchor date for the Period/Delta **RDATE** type (recurring date).
	TRUE	End device accepts a separate telephone schedule anchor date for the Period/Delta **RDATE** type (recurring date).
OFFHOOK_DETECT_FLAG	FALSE	End device does not currently detect off hook conditions.
	TRUE	End device currently detects off hook conditions.
BIT_RATE	0	Bit rate can not be controlled.
	1	Answering and originating bit rate can be controlled globally in **GLOBAL_PARAMETERS_TBL (Table 92).**
	2	Answering and originating bit rate can be controlled separately in **ORIGINATE_PARAMETERS_TBL (Table 93)** and **ANSWER_PARAMETERS_TBL (Table 95).**
	3	Reserved.
ID_IN_PURPOSE	**FALSE**	End device does not currently include an ID field in **CALL_PURPOSE_TBL (Table 96).**
	TRUE	End device currently includes an ID field in **CALL_PURPOSE_TBL (Table 96).**
NO_LOCKOUT_PARM	**FALSE**	End device currently includes lockout parameters in table **ANSWER_PARAMETERS_TBL** (Table 95).
	TRUE	End device does not currently include lockout parameters in table **ANSWER_PARAMETERS_TBL** (Table 95).
TELEPHONE_RCD		
TELEPHONE_FLAGS		See definition above.
NBR_ORIGINATE_WINDOWS	0..255	Actual number of originate time window descriptions used by the end device.
NBR_SETUP_STRINGS	0..255	Actual number of setup strings used by the end device.
SETUP_STRING_LENGTH	0..255	Actual length (in bytes) of setup strings used by the end device.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

PREFIX_LENGTH	0..255	Actual length (in bytes) of phone number prefix used by the end device.
NBR_ORIGINATE_NUMBERS	0..7	Actual number of originate phone numbers used by the end device.
PHONE_NUMBER_LENGTH	0..255	Actual length (in bytes) of phone numbers used by the end device.
NBR_RECURRING_DATES	0..255	Actual number of recurring dates used to create the origination schedule.
NBR_NON_RECURRRING_DATES		0..255 Actual number of non-recurring dates used to create the origination schedule.
NBR_EVENTS	0..255	Actual number of events used to create the origination schedule.
NBR_WEEKLY_SCHEDULES	0..255	Actual number of weekly schedules used to create the origination schedule.
NBR_ANSWER_WINDOWS	0..255	Actual number of answer time window descriptions used by the end device.
NBR_CALLER_IDS	0..255	Actual number of Caller ID strings used by the end device.
CALLER_ID_LENGTH	0..255	Actual length (in bytes) of Caller ID strings used by the end device.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

TABLE 92 Global Parameters Table

Table 92 Data description

GLOBAL_PARAMETERS_TBL (Table 92) defines general parameters used both for answering and originating a call.

```
TYPE SETUP_STRING_RCD = PACKED RECORD
    SETUP_STRING          : ARRAY[ACT_TELEPHONE_TBL.SETUP_STRING_LENGTH] OF CHAR;
END;

TYPE GLOBAL_PARAMETERS_RCD = PACKED RECORD
    PSEM_IDENTITY         : UINT8;
    IF ACT_TELEPHONE_TBL.BIT_RATE = 1 THEN
            BIT_RATE      : UINT32;
    END ;
    MODEM_SETUP_STRINGS       : ARRAY[ACT_TELEPHONE_TBL.NBR_SETUP_STRINGS]
                              OF SETUP_STRING_RCD;
END;

TABLE GLOBAL_PARAMETERS_TBL = GLOBAL_PARAMETERS_RCD;
```

Identifier	Value	Definition
SETUP_STRING_RCD		
SETUP_STRING		Modem setup string used to configure the modem.
GLOBAL_PARAMETERS_RCD		
PSEM_IDENTITY	0..255	Value used by the <identity> field in layer 2.
BIT_RATE		Bit rate at which the end device will originate or answer a call. Zero denotes an externally defined bit rate.
MODEM_SETUP_STRINGS		**An array of Modem Setup Strings**

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

TABLE 93 Originate Parameters Table

Table 93 Data description

ORIGINATE_PARAMETERS_TBL (Table 93) defines originate configuration parameters. Originate refers to the end device placing calls to a utility host.

```
TYPE STIME = PACKED RECORD
    CASE GEN_CONFIG_TBL.TM_FORMAT OF
            0: NIL          : NIL;
            1: HOUR         : BCD;
               MINUTE       : BCD;
            2: HOUR         : UINT8;
               MINUTE       : UINT8;
            3: D_TIME       : UINT32;
    END;
END;

TYPE DAYS_BFLD = BIT FIELD OF UINT8
    SUNDAY                  : BOOL(0);
    MONDAY                  : BOOL(1);
    TUESDAY                 : BOOL(2);
    WEDNESDAY               : BOOL(3);
    THURSDAY                : BOOL(4);
    FRIDAY                  : BOOL(5);
    SATURDAY                : BOOL(6);
    FILLER                  : FILL(7..7);
END;

TYPE WINDOW_RCD = PACKED RECORD
    BEGIN_WINDOW_TIME   : STIME;
    WINDOW_DURATION     : STIME;
    WINDOW_DAYS         : DAYS_BFLD;
END;

TYPE PHONE_NUMBER_RCD = PACKED RECORD
    PHONE_NUMBER            : ARRAY[ACT_TELEPHONE_TBL.PHONE_NUMBER_LENGTH] OF
CHAR;
END;

TYPE PHONE_NUMBERS_RCD = PACKED RECORD
    PREFIX              : ARRAY[ACT_TELEPHONE_TBL.PREFIX_LENGTH] OF CHAR;
    PHONE_NUMBERS           : ARRAY[ACT_TELEPHONE_TBL.NBR_ORIGINATE_NUMBERS]
                          OF PHONE_NUMBER_TYPE;
END;

TYPE ORIGINATE_PARAMETERS_RCD = PACKED RECORD
    IF ACT_TELEPHONE_TBL.BIT_RATE = 2 THEN
            ORIGINATE_BIT_RATE  : UINT32;
    END;
    DIAL_DELAY              : UINT8;
    ORIGINATE_PHONE_NUMBERS : PHONE_NUMBERS_RCD;
    WINDOWS                 : ARRAY[ACT_TELEPHONE_TBL.NBR_ORIGINATE_WINDOWS]
                              OF WINDOW_RCD;
END;

TABLE ORIGINATE_PARAMETERS_TBL = ORIGINATE_PARAMETERS_RCD;
```


Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

Identifier	Value	Definition
STIME		
HOUR	0..23	Hour of the day.
	24..31	Unassigned.
MINUTE	0..59	Minute of hour.
	60..63	Unassigned.
D_TIME		Time in seconds since 00:00:00 local time.
DAYS_BFLD		
SUNDAY	FALSE	This window does not start each Sunday.
	TRUE	This window does start each Sunday.
MONDAY	FALSE	This window does not start each Monday.
	TRUE	This window does start each Monday.
TUESDAY	FALSE	This window does not start each Tuesday.
	TRUE	This window does start each Tuesday.
WEDNESDAY	FALSE	This window does not start each Wednesday.
	TRUE	This window does start each Wednesday.
THURSDAY	FALSE	This window does not start each Thursday.
	TRUE	This window does start each Thursday.
FRIDAY	FALSE	This window does not start each Friday.
	TRUE	This window does start each Friday.
SATURDAY	FALSE	This window does not start each Saturday.
	TRUE	This window does start each Saturday.
WINDOW_RCD		
BEGIN_WINDOW_TIME		**The time of day the window starts.**
WINDOW_DURATION		The duration of the window.
WINDOW_DAYS		See DAYS_BFLD above. Defines the days that a window can start.
PHONE_NUMBER_RCD		
PHONE_NUMBER		Phone number to use to place a call.
PHONE_NUMBERS_RCD		
PREFIX		This number will be inserted at the beginning of every phone number before beginning dialing.
PHONE_NUMBERS		An array of originate phone numbers.
ORIGINATE_PARAMETERS_RCD		
ORIGINATE_BIT_RATE		Bit rate at which the end device will originate a call. Zero denotes an externally defined bit rate.
DIAL_DELAY	0..255	Delay in seconds the end device will wait before dialing after going off-hook.
ORIGINATE_PHONE_NUMBERS		See **PHONE_NUMBER_RCD** above.
WINDOWS		An array of call originate windows.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

TABLE 94 Originate Schedule Table

Table 94 Data description

ORIGINATE_SCHEDULE_TBL (Table 94) is used to configure when and why a metering device originates a call.

```
TYPE CTRL_BFLD = BIT FIELD OF UINT8
    PRIMARY_PHONE_NUMBER      : UINT(0..2);
    FILLER                    : FILL(3..3);
    SECONDARY_PHONE_NUMBER    : UINT(4..6);
    USE_WINDOWS               : BOOL(7);
END;

TYPE RECURRING_DATES_RCD = PACKED RECORD
    ORIGINATE_DATE            : RDATE;
    START_TIME                : TIME;
    ORIGINATE_CTRL            : CTRL_BFLD;
END;

TYPE NON_RECURRRING_DATES_RCD = PACKED RECORD
    ORIGINATE_DATE            : DATE;
    START_TIME                : TIME;
    ORIGINATE_CTRL            : CTRL_BFLD;
END;

TYPE EVENTS_RCD = PACKED RECORD
    ED_STD_STATUS1            : ED_MODE_STATUS_TBL.ED_STD_STATUS1_BFLD;
    ED_STD_STATUS2            : ED_MODE_STATUS_TBL.ED_STD_STATUS2_BFLD;
    ED_MFG_STATUS                   : ED_MODE_STATUS_TBL.ED_MFG_STATUS_RCD;
    ORIGINATE_CTRL            : CTRL_BFLD;
END;

TYPE WEEKLY_SCHEDULE_RCD = PACKED RECORD
    DAYS                      : ORIGINATE_PARAMETERS_TBL.DAYS_BFLD;
    START_TIME                : TIME;
    ORIGINATE_CTRL            : CTRL_BFLD;
END;

TYPE ORIGINATE_SCHEDULE_RCD = PACKED RECORD
    IF ACT_TELEPHONE_TBL.S_ANCHOR_DATE_FLAG THEN
        S_ANCHOR_DATE               : DATE;
    END;

    RECURRING_DATES           : ARRAY[ACT_TELEPHONE_TBL.NBR_RECURRING_DATES]
                                OF RECURRING_DATES_RCD;
    NON_RECURING_DATES              :
ARRAY[ACT_TELEPHONE_TBL.NBR_NON_RECURRRING_DATES]
                                OF NON_RECURRRING_DATES_RCD;
    EVENTS                          : ARRAY[ACT_TELEPHONE_TBL.NBR_EVENTS]
                                OF EVENTS_RCD;
    WEEKLY_SCHEDULES          : ARRAY[ACT_TELEPHONE_TBL.NBR_WEEKLY_SCHEDULES]
                                OF WEEKLY_SCHEDULE_RCD;
END;

TABLE ORIGINATE_SCHEDULE_TBL = ORIGINATE_SCHEDULE_RCD;
```


Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

Identifier	Value	Definition
CTRL_BFLD		
PRIMARY_PHONE_NUMBER	0..7	Selection of the primary phone number to use to originate the call. This value is an index into the PHONE_NUMBERS array in the ORIGINATE_PARAMETERS_TBL (Table 93). Seven (7) indicates that no call should take place.
SECONDARY_PHONE_NUMBER	0..7	Selection of the secondary phone number to use to originate a call. This value is an index into the PHONE_NUMBERS array in the ORIGINATE_PARAMETERS_TBL (Table 93). Seven (7) indicates that no secondary number is provided.
USE_WINDOWS	FALSE	An originate time window will not be used to restrict the period available to place this call. The call will be placed immediately.
	TRUE	An Originate time window will be used to restrict the period available to place this call. Calls originating outside a window will be delayed until the next open window.
RECURRING_DATES_RCD		
ORIGINATE_DATE		Recurring date at which call originate procedures should begin.
START_TIME		Time at which call originate procedures should begin.
ORIGINATE_CTRL		See CTRL_BFLD above.
NON_RECURRRING_DATES_RCD		
ORIGINATE_DATE		Specific date at which a call originate procedure should begin.
START_TIME		Time at which a call originate procedure should begin.
ORIGINATE_CTRL		See CTRL_BFLD above.
EVENTS_RCD		
ED_STD_STATUS1		Standard status events which initiate a call originate procedure.
	FALSE	No call should take place for this status bit.
	TRUE	A call should take place if an event is detected for this status bit.
ED_STD_STATUS2		Standard status events which initiate a call originate procedure.
	FALSE	No call should take place for this status bit.
	TRUE	A call should take place if an event is detected for this status bit.
ED_MFG_STATUS		Manufacturer status events which initiate a call originate procedure.
	FALSE	No call should take place for this status bit.
	TRUE	A call should take place if an event is detected for this status bit.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

ORIGINATE_CTRL	See CTRL_BFLD above.
WEEKLY_SCHEDULE_RCD	
DAYS	Days of the week in which the call originate procedure should begin.
START_TIME	Time at which the call originate procedure should begin.
ORIGINATE_CTRL	See CTRL_BFLD above.
ORIGINATE_SCHEDULE_RCD	
S_ANCHOR_DATE	A separate telephone schedule anchor date used with an **RDATE** recurring date using the **PERIOD/OFFSET** mechanism. If this table contains an anchor date it is used with RDATE.If not present, and an anchor date is provided in table 54 (**CALENDAR_TBL**) then RDATE will use that anchor date. If neither anchor date is present then the anchor date is defined by the manufacturer.
RECURRING_DATES	Array of recurring call origination dates.
NON_RECURING_DATES	Array of non recurring call origination dates.
EVENTS	Array of call origination trigger events.
WEEKLY_SCHEDULES	Array of call origination days.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

TABLE 95 Answer Parameters

Table 95 Data description

ANSWER_PARAMETERS_TBL (Table 95) defines the answer configuration for receiving calls placed by a utility host.

```
TYPE CALLER_ID_RCD = PACKED RECORD
    CALLER_ID                       : ARRAY[ACT_TELEPHONE_TBL.CALLER_ID_LENGTH] OF CHAR;
END;

TYPE ANSWER_PARAMETERS_RCD = PACKED RECORD
    IF ACT_TELEPHONE_TBL.BIT_RATE = 2 THEN
            ANSWER_BIT_RATE     : UINT32;
    END;
    IF NOT ACT_TELEPHONE_TBL.NO_LOCKOUT_PARM THEN
            LOCKOUT_DELAY               : UINT8;
            RETRY_ATTEMPTS      : UINT8;
            RETRY_LOCKOUT_TIME: UINT8;
    END;
    NBR_RINGS                       : UINT8;

    IF ACT_TELEPHONE_TBL.NBR_ANSWER_WINDOWS > 0 THEN
            NBR_RINGS_OUTSIDE   : UINT8;
    END;

    CALLER_IDS                      : ARRAY[ACT_TELEPHONE_TBL.NBR_CALLER_ID]
                                      OF CALLER_ID_RCD;
    WINDOWS                         : ARRAY[ACT_TELEPHONE_TBL.NBR_ANSWER_WINDOWS]
                                      OF ORIGINATE_PARAMETERS_TBL.WINDOW_RCD;
END;

TABLE ANSWER_PARAMETERS_TBL = ANSWER_PARAMETERS_RCD;
```

Identifier	Value	Definition
CALLER_ID_RCD		
CALLER_ID		Caller ID string. Incoming calls must have a valid caller ID for the call to be answered.
ANSWER_PARAMETERS_RCD		
ANSWER_BIT_RATE	$0..2^{32}-1$	Bit rate at which the end device will answer a call. Zero indicates an externally defined bit rate.
LOCKOUT_DELAY	0..255	Number of minutes to lockout call answering after detecting a voice call.
RETRY_ATTEMPTS	0..255	Maximum number of consecutive security violations allowed before lockout.
RETRY_LOCKOUT_TIME	0..255	Minutes to lockout logons after security retry attempts exceeded.
NBR_RINGS	0..255	The number of rings to wait before answering while inside a time window. A value of 0 means do not answer.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

NBR_RINGS_OUTSIDE	0..255	The number of rings to answer while outside a time window. A value of 0 means do not answer.
CALLER_IDS		An array of caller IDs.
WINDOWS		An array of answer time windows.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS

Not for Resale

TABLE 96 Call Purpose

Table 96 Data description

CALL_PURPOSE_TBL (Table 96) indicates the purpose for the current call originated by the end device.

```
TYPE CALL_PURPOSE_BFLD = BIT FIELD OF UINT16
    POWER_OUTAGE                    :BOOL(0);
    POWER_RESTORAL                  :BOOL(1);
    SCHEDULED_CALL                  :BOOL(2);
    STATUS_CALL                     :BOOL(3);
    IMMEDIATE_CALL                  :BOOL(4);
    FILLER                          :FILL(5..11);
    MANUFACTURER_PURPOSES           :UINT(12..15);
END;

TYPE CALL_PURPOSE_RCD = PACKED RECORD
    CALL_PURPOSE                    : CALL_PURPOSE_BFLD;
    CALL_TRIGGER_STATUS             : ED_MODE_STATUS_TBL.ED_MODE_STATUS_RCD;
    IF ACT_TELEPHONE_TBL.ID_IN_PURPOSE THEN
        IDENT                       : DEVICE_IDENT_TBL.IDENT_RCD;
    END;
END;

TABLE CALL_PURPOSE_TBL = CALL_PURPOSE_RCD;
```

Identifier	Value	Definition
CALL_PURPOSE_BFLD		
POWER_OUTAGE		This phone call was triggered by the detection of a power outage.
POWER_RESTORAL		This phone call was triggered by the detection of a power restoration.
SCHEDULED_CALL		This phone call is a normal call.
STATUS_CALL		This phone call was triggered by an change in the status record.
IMMEDIATE_CALL		This phone call was triggered by an immediate call procedure.
MANUFACTURER_PURPOSES		This field if non-zero indicates that the phone call was triggered by a manufacturer defined purpose.
CALL_PURPOSE_RCD		
CALL_PURPOSE		See definitions of CALL_PURPOSE_BFLD above.
IDENT		Identity of the call originating device (Table 5).
CALL_TRIGGER_STATUS		If the STATUS_CALL bit is TRUE then this field indicates which status bit change(s) caused the call. The field is in the image of ED_MODE_STATUS_TBL.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

TABLE 97 Call Status

Table 97 Data description

CALL_STATUS_TBL (Table 97) indicates the status of the most recent call to each defined phone number.

```
TYPE CALL_STATUS_RCD = PACKED RECORD
    CALL_STATUS_ARRAY  : ARRAY[ACT_TELEPHONE_TBL.NBR_ORIGINATE_NUMBERS]
                         OF UINT8;
END;

TABLE CALL_STATUS_TBL = CALL_STATUS_RCD;
```

Identifier	Value	Definition
CALL_STATUS__RCD		
CALL_STATUS_ARRAY		Status or disposition of the most recent call to each phone number. Entries are defined as follows:
	0	No phone call made
	1	Phone call in progress
	2	Dialing
	3	Waiting for a connection
	4	Communicating
	5	Completed normally
	6	Not completed
	7	Not completed, Line busy
	8	Not completed, No dial tone
	9	Not completed, Line cut
	10	Not completed, No connection
	11	Not completed, No modem response

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

E.5 History and event log codes

Following codes extend those already defined in ANNEX B "HISTORY & EVENT LOG CODES" of ANSI C12.19-1997 standard.

Code	Event	Argument
30	Metering mode started	None
31	Metering mode stopped	None
32	Test mode started	None
33	Test mode stopped	None
34	Meter shop mode started	None
35	Meter shop mode stopped	None
36	Meter reprogrammed	None
37	Configuration error detected	None
38	Self check error detected	None
39	RAM failure detected	None
40	ROM failure detected	None
41	Nonvolatile memory failure detected	None
42	Clock error detected	None
43	Measurement error detected	None
44	Low battery detected	None
45	Low loss potential detected	None
46	Demand overload detected	None
47	Tamper attempt detected	None
48	Reverse rotation detected	None

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

E.6 Default sets for decade tables

Addition to C12.19 ANNEX C.

The following values are assigned to the decade and decade+1 table values based upon the value of the TABLE 00 variable **DEFAULT_SET_USED**.

Variable	DEFAULT_SET_USED			
	1	2	3	4
TABLE 91				
ANSWER_FLAG	0	0	1	1
S_ANCHOR_DATE_FLAG	0	0	0	0
OFFHOOK_DETECT_FLAG	1	1	1	1
BIT_RATE	1	1	1	1
ID_IN_PURPOSE	1	1	1	1
NO_LOCKOUT_PARM	0	0	0	0
NBR_ORIGINATE_WINDOWS	2	2	2	2
NBR_SETUP_STRINGS	1	1	1	1
SETUP_STRING_LENGTH	30	30	30	30
PREFIX_LENGTH	6	6	6	6
NBR_ORIGINATE_NUMBERS	1	2	2	3
PHONE_NUMBER_LENGTH	30	30	30	30
NBR_RECURRING_DATES	0	0	0	0
NBR_NON_RECURRING_DATES	1	2	2	3
NBR_EVENTS	1	2	2	3
NBR_WEEKLY_SCHEDULES	1	2	2	3
NBR_ANSWER_WINDOWS	2	2	2	2
NBR_CALLER_IDS	0	0	0	0
CALLER_ID_LENGTH	0	0	0	0

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS Not for Resale

E.7 Indicies for partial table access

For purposes of this standard the indices for "PARTIAL TABLE ACCESS" defined in C12.19-1997 are considered informative only. The following values are for use with the partial table access method identified as index-count.

TABLE 090 - DIM_TELEPHONE_TBL								
TELEPHONE_FLAGS	0							
NBR_ORIGINATE_WINDOWS	1							
NBR_SETUP_STRINGS	2							
SETUP_STRING_LENGTH	3							
PREFIX_LENGTH	4							
NBR_ORIGINATE_NUMBERS	5							
PHONE_NUMBER_LENGTH	6							
NBR_RECURRING_DATES	7							
NBR_NON_RECURRING_DATES	8							
NBR_EVENTS	9							
NBR_WEEKLY_SCHEDULES	10							
NBR_ANSWER_WINDOWS	11							
NBR_CALLER_IDS	12							
CALLER_ID_LENGTH	13							
TABLE 091 - ACT_TELEPHONE_TBL								
TELEPHONE_FLAGS	0							
NBR_ORIGINATE_WINDOWS	1							
NBR_SETUP_STRINGS	2							
SETUP_STRING_LENGTH	3							
PREFIX_LENGTH	4							
NBR_ORIGINATE_NUMBERS	5							
PHONE_NUMBER_LENGTH	6							
NBR_RECURRING_DATES	7							
NBR_NON_RECURRING_DATES	8							
NBR_EVENTS	9							
NBR_WEEKLY_SCHEDULES	10							
NBR_ANSWER_WINDOWS	11							
NBR_CALLER_IDS	12							
CALLER_ID_LENGTH	13							
TABLE 092 - GLOBAL_PARAMETERS_TBL								
BIT_RATE	0							
PSEM_IDENTITY	1							
MODEM_SETUP_STRINGS	2							
SETUP_STRING	2	[n]						
TABLE 093 - ORIGINATE_PARAMETERS_TBL								
ORIGINATE_BIT_RATE	0							
DIAL_DELAY	1							
ORIGINATE_PHONE_NUMBERS	2							
PREFIX	2	0						
PHONE_NUMBERS	2	1						

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS Not for Resale

PHONE_NUMBER	2	1	[n]					
WINDOWS	3							
BEGIN_WINDOW_TIME	3	[n]	0					
WINDOW_DURATION	3	[n]	1					
WINDOW_DAYS	3	[n]	2					
TABLE 094 - ORIGINATE_SCHEDULING_TBL								
S_ANCHOR_DATE	0							
RECURRING_DATES	1							
RECURRING_DATES (Item)	1	[n]						
ORIGINATE_DATE	1	[n]	0					
START_TIME	1	[n]	1					
ORIGINATE_CTRL	1	[n]	2					
NON_RECURRING_DATES	2							
NON_RECURRING_DATES (Item)	2	[n]						
ORIGINATE_DATE	2	[n]	0					
START_TIME	2	[n]	1					
ORIGINATE_CTRL	2	[n]	2					
EVENTS	3							
EVENTS (Item)	3	[n]						
ED_STD_STATUS1	3	[n]	0					
ED_STD_STATUS2	3	[n]	1					
ED_MFG_STATUS	3	[n]	2					
ORIGINATE_CTRL	3	[n]	3					
WEEKLY_SCHEDULES	4							
WEEKLY_SCHEDULES (Item)	4	[n]						
DAYS	4	[n]	0					
START_TIME	4	[n]	1					
ORIGINATE_CTRL	4	[n]	2					
TABLE 095 - ANSWER_PARAMETERS_TBL								
ANSWER_BIT_RATE	0							
LOCKOUT_DELAY	1							
RETRY_ATTEMPTS	2							
RETRY_LOCKOUT_TIME	3							
NBR_RINGS	4							
NBR_RINGS_OUTSIDE	5							
CALLER_IDS	6							
CALLER_ID	6	[n]						
WINDOWS	7							
WINDOWS (Item)	7	[n]						
BEGIN_WINDOW_TIME	7	[n]	0					
WINDOW_DURATION	7	[n]	1					
WINDOW_DAYS	7	[n]	2					
TABLE 096 - CALL_PURPOSE_TBL								
CALL_PURPOSE	0							
CALL_TRIGGER_STATUS	1							
ED_MODE	1	0						
ED_STD_STATUS1	1	1						
ED_STD_STATUS2	1	2						

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS Not for Resale

ED_MFG_STATUS	1	3						
IDENTIFICATION (BCD)	1	4	0					
IDENTIFICATION (CHAR)	1	4	1					
TABLE 097 - CALL_STATUS_TBL								
CALL_STATUS_ARRAY	0							
CALL_STATUS_ARRAY (Item)	0	[n]						

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

ANNEX F — CRC examples

Informative

Trace

This example shows the actual bit manipulations performed in calculation of the frame check sequence (CRC) for an example PSEM packet in compliance with ANSI C12.18 and this document. In order to be in compliance with the CRC calculations, the bit ordering shall be consistent with this example.

```
packet without crc      = ee 00 00 00 00 01 20
                = 11101110 00000000 00000000 00000000 00000000 00000001 00100000 00000000 00000000
LSBit First       01110111 00000000 00000000 00000000 00000000 10000000 00000100 00000000 00000000
XOR FF            11111111 11111111

Start Tx        = 10001000 11111111 00000000 00000000 00000000 10000000 00000100 00000000 00000000
Apply P(x)        10001000 00010000 1
                  00000000 11101111 10000000 00000000 00000000 10000000 00000100 00000000 00000000
Apply P(x)                 10001000 00010000 1
                  00000000 01100111 10010000 10000000 00000000 10000000 00000100 00000000 00000000
Apply P(x)                  1000100 00001000 01
                  00000000 00100011 10011000 11000000 00000000 10000000 00000100 00000000 00000000
Apply P(x)                   100010 00000100 001
                  00000000 00000001 10011100 11100000 00000000 10000000 00000100 00000000 00000000
Apply P(x)                         1 00010000 00100001
                  00000000 00000000 10001100 11000001 00000000 10000000 00000100 00000000 00000000
Apply P(x)                         10001000 00010000 1
                  00000000 00000000 00000100 11010001 10000000 10000000 00000100 00000000 00000000
Apply P(x)                               100 01000000 100001
                  00000000 00000000 00000000 10010001 00000100 10000000 00000100 00000000 00000000
Apply P(x)                                   10001000 00010000 1
                  00000000 00000000 00000000 00011001 00010100 10000000 00000100 00000000 00000000
Apply P(x)                                      10001 00000010 0001
                  00000000 00000000 00000000 00001000 00010110 10000000 00000100 00000000 00000000
Apply P(x)                                        1000 10000001 00001
                  00000000 00000000 00000000 00000000 10010111 10000000 00000100 00000000 00000000
Apply P(x)                                           10001000 00010000 1
                  00000000 00000000 00000000 00000000 00011111 10000000 10000100 00000000 00000000
Apply P(x)                                              10001 00000010 0001
                  00000000 00000000 00000000 00000000 00001110 00001010 10010100 00000000 00000000
Apply P(x)                                               1000 10000001 00001
                  00000000 00000000 00000000 00000000 00000110 10001011 10011100 00000000 00000000
Apply P(x)                                                100 01000000 100001
                  00000000 00000000 00000000 00000000 00000010 11001011 00011000 00000000 00000000
Apply P(x)                                                  10 00100000 0100001
                  00000000 00000000 00000000 00000000 00000000 11101011 01011010 00000000 00000000
Apply P(x)                                                     10001000 00010000 1
                  00000000 00000000 00000000 00000000 00000000 01100011 01001010 10000000 00000000
Apply P(x)                                                      1000100 00001000 01
                  00000000 00000000 00000000 00000000 00000000 00100111 01000010 11000000 00000000
Apply P(x)                                                        100010 00000100 001
                  00000000 00000000 00000000 00000000 00000000 00000101 01000110 11100000 00000000
Apply P(x)                                                           100 01000000 100001
                  00000000 00000000 00000000 00000000 00000000 00000001 00000110 01100100 00000000
Apply P(x)                                                             1 00010000 00100001
                  00000000 00000000 00000000 00000000 00000000 00000000 00010110 01000101 00000000
Apply P(x)                                                               10001 00000010 0001
                  00000000 00000000 00000000 00000000 00000000 00000000 00000111 01000111 00010000
Apply P(x)                                                                  100 01000000 100001
                  00000000 00000000 00000000 00000000 00000000 00000000 00000011 00000111 10010100
Apply P(x)                                                                    10 00100000 0100001
                  00000000 00000000 00000000 00000000 00000000 00000000 00000001 00100111 11010110
Apply P(x)                                                                     1 00010000 00100001
                  00000000 00000000 00000000 00000000 00000000 00000000 00000000 00110111 11110111
XOR FF                                                                           11001000 00001000
MSBit First                                                                      00010011 00010000
crc                                                                              = 1310
```


Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

C Code Example

The following is an example of C code which calculates the <crc> field in a manner compliant with C12.18 section 4.7 and figure C-1. This code is provided as an example only.

```
#include <stdio.h>

unsigned short crc16(unsigned char octet, unsigned short crc);
unsigned short crc(int size, unsigned char *packet);


unsigned short crc16(unsigned char octet, unsigned short crc)
{
   int i;

   for (i = 8; i; i--)
   {
      if (crc & 0x0001)
      {
         crc >>= 1;
         if (octet & 0x01)
            crc |= 0x8000;
         crc = crc ^ 0x8408;   /* 0x1021 inverted = 1000 0100 0000 1000 */
         octet >>= 1;
      }
      else
      {
         crc >>= 1;
         if (octet & 0x01)
            crc |= 0x8000;
         octet >>= 1;
      }
   }

   return crc;
}

unsigned short crc(int size, unsigned char *packet)
{
   int i;
   unsigned short crc;

   crc = (~packet[1] << 8) | (~packet[0] & 0xFF);

   for (i=2 ; i<size; i++)
      crc = crc16(packet[i], crc);

   crc = crc16(0x00, crc);
   crc = crc16(0x00, crc);
   crc = ~crc;
   crc = crc >> 8 | crc << 8;

   return crc;
}

int main()
{
   unsigned char packet[] =  { 0xEE, 0x00, 0x00, 0x00, 0x00, 0x01, 0x20 };

   printf("Crc   = %04x \n", crc(sizeof(packet), packet));
   return 0;
}
```

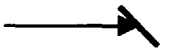
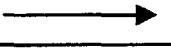

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS

Not for Resale

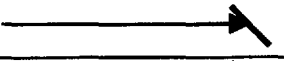
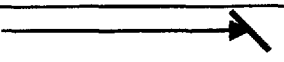
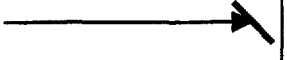
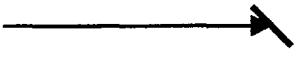
ANNEX G — Error handling

Informative

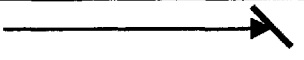
Example 1: Lost packet — Retry successful

Sender	Traffic	Receiver
Send Packet		Packet Lost
Response Timeout Retry 1		Packet Received
Ack Received		Send ACK

Example 2: Lost packet — Retry unsuccessful

Sender	Traffic	Receiver
Send Packet		Packet Lost
Response Timeout Retry 1		Packet Lost
Response Timeout Retry 2		Packet Lost
Response Timeout Retry 3		Packet Lost
Go to Base State		

Example 3: Corrupted packet — Retry successful

Sender	Traffic	Receiver
Send Packet		Packet Corrupted
NAK received		Send NAK
Retry 1		Packet Received
ACK received		Send ACK

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

Example 4: Corrupted packet — Retry unsuccessful

Sender	Traffic	Receiver
Send Packet	→	Packet Corrupted
NAK received	←	Send NAK
Retry 1	→	Packet Corrupted
NAK received	←	Send NAK
Retry 2	→	Packet Corrupted
NAK received	←	Send NAK
Retry 3	→	Packet Corrupted
NAK received	←	Send NAK
Go to Base State		

Example 5: Lost ACK

Sender	Traffic	Receiver
Send Packet	→	Packet Received
ACK Lost	←	Send ACK
Response Timeout Retry 1	→	Duplicate Packet Received Discard Packet
Ack Received	←	Send ACK

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

ANNEX H — Data encryption standard

Normative

This annex gives an implementation example of the Data Encryption Standard (ANSI Std X3.92-1981) and explains the way to use it with ANSI C12.21 services.

Usage

The identification service is used to notify the initiator of the communication of the usage of authentication algorithm. The first step done by the initiator is the transmission of an identification request to the target. The target responds with 0 as <auth_alg_id> and a random value as <ticket>.

To be authenticated, the initiator transmits an authenticate request with the key id selected and an encrypted value. This value is the result of applying the DES algorithm to the <ticket> and the selected key.

The target will decrypt this value with its own key and validate the result. If this validation fails, an <isc> is returned in the authenticate response. Otherwise this encrypted value is used as a ticket to authenticate itself. This ticket is encrypted to produce an other encrypt value. The result is then transmitted back to the initiator in the authenticate response.

The following definitions clarify the use of the identification and authentication service.

In Identification response

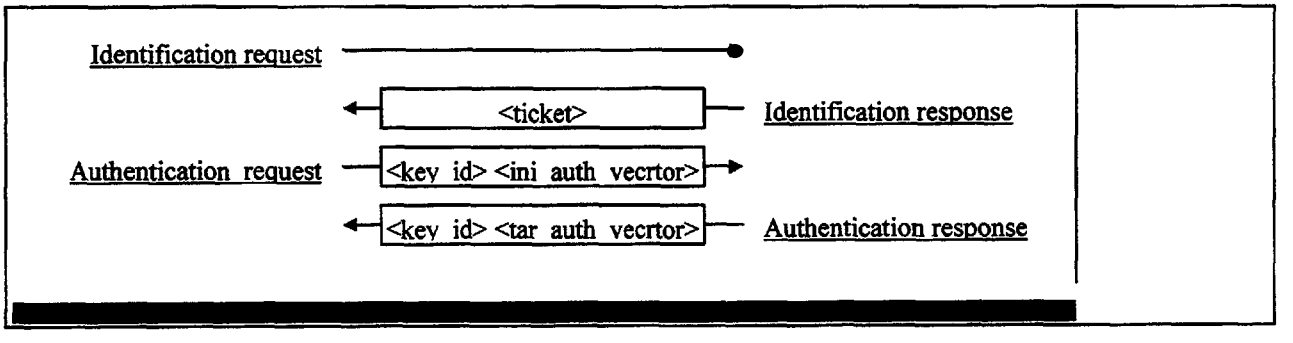
<auth_algm_id>	::=	00	
<ticket>	::=	<byte>*	{8 bytes random value}

In Authentication request

<auth_request>	::=	<key id> <ini_auth_vector>	
<key_id>	::=	<byte>	{Index of the key selected.}
<ini_auth_vector>	::=	<byte>*	{8 bytes, <ticket> encrypted.}

In Authentication response

<auth_response>	::=	<key id> <tar_auth_vector>	
<key_id>	::=	<byte>	{Index of the key selected.}
<tar_auth_vector>	::=	<byte>*	{8 bytes, <ini_auth vector> encrypted.}



Legal issues

Cryptographic devices implementing this standard may be covered by U.S. and foreign patents issued to the International Business Machines Corporation. However, IBM has granted nonexclusive, royalty-free

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

licenses under the patents to make, use, and sell apparatus which complies with the standard. The terms, conditions, and scope of the license are set out in notices published in the May 13, 1975 and August 31, 1976 issues of the Official Gazette of the United States Patent and Trademark Office (9434 O"G" 452 and 949 O.G. 1717).

Implementation

The Data Encryption Standard (DES) algorithm, adopted by the U.S. government in 1977, is a block cipher that transforms 64-bit data blocks under a 56-bit secret key, by means of permutation and substitution. The following is a description of how to use the DES algorithm to encrypt one 64-bit block.

Step 1

Get a 64-bit key.

Step 2

Perform the following permutation on the 64-bit key. The most significant bit of each bytes is discarded, reducing the key to 56 bits. Bit 1 of the permuted block is bit 57 of the original key, bit 2 is bit 49, and so on with bit 56 being bit 4 of the original key.

```
57 49 41 33 25 17  9  1 58 50 42 34 26 18
10  2 59 51 43 35 27 19 11  3 60 52 44 36
63 55 47 39 31 23 15  7 62 54 46 38 30 22
14  6 61 53 45 37 29 21 13  5 28 20 12  4
```

Split the permuted key into two halves. The first 28 bits are called C[0] and the last 28 bits are called D[0].

Start with i = 1.

Step 3, 4

Perform one or two circular left shifts on both C[i-1] and D[i-1] to get C[i] and D[i], respectively. The number of shifts per iteration are given in the table below.

```
Iteration #   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
Left Shifts   1  1  2  2  2  2  2  2  1  2  2  2  2  2  2  1
```

Step 5

Permute the concatenation C[i]D[i] as indicated below. This will yield K[i], which is 48 bits long.

```
14 17 11 24  1  5  3 28 15  6 21 10
23 19 12  4 26  8 16  7 27 20 13  2
41 52 31 37 47 55 30 40 51 45 33 48
44 49 39 56 34 53 46 42 50 36 29 32
```

Loop back to Step 3 until K[16] has been calculated.

Step 6

Get a 64-bit data block. If the block is shorter than 64 bits, it should be padded as appropriate for the application.

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

Step 7

Perform the following permutation on the data block.

```
58 50 42 34 26 18 10 2 60 52 44 36 28 20 12 4
62 54 46 38 30 22 14 6 64 56 48 40 32 24 16 8
57 49 41 33 25 17  9 1 59 51 43 35 27 19 11 3
61 53 45 37 29 21 13 5 63 55 47 39 31 23 15 7
```

Split the block into two halves. The first 32 bits are called L[0], and the last 32 bits are called R[0].

Start with i = 1.

Step 8

Expand the 32-bit R[i-1] into 48 bits according to the bit-selection function below.

```
32  1  2  3  4  5  4  5  6  7  8  9
 8  9 10 11 12 13 12 13 14 15 16 17
16 17 18 19 20 21 20 21 22 23 24 25
24 25 26 27 28 29 28 29 30 31 32  1
```

Step 9

Exclusive-or E(R[i-1]) with K[i].

Step 10

Break E(R[i-1]) xor K[i] into eight 6-bit blocks. Bits 1-6 are B[1], bits 7-12 are B[2], and so on with bits 43-48 being B[8].

Substitute the values found in the S-boxes for all B[j]. Start with j = 1. All values in the S-boxes should be considered 4 bits wide. Take the 1st and 6th bits of B[j] together as a 2-bit value indicating the row in S[j]. Take the 2nd through 5th bits of B[j] together as a 4-bit value indicating the column in S[j].

```
S[1]
14  4 13  1  2 15 11  8  3 10  6 12  5  9  0  7
 0 15  7  4 14  2 13  1 10  6 12 11  9  5  3  8
 4  1 14  8 13  6  2 11 15 12  9  7  3 10  5  0
15 12  8  2  4  9  1  7  5 11  3 14 10  0  6 13

S[2]
15  1  8 14  6 11  3  4  9  7  2 13 12  0  5 10
 3 13  4  7 15  2  8 14 12  0  1 10  6  9 11  5
 0 14  7 11 10  4 13  1  5  8 12  6  9  3  2 15
13  8 10  1  3 15  4  2 11  6  7 12  0  5 14  9

S[3]
10  0  9 14  6  3 15  5  1 13 12  7 11  4  2  8
13  7  0  9  3  4  6 10  2  8  5 14 12 11 15  1
13  6  4  9  8 15  3  0 11  1  2 12  5 10 14  7
 1 10 13  0  6  9  8  7  4 15 14  3 11  5  2 12

S[4]
 7 13 14  3  0  6  9 10  1  2  8  5 11 12  4 15
13  8 11  5  6 15  0  3  4  7  2 12  1 10 14  9
10  6  9  0 12 11  7 13 15  1  3 14  5  2  8  4
 3 15  0  6 10  1 13  8  9  4  5 11 12  7  2 14
```


Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

```
S[5]
 2 12  4  1  7 10 11  6  8  5  3 15 13  0 14  9
14 11  2 12  4  7 13  1  5  0 15 10  3  9  8  6
 4  2  1 11 10 13  7  8 15  9 12  5  6  3  0 14
11  8 12  7  1 14  2 13  6 15  0  9 10  4  5  3

S[6]
12  1 10 15  9  2  6  8  0 13  3  4 14  7  5 11
10 15  4  2  7 12  9  5  6  1 13 14  0 11  3  8
 9 14 15  5  2  8 12  3  7  0  4 10  1 13 11  6
 4  3  2 12  9  5 15 10 11 14  1  7  6  0  8 13

S[7]
 4 11  2 14 15  0  8 13  3 12  9  7  5 10  6  1
13  0 11  7  4  9  1 10 14  3  5 12  2 15  8  6
 1  4 11 13 12  3  7 14 10 15  6  8  0  5  9  2
 6 11 13  8  1  4 10  7  9  5  0 15 14  2  3 12

S[8]
13  2  8  4  6 15 11  1 10  9  3 14  5  0 12  7
 1 15 13  8 10  3  7  4 12  5  6 11  0 14  9  2
 7 11  4  1  9 12 14  2  0  6 10 13 15  3  5  8
 2  1 14  7  4 10  8 13 15 12  9  0  3  5  6 11
```

Step 11

Permute the concatenation of B[1] through B[8] as indicated below.

```
16  7 20 21 29 12 28 17
 1 15 23 26  5 18 31 10
 2  8 24 14 32 27  3  9
19 13 30  6 22 11  4 25
```

Step 12

Exclusive-or the resulting value with L[i-1]. Thus, all together, your R[i] = L[i-1] x or P(S[1](B[1])...S[8](B[8])), where B[j] is a 6-bit block of E(R[i-1]) x or K[i].

Step 13

L[i] = R[i-1].

Loop back to Step 8 until K[16] has been applied.

Step 14

Perform the following permutation on the block R[16]L[16].

```
40  8 48 16 56 24 64 32 39  7 47 15 55 23 63 31
38  6 46 14 54 22 62 30 37  5 45 13 53 21 61 29
36  4 44 12 52 20 60 28 35  3 43 11 51 19 59 27
34  2 42 10 50 18 58 26 33  1 41  9 49 17 57 25
```

To decrypt, use the same process, but just use the keys K[i] in reverse order. That is, instead of applying K[1] for the first iteration, apply K[16], and then K[15] for the second, on down to K[1].

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

Code example

The following is an example of C code which encrypts a data block of 64 bits using a key of 56 bits. This code is provided as an example only, and is not required for compliance with the Data Encryption Standard.

```
#include <stdio.h>
typedef unsigned char uint8;

static uint8   key[64] = {
   1,1,0,0,1,1,0,1,0,0,1,1,1,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,1,1,1,1,1,1,
   1,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,1,0,0,0,1,1,0,0,0,0,0,1,1,1,0
};

static uint8   data[64] = {
   1,1,0,0,1,1,0,1,0,0,1,1,1,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,1,1,1,1,1,1,
   1,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,1,0,0,0,1,1,0,0,0,0,0,1,1,1,0
};

static uint8   perm1[56] = {
   57, 49, 41, 33, 25, 17,  9,  1, 58, 50, 42, 34, 26, 18,
   10,  2, 59, 51, 43, 35, 27, 19, 11,  3, 60, 52, 44, 36,
   63, 55, 47, 39, 31, 23, 15,  7, 62, 54, 46, 38, 30, 22,
   14,  6, 61, 53, 45, 37, 29, 21, 13,  5, 28, 20, 12, 4
};

static uint8   perm2[56] = {
    2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15,
   16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28,  1,
   30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43,
   44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 29
};

static uint8   perm3[48] = {
   14, 17, 11, 24,  1,  5,  3, 28, 15,  6, 21, 10, 23, 19, 12,  4,
   26,  8, 16,  7, 27, 20, 13,  2, 41, 52, 31, 37, 47, 55, 30, 40,
   51, 45, 33, 48, 44, 49, 39, 56, 34, 53, 46, 42, 50, 36, 29, 32
};

static uint8   perm4[64] = {
   58, 50, 42, 34, 26, 18, 10,  2, 60, 52, 44, 36, 28, 20, 12,  4,
   62, 54, 46, 38, 30, 22, 14,  6, 64, 56, 48, 40, 32, 24, 16,  8,
   57, 49, 41, 33, 25, 17,  9,  1, 59, 51, 43, 35, 27, 19, 11,  3,
   61, 53, 45, 37, 29, 21, 13,  5, 63, 55, 47, 39, 31, 23, 15,  7,
};

static uint8   perm5[48] = {
   32,  1,  2,  3,  4,  5,  4,  5,  6,  7,  8,  9,  8,  9, 10, 11,
   12, 13, 12, 13, 14, 15, 16, 17, 16, 17, 18, 19, 20, 21, 20, 21,
   22, 23, 24, 25, 24, 25, 26, 27, 28, 29, 28, 29, 30, 31, 32,  1,
};

static uint8   perm6[32] = {
   16,  7, 20, 21, 29, 12, 28, 17,  1, 15, 23, 26,  5, 18, 31, 10,
    2,  8, 24, 14, 32, 27,  3,  9, 19, 13, 30,  6, 22, 11,  4, 25,
};

static uint8   perm7[64] = {
   40,  8, 48, 16, 56, 24, 64, 32, 39,  7, 47, 15, 55, 23, 63, 31,
   38,  6, 46, 14, 54, 22, 62, 30, 37,  5, 45, 13, 53, 21, 61, 29,
   36,  4, 44, 12, 52, 20, 60, 28, 35,  3, 43, 11, 51, 19, 59, 27,
   34,  2, 42, 10, 50, 18, 58, 26, 33,  1, 41,  9, 49, 17, 57, 25,
};
```


Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

```
static uint8    sboxes[8][64] = {
{
   14,4,13,1,2,15,11,8,3,10,6,12,5,9,0,7,0,15,7,4,14,2,13,1,10,6,12,11,9,5,3,8,
   4,1,14,8,13,6,2,11,15,12,9,7,3,10,5,0,15,12,8,2,4,9,1,7,5,11,3,14,10,0,6,13,
},{
   15,1,8,14,6,11,3,4,9,7,2,13,12,0,5,10,3,13,4,7,15,2,8,14,12,0,1,10,6,9,11,5,
   0,14,7,11,10,4,13,1,5,8,12,6,9,3,2,15,13,8,10,1,3,15,4,2,11,6,7,12,0,5,14,9,
},{
   10,0,9,14,6,3,15,5,1,13,12,7,11,4,2,8,13,7,0,9,3,4,6,10,2,8,5,14,12,11,15,1,
   13,6,4,9,8,15,3,0,11,1,2,12,5,10,14,7,1,10,13,0,6,9,8,7,4,15,14,3,11,5,2,12,
},{
   7,13,14,3,0,6,9,10,1,2,8,5,11,12,4,15,13,8,11,5,6,15,0,3,4,7,2,12,1,10,14,9,
   10,6,9,0,12,11,7,13,15,1,3,14,5,2,8,4,3,15,0,6,10,1,13,8,9,4,5,11,12,7,2,14,
},{
   2,12,4,1,7,10,11,6,8,5,3,15,13,0,14,9,14,11,2,12,4,7,13,1,5,0,15,10,3,9,8,6,
   4,2,1,11,10,13,7,8,15,9,12,5,6,3,0,14,11,8,12,7,1,14,2,13,6,15,0,9,10,4,5,3,
},{
   12,1,10,15,9,2,6,8,0,13,3,4,14,7,5,11,10,15,4,2,7,12,9,5,6,1,13,14,0,11,3,8,
   9,14,15,5,2,8,12,3,7,0,4,10,1,13,11,6,4,3,2,12,9,5,15,10,11,14,1,7,6,0,8,13,
},{
   4,11,2,14,15,0,8,13,3,12,9,7,5,10,6,1,13,0,11,7,4,9,1,10,14,3,5,12,2,15,8,6,
   1,4,11,13,12,3,7,14,10,15,6,8,0,5,9,2,6,11,13,8,1,4,10,7,9,5,0,15,14,2,3,12,
},{
   13,2,8,4,6,15,11,1,10,9,3,14,5,0,12,7,1,15,13,8,10,3,7,4,12,5,6,11,0,14,9,2,
   7,11,4,1,9,12,14,2,0,6,10,13,15,3,5,8,2,1,14,7,4,10,8,13,15,12,9,0,3,5,6,11,
}};

static uint8    keys[16][48];

/*****************************************************************************/
void            Permutation(uint8 *dst, uint8 *src, uint8 lgn, uint8 *perm_table)
{
   uint8          tmp[64];

   if (src == NULL)
   {
       src = tmp;
       memcpy(src, dst, 64);
   }

   for (; lgn > 0; lgn--, dst++, perm_table++)
      *dst = src[*perm_table - 1];
}

/*****************************************************************************/
void            Xor(uint8 *dst, uint8 *src, uint8 lgn)
{
   for (; lgn > 0; lgn--, dst++, src++)
   *dst ^= *src;
}

/*****************************************************************************/
void            Copy(uint8 *dst, uint8 *src, uint8 lgn)
{
   for (; lgn > 0; lgn--, dst++, src++)
   *dst = *src;
}

/*****************************************************************************/
void            SBoxes(uint8 *dst, uint8 *src, uint8 *sbox)
{
   int         i;

   i = src[4];
   i |= src[3] << 1;
   i |= src[2] << 2;
   i |= src[1] << 3;
   i |= src[5] << 4;
   i |= src[0] << 5;
```


Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

```
    i = sbox[i];

    dst[3] = i & 1;
    dst[2] = i >> 1 & 1;
    dst[1] = i >> 2 & 1;
    dst[0] = i >> 3 & 1;
}

/*****************************************************************/
void        des(uint8 *_key, uint8 *_data, int _encrypt)
{
    uint8       key[64], data[64], right[48];
    int         i, j;

    Permutation(key, _key, 56, perm1);

    for (i = 1; i <= 16; i++)
    {
        Permutation(key, NULL, 56, perm2);

        if (i != 1 && i != 2 && i != 9 && i != 16)
            Permutation(key, NULL, 56, perm2);

        Permutation(keys[_encrypt ? i - 1 : 16 - i], key, 48, perm3);
    }

    Permutation(data, _data, 64, perm4);

    for (i = 1; i <= 16; i++)
    {
        Permutation(right, data + 32, 48, perm5);
        Xor(right, keys[i - 1], 48);

        for (j = 0; j < 8; j++)
            SBoxes(right + 4 * j, right + 6 * j, sboxes[j]);

        Permutation(right, NULL, 32, perm6);
        Xor(right, data, 32);
        Copy(data, data + 32, 32);
        Copy(data + 32, right, 32);
    }

    Copy(_data, data + 32, 32);
    Copy(_data + 32, data, 32);
    Permutation(_data, NULL, 64, perm7);
}

/*****************************************************************/
int main()
{
    des(key, data, 1);
    des(key, data, 0);
    return 0;
}
```


Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

Trace example

This example shows the bit manipulations performed in each step in compliance with the Data Encryption Standard.

```
Initialize keys
        Step 1: Key = 1100110100111000111110000001111110000000111111100010001100001110
        Step 2: Permutation = 0011010100100101011001100010111010001010100110101111111 0
  Step 3: Rotation = 0110101001001010110011000100110100010101001101011111101
   Step 5: keys[1] = 0100011010110000100100110100001110011101011 01111
  Step 3: Rotation = 1101010010010101100110001000101000101010011010111111011
   Step 5: keys[2] = 1100100001100011001001010111110011010111010110 10
  Step 3: Rotation = 1010100100101011001100010001010001010100110101111111 0111
  Step 4: Rotation = 0101001001010110011000100011100010101001101011111110 1110
   Step 5: keys[3] = 1000000110011111000110011001110111110100011010 10
  Step 3: Rotation = 1010010010101100110001000110000101010011010111111101 1101
  Step 4: Rotation = 0100100101011001100010001101001010100110101111111011 1010
   Step 5: keys[4] = 0100010100110010111000111110110011111110011000 00
   ...
   Skipping keys[5] to keys[15]
   ...
  Step 3: Rotation = 0011010100100101011001100010111010001010100110101111111 0
  Step 5: keys[16] = 1010001001001101011010001111111011101110010010 10

Process data block
     Step 6: Data = 1100110100111000111110000001111110000000111111100010001100001110
 Step 7: Permutation = 0010010100101110101010010100100100110101011001101010111111101000
 Step 8: Permutation = 0001101010101011000011010101010111111111101010000
      Step 9: Xor = 0101110000011011100111100001011001100010001111 11
   Step 10: SBoxes = 1011001100001111001001011111 1011
Step 11: Permutation = 1100101011010010011111101100 1011
     Step 12: Xor = 1110111111111100110101111000 0010
    Step 13: Copy = 0011010101100110101011111110100011101111111111001101011110000010
 Step 8: Permutation = 0111010111111111111111001011010101111110000000101
      Step 9: Xor = 1011110110011100110111000001011000101011010111 11
   Step 10: SBoxes = 0111011011110100001011101010 0010
Step 11: Permutation = 0101010000100011100101111001 1111
     Step 12: Xor = 0110000101000101001110000111 0111
    Step 13: Copy = 1110111111111100110101111000001001100001010001010011100001110111
 Step 8: Permutation = 1011000000101010000010101001111100000011101 01110
      Step 9: Xor = 0011000110110101000100110000001011110111110 00100
   Step 10: SBoxes = 1011100111000111001010100110 1000
Step 11: Permutation = 1001100011111001010101111000 0010
     Step 12: Xor = 0111011100000101100000000000 0000
    Step 13: Copy = 0110000101000101001110000111011101110111000001011000000000000000
 Step 8: Permutation = 0011101011101000000010111100000000000000000000000
      Step 9: Xor = 0111111111011010111010000010110011111110011000 00
   Step 10: SBoxes = 1000111010011100011101011110 0111
Step 11: Permutation = 0110010010011110001111011111 1001
     Step 12: Xor = 0000010111011011000001011000 1110
    Step 13: Copy = 0111011100000101100000000000000000000010111011011000001011000 1110
   ...
   Skipping round 5 to 15
   ...
 Step 8: Permutation = 0110101011010111010100101010101001111111101 01101
      Step 9: Xor = 1100100010011010001110100101010010010001111 00111
   Step 10: SBoxes = 1100111110000010111101110111 0111
Step 11: Permutation = 0110001111111110111011011011 1000
     Step 12: Xor = 0110111011011011011001000100 0010
    Step 13: Copy = 1101011011101001010100111111011001101110110110110110010001000010
   Step 14: Result = 0011100011011011110001100111000010011010011001101111111110110010
```


Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale

ANNEX I — I Command operational description

Informative

This annex describes the "I command" currently used in many heterogeneous protocol environments. This annex is informative and is up to each manufacturer to implement or not.

The problem

End devices that are calling into a host application, may not all support the ANSI C12.21 protocol. This makes it extremely difficult for the host application to initiate a communication session with the end device if the protocol is not known.

The solution

When an end device calls into a host application and the communication channel is established, the host application will transmit a single ASCII uppercase "I." The end device will respond with a 15 character ASCII string that identifies the protocol. This string may be followed by an optional carriage return. From this information, the host application will be able to properly perform a communication session with the meter.

The "I" command does not use the data link layer as described in this standard. The meter will only recognize this command during the initial portion of a call-originate procedure. It will not respond to the "I" command after an ANSI C12.21 session is established.

Request:

<I_command> ::= 49_H

Response:

<I_command_r> ::= <byte>$^{+15}$ [$0D_H$] {ASCII text "PSEM "}

Copyright National Electrical Manufacturers Association
Provided by IHS under license with NEMA
No reproduction or networking permitted without license from IHS
Not for Resale