

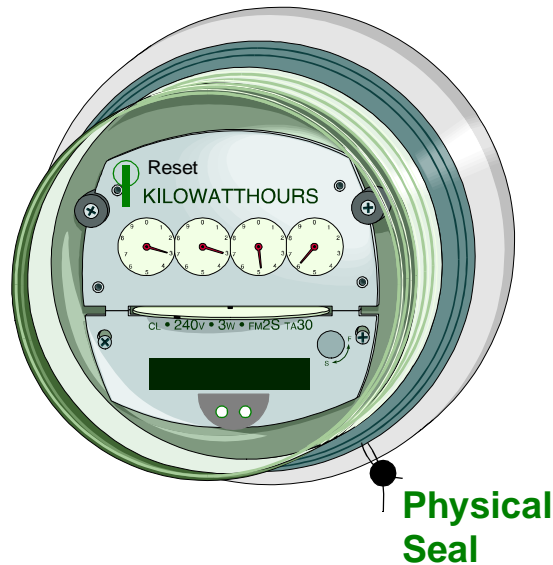
# AUDIT TRAIL IMPLEMENTATION GUIDE FOR ANSI C12.19 / IEEE 1377, UTILITY INDUSTRY STANDARD TABLES

---

This is the guide for implementing Measurement Canada “Interim Specifications (/ Procedures) Relating to Event Loggers for Electricity Metering Devices and Systems” , IS-E-01-E / IP-E-01-E and PS-EGMV-XX-E, for re-programming ANSI C12.19 / IEEE 1377 Standard based metering devices, which operate an event logger or event counters.

---

## Approved & Verified Electricity Meter



---

**Written By:** Dr. Avygdor Moise, Future DOS Research & Development Inc.  
Chairperson of the Data Communication Working Group of the Measurement Canada Task  
force on Data Communications Protocol for Electronic Metering Devices.

**C12.19 Technical submissions and recommendations by**

Michel Veillelte, Nertec Design Inc.  
Editor, ANSI Standard C12.22 subcommittee, Protocol Specification for Interfacing to Data  
Communication Networks.

**Version** 2.0

**Date:** March 21, 2002

**Contributors:** Members of the Measurement Canada Task Force for Data Communication Protocol,  
Members of the ANSI ASC12 Subcommittee 17,  
Members of AMRA/IEEE SCC31 End Device/TIU Subcommittee,  
Measurement Canada "Sealing Team".

# TABLE OF CONTENT

---

---

## CHAPTER 1

### AUDIT TRAIL IMPLEMENTATION SPECIFICATION FOR ANSI C12.19 / IEEE 1377 METERS

|        |  |    |
|--------|--|----|
| 1.1    | Introduction   | 7  |
| 1.2    | Audit Trail Implementation Models for Verified, Approved Sealed Meters   | 8  |
| 1.2.1  | Special Considerations   | 9  |
| 1.2.2  | Meters Implementing Event Counters   | 10 |
| 1.2.3  | Meters Implementing an Event Logger  | 15 |
| 1.3    | Rules for Creating, Managing and Maintaining Embedded Event Loggers  | 18 |
| 1.3.1  | Rule for Table Mapping to Event Logger Parameters  | 18 |
| 1.3.2  | Rule for Identifying the Embedded Event Logger Tables  | 19 |
| 1.3.3  | Rule for Creating an Entry In the Embedded Event Log Table   | 25 |
| 1.3.4  | Rule for Identifying Event Log Triggers  | 26 |
| 1.3.5  | Rule for Setting Event Logger List Size  | 26 |
| 1.3.6  | Rule for Overwriting Event Logger Entries  | 26 |
| 1.3.7  | Rule for Reserving Event Logger Entries  | 27 |
| 1.3.8  | Rule for Event Logger List Management  | 28 |
| 1.3.9  | Rule for Embedded Event Log Entry Format   | 30 |
| 1.3.10 | Rule for Option 1, EVENT_CODE and EVENT_ARGUMENT Usage in Self-contained Loggers                               | 33 |
| 1.3.11 | Rule for Option 2, EVENT_CODE and EVENT_ARGUMENT Usage by Signed Downloadable Event Loggers With No NEW_VALUES | 36 |
| 1.3.12 | Rule for Option 3, EVENT_CODE and EVENT_ARGUMENT Usage by Signed Event Loggers With NEW_VALUES                 | 38 |

---

---

|        |   |    |
|--------|---|----|
| 1.3.13 | Rule for Verification Event                                 | 42 |
| 1.3.14 | Rule for Issuing a Re-verification Event                    | 42 |
| 1.3.15 | Rule for Downloading The Event Logger                       | 43 |
| 1.3.16 | Rule for Updating Last Read Entry or Resetting the Pointers | 43 |
| 1.4    | Event Logger Implementation Algorithm                       | 44 |
| 1.4.1  | Event Signature Computation - EVENT_CHECK_SIG               | 44 |
| 1.4.2  | Event Argument Construction - EVENT_ARGUMENT                | 47 |
| 1.4.3  | Event Log Entry Construction, a Complete Example            | 50 |

---

## CHAPTER 2

### ANSI C12.19 / IEEE 1377

#### UTILITY INDUSTRY STANDARD METROLOGICAL TABLES

|        |  |    |
|--------|--|----|
| 2.1    | Introduction   | 53 |
| 2.2    | The Utility Standard Tables, in a nut shell                              | 54 |
| 2.2.1  | Decade 0 - End Device Configuration, Identification and Procedure Tables | 55 |
| 2.2.2  | Decade 1 - Sensor Data Source Selection Tables                           | 55 |
| 2.2.3  | Decade 2 - Metrological Data Register Tables                             | 56 |
| 2.2.4  | Decade 4 - Access and Change Security Tables                             | 56 |
| 2.2.5  | Decade 5 - Clock, Calendar, Time and Time of Use Tables                  | 56 |
| 2.2.6  | Decade 6 - Load Profile Control and Recorded Tables                      | 56 |
| 2.2.7  | Decade 7 - History Log and Event Log                                     | 56 |
| 2.2.8  | Decade 8 - User Defined (selections) Tables                              | 57 |
| 2.2.9  | Decade 9 - Telephone Control   | 57 |
| 2.2.10 | Communication Protocols  | 57 |
| 2.3    | Metrological Tables  | 60 |
| 2.3.1  | ANSI C12.19 / IEEE 1377 Metrological Tables Identification Criteria      | 60 |
| 2.3.2  | The ANSI C12.19 / IEEE 1377 Metrological Tables                          | 61 |

---

## CHAPTER 3

### DECADE 7 - HISTORY AND EVENT LOG TABLES

|       |                                       |    |
|-------|---------------------------------------|----|
| 3.1   | Overview                              | 69 |
| 3.2   | Std Table 70, Limiting Log Dimensions | 71 |
| 3.2.1 | Description                           | 71 |
| 3.2.2 | Synopsis                              | 71 |
| 3.2.3 | Data Definition                       | 71 |

3.3 Std Table 71, Actual Log Dimensions ..... 73

    3.3.1 Description ..... 73

    3.3.2 Synopsis ..... 73

    3.3.3 Data Definition ..... 73

3.4 Std Table 72, Event Identification ..... 75

    3.4.1 Description ..... 75

    3.4.2 Synopsis ..... 75

    3.4.3 Data Definition ..... 75

3.5 Std Table 73, History Log Control ..... 76

    3.5.1 Description ..... 76

    3.5.2 Synopsis ..... 76

    3.5.3 Data Definition ..... 76

3.6 Std Table 74, History Log Data ..... 78

    3.6.1 Description ..... 78

    3.6.2 Synopsis ..... 78

    3.6.3 Data Definition ..... 79

3.7 Std Table 75, Event Log Control ..... 81

    3.7.1 Description ..... 81

    3.7.2 Synopsis ..... 81

    3.7.3 Data Definition ..... 81

3.8 Std Table 76, Event Log Data ..... 83

    3.8.1 Description ..... 83

    3.8.2 Synopsis ..... 83

    3.8.3 Data Definition ..... 84

3.9 Limiting Log Dimensions (Proposed Table 70, Std Rev 2.0) ..... 86

    3.9.1 Description ..... 86

    3.9.2 Synopsis ..... 86

    3.9.3 Data Definition ..... 87

3.10 Actual Log Dimensions (Proposed Table 71, Std Rev 2.0) ..... 88

    3.10.1 Description ..... 88

    3.10.2 Synopsis ..... 88

    3.10.3 Data Definition ..... 88

3.11 Event Log and Signatures Enable Table (Proposed Std Table 77, Rev 2.0) ..... 90

    3.11.1 Description ..... 90

    3.11.2 Synopsis ..... 90

    3.11.3 Data Definition ..... 91

3.12 Signatures Table (Proposed Std Table 78, Rev 2.0) ..... 92

    3.12.1 Description ..... 92

    3.12.2 Synopsis ..... 92

    3.12.3 Data Definition ..... 93

---

---

## **Appendix A**

### **ANSI STANDARD**

#### **C12.19 HISTORY LOG CODES**

|     |   |     |
|-----|---|-----|
| A.1 | ANSI Standard C12.19-1997 History Log Codes                   | I   |
| A.2 | ANSI Standard C12.21-1999 Extensions to the History Log Codes | III |

---

## **Appendix B**

### **EVENT LOG CODES**

|       |  |     |
|-------|--|-----|
| B.1   | Event Log Codes for Use by Canadian Industry         | V   |
| B.1.1 | Event Codes for Self-contained Event Loggers         | V   |
| B.1.2 | Event Codes for Signed Event Logs with no New Values | VI  |
| B.1.3 | Event Codes for Signed Event Logs with New Values    | VII |

---

## **Appendix C**

### GLOSSARY

---

## **Appendix D**

### REFERENCES

---

# CHAPTER

# 1

## **AUDIT TRAIL IMPLEMENTATION SPECIFICATION FOR ANSI C12.19 / IEEE 1377 METERS**

---

---

### **1.1 Introduction**

---


The guide is a consolidation of separate papers provided to Measurement Canada or Sealing Team in particular to provide its members the understanding of implementation of the audit trail according to the ANSI C12.19 Standard. The implementation is complied with their draft Specifications Relating to Metrological Audit Trails “PS-EGMV-XX-E”. To do so, it is necessary to map between the draft architecture and the actual implementation (this document). The Glossary is a valuable resource that needs to be used frequently to map these terms. Chapter 2 of this guide is not only based on the Measurement Canada draft but it also incorporates four years of work product of the Task Force members, in which the critic had frequently asked the meetings schedule to be changed to ensure he/she could contribute. The Measurement Canada draft on event logger just provides a blanket statement on metrological parameters. This approach is not acceptable to the industry, hence the Task Force has relentlessly formed the list according to the Standard as well as Measurement Canada current practice. Therefore it might appear as policy making, but it aims to provide a road map of legal metrological parameters in the Standard to assist the Sealing Team to form a Measurement Canada policy in satisfying the industry desire and the Standard. The final decision is theirs.

To accomplish the engineering objective this guide defines an implementation model for the audit trail using the Utility Industry Standard Table. This model is entirely consistent with ANSI C12.19 / IEEE 1377, and with all of the implementation principles that were proposed by “Measurement Canada Task Force on Data Communication Protocol for Electronic Metering Devices”. It also describes a number of implementation strategies, mechanisms and procedures, that can be incorporated in a metering devices, that are consistent with

---

Measurement Canada “*Interim Specifications Relating to Event Loggers for Electricity Metering Devices and Systems*”, IS-E-01-E; and “*Interim Procedures Relating to Event Loggers for Electricity Metering Devices and Systems Pursuant to the Requirements of IS-E-01*”, IP-E-01-E, October 18, 2001.

The implementation architecture of the audit trail is described in terms that should be familiar to an electrical engineer who is not necessarily a programmer. Basic understanding of AMR, security, and Measurement Canada audit trail requirements is essential. However, the architecture is exposed with a sufficient level of detail to also enable the software or hardware engineer to implement a Measurement Canada and ANSI C12.19 / IEEE 1377 Standard conforming event loggers. For this reason the discussion exposes audit trail implementation principles, it identifies the C12.19 / IEEE 1377 event-logger related table elements and their exact interpretation. In addition the correct element operating states, modes and data format are shown for each audit trail implementation model.

A number of variations are exposed, and the implementor is free to choose any one of them. However, the used of this guide strictly and without consultation with Measurement Canada is discourage as it is possible that within the context of a specific implementation Measurement Canada may consider the solution non-compliant. Such implementation will be flagged with the hand icon,  , on the left hand side of the paragraph to indicate the need for additional consideration.

All the procedures and constraints described herein assume “verified, approved and sealed” metering systems. This simply implies that the meters are assumed to be verified for function and accuracy, by whatever criteria, against the claims made by their manufacturer and as approved usage. The actual mechanisms for obtaining such approvals and the technology used to prevent alteration to internal working mechanisms are not considered nor discussed in this document. It is simply assumed that such mechanisms are in effect to the satisfaction of Measurement Canada.

---

## **1.2 Audit Trail Implementation Models for Verified, Approved Sealed Meters**

---

There are five possible implementation scenarios for the audit trail.

1. Meters Implementing Event Counters (See sub-section [1.2.1.1](#) below)
2. Event logger without values (See sub-sections [1.2.1.2-3](#) and [1.2.1.3-3](#) below).
3. Event logger with new values (See sub-section [1.2.1.2-2](#) below).
4. Event logger with metrological signature (See sub-section [1.2.1.3-1](#) below).
5. Event logger with new values & metrological signature (See sub-sections [1.2.1.2-1](#) and [1.2.1.3-2](#) below).

The selection of the correct implementation model depends on whether the end-device can be programmed by remote means after it has been verified, approved and physically sealed. The key distinction is made on whether the end-device is protected by a software lock that can be unlocked locally or remotely. When unlocked remotely, an the event logger data shall be used.

Meters with event counters are typically very simple. Event counters are more likely to be found in water or gas meters. They are seldomly used in electricity meters.

Meters with ANSI C12.19 event loggers (also referred to by Measurement Canada as audit trails) maintain information that can be subsequently used to reconstruct the programming change history of the meters. As you will later see in this chapter, in some cases the information can be stored entirely inside the meter and in other instances the information may be distributed partly inside the meter and partly inside the owner's record keeping systems. The four possible variations of the event logger are presented to provide a consistent and strict engineering instructions, so that all C12.19 Event Loggers can be implemented and operated consistently for each of possible variations.

## 1.2.1 Special Considerations

This guide does not preclude implementations which may be deemed invalid by Measurement Canada under some conditions. As such this guide is quite neutral. However, the implementer is encouraged to pay special attention and consideration to the guiding principles set forth by IS-E-01-E and SP-E-01-E, as follows:

### 1.2.1.1 C12.19-1997 Meters Implementing Event Counters

Its use is more likely to be acceptable<sup>1</sup> when it is limited to "Category 2 end-device" (See GLOSSARY) with remote configuration capability, when access is strictly controlled by physical hardware.

### 1.2.1.2 C12.19-1997 Meters Implementing A Self-contained Event Logger

These are likely to be Measurement Canada Category 3, Type A end-devices (See GLOSSARY) end-device like one of those described follow:

1. The event logger data (new value) is encrypted or secured by the manufacturer to prevent unauthorized access to the data or manipulation of the data. Its use is likely to be acceptable<sup>1</sup> without additional qualification or stipulation.
2. The event logger data (new value) is not encrypted and it is not secured by the manufacturer to prevent unauthorized access to the data or manipulation of the data. Its use is less likely to be acceptable<sup>1</sup> without additional qualification or stipulation.
3. The event logger data (new value) is implemented using only strict C12.19-1997 (US) Standard event codes and values. Its use is less likely to be acceptable<sup>1</sup> (since the 1997 published event codes and associated parameter formats are weaker than what may be deemed sufficient for used in trade). This deficiency has been documented in this guide as corrected in the second revision of ANSI Standard C12.19.



### 1.2.1.3 C12.19-1997 Meters Implementing Downloadable Event Logger

These are likely to be Measurement Canada Category 3, Type C end-devices (See GLOSSARY) end-device like one of those described follow:

<sup>1</sup> In accordance with all conditions listed in Section 1.0, Scope, of IS-E-01, these restrictions do not apply to devices, which are appropriately marked as stipulated by legislation as being not for use in trade.

1. The event logger data is encrypted or secured by the manufacturer to prevent unauthorized access to the data or manipulation of the data. Event log data (new value) is not embedded in the meter, but event log metrological signature is embedded. Its use is likely to be acceptable<sup>1</sup> without additional qualification or stipulation.
2. The event logger data is encrypted or secured by the manufacturer to prevent unauthorized access to the data or manipulation of the data. Event log data (new value) is embedded in the meter together with event log metrological signature is embedded. Its use is likely to be acceptable<sup>1</sup> without additional qualification or stipulation.
3. The event logger data (new value) is implemented using only strict C12.19-1997 (US) Standard event codes and values. Its use is less likely to be acceptable<sup>1</sup> (since the 1997 published event codes and associated parameter formats are weaker than what may be deemed sufficient for used in trade). This deficiency has been documented in this guide as corrected in the second revision of ANSI Standard C12.19.



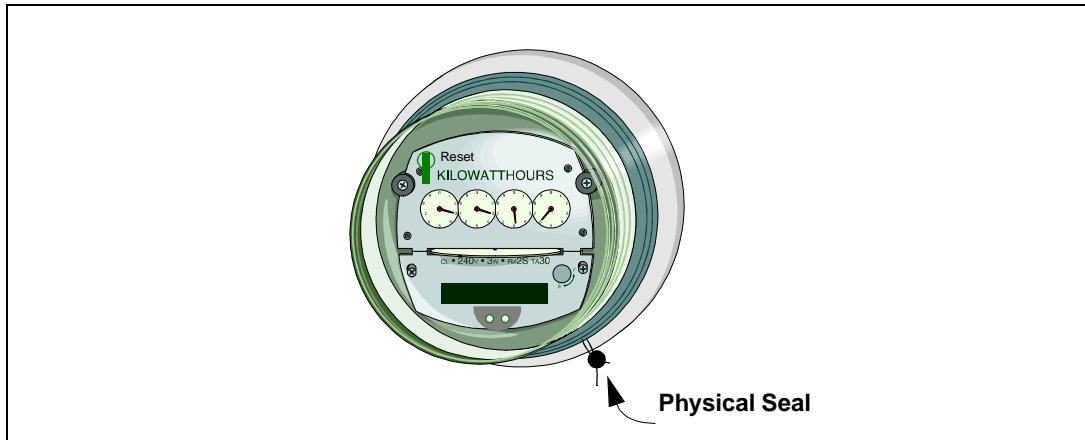
### 1.2.2 Meters Implementing Event Counters

Once an approved metering device is verified and sealed with a physical seal no change to the programmable parameters/functions of the metering device are allowed to be performed remotely regardless of remote technology or remote means used to provide access to end-device re-programming. Therefore, an end-device without an event logger shall never be remotely unlocked.

---

Figure 1.1

Approved, verified and sealed meter without an event logger.



Such end-devices are considered Measurement Canada Category 1 devices, since they do not have remote configuration capabilities for their sealable parameters and they do not have an event logger. These end-devices can also be considered Measurement Canada Category 2 devices, if a remote configuration capability exists, but **access to program or configuration locks is not remotely available**.

As such, approved end-devices shall minimally have:

1. A physical seal, that prevents all programming or;
2. Two event counters:
  - one for calibration parameters; and
  - one for change in configuration or metrological parameters, where the access to these is controlled by a software switch.

### 1.2.2.1 UOM Table Construction Details for End-devices Using Event Counters

When event counters are implemented, instead of an event logger, the event counters shall be described consistently as program counters using ID\_CODE=52 in the UOM\_ENTRY\_BFLD of Table 12, "Unit Of Measure", as follows:

**Chart 1.1** UOM Description for the configuration and metrological parameters, access controlled by a software switch counter.

| Table Element Name | Description  | Value        | Interpretation   |
|--------------------|--|--------------|--|
| ID_CODE            | This is the UOM identifier of the physical quantity being recorded or measured. When used to count the number of times the end-device was re-programmed or re-configured this UOM code shall describe a programming counter. | 52           | This UOM identifies the configuration and metrological parameters, access controlled by a software switch counter. |
| TIME_BASE          | This sub-element describes the measurement's time reference. This measurement shall be assigned as an event counter quantity.  | 7            | Number of occurrences of changes to configuration, metrological parameters.  |
| MULTIPLIER         | The multiplier identifies the scaling value to apply to the reported. This is the counter scaling factor.  | 0            | 10 <sup>0</sup> , i.e. unity multiplier.   |
| Q1_ACCOUNTABILITY  | Indication that measurement/counter operates while the commodity flows in Quadrant 1   | <b>FALSE</b> | Energy direction of flow does not affect the counter.  |
| Q2_ACCOUNTABILITY  | Indication that measurement/counter operates while the commodity flows in Quadrant 2   | <b>FALSE</b> | Energy direction of flow does not affect the counter.  |

**Chart 1.1** UOM Description for the configuration and metrological parameters, access controlled by a software switch counter.

| Table Element Name      | Description  | Value        | Interpretation  |
|-------------------------|--|--------------|---|
| Q3_ACCOUNTABILITY       | Indication that measurement/counter operates while the commodity flows in Quadrant 3   | <b>FALSE</b> | Energy direction of flow does not affect the counter.                                     |
| Q4_ACCOUNTABILITY       | Indication that measurement/counter operates while the commodity flows in Quadrant 4   | <b>FALSE</b> | Energy direction of flow does not affect the counter.                                     |
| NET_FLOW_ACCOUNTABILITY | Identifies the manner in which the quadrants influence the measurement.  | <b>FALSE</b> | Direction of flow does not affect counters.   |
| SEGMENTATION            | When the ID_CODE field represents the electric utility industry unit of measures, this sub-element indicates phase-to-phase or line-to-neutral measurement relation. | 0            | The counter is not a phase related measurement quantity, so this field is not applicable. |
| HARMONIC                | Indicates whether the reported measurement is the entire signal or a harmonic component of the signal.   | 0            | The counter is an absolute count, so harmonic content is not applicable.                  |
| RESERVED                |  | 0            | Filled with zeros.  |
| NSF                     | Not a Standard Flag  | <b>FALSE</b> | This is a Standard counter.   |

### 1.2.2.2 Source Selection Table Construction Details for End-devices Using Event Counters

The corresponding counter elements of SOURCE\_LINK\_BFLD in Table 16, “Source Definition”, shall be set as follows:

**Chart 1.2** Setting for counter SOURCE\_LINK\_BFLD sub-elements in Table 16, “Source Definition”.

| Table Element Name | Description   | Value        | Interpretation   |
|--------------------|---|--------------|--|
| UOM_ENTRY_FLAG     | Identifies the presence or absence of a UOM entry for this measurement in Table 12, “Unit Of Measure”.    | <b>FALSE</b> | An entry does not exist for this counter in Unit of Measure Table.   |
|                    |   | <b>TRUE</b>  | An entry exists for this counter in the Unit Of Measure table. This is the preferred implementation for Table 12, “Unit of Measure”.   |
| DEMAND_CTRL_FLAG   | Demand interval selector.   | <b>FALSE</b> | There is no relevant information for this counter in Table 13, “Demand Control”.   |
| DATA_CTRL_FLAG     | Manufacturer proprietary data record may be used to further qualify and control how the counter operates. | <b>TRUE</b>  | Additional information regarding the operation of this counter is optionally supplied by the manufacturer in Table 14, “Data Control”.   |
|                    |   | <b>FALSE</b> | No additional information is supplied by the manufacturer in Table 14, “Data Control”.   |
| CONSTANTS_FLAG     | The constants, offsets and multiplier enabling sub-elements for this measurement.                         | <b>FALSE</b> | There is no multipliers, offsets of constants provided in Table 15, “Constants” for this counter. The offset is assumed to be zero and multiplier/scalars are assumed to be 1.   |
|                    |   | <b>TRUE</b>  | Multipliers, offsets of constants are provided in Table 15, “Constants” for this counter. All offsets shall be set to 0 and all multipliers shall be set to 1 such that the counters increment by one and have a unity scaling factor. i.e. the counter is a true count. |

**Chart 1.2**      Setting for counter SOURCE\_LINK\_BFLD sub-elements in Table 16, “Source Definition”.

| Table Element Name     | Description  | Value        | Interpretation  |
|------------------------|--|--------------|---|
| PULSE_ENGR_FLAG        | Pulse engineering metering mode control flag.  | <b>FALSE</b> | Not applicable. The values shall be set to <b>FALSE</b> to indicate that the counter source is in pulse/counts units. |
| CONSTANT_TO_BE_APPLIED | The corresponding offsets and multipliers from Table 15, “Constants”, have already been applied. The measurement reflects actual physical value. | <b>FALSE</b> | The counter multipliers have all been applied. The counter reports true counts.                                       |
|                        |  | <b>TRUE</b>  | Invalid state for configuration and re-programming event-counter UOMs   |

### 1.2.2.3 Location of Event Counters

Event counters shall be placed and read from the elements of the PRESENT\_VALUE array of Table 28, “Present Register Data”. The first element in array PRESENT\_VALUE that is associated with a data source with ID\_CODE=52 (programmed counter) is the calibration counter. The second element in array PRESENT\_VALUE that is associated with a data source with ID\_CODE=52 (programmed counter) is the metrological parameters change counter

Table 28 placement order of the counters is programmed into elements of the array PRESENT\_VALUE\_SELECT of Table 27, “Present Register Selection”. This is done by entering the appropriate indices of calibration or configuration / metrological parameters counter selections from array SOURCES\_LINK found in Table 16, “Source Definition”.

The event counters must operate as follows:

1. The full positive integral numeric range of the **NI\_FORMATS** selected shall be supported when implementing an event counter.
2. Event counters shall increment only once regardless of the number of changes made while the end-device is in the adjustment mode.
3. Where access is controlled by a software switch, if the mode is accessed and no changes are made, this shall not constitute an event and the counter shall not increment.
4. Once an event counter has reached its capacity (the positive upper limit of the of the **NI\_FORMAT** type selected), it shall not be possible to make further changes to the end-device.

### 1.2.3 Meters Implementing an Event Logger

End-devices that fall into Measurement Canada Category 3 are devices which have remote configuration capability. Access to the metrological parameters is controlled by a software switch (possibly in addition to a logical or physical seal). These end-devices shall be equipped with an event logger.

The event logger implementation needs to meet certain audit trail reliability and security requirements and the event-logger shall be minimally:

1. Stored in non-volatile memory; and
2. Protected from alteration, modification, replacement, seizure, substitution, and unauthorized erasure or deletion.

There are three possible variations for the implementation of an event logger:

**Option 1:** A self-contained event logger, where the end-device is expected to be able to store all event entries for its operating life (or at least store its initial configuration and all the metrological change events that occur after verification and sealing). The optional event argument (`EVENT_ARGUMENT` portion of the event entry) contains only the changes made to the end-device.

**Option 2:** A downloadable event logger, where it is possible to download and securely transfer the event log records to a remote centralized host system; and subsequently to the secure transfer, the embedded event logger space can be reused for storing new events. The main difference between option 1 and option 2 is that in option 1 (the self-contained event logger) the end-device may record the new values of the elements that were changed. Whereas in option 2 the values of the elements (tables) may reside on a central host computer system and only the MD5 Signatures (`EVENT_CHECK_SIG`) are recorded in the `EVENT_ARGUMENT` portion of the event log entry.

**Option 3:** A downloadable (as in option 2) or self-contained event logger (as in option 1), which stores both new values (`NEW_VALUES`) and the MD5 Signatures (`EVENT_CHECK_SIG`) in the `EVENT_ARGUMENT` portion of the event entry.

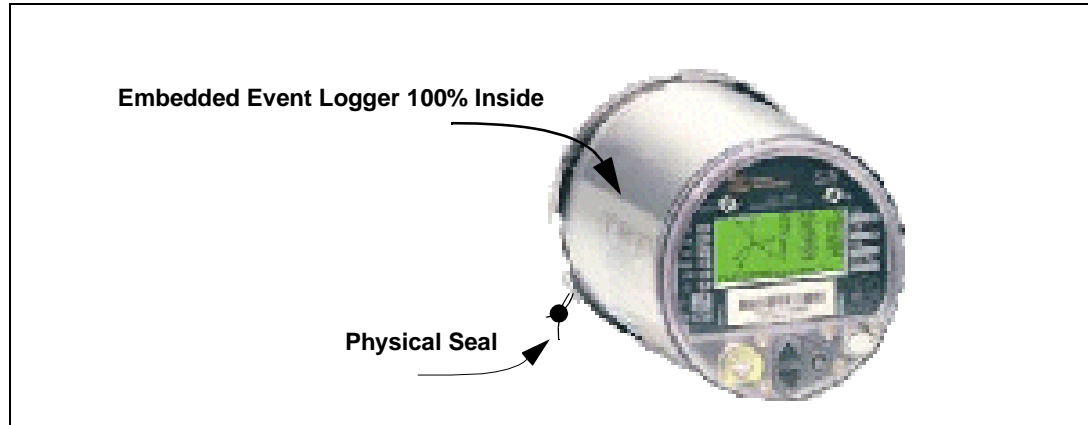


### 1.2.3.1 Meters With Self-contained Event Logger (Option 1)

Approved and verified functions/parameters of a metering device are permitted to be changed (reprogrammed) while the device is under physical seal, provided that all the changes are recorded in an embedded event logger.

Figure 1.2

Approved, verified and sealed end-device with a self-contained event logger.



When the event logger is self-contained inside the end-device it cannot be transferred out of the end-device (i.e. after a successful download of the event-list, the event-logger list pointers cannot be modified in any way that results in the release space inside the end-device or the erasure of event data). The manufacturer of the end-device shall ensure that the following general functional requirements are correctly implemented by the event logger:

1. The implementation may optionally capture of the entire program state of the end-device, prior to verification and sealing, in the event logger.
2. Upon capture of event state, if more than one event record is needed to describe the end-device configuration then each record shall contain the new values that were configured or programmed or activated simultaneously.
3. The log shall always contain a verification event (code 61). This event marks or captures the sealed stated of the end-device.
4. The implementation may store in each event-record the new values of the metrological parameters programmed in the end-device, subsequent to the verification event.
5. All subsequent event records may contain only the new values programmed.
6. When a re-verification event occurs (event code 62) it is an indication that the end-device needs to be re-verified, it is equivalent to the breaking of a physical seal. However, the end-device shall continue to meter correctly using its active program; and there shall be a clear status indication provided in the program status flags of Table 3, "End-Device Mode Status", as prescribed by ["Rule for Issuing a Re-verification Event" on page 1-42.](#)
7. When ANSI C12.19 / IEEE 1377 Standard Procedure 4, "Reset List Pointers", or Standard Procedure 5, "Update Last Read Entry" are invoked with a the LIST procedure parameter value=1, the end-device shall cause a response code=3 to be returned in Standard Table 8, "Procedure response". This indicates that the request



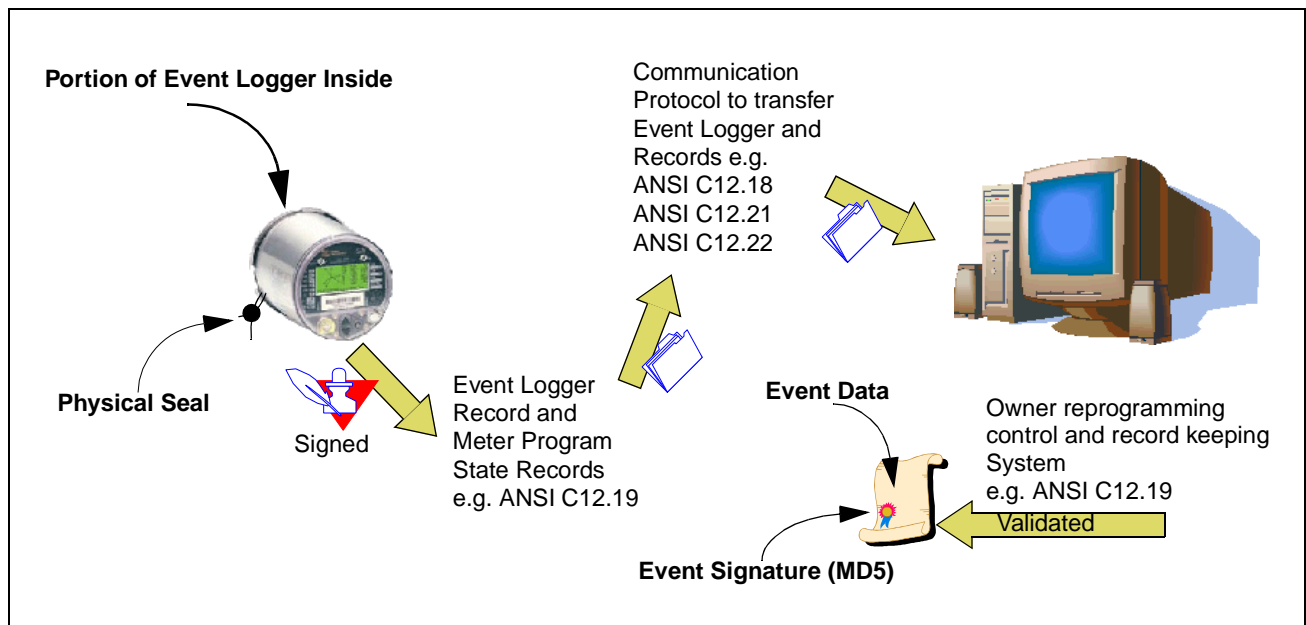
conflicts with current device settings (i.e. the end-device is verified and sealed, and the event logger cannot be manipulated). The request shall be ignored and an event log entry shall not be created.

8. There shall be no means, other than those which cause a re-verification event to be recorded, to alter, modify, replace, seize operation, substitute, erase or delete an event log entry.

### 1.2.3.2 Meters with a Downloadable Signed Event Logger (Options 2,3)

Approved and verified functions/parameters of a metering device are permitted to be changed (reprogrammed), while the device is under physical seal, providing the changes are recorded in the event logger and can be securely and reliably transferred to a remote record keeping centralized host system.

**Figure 1.3** Approved, verified and sealed end-devices with an event logger that is located in part inside and can be transferred in part to an external record keeping system.



This type of device stores an event signature of the new metrological parameters that govern the operation of the end-device. This required signature (MD5 hash code) is implemented to enable the detection of any discontinuity in the embedded and downloaded event logger records for the entire life of the meter (one life cycle starts after approval, verification and sealing).

Under this scenario, the role of the meter agent is to manage the end-device operations and programs, and to maintain the records. Specifically, the meter agent is responsible for keeping a record of all end-device programming parameters, constants, change history and event logger records. The MD5 unique signatures and identifiers that are embedded in the end-device enable the device's agent to fulfil its responsibility in maintaining and demonstrating

continuity between metrological data records and end-device programs that are stored within. This is accomplished exclusively through the use of the end-device embedded portion of the event logger tables, the downloaded metrological tables and the computer-hosted portion of the event logger tables, as prescribed by ANSI C12.19 / IEEE 1377.

An element of complexity is introduced, by virtue of the fact that the event logger tables may fill up. This is avoided by methodical meter reading and downloading of event records, followed by the execution of Standard Procedure 5, "Update Last Read Entry". The exact formalism for dealing with this operation is described in the following sections, which deal with the governing rules for a compliant design and operation of an event logger.

Implementing option 2 (Event Logger with signature only) has specific restriction compared to option 3 (Event Logger with signatures and new values). For example, with option 2, each table can be modified only once between downloads. If this constrain is not observed, raw table information will be lost and the signature computed by the record keeping system will no longer match the signature of the meter. The end-device can enforce such a constraint to prevent accidental errors due to improper practice and operations. Option 3 has also the advantage that it does not necessitates the download of changed tables; and it can also be used without a record keeping system in a self-contained event logger.

## **1.3 Rules for Creating, Managing and Maintaining Embedded Event Loggers**

---

The event logger operation needs to be designed in such a manner as to establish and maintain consumer confidence in the metering device technology. For that reason an ANSI C12.19 / IEEE 1377 Standard based event logger needs to adhere to strict guidelines and uniform rules. The Tables implementation rules, outlined in subsequent sub-sections, are consistent with both the Table construction rules set forth in ANSI C12.19 / IEEE 1377 Standard and the Event Logger requirements principles published by Measurement Canada.

### **1.3.1 Rule for Table Mapping to Event Logger Parameters**

When any element of a Table is a calibration parameter, a configuration parameter or a metrological parameter then the entire Table is considered to be a metrological table.

#### **1.3.1.1 Mapping of Metrological Elements to Metrological Tables**

The Standard tables can be rather large, complex, and contain many related elements. This fact alone makes it prohibitively difficult (impossible) to manage an event logger on a per table element basis. For this reason when an ANSI C12.19 / IEEE 1377 table element is considered by the manufacturer or by Measurement Canada to be a calibration parameter, a configuration parameter, or a metrological parameter then any change to that element, or change to other elements that reside in the same Table, shall cause the creation of a new event log entry for that Table.

**1.3.1.2 Event Logger and End-Device State Synchronization**

In creating the event logger entry, the implementer shall exercise care to ensure that all programmed changes, which take place simultaneously are recorded simultaneously in the embedded event logger. Therefore, if more than one element inside one table is changed, and the changes take effect simultaneously then only one event logger record shall be created.

If the changes take effect step-wise, or the elements that are re-programmed or modified do not occupy the same table, then one event log entry shall be created for each changed table or changed element as needed.

The manufacturer may facilitate this operation through the use of Standard Procedure 2, “Save Configuration”. When the “Save Configuration” Procedure is made available by the manufacturer then it shall be used to activate all table changes that were introduced during one session. This shall result in the creation of one (or more) event log entry, as appropriate, only if all of the Table programming or changes performed can be activated in totality. Otherwise all the changes will be discarded and an event log entry shall not be created.

**1.3.2 Rule for Identifying the Embedded Event Logger Tables**

**1.3.2.1 Event Logger Data Tables**

The embedded event logger records, including identification elements, security elements, program / parameter change elements and associated data shall be embedded and retrieved by requesting **Standard Table 76 - “Event Log Data”**.

**1.3.2.2 Event Logger Configuration and Control Tables**

The event logger configuration shall be programmed into Standard Table 71, “Actual Log Dimensions”, and Table 72, “Event Identification”, prior to end-device verification and sealing. The Table elements listed below shall be configured as shown below:

**Chart 1.3** Event Log Dimensions Parameters, Table 71

| Table 71 Element Name | Description  | Value        | Interpretation   |
|-----------------------|--|--------------|--|
| EVENT_NUMBER_FLAG     | An optional event number is used to synchronize Table 74, “History Log”. | <b>FALSE</b> | A shared event number is not used.   |
|                       |  | <b>TRUE</b>  | A shared event number is used to maintain synchronization between the optional history log entries and the required event log entries. |

**Chart 1.3**      Event Log Dimensions Parameters, Table 71 (Continued)

| Table 71 Element Name  | Description  | Value        | Interpretation   |
|------------------------|--|--------------|--|
| EVENT_INHIBIT_OVF_FLAG | Event logger list overflow control flag.   | <b>FALSE</b> | The event logger processor is not inhibiting the creation of new entries when an overflow condition exists. This flag shall always be set to <b>TRUE</b> in verified and sealed end-device.  |
|                        |  | <b>TRUE</b>  | The event logger processor is inhibiting the creation of new entries when an overflow condition exists. This is the required state of the control flag in a verified and sealed end-device.  |
| NBR_STD_EVENTS         | Number of octets (maximum 255) in the set <code>STD_EVENTS_SUPPORTED</code> element found in Table 72, "Event Identification". The value 1 represents 8 events since there are 8 bits per octet. | 1..255       | This value shall be computed based on the verified and approved number of event types supported by the end-device divided by 8.  |
|                        |  | 0            | Invalid if Standard event logger is implemented.   |
| NBR_MFG_EVENTS         | Number of octets (maximum 255) in the set <code>MFG_EVENTS_SUPPORTED</code> element found in Table 72, "Event Identification". The value 1 represents 8 events since there are 8 bits per octet. | 0..255       | This value shall be computed based on the case by case, documented, verified and approved number of manufacturer's proprietary event types supported by the end-device divided by 8.   |
| EVENT_DATA_LENGTH      | Number of octets in the <code>EVENT_ARGUMENT</code> found in Table 76, "Event Log Data.  | 0..255       | The transmitted size of each <code>EVENT_ARGUMENT</code> . The size should be large enough to include the new values used to affect a change to the metrological elements (using a PSEM write service request sequences) and the signature, as required. |

**Chart 1.3** Event Log Dimensions Parameters, Table 71 (Continued)

| Table 71 Element Name | Description  | Value    | Interpretation  |
|-----------------------|--|----------|---|
| NBR_EVENT_ENTRIES     | The actual maximum number of entries available in the event log. | 0..65535 | <p>The event logger shall not be allowed to overflow or roll over when the number of unread entries equal this value.</p> <p>When the event logger is self-contained, it shall not be possible to create more than NBR_EVENT_ENTRIES entries in the log. And when the number of entries in the event logger equal to NBR_EVENT_ENTRIES, the end-device shall reject all changes to metrological elements.</p> <p>When the event logger is downloadable it shall be possible to invoke Standard procedure 5, "Update Last Read Entry" to unblock the event logger (after securely downloading the Event Log Table data).</p> |

Chart 1.4 Event Log Available Triggers Identification, Table 72

| Table 72 Element Name | Description   | Value | Interpretation   |
|-----------------------|---|-------|--|
| STD_EVENTS_SUPPORTED  | <p>This set of boolean flags identifies positionally the Standard events triggers that are available in the end-device. <b>These triggers are available in addition to metrological table element change events.</b> The total number of event trigger entries enabled equals to 8 X NBR_STD_EVENTS. With a one (1) representing a <b>TRUE</b> or an available event trigger capability and a zero (0) representing a <b>FALSE</b> or an unimplemented event capability.</p> <p>The indicator at bit offset 0 represents the availability of event code 0, the value at bit offset 1 represents the availability of event code 1, and so on, up to NBR_STD_EVENTS - 1, being the largest possible event code.</p> |       | <p>The values and parameters listed in the <a href="#">ANSI Standard C12.19-1997 History Log Codes</a> are applicable only for use in the history tables or event log tables, <b>when security codes and new element values are not recorded in the event logger.</b> The actual event codes required for operating a validated, approved and sealed end-device are discussed in <a href="#">Section 1.3.9, "Rule for Embedded Event Log Entry Format,"</a> on page 1-30. These are all related to sealing and the protection of metrological elements that are expressed as tables. The implication is that all operating metrological parameters of an end-device need to be expressed as ANSI C12.19 / IEEE 1377 tables (Standard or Manufacturer) so that they can be captured and recorded in the event logger.</p> |

Chart 1.4 Event Log Available Triggers Identification, Table 72 (Continued)

| Table 72 Element Name | Description  | Value | Interpretation   |
|-----------------------|--|-------|--|
| MFG_EVENTS_SUPPORTED  | <p>This set of boolean flags identifies positionally of the Manufacturer proprietary events triggers that are available for selection in the end-device.</p> <p>The total number of event entries available equals to 8 X MFG_STD_EVENTS. With a one (1) representing a <b>TRUE</b> or an implemented event trigger capability and a zero (0) representing a <b>FALSE</b> or an unimplemented event capability.</p> <p>The indicator at bit offset 0 represents the availability of event code 0, the value at bit offset 1 represents the availability of event code 1, and so on, up to MFG_STD_EVENTS - 1, being the largest possible event code.</p> |       | <p>If a manufacturer event may be recorded in the event logger, then a corresponding table element needs to be associated with that trigger to enable the storage of its "new value", when so desired.</p> |

**Chart 1.5**      Event Log Control Parameters, Table 75

| Table 75 Element Name      | Description   | Value        | Interpretation  |
|----------------------------|---|--------------|---|
| STD_EVENTS_MONITORED_FLAGS | <p>This set of boolean flags identifies positionally the Standard events triggers that are actually monitored in the end-device. <b>These triggers are in addition to metrological table element change events.</b> The total number of event trigger entries enabled equals to 8 X NBR_STD_EVENTS. With a one (1) representing a <b>TRUE</b> or an implemented event trigger capability and a zero (0) representing a <b>FALSE</b> or an un-implemented event capability.</p> <p>The indicator at bit offset 0 represents the enabling state of event code 0, the value at bit offset 1 represents the enabling state of event code 1, and so on, up to NBR_STD_EVENTS - 1, being the largest possible event code.</p> | <b>FALSE</b> | Turns off event recording for associated event code.                  |
|                            |   | <b>TRUE</b>  | Turns on event recording for associated event code.                   |
| MFG_EVENTS_MONITORED_FLAGS | A set of boolean flags that operate in a similar way to STD_EVENTS_MONITORED_FLAGS, except that they select manufacturer monitored events.  | <b>FALSE</b> | Turns off event recording for associated event code.                  |
| STD_TBLS_MONITORED_FLAGS   | A set of boolean flags that operate in a similar way to STD_EVENTS_MONITORED_FLAGS, except that they identify all Standard tables that are monitored for table change events.   | <b>FALSE</b> | Turns off event recording for associated Standard table change event. |
|                            |   | <b>TRUE</b>  | Turns on event recording for associated Standard table change event.  |

Chart 1.5 Event Log Control Parameters, Table 75 (Continued)

| Table 75 Element Name    | Description  | Value | Interpretation  |
|--------------------------|--|-------|---|
| MFG_TBLS_MONITORED_FLAGS | A set of boolean flags that operate in a similar way to STD_TBLS_MONITORED_FLAGS, except that they identify all Manufacturer tables that are monitored for table change events.          | FALSE | Turns off event recording for associated Manufacturer table change event.         |
|                          |  | TRUE  | Turns on event recording for associated Manufacturer table change event.          |
| STD_PROC_MONITORED_FLAGS | As set of boolean flags that operate in a similar way to STD_TBLS_MONITORED_FLAGS, except that they identify all Standard procedures that are monitored for a procedure invoke event.    | FALSE | Turns off event recording for associated Standard procedure invocation event.     |
|                          |  | TRUE  | Turns on event recording for associated Standard procedure invocation event.      |
| MFG_PROC_MONITORED_FLAGS | A set of boolean flags that operate in a similar way to STD_PROC_MONITORED_FLAGS, except that they identify all Manufacturer procedures that are monitored for a procedure invoke event. | FALSE | Turns off event recording for associated Manufacturer procedure invocation event. |
|                          |  | TRUE  | Turns on event recording for associated Manufacturer procedure invocation event.  |

### 1.3.3 Rule for Creating an Entry In the Embedded Event Log Table

The following conditions govern and enable the end-device event logger entry creation. These are applicable following the end-device approval, verification and sealing.

1. While the event log list is full, the end-device shall not permit any changes to take place to metrological tables, except for a download event (i.e. “Update Last Read Entry” Standard Procedures).
2. Event log entries shall only be created as a result of changes, by what-ever-means, to metrological parameters.
3. Changes to metrological parameters shall be reflected as changes in related metrological table elements.
4. Standard procedure “Update Last Read Entry” shall be used following the successful download of the event log table to release space for new event logger entries.
5. Upon successful invocation of the “Update Last Read Entry” Standard Procedure the end-device shall create an Event Log Table entry confirming this action using an event code from [Chart 1.9, “EVENT\\_CODES for downloadable event loggers that](#)

are only signed,” on page 1-37 or from Chart 1.10, “EVENT\_CODES for downloadable event loggers that are signed with new values,” on page 1-39.

6. Whenever an event log entry is created the event sequence numbers in Table 76, `LAST_ENTRY_SEQ_NBR` and `EVENT_SEQ_NBR`, shall auto-increment by one and simultaneously (with initial values set to 0 or 1).
7. When the `LAST_ENTRY_SEQ_NBR` reaches its maximum operating limit of 4,294,967,295, and the event logger is 100% embedded, the end-device shall not permit any changes to take place to metrological tables.
8. When the `LAST_ENTRY_SEQ_NBR` reaches its maximum operating limit of 4,294,967,295, and the event logger is downloadable, the `LAST_ENTRY_SEQ_NBR` shall reset to 0 and the end-device shall continue to permit changes to metrological tables.
9. When the `EVENT_SEQ_NBR`, reaches its maximum value of 65535 it shall reset to 0.
10. The event record sequence number, `EVENT_SEQ_NBR`, of the last event entry shall be verified to be  $EVENT\_SEQ\_NBR = LAST\_ENTRY\_SEQ\_NBR \text{ MOD } 65536$ .
11. The Table `MODIFIED_FLAG` in “Signatures Table (Proposed Std Table 78, Rev 2.0)” on page 3-92, shall be set to **FALSE**. for all tables that were modified when the event-log entry is created.

#### 1.3.4 Rule for Identifying Event Log Triggers

It is not permitted to use the event logger as a general purpose information logging mechanism. Approved, verified and sealed devices shall only log changes to tables, which have been identified to contain metrological elements by Measurement Canada or the Manufacturer of the end-device.

#### 1.3.5 Rule for Setting Event Logger List Size

The actual size of the embedded event logger is established by setting the Standard elements `NBR_EVENT_ENTRIES` to the maximum number of entries supported by the end device. This element is defined in Table 71, the “Actual Log”. In addition the following conditions shall be met:

1. The size of the event log table shall be sufficiently large to enable the execution of the “Update Last Read Entry” Standard Procedure, which shall also create a new event entry, without the erasure of any existing event log entries.
2. The minimum list size shall be approved by Measurement Canada. The minimum value required for correct firmware operation of the Standard Event Log Tables is 2 entries.

#### 1.3.6 Rule for Overwriting Event Logger Entries

Event logger re-writing shall be performed in accordance with the following constraints:

1. Standard Procedure 5, “Update Last Read Entry”, shall be used to update the event logger list pointers to “free” space following a successful download of the event logger entries.

2. When the embedded event log list is “full”, and when not fully embedded, it shall also be possible to execute Standard Procedure 5, “Update Last Read Entry” when the end-device is physically sealed, to “free” space in the event logger.
3. The execution of “Update Last Read Entry” procedure shall not cause any existing event log entries to be erased or overwritten.
4. The successful execution of “Update Last Read Entry” procedure shall be recorded in the Event Log Table.
5. Following the successful execution of “Update Last Read Entry” procedure the MODIFIED\_FLAG in “Signatures Table (Proposed Std Table 78, Rev 2.0)” on page 3-92 (if present), shall be set to **FALSE**.
6. When reading the Event Log Table after the execution of the “Update Last Read Entry” procedure, it shall be possible to access the older entries in the Event Log Table.
7. When new Event Log Table entries are created, the new entries may overwrite existing event log records, starting with the oldest. These should have previously been securely downloaded to the record keeping system.
8. The interlocking principles described in [Section 1.3.3, “Rule for Creating an Entry In the Embedded Event Log Table,” on page 1-25](#), shall be observed. Any violation of the interlock principle shall result in a re-verification event.

### 1.3.7 Rule for Reserving Event Logger Entries

Room for event log entries shall be created upon programming the end-device via pending (deferred program activation) tables as follows:

1. One event log entry space shall be reserved upon successful programming of a pending metrological table or procedure invocation.
2. Upon activation of pending tables or procedure (invocation of pending metrological program) the event log entry shall be created.
3. The event time stamp shall be that of the time of activation.
4. The event entry created shall be next in sequence, the newest entry available, as if the end-device was programmed at the time of the table or procedure activation.
5. One event-log entry shall be created for each pending table or procedure activation.
6. Upon cancellation, without activation, of a pending metrological table or procedure the reserved slot event logger entry shall be freed.
7. The interlocking rule described in [Section 1.3.3, “Rule for Creating an Entry In the Embedded Event Log Table,” on page 1-25](#) shall be observed. A pending activation shall not activate a metrological parameter without creating an event logger entry; and an event logger entry shall not be created upon cancellation, by what-ever-means, of a pending metrological program activation.

### 1.3.8 Rule for Event Logger List Management

The Event Log Data Table (76) provides event logger list management elements that shall be implemented and interpreted as follows:

Chart 1.6 Event Log List Description Elements, Table 76

| Table Element Name | Description  | Value | Interpretation  |
|--------------------|--|-------|---|
| ORDER              | An indicator for the order in which the event logger entries are stored and transported by the end device.   | 0     | The embedded event log entries are stored and transported in ascending order (i.e. element N is older than element N+1).  |
|                    |  | 1     | The embedded event log entries are stored and transported in descending order (i.e. element N is newer than element N+1)  |
| OVERFLOW_FLAG      | Indicates whether the event logger reached its operating limits. When the event logger operating limits are reached it is not permitted to create new entries and all changes to metrological tables shall be inhibited, according to the interlocking principle. This condition shall be cleared by executing the "Update Last Read Entry" Procedure. | FALSE | Event logger overflow has not occurred.   |
|                    |  | TRUE  | An attempt was made to create a new event log entry, such that the number of unread entries stored in the NBR_UNREAD_ENTRIES element would have exceeded the actual number of permissible entries in the log. The attempt failed. |
| LIST_TYPE          | The event logger list buffer storage management type.  | 0     | The event logger list elements are stored and transmitted using a FIFO (First In First Out) buffer. The placement of the first/last entry shall be determined from the value of the ORDER element and LAST_ENTRY_ELEMENT.         |
|                    |  | 1     | The event logger list elements are stored and transmitted using a circular (ring) buffer. The placement of the last entry shall be determined from the value of the LAST_ENTRY_ELEMENT.   |

Chart 1.6 Event Log List Description Elements, Table 76 (Continued)

| Table Element Name    | Description   | Value        | Interpretation   |
|-----------------------|---|--------------|--|
| INHIBIT_OVERFLOW_FLAG | Indicator for the state of the inhibit overflow control bit, <code>EVENT_INHIBIT_OVF_FLAG</code> , in Table 71, "Actual Log Dimensions".  | <b>FALSE</b> | Event Logger processor is not inhibiting new entries when an overflow condition exists. <b>This state is invalid and shall not exist in an approved, verified, and sealed end-device.</b>  |
|                       |   | <b>TRUE</b>  | Event Logger processor is inhibiting new entries when an overflow condition exists. This is the correct operating mode for an approved, verified, and sealed end-device  |
| NBR_VALID_ENTRIES     | The number of valid event entries in the embedded event logger. This includes the read and unread entries totals.<br><br>This value is set to 0, initially, when the end-device is configured. It increments up to the value of <code>NBR_EVENT_ENTRIES</code> as new events entries are recorded.      | 0            | The embedded event log list is completely empty.   |
|                       |   | Non Zero     | Actual number of entries in the embedded event logger list.  |
| LAST_ENTRY_ELEMENT    | The embedded event logger array element number of the newest entry in the log.  |              |  |
| LAST_ENTRY_SEQ_NBR    | The sequence number of the newest (last) entry in the embedded event logger. The range of the sequence number is form 0 (following initial configuration) to 4,294,967,295. This element auto-increments by one each time an event log entry is created, during the operational life of the end-device. |              | The corresponding event logger entry sequence number, <code>EVENT_SEQ_NBR</code> , is the least significant 16 bits of <code>LAST_ENTRY_SEQ_NBR</code> at the time when the event entry is created. This synchronization shall be maintained and persisted for the life of the end-device. |

**Chart 1.6** Event Log List Description Elements, Table 76 (Continued)

| Table Element Name | Description  | Value    | Interpretation  |
|--------------------|--|----------|---|
| NBR_UNREAD_ENTRIES | The unread number of entries in the Event Log Table.   | 0        | All entries in the Event Log Tables were successfully read and downloaded to the central record keeping host computer by the end-device's agent.  |
|                    | This value shall be set initially to 0, when the end-device is configured, and increment each time a new event is recorded. When an attempt is made to modify a metrological value when this field equals to NBR_EVENT_ENTRIES, the attempt shall fail and OVERFLOW_FLAG shall be set to <b>TRUE</b> . | non-zero | This element auto-increments by one each time an event log entry is created. It will decrement by the invocation of the "Update Last Read Entry" Standard Procedure, after the successful and secure download of the event logger records. There is no software provision for ensuring that the events where downloaded and securely places in the remote host computer system. <b>This process shall be documented and managed reliably by the end-device's agent.</b> |

In addition, when the ["Signatures Table \(Proposed Std Table 78, Rev 2.0\)"](#) on page 3-92, is available, the `MODIFIED_FLAG` will be set to **FALSE** upon invocation of Procedure 5, "Update Last Read Entry".

### 1.3.9 Rule for Embedded Event Log Entry Format

The embedded event logger follows the prescribed format of the Standard `EVENT_ENTRY_RCD` record. All `EVENT_ENTRY_RCD` records are collected into an array, `ENTRIES`, whose maximum capacity and operation is determined by the `NBR_EVENT_ENTRIES` element of Table 71, "Actual Log Dimensions", and controlled by Table 72, "Event Identification" and Table 75, "Event Log Control" in accordance with Measurement Canada minimal size requirements for the event logger. Each embedded event logger record shall contain the following elements in its preamble:

**Chart 1.7** Event Log Table entry elements' preamble

| Table Element Name | Description   | Value     | Interpretation   |
|--------------------|---|-----------|--|
| EVENT_TIME         | Date and time of Event Log entry creation.  |           | This entry includes the year, month, date, hour, minute and second in accordance with ANSI C12.19 / IEEE 1377 LTIME_DATE Standard format.  |
| EVENT_SEQ_NBR      | Event log entry sequence number.  |           | This is the least significant 16 bits of the LAST_ENTRY_SEQ_NBR at the time, EVENT_TIME, when the entry is created. Synchronization shall be maintained between EVENT_SEQ_NBR and LAST_ENTRY_SEQ_NBR for the life of the end-device.   |
| USER_ID            | The User ID associated with this Event Log entry. It comes from the Log In Procedure or from a communication session initiation sequence. | 0         | A USER_ID of zero indicates that an end-device internal process initiated the event. (e.g. as a result of a programmed or scheduled event).  |
|                    |   | 1         | A USER_ID of one indicates an event that was manually initiated (e.g. an external mechanical adjustment, non communication based tamper).  |
|                    |   | 2..15     | USER_IDS reserved for ANSI C12.22 communication module related events.   |
|                    |   | 16..65535 | USER_IDS assigned by the end-device agent (e.g. the Utility).  |
| EVENT_CODE         | Measurement Canada approved event codes used to transport end-device NEW_VALUES as described below.                                       |           | These codes extend the ANSI C12.19-1997 History Log & Event Log Codes. They will be adopted into version 2.0 of the C12.19 / IEEE 1377 standard, following Measurement Canada acceptance of these principles. An explanation of the EVENT_ARGUMENT format is provided below. |

**Chart 1.7**      Event Log Table entry elements' preamble (Continued)

| Table Element Name | Description  | Value | Interpretation   |
|--------------------|--|-------|--|
| EVENT_ARGUMENT     | The event trigger program data elements describing the new event value and related parameters and signature. |       | <p>An ANSI C12.19 / IEEE 1377 sub-record which identifies the affected metrological program, constant, data table number or procedure number. It also marks whether the action was communication triggered or as a result of a pending program activation. The exact content of the EVENT_ARGUMENT depends on the end-device implementation model.</p> <p>This is described in <a href="#">Section 1.3.10, "Rule for Option 1, EVENT_CODE and EVENT_ARGUMENT Usage in Self-contained Loggers"</a>, <a href="#">Section 1.3.11, "Rule for Option 2, EVENT_CODE and EVENT_ARGUMENT Usage by Signed Downloadable Event Loggers With No NEW_VALUES"</a> and <a href="#">Section 1.3.12, "Rule for Option 3, EVENT_CODE and EVENT_ARGUMENT Usage by Signed Event Loggers With NEW_VALUES"</a> next.</p> <p>When SIGNATURE is provided then it is encoded using the MD5 encryption algorithm.</p> <p>When Table program NEW_VALUE elements are provided they are encoded using PSEM write request sequences.</p> <p>When the EVENT_ARGUMENT does not fill the entire space provided it shall be padded with binary zeros, and not included in the MD5 digest (signature)</p> |

### 1.3.10 Rule for Option 1, EVENT\_CODE and EVENT\_ARGUMENT Usage in Self-contained Loggers

Event logger codes 58, 59, 60, 61 and 62 shall be used in implementations where the event logger is self-contained inside the end-device. The main feature of the EVENT\_ARGUMENT is that it always contains NEW\_VALUES. The NEW\_VALUES are stored in the EVENT\_ARGUMENT portion of each event entry and formatted using the PSEM write service or the PSEM partial write service sequence of requests.

The EVENT\_ARGUMENTS that are associated with the these event codes do not include an authentication mechanism (signature). The use of NEW\_VALUES without signatures has not yet been approved for use by Measurement Canada.

**The manufacturer is advised to obtain specific approval  
from Measurement Canada  
when using event codes 58, 59, 60, 61 or 62.**



**Chart 1.8** EVENT\_CODES for 100% embedded event loggers (Continued)

| Event Code | Event Description  | Argument Type                      | EVENT_ARGUMENT Description  |
|------------|--|------------------------------------|---|
| 59         | A single metrological Table modification event.  | TABLE_IDA_BFLD +<br><br>NEW_VALUES | <p>TABLE_IDA_BFLD identifies the metrological Table change that triggered this event. All the fields are identical to that of event code 58, except that TBL_PROC_NBR subelement is the table number.</p> <p>NEW_VALUES represent the collection of all changes that occurred as a result of the invocation of the metrological Table. These are encoded as PSEM write service sequence.</p> <p>The first PSEM write service request shall be a write service to the metrological table that triggered these changes.</p> |
| 60         | Metrological tables modified or programmed event.  | NEW_VALUES                         | <p>NEW_VALUES represent the collection of all changes that occurred in more than one metrological table. These are encoded as PSEM write service sequence.</p> <p>The PSEM write service requests shall be assembled so that they express all of the changes (new values) that took place as a result of the Table update.</p>  |
| 61         | Verification Event.<br>End-device was verified, tested and physically sealed. The end-device can now be used for revenue metering. | NEW_VALUES                         | <p>NEW_VALUES represent the content of all metrological tables that were configured prior to sealing the end-device. These are encoded as PSEM write service sequence.</p> <p>If the EVENT_ARGUMENT is too small to hold a copy of all metrological tables at the time of the verification, then it is expected that this event code shall be preceded by one or more instances of codes 58, 59 or 60 as needed to capture the state of all metrological tables as verified, approved and sealed.</p>                     |

**Chart 1.8**      `EVENT_CODES` for 100% embedded event loggers (Continued)

| Event Code | Event Description   | Argument Type           | <code>EVENT_ARGUMENT</code> Description  |
|------------|---|-------------------------|--|
| 62         | <p>Re-verification event.</p> <p>The state of the end-device was changed in such a manner that effectively (or actually) the seal was broken.</p> | <code>NEW_VALUES</code> | <p><code>NEW_VALUES</code> represent the collection of all changes that occurred to cause the re-verification event. These are encoded as PSEM write service sequence.</p> <p>If the <code>EVENT_ARGUMENT</code> is too small to hold a copy of all changes that triggered this event, this code may be preceded by one or more instances of codes 58, 59 or 60 as needed to capture the changes to the end-device metrological tables elements.</p> |

### 1.3.11 Rule for Option 2, `EVENT_CODE` and `EVENT_ARGUMENT` Usage by Signed Downloadable Event Loggers With No `NEW_VALUES`

Event codes 63, 64, 65, 66 and 67 are only applicable in implementations where the event logger is downloadable. The main feature of the `EVENT_ARGUMENT` is that it does not contain `NEW_VALUES` as a component of the `EVENT_ARGUMENT`. Instead the `EVENT_ARGUMENT` contains only an identification of the action that triggered the metrological change and an event entry signature (hash code). This signature (`EVENT_CHECK_SIG`) is computed using the MD5 algorithm.

The signature is 128 bits in length. It authenticates all elements of the event-logger record and links to the signature of the previous event logger record entry to the signature of all the metrological tables of the end device. Event signatures provide means to detect discontinuity in the recording process, under the assumption that the end-device meter agent shall keep a copy of the content of all metrological tables in its central record keeping system and a copy of all downloaded event log entries over the entire life of the end-device (life starts after a verification event).

The signature essentially signs the event-logger incrementally for the life of the end-device, thus it provides the means to demonstrate the event logger's integrity for the life of the end-device. All downloaded event logger records shall be signed with this algorithm ensuring that both the end-device program state and the event logger were securely downloaded and preserved in the central record keeping system, without accidental or intentional corruption or alteration.

When this event logger model is implement, the first entry created in the event log shall be the verification event (67).

**Chart 1.9** EVENT\_CODES for downloadable event loggers that are only signed

| Event Code | Event Description  | Argument Type                       | EVENT_ARGUMENT Description  |
|------------|--|-------------------------------------|---|
| 63         | A single metrological procedure invocation event.  | EVENT_CHECK_SIG +<br>TABLE_IDA_BFLD | <p>EVENT_CHECK_SIG is the event check signature for this event entry.</p> <p>TABLE_IDA_BFLD identifies the metrological Procedure that triggered this event. The TABLE_IDA_BFLD sub-elements are set as follows:</p> <p>The TBL_PROC_NBR sub-element is the procedure number.</p> <p>The STD_VS_MFG_FLAG sub-element is set to <b>FALSE</b> if the procedure is a Standard procedure; and it is set to <b>TRUE</b> if the Procedure is a manufacturer procedure.</p> <p>The PENDING_FLAG sub-element is set to <b>FALSE</b> if this procedure was invoked directly; it is set to <b>TRUE</b> if it is a pending procedure that was activated.</p> |
| 64         | A single metrological Table modification event.  | EVENT_CHECK_SIG +<br>TABLE_IDA_BFLD | <p>EVENT_CHECK_SIG is the event check signature for this event entry.</p> <p>TABLE_IDA_BFLD identifies the metrological Table change that triggered this event. All the fields are identical to that of event code 63, except that TBL_PROC_NBR sub-element is the table number.</p>  |
| 65         | Metrological tables modified or programmed event.  | EVENT_CHECK_SIG                     | <p>EVENT_CHECK_SIG is the event check signature. This event is triggered when more than one metrological table was changed, or when the reason for the change cannot simply be expressed using a single table identifier.</p>   |
| 66         | Verification Event.<br>End device was verified, tested and physically sealed. The end-device can now be used for revenue metering. | EVENT_CHECK_SIG                     | <p>EVENT_CHECK_SIG is the event check signature. Upon a verification event, the initial seed value for the EVENT_CHECK_SIG shall be set to zero. The initial EVENT_CHECK_SIG is then computed using the state of all metrological tables and the verification event preamble at the time of the verification event.</p>   |

**Chart 1.9**

EVENT\_CODES for downloadable event loggers that are only signed (Continued)

| Event Code | Event Description  | Argument Type   | EVENT_ARGUMENT Description   |
|------------|--|-----------------|--|
| 67         | Re-verification event.<br>The state of the end-device was changed in such a manner that effectively (or actually) the seal was broken. | EVENT_CHECK_SIG | EVENT_CHECK_SIG is the event check signature. This event is triggered when any metrological element was changed, in such a manner that it triggers a re-verification indication for this end-device. |

### **1.3.12 Rule for Option 3, EVENT\_CODE and EVENT\_ARGUMENT Usage by Signed Event Loggers With NEW\_VALUES**

Event codes 68, 69, 70, 71 and 72 are only applicable in implementations where the event logger is downloadable or self-contained. The main feature of the EVENT\_ARGUMENT is that it also contains NEW\_VALUES as a component of the EVENT\_ARGUMENT, in addition to the event entry signature (hash code). This signature is identical to that described in [Section 1.3.11](#), “Rule for Option 2, EVENT\_CODE and EVENT\_ARGUMENT Usage by Signed Downloadable Event Loggers With No NEW\_VALUES,” on page 1-36.

**Chart 1.10** EVENT\_CODES for downloadable event loggers that are signed with new values

| Event Code | Event Description                                 | Argument Type  | EVENT_ARGUMENT Description   |
|------------|---|--|--|
| 68         | A single metrological procedure invocation event. | <p>EVENT_CHECK_SIG +</p> <p>TABLE_IDA_BFLD +</p> <p>NEW_VALUES</p> | <p>EVENT_CHECK_SIG is the MD5 event check signature for this entry.</p> <p>TABLE_IDA_BFLD identifies the metrological Procedure that triggered this event. The TABLE_IDA_BFLD sub-elements are set as follows:</p> <p>The TBL_PROC_NBR sub-element is the procedure number.</p> <p>The STD_VS_MFG_FLAG sub-element is set to <b>FALSE</b> if the procedure is a Standard procedure; and it is set to <b>TRUE</b> if the Procedure is a manufacturer procedure.</p> <p>The PENDING_FLAG sub-element is set to <b>FALSE</b> if this procedure was invoked directly; it is set to <b>TRUE</b> if it is a pending procedure that was activated.</p> <p>NEW_VALUES represent the collection of all changes that occurred as a result of the invocation of the metrological procedure. These are encoded as PSEM write service sequence.</p> <p>The first PSEM write request shall be a write service request to Table 7, "Procedure Initiate", and it will minimally include the PROC element (formatted as TABLE_IDB_BFLD), which is the first element of Table 7.</p> <p>Additional PSEM write service requests shall follow, as needed, to document all of the metrological changes (new values) that resulted from this Procedure invocation.</p> |

**Chart 1.10**      `EVENT_CODES` for downloadable event loggers that are signed with new values (Continued)

| Event Code | Event Description  | Argument Type  | EVENT_ARGUMENT Description  |
|------------|--|--|---|
| 69         | A single metrological Table modification event.  | <code>EVENT_CHECK_SIG +</code><br><br><code>TABLE_IDA_BFLD +</code><br><br><code>NEW_VALUES</code> | <p><code>EVENT_CHECK_SIG</code> is the MD5 event check signature for this entry.</p> <p><code>TABLE_IDA_BFLD</code> identifies the metrological Table change that triggered this event. All the fields are identical to that of event code 63, except that the <code>TBL_PROC_NBR</code> sub-element is the table number.</p> <p><code>EVENT_CHECK_SIG</code> is the event check signature for this event entry.</p> <p><code>NEW_VALUES</code> represent the collection of all changes that occurred as a result of the invocation of the metrological Table. These are encoded as PSEM write service sequence.</p> <p>The first PSEM write service request shall be a write service to the metrological table that triggered these changes.</p> <p>The PSEM write service requests shall be assembled so that they express all of the changes (new values) that took place as a result of the Table update.</p> |
| 70         | Metrological tables modified or programmed event. This event is triggered when more than one metrological table was changed, or when the reason for the change cannot be simply expressed using a single table identifier. | <code>EVENT_CHECK_SIG +</code><br><br><code>NEW_VALUES</code>                                      | <p><code>EVENT_CHECK_SIG</code> is the MD5 event check signature for this entry.</p> <p><code>NEW_VALUES</code> represent the collection of all changes that occurred as a result of the modification to the metrological tables. These are encoded as PSEM write service sequence.</p> <p>The PSEM write service requests shall be assembled so that they express all of the changes (new values) that took place as a result of the Table updates.</p>  |

**Chart 1.10** EVENT\_CODES for downloadable event loggers that are signed with new values (Continued)

| Event Code | Event Description  | Argument Type                       | EVENT_ARGUMENT Description  |
|------------|--|-------------------------------------|---|
| 71         | Verification Event.<br>End device was verified, tested and physically sealed. The end-device can now be used for revenue metering.     | EVENT_CHECK_SIG +<br><br>NEW_VALUES | <p>Upon a verification event, the initial seed value for the EVENT_CHECK_SIG shall be set to zero. The initial EVENT_CHECK_SIG is then computed using the state of all metrological tables and the verification event preamble at the time of the verification event.</p> <p>NEW_VALUES represent the content of all metrological tables that were configured prior to sealing the end-device. These are encoded as PSEM write service sequence.</p> <p>If the EVENT_ARGUMENT is too small to hold a copy of all metrological tables at the time of the verification, then it is expected that this event code shall be preceded by one or more instances of codes 68, 69 or 70 as needed to capture the state of all metrological tables as verified, approved and sealed.</p> |
| 72         | Re-verification event.<br>The state of the end-device was changed in such a manner that effectively (or actually) the seal was broken. | EVENT_CHECK_SIG +<br><br>NEW_VALUE  | <p>EVENT_CHECK_SIG is the event check signature. This event is triggered when any metrological element was changed, in such a manner that it triggers a re-verification indication for this end-device.</p> <p>NEW_VALUES represent the collection of all changes that occurred to cause the re-verification event. These are encoded as PSEM write service sequence.</p> <p>If the EVENT_ARGUMENT is too small to hold a copy of all changes that triggered this event, this code may be preceded by one or more instances of codes 68, 69 or 70 as needed to capture the changes to the end-device metrological tables elements.</p>  |

### 1.3.13 Rule for Verification Event

The manufacturer of the end-device shall provide means to generate a verification event to indicate that the end-device has been verified, approved and physically sealed. The technique used to generate a verification event can be mechanical or used configuration software. The verification event shall generate an event log entry with `EVENT_CODE=61, 66 or 71`.

### 1.3.14 Rule for Issuing a Re-verification Event

Changes to any sealed parameter or access to metrological adjustments that are not recorded by the event logger shall be considered a verification triggering event (unless specifically excluded by Measurement Canada).

An event-logger entry of a re-verification event is analogous to the breaking of a physical seal. For this reason, the end device shall provide direct and clear re-verification status indicator flags, in addition to all other visual indicators.

The program status flags in Table 3, “End-Device Mode Status”, shall be set upon a re-verification event, as follows:

**Chart 1.11** Re-verification status indicators

| Table Element Name                | Description  | Value        | Interpretation   |
|-----------------------------------|--|--------------|--|
| <code>METERING_FLAG</code>        | Indicates whether the end-device is metering and verified for trade use.         | <b>FALSE</b> | A re-verification event shall set this flag to <b>FALSE</b> , but the end-device shall continue to meter according to its active program.  |
| <code>TEST_MODE_FLAG</code>       | Indicates end-device is running in test mode.                                    | <b>FALSE</b> | This flag may be set to <b>TRUE</b> only when the end-device is physically sealed and the Manufacturer of the end device demonstrates to the satisfaction of Measurement Canada that “test mode” does not impact on the end-device’s ability to operate correctly for trade use. |
| <code>METER_SHOP_MODE_FLAG</code> | Indicates that the end-device is running in meter-shop mode.                     | <b>FALSE</b> | This flag is not permitted to be set to <b>TRUE</b> when the end-device is physically sealed.  |
| <code>UNPROGRAMMED_FLAG</code>    | Indicates that the end-device has lost its program or has never been programmed. | <b>TRUE</b>  | When the end-device is approved, verified, sealed and <code>METERING_FLAG</code> is set to <b>FALSE</b> then a re-verification event was triggered.  |

Chart 1.11 Re-verification status indicators (Continued)

| Table Element Name       | Description  | Value       | Interpretation   |
|--------------------------|--|-------------|--|
| CONFIGURATION_ERROR_FLAG | Indicates that the active program or configuration of the end-device is not consistent with its uses, as protected by a physical seal. | <b>TRUE</b> | When the end-device is approved, verified, sealed and METERING_FLAG is set to <b>FALSE</b> then a re-verification event was triggered. |
| SELF_CHK_ERROR_FLAG      | Indication that an inconsistency exists between the physical end-device operating requirements and its active program or data values.  | <b>TRUE</b> | When the end-device is approved, verified, sealed and METERING_FLAG is set to <b>FALSE</b> then a re-verification event was triggered. |

### 1.3.15 Rule for Downloading The Event Logger

The act of Downloading is the confirmed and validated process of transferring a copy of the content of end-device Table elements into the end-device-agent’s record-keeping system. This process shall be reliable, secure, and demonstratively verifiable.

Event-log Table or Table elements downloading is in addition requires the following steps:

1. Minimally downloading the newly created Event Log Table entries (Table 76, ENTRIES), as indicated by the element NBR\_UNREAD\_ENTRIES in Table 76, “Event Log Data”; and
2. Shall securely store the changed table values (event loggers with no new values) or the NEW\_VALUES (event loggers with new values) in the record keeping system in a manner that enables the reconstruction of the modified tables as they exist inside the end-device; and
3. Shall download the end-device IDENTIFICATION element from Table 5, “Device Identification” or equivalently the element DEVICE\_ID of Table 6, “Utility Information”; and
4. Shall preserve the relationship between LAST\_ENTRY\_SEQ\_NBR and the entry’s EVENT\_SEQ\_NBR, effectively extending the EVENT\_SEQ\_NBR to 32 bits; and
5. Shall invoke the “Update Last Read Entry” procedure after the successful verified completion of the download process.

### 1.3.16 Rule for Updating Last Read Entry or Resetting the Pointers

#### 1.3.16.1 Resetting Any List Pointers

If the end-device is verified, approved and physically sealed, this action shall trigger a re-verification event.

### 1.3.16.2 Updating Non Event-Log Table List Pointers

This is a normal operation and it shall not trigger the creation of an event log entry to be created.

### 1.3.16.3 Updating The Event-Log Table List Pointers

The invocation of this procedure is permitted only by the successful download of the event logger as outlined in [Section 1.3.15, “Rule for Downloading The Event Logger,” on page 1-43](#). The following shall be the final state of the Event Log Table list management elements after invocation of the Standard “Update Last Read Entry” Procedure with a non zero ENTRIES\_READ parameter:

---

**Chart 1.12** Event log list management elements

| Event Log Table Element Name  | Value After List Pointers Update   |
|-------------------------------|--|
| ORDER                         | Value unchanged.   |
| OVERFLOW_FLAG                 | Set to <b>FALSE</b> to clear overflow indication.  |
| LIST_TYPE                     | Value unchanged.   |
| INHIBIT_OVERFLOW_FLAG         | Value unchanged.   |
| NBR_VALID_ENTRIES             | Increments by one if the event log tables contains entries that were never populated. Otherwise the value remains unchanged. |
| LAST_ENTRY_ELEMENT            | Assigned the index value of the “Update List Pointers” event record.   |
| LAST_ENTRY_SEQ_NBR            | Increments by one.   |
| NBR_UNREAD_ENTRIES            | Decremented by (ENTRIES_READ - 1).   |
| ENTRIES[ LAST_ENTRY_ELEMENT ] | The actual “Update List Pointers” event record.  |

In addition all of the MODIFIED\_FLAG in “Signatures Table (Proposed Std Table 78, Rev 2.0)” on page 3-92, shall be set to **FALSE**.

---

## 1.4 Event Logger Implementation Algorithm

---

### 1.4.1 Event Signature Computation - EVENT\_CHECK\_SIG

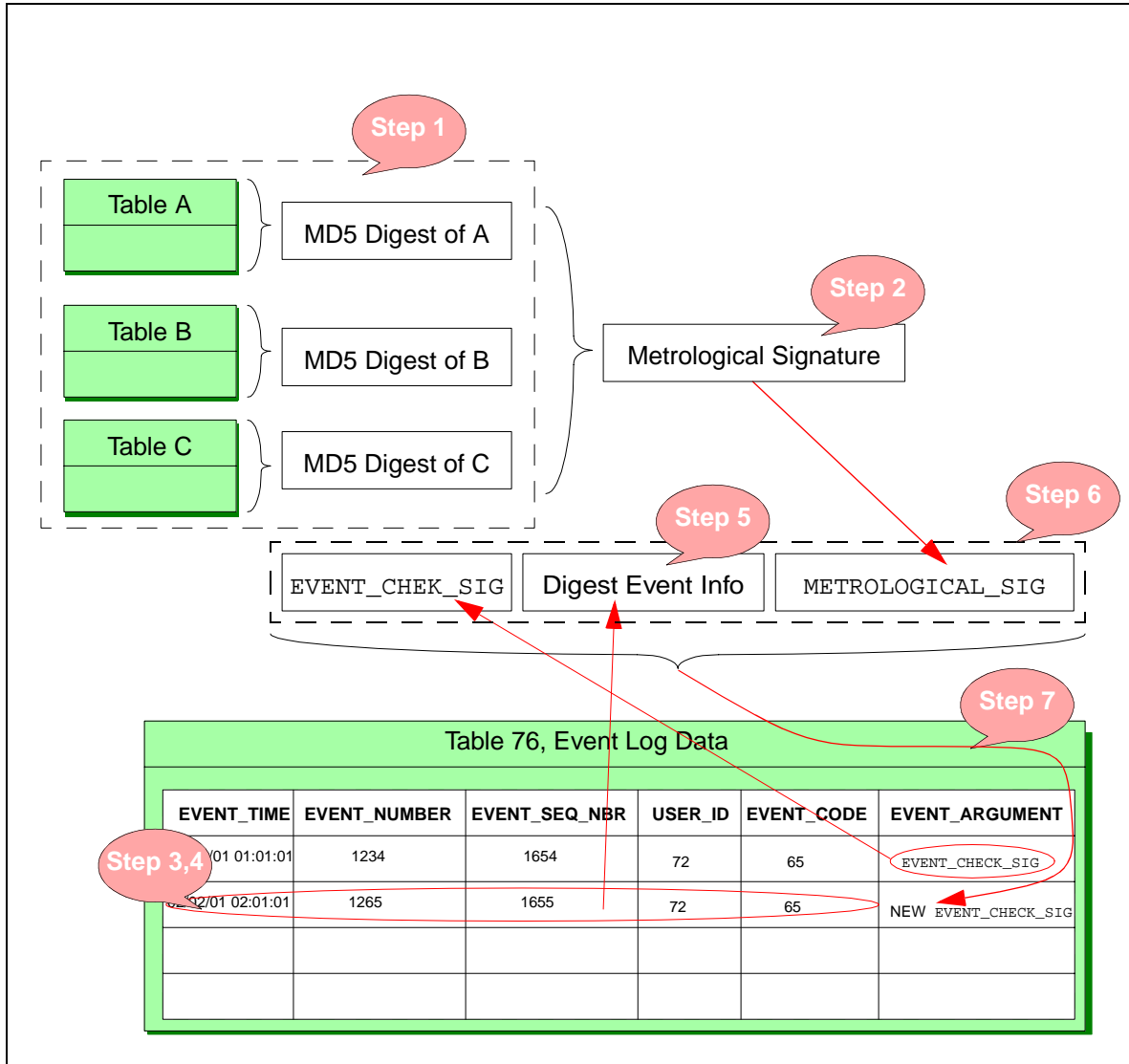
The event entry signature is computed as follows:

1. Using the MD5 algorithm, compute and save the individual Table Signatures (digests) of each metrological Table separately, by placing them into an array that is ordered by table number. When Table 77, “Signatures”, is available, these signatures are stored in the elements TABLE\_SIG. This step can be easily optimized by computing a Table’s signature only when a table is programmed or updated.
2. Again using the MD5 algorithm, compute a Metrological Signature, which is the digest of the concatenated signatures produced in step 1, above. The computation

shall be performed sequentially by concatenating all the signatures of the metrological tables in increasing table number. Standard tables are processed first followed by Manufacturer tables. Only the signatures of metrological tables shall be included in the computation. When Table 77, "Signatures", is available, this signature is stored in `METROLOGICAL_SIG`.

3. Create a new event entry in Table 76, "Event Log Data" and initialize its `EVENT_ARGUMENT` element to all binary 0.
4. Fill in the `EVENT_TIME`, the optional `EVENT_NUMBER`, and the required `EVENT_SEQ_NBR`, `USER_ID` and `EVENT_CODE` and then the `NEW_VALUES` portion of `EVENT_ARGUMENT` (`TABLE_IDAS` and `NEW_VALUES`, are placed after the `EVENT_CHECK_SIG`).
5. Again, using the MD5 algorithm, digest all the new event information that was placed in the `EVENT_ENTRY_RCD`. This includes all non `EVENT_CHECK_SIG` elements of the `EVENT_ARGUMENT`, as applicable, based on the value of the `EVENT_CODE` (for example: when `EVENT_CODE=65`, the `EVENT_ARGUMENT` will not be digested; when `EVENT_CODE=64`, only the `TABLE_IDA` field of the `EVENT_ARGUMENT` will be digested; when `EVENT_CODE=70`, then all octets starting at offset 16 to offset `ACT_LOG_TBL.EVENT_DATA_LENGTH-1` of the `EVENT_ARGUMENT` will be digested).
6. Fetch the previous entry's `EVENT_CHECK_SIG` (or use a signature of binary 0 if this is the first event entry being created), append to it the `EVENT_ENTRY_RCD` signature (computed in step 5 above), then the metrological signature, that was computed in step 2 and digest the concatenated signature with the MD5 algorithm to produce the event logger entry signature. When Table 77, "Signatures", is available, this value should be the same one found in `EVENT_CHECK_SIG`.
7. Store the resulting signature in the octet 0..15 of the `EVENT_ARGUMENT` in the new event entry; and when Table 77, "Signatures", is available, this value shall also be stored in the `EVENT_CHECK_SIG`.
8. Done

Figure 1.4 Event Log Signature Computation



## 1.4.2 Event Argument Construction - EVENT\_ARGUMENT

### 1.4.2.1 Construction of EVENT\_ARGUMENT with NEW\_VALUES

NEW\_VALUES are added to the tail end of the EVENT\_ARGUMENT when EVENT\_CODE=58, 59, 60, 61, 62 (self-contained event loggers) or EVENT\_CODE=68, 69, 70, 71, 72 (downloadable and downloadable event loggers). The NEW\_VALUES presented as a list of PSEM full or partial Write Service Request. For example, let's assume that Table 11, "Actual Sources Limiting", and Table 15, "Constants" have been set up as follows:

Chart 1.13

Example of embedding Tables 11 and 15 as NEW\_VALUES

| Table Number | Element Name         | Value  |
|--------------|----------------------|--------|
| Table 11     | NBR_CONSTANT_ENTRIES | 1      |
| Table 15     | MULTIPLIER           | 0.0072 |
|              | OFFSET               | 0.0    |

Using the above information, the EVENT\_ARGUMENT shall have the following binary value (expressed in hexadecimal notation)

```
0x4F000B000005000101FF40000F00080D0072BB0D0BBBBB38
```

Where

- 4F is the PSEM partial write with offset service request code;
- 000B is the table identifier (0x000B = 11 decimal);
- 000005 is the offset = 5 to NBR\_CONSTANT\_ENTRIES, in octets;
- 0001 is a count = 1 octet, representing the number of data octets stored;
- 01 is the value of NBR\_CONSTANT\_ENTRIES = 1;
- FF is the PSEM request fragment's checksum;

and

- 40 is the next PSEM full write service request code.
- 000F is the table identifier (0x000F = 15 decimal).
- 0008 is a count = 8 octets, representing the number of data octets stored.
- 0D0072BB is the value of the MULTIPLIER=0.0072 (encoded in BCD, i.e. this assumes that the element NI\_FORMAT1= 6, in Table 0, "General Configuration");
- 0D0BBBBB is the value of the OFFSET = 0.0 in BCD (also encoded in BCD, i.e. this assumes that the element NI\_FORMAT1= 6, in Table 0, "General Configuration");
- 38 is the PSEM request fragment's checksum.

All unused trailing octets of the EVENT\_ARGUMENT will be set to zero. This also ensures clear and ambiguous PSEM request sequence termination, as PSEM requests cannot be zero.

**1.4.2.2 Construction of EVENT\_ARGUMENT with EVENT\_CHECK\_SIG without NEW\_VALUES**

This construction is relevant when the EVENT\_CODE=63, 64, 65, 66 or 67, as discussed in [Section 1.3.11 on page 1-36](#). The EVENT\_ARGUMENT can have the following components:

1. An EVENT\_CHECK\_SIG is an **ARRAY[16] OF UINT8** used to store the hash code produced by the MD5 algorithm. The EVENT\_CHECK\_SIG is placed first in the EVENT\_ARGUMENT (i.e. at octets 0..15).
2. TABLE\_IDA\_BFLD, a 16 bits element, **UINT16**. that provides for a Procedure or Table identification as follows:

**Chart 1.14**

TABLE\_IDA Field Description

| Sub-Element Name | Bit Range | Description  |
|------------------|-----------|--|
| TBL_PROC_NBR     | 0..10     | Table or Procedure number  |
| STD_VS_MFG_FLAG  | 11..11    | Standard or Manufacturer Table or Procedure.<br><br>This bit is set to <b>FALSE</b> if this is a Standard Table or Procedure; and it is set to <b>TRUE</b> if this is a Manufacturer defined Table or Procedure. |
| PENDING_FLAG     | 12..12    | This bit is set to <b>FALSE</b> if this Table or Procedure was invoked directly; it is set to <b>TRUE</b> if it is a pending Table or Procedure that was activated.  |
| Filler           | 13..15    | Filled with binary 0's.  |

When the TABLE\_IDA element is present, it follows the EVENT\_CHECK\_SIG element.

For example, let's assume that Table 15, "Constants", was modified so that new EVENT\_CHECK signature for this event entry is:

0X9D47913E0C8F797AD4A7233E407C8794

As a result of this change an event log entry shall be created with EVENT\_CODE=64 and the EVENT\_ARGUMENT will be:

0X9D47913E0C8F797AD4A7233E407C87940F00

Where

9D47913E0C8F797AD4A7233E407C8794

Occupies the first 16 octets of the EVENT\_ARGUMENT, and is the EVENT\_CHECK\_SIG hash code for this event;

0F00

is a Standard Table 15 identifier (using TABLE\_IDA, where DATA\_ORDER=0 in Table 0, "General Configuration", i.e. little-endian architecture, in this example).

All unused trailing octets of the EVENT\_ARGUMENT will be set to zero.

### 1.4.2.3 Construction of EVENT\_ARGUMENT with EVENT\_CHECK\_SIG with NEW\_VALUES

This construction is relevant when the EVENT\_CODE=68, 69, 70, 71 or 72, as discussed in [Section 1.3.12 on page 1-38](#). The EVENT\_ARGUMENT can have the following components:

1. An EVENT\_CHECK\_SIG is an **ARRAY[16] OF UINT8** used to store the hash code produced by the MD5 algorithm. The EVENT\_CHECK\_SIG is placed first in the EVENT\_ARGUMENT (i.e. at octets 0..15).
2. TABLE\_IDA\_BFLD, a 16 bits element, **UINT16**, that provides for a Procedure or Table identification as shown in [Chart 1.14, "TABLE\\_IDA Field Description," on page 1-48](#). When the TABLE\_IDA element is present, it follows the EVENT\_CHECK\_SIG element.
3. NEW\_VALUES that are added to the tail end of the EVENT\_ARGUMENT, as discussed in [Section 1.4.2.1, "Construction of EVENT\\_ARGUMENT with NEW\\_VALUES," on page 1-47](#).

For example, let's assume that Table 11, "Actual Sources Limiting", and Table 15, "Constants" were modified with the exact same values that were used in the example of [Section 1.4.2.1 on page 1-47](#), but with the addition of an EVENT\_CHECK\_SIG value of:

```
0x9D47913E0C8F797AD4A7233E407C8794
```

As a result of this change an event log entry shall be created with EVENT\_CODE=70 and the EVENT\_ARGUMENT will be:

```
0X9D47913E0C8F797AD4A7233E407C87944F000B000005000101FF40000F00080D0
072BB0D0BBBBB38
```

Where

```
9D47913E0C8F797AD4A7233E407C8794
```

Occupies the first 16 octets of the EVENT\_ARGUMENT, and is the EVENT\_CHECK\_SIG hash code for this event; and

and where

```
4F is the PSEM partial write with offset service request code;
000B is the table identifier (0x000B = 11 decimal);
000005 is the offset = 5 to NBR_CONSTANT_ENTRIES, in octets;
0001 is a count = 1 octet, representing the number of data octets stored;
01 is the value of NBR_CONSTANT_ENTRIES = 1;
FF is the PSEM request fragment's checksum;
```

and

```
40 is the next PSEM full write service request code.
000F is the table identifier (0x000F = 15 decimal).
0008 is a count = 8 octets, representing the number of data octets stored.
0D0072BB is the value of the MULTIPLIER=0.0072 (encoded in BCD, i.e. this assumes that the
element NI_FORMAT1= 6, in Table 0, "General Configuration");
0D0BBBBB is the value of the OFFSET = 0.0 in BCD (also encoded in BCD, i.e. this assumes
that the element NI_FORMAT1= 6, in Table 0, "General Configuration");
```

38 is the PSEM request fragment's checksum.

All unused trailing octets of the `EVENT_ARGUMENT` will be set to zero. This also ensures clear and ambiguous PSEM request sequence termination, as PSEM requests cannot be zero.

### 1.4.3 Event Log Entry Construction, a Complete Example

This is a complete example of the computation of the event logger signature of an end-device that contains two metrological tables: Table 11, "Actual Sources Limiting", and Table 13, "Demand Control". In this example we assume that the present value of the event logger signature is: 0x4AE71336E44BF9BF79D2752E234818A5.

#### 1.4.3.1 Computing the Signatures of the Metrological Tables.

We assume that Table 11 elements are populated with the following values:

| Table Element Name      | Value        |
|-------------------------|--------------|
| PF_EXCLUDE_FLAG         | <b>FALSE</b> |
| RESET_EXCLUDE_FLAG      | <b>TRUE</b>  |
| BLOCK_DEMAND_FLAG       | <b>FALSE</b> |
| SLIDING_DEMAND_FLAG     | <b>TRUE</b>  |
| THERMAL_DEMAND_FLAG     | <b>FALSE</b> |
| SET1_PRESENT_FLAG       | <b>FALSE</b> |
| SET2_PRESENT_FLAG       | <b>FALSE</b> |
| NBR_UOM_ENTRIES         | 2            |
| NBR_DEMAND_CTRL_ENTRIES | 1            |
| DATA_CTRL_LENGTH        | 0            |
| NBR_DATA_CTRL_ENTRIES   | 0            |
| NBR_CONSTANTS_ENTRIES   | 2            |
| CONSTANTS_SELECTOR      | 2            |
| NBR_SOURCES             | 2            |

When Table 11 is binary encoded it shows as: 0x0A02010000020202. The MD5 signature of this table is: 0X317F4AEA0D462DFC7B59A0614D2E82FD.

We also assume that Table 13 elements are populated with the following values:

| Table Element Name | Value |
|--------------------|-------|
| RESET_EXCLUSION    | 60    |
| SUB_INT            | 5     |
| INT_MULTIPLIER     | 3     |

When Table 13 is binary encoded it shows as: 0x3C0503. The MD5 signature of this table is: 0X4412FCB078308A391DF6CE481D6DE544.

### 1.4.3.2 Computing the Metrological Signature.

The metrological signature is computed by concatenating the signatures of Tables 11 and 13, which were computed on [page 1-50](#) as follows:

```
0X317F4AEA0D462DFC7B59A0614D2E82FD+4412FCB078308A391DF6CE481D6DE544
```

Then digesting it with the MD5 algorithm to produce the metrological signature:  
0X182AB8148AF56F7F23833224E4CE29AA.

### 1.4.3.3 Assembling the Event Logger Preliminary Entry.

An event entry is created in Table 76, “Event Log Data”, with the following sample information:

| Table Element Name | Value               |
|--------------------|---------------------|
| EVENT_TIME         | 2001/09/19 18:29:01 |
| EVENT_SEQ_NBR      | 847                 |
| USER_ID            | 43                  |
| EVENT_CODE         | 65                  |

This event (according to the `EVENT_CODE=65`), does not provide additional embedded information beyond the event signature.

The preliminary event record is binary encoded as follows:  
0X0109191829014F032B004100.

### 1.4.3.4 Computing the Event Logger Entry Final Signature.

The `EVENT_CHECK_SIG` is computed by the concatenation of:

1. The previous value of the event logger signature, 0X4AE71336E44BF9BF79D2752E234818A5; and
2. The preliminary event logger record value, 0x0109191829014F032B004100 and;
3. The metrological signature, 0x182AB8148AF56F7F23833224E4CE29AA.

```
0X4AE71336E44BF9BF79D2752E234818A5+0109191829014F032B004100+182AB8148AF56F7F23833224E4CE29AA
```

To yield the final event logger signature: 0x68B35CDFE403C02F51CEA4427B9B2272. This value is inserted as the `EVENT_CHECK_SIG` starting of offset 0 of the `EVENT_ARGUMENT`, thus completing the creation of the event log entry.



---

# CHAPTER **2**

## **ANSI C12.19 / IEEE 1377 UTILITY INDUSTRY STANDARD METROLOGICAL TABLES**

---

---

### **2.1 Introduction**

---

This Chapter provides implementation guidelines for the classification of ANSI C12.19 / IEEE 1377 Utility Industry End Device Standard Program Tables as Metrological Table. It starts with a short description of the Utility Industry Standard Table (Section 2.2 on page 2-54). This introduction is sufficient for gaining a broad understanding of the table allocation of metrological services. However, the ANSI C12.19 / IEEE 1377 Standard document is over two hundred pages in length and, like most standards, very technical.

The more technically inclined individual, an engineer or an implementer testing apparatus is encouraged to:

1. Attend training on this subject. Training is offered by Future DOS R&D Inc., Nertec Design Inc. and Tucker Engineering Associates Inc.;
2. Attend a one or two-day tutorial that is offered by AMRA.
3. Study the Utility Industry Standard Tables User's Guide, written by Dr. A. Moise and published by Measurement Canada.
4. Read the actual Standards as published by ANSI/NEMA and AMRA/IEEE.

If you are already familiar with ANSI C12.19 / IEEE 1377, Tables Standard and related communication protocols skip Section 2.2, "The Utility Standard Tables, in a nut shell," on page 2-54, and continue reading Section 2.3, "Metrological Tables," on page 2-60.

---

## 2.2 The Utility Standard Tables, in a nut shell

---

Advances in metering and monitoring technology over the last two decades have led to the introduction of a bewildering array of proprietary communications equipment and protocols. Each vendor produced products which best fit their style and philosophy of manufacturing and fiercely protected their protocols and data formats. The users of this technology were faced with the equally onerous choices of duplicating data retrieval and processing products and services in order to take advantage of competitive purchasing practices, or locking in with one manufacturer, and thus giving up the benefits of competition, in order to simplify the retrieval and processing of data. In an effort to overcome this impasse, industry and trade organizations began to investigate the possibility of a communications protocol standard, the result of which was the establishment of the ANSI C12.19 / IEEE 1377 Standards.

The ANSI C12.19 / IEEE 1377 tables are nothing more than templates for the transportation of data. The best analogy found to date for use of tables by AMR (Automated Meter Reading) tools is that of Revenue Canadian Tax Form T1 (or US Federal General Form 1040) of the Individual Income Tax Return. The form represents an ordered list of the information required by internal revenue services to determine the extent and disposition of the tax obligation.

The authorities expect one to:

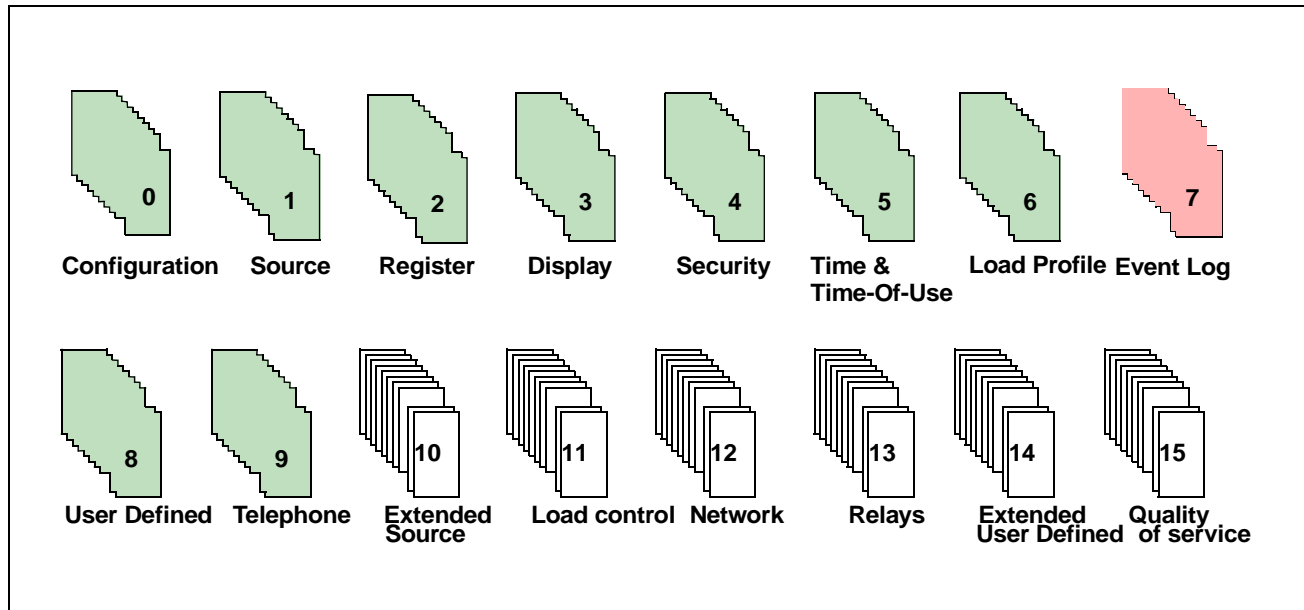
1. Collate the data requested in a specific order and presentation format.
2. Deliver the data by any available means to the processing center before the tax deadline.

In a similar manner:

1. The end-device (meter) only needs to be able to encapsulate the data requested, in the proper order and format; and
2. when requested to do so by the end-device reader, for the purpose of transportation.

The Utility Industry Tables are, therefore, a template for describing end-device related functions and the encapsulation details of program and data for use to program, re-program and to deliver metering data. The related program or data elements are grouped together in a single structure for transport (e.g. Table 23, current registers table co-locates kWh and kVARh readings). The tables are further grouped by function, also known as Decades, ten tables per Decade. These are explained next:

**Figure 2.1** ANSI C12.19 / IEEE 1377 Utility Industry Data Tables Versions 1 & 2, decades functional description.



Decades 0-8, are defined in Version 1.0 of ANSI C12.19-1997. Decade 9 is described in ANSI C12.21-1999. Decades 10-15 are proposed in version 2.0 of ANSI C12.19. The above extensions do not impact on the implementation of the event logger tables, therefore they will not be discussed here. The following is a description of the standard decades:

### 2.2.1 Decade 0 - End Device Configuration, Identification and Procedure Tables

Decade 0 contains tables associated with end device capabilities, configuration and identification. It also contains the special procedure and response tables. Decade 0 is somewhat special since it contains the General Configuration Table, which is the program and data tables master directory.

### 2.2.2 Decade 1 - Sensor Data Source Selection Tables

Decade 1 contains tables associated with end device sensors input source identification and characterization. These include the description of the physical inputs, input descriptors, input multipliers and meter constants and demand interval descriptors.

### **2.2.3 Decade 2 - Metrological Data Register Tables**

Decade 2 contains the control and data tables associated with measurements (data registers). These tables contain selections into recorded values and controls for the registration of the recorded data as total energy readings, peak demands and coincidental readings. The registers are also segmented into present data (real time) and current data (associated with billing cycle), previous season data, previous demand data and self read data.

When event counters are implemented then they will manifest themselves in the present data (real-time) registers Table 28.

#### **2.2.3.1 Decade 3 - Local Display Tables**

Decade 3 contains tables that control the selections, sequences and timing of displayed metered quantities. Selections can be made from any table element of an end-device.

### **2.2.4 Decade 4 - Access and Change Security Tables**

Decade 4 is a collection of tables, which contain passwords, authentication codes, and programming permissions used to manage access to metering data tables and re-programming any table in the end-device. The table re-programming master controls are found in Table 0, General Configuration, which define the list of tables and procedures implemented in the end-device and the global tables access mode: read only or read/write.

### **2.2.5 Decade 5 - Clock, Calendar, Time and Time of Use Tables**

Decade 5 contains the clock, clock-state, calendar, operating schedules, control functions and time-of-use register time allocations for end-devices that are equipped with a clock/calendar and/or time-of-use functions. The actual data values are recorded metrological data register Tables found in Decade 2.

### **2.2.6 Decade 6 - Load Profile Control and Recorded Tables**

Decade 6 contains tables for programming and operating up to four independent load profile recorders. Each recorder operates independently and is multi-channel. The recorders are not only independent from each other, but they also operate independently from Decade 2 and Decade 5, with the exception that they share the Clock when time-stamping the interval data blocks.

### **2.2.7 Decade 7 - History Log and Event Log**

Decade 7 contains two independent audit trail recorders: the History Log and Event Log.

Table 76, Event Log Data Table, contains the embedded portion of the event logger data.

### 2.2.7.1 History Log

The History Log contains programming instructions and selections of events and stores historical data that is useful as a general purpose logging mechanism. The long term storage and access to this data is not necessarily reliable or secured.

**Table 74, History Log Data Table, is the general purpose history log table.**

### 2.2.7.2 Event Log

The Event Log, as mandated by Measurement Canada, is a reliable and secured recorder of events to be used as the audit trail for reprogramming of approved re-programmable metering device while under physical seal.

**Table 76, Event Log Data Table, is the embedded portion of the event logger data.**

This table realizes Measurement Canada "Interim Specifications (/ Procedures) Relating to Event Loggers for Electricity Metering Devices and Systems", IS-E-01-E / IP-E-01 and EPS-EGMV-XX-E, for re-programming ANSI C12.19 / IEEE 1377 Standard metering devices, which operate an event logger; and implements the event logger principles proposed by the Measurement Canada Task Force on Metering Data Communication Protocol.

## 2.2.8 Decade 8 - User Defined (selections) Tables

Decade 8 contains the User Defined Tables, which provide the means of customizing the data table retrieval process. This process allows the end-device to select elements from any tables (readings and values) and collect them into one table, in sequence. As a result it is possible to access or modify a single table, instead of having to program or access the individual elements that may be scatter across many tables.

This is an optimization process and it cannot be used to circumvent the basic security mechanisms of the end-device or compromise the integrity of the event-logger or the event-counter or the physical seal.

## 2.2.9 Decade 9 - Telephone Control

Decade 9 contains the Tables associated with the use of a telephone modem communication. These tables are used in configuration, control of call and access windows, the report of call status and the transport of Tables using PSEM. This decade has been introduced into Standard as ANSI C12.21 in 1999.

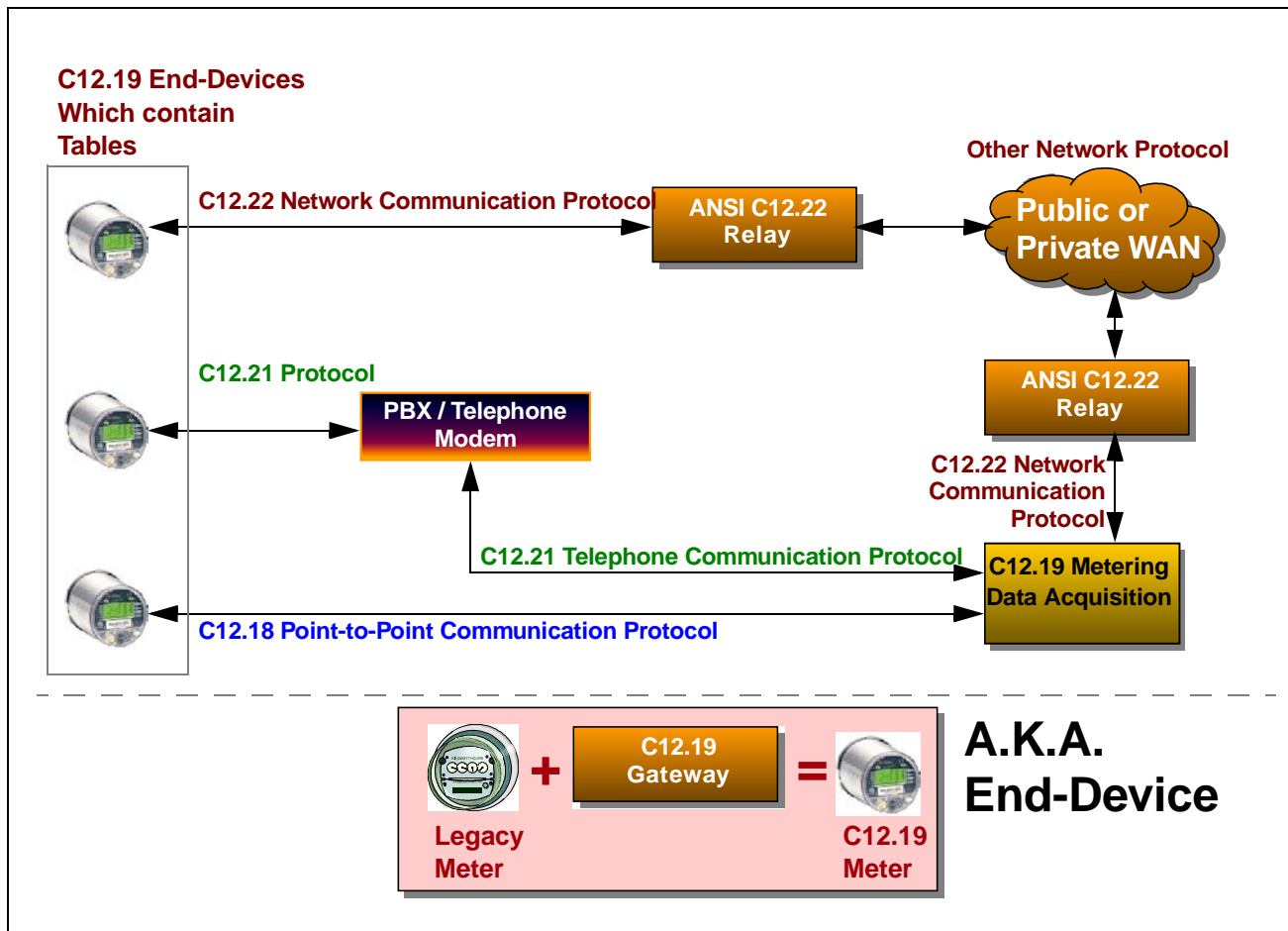
## 2.2.10 Communication Protocols

The ANSI C12.19 / IEEE 1377 Standard was developed in conjunction with a set of data transmission standards, especially designed to accommodate the transmission of the Utility Industry Standard Tables over as many communication mediums as possible. The communication protocols suite includes:

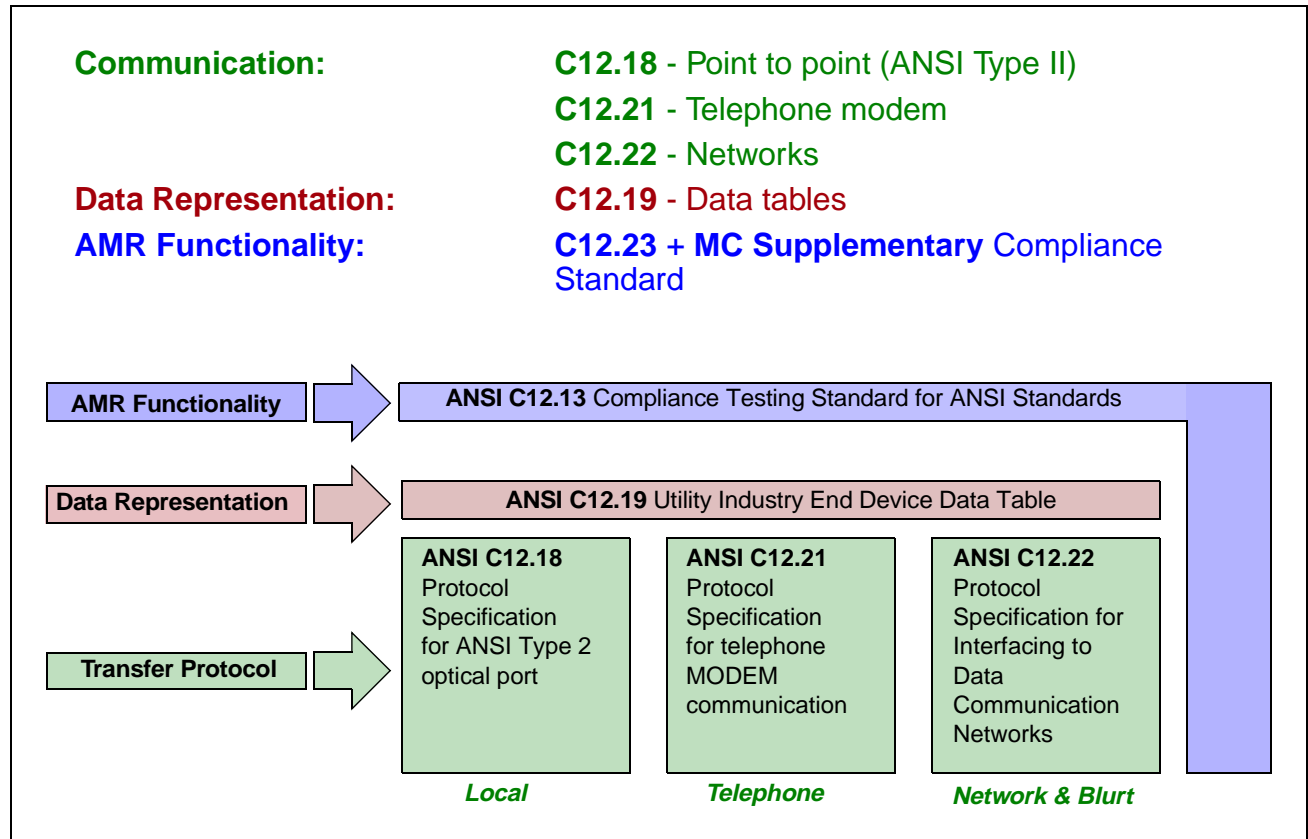
- ANSI C12.18-1996, Protocol Specifications for ANSI Type 2 Optical Port, NEMA, April 1996.
- ANSI C12.21-1999, Protocol Specification for Telephone MODEM Communication, NEMA, October 1999.
- ANSI C12.22-200x, Extended PSEM, Protocol Specification for Networks, NEMA, 200x.

Efforts are now under way to develop a Standard test suite under the sponsorship of Measurement Canada, NEMA and IEEE/AMRA SCC31. This work will be published as “ANSI C12.23 Compliance Testing for Standards Protocols C12.19 and C12.18/21/22”.

**Figure 2.2** Where are the ANSI C12.19 / IEEE 1377 data tables residing and how do they get transported?



**Figure 2.3** Utility Industry Standard Tables protocol suite, for energy market participants



## 2.3 Metrological Tables

---

The objective of this Section is to assist the implementor of an ANSI C12.19 / IEEE 1377 Standard based end-device and the Regulator (e.g. Measurement Canada) to identify and associate Standard Tables with one of the three types listed below:

1. Metrological Program Tables;
2. Metrological Data Tables;
3. Program or Data that are Non-Metrological Tables;



The identified tables, elements and procedures are provided as a general guide. This is not a statement of policy of Measurement Canada. What is or is not metrological is a purely technical issue that ultimately needs to be addressed when a manufacturer submits an end-device for approval. However, given that the Standards was designed to describe data tables that may be used in trade, and when the intended of use of the tables is clear this guide provide an indication of what tables are likely to be considered Metrological by Measurement Canada. However, the final selection is up to Measurement.

### 2.3.1 ANSI C12.19 / IEEE 1377 Metrological Tables Identification Criteria

Programming of ANSI C12.19 / IEEE 13777 based meters is accomplished through writing to tables. It is important to note that generally it is not permitted to re-program an end-device after it has been verified, approved and physically sealed. However, special exceptions shall be issued by Measurement Canada that allow the re-programming of specified Standard defined (or Manufacturer defined) Tables. These exceptions will depend on whether:

1. The end-device is protected by a Seal; and/or
2. The end-device contains event counters; and/or
3. The end-device contains an event logger.

Any table that contains one or more Metrological element shall be considered to be a Metrological Table. Therefore, only whole Tables can be subject to these restrictions on re-programming. In this context a metrological Table is one that contains constants, factors or algorithm that are used for revenue metering. As such a Table may be considered Metrological in one end-device and not Metrological in another, subject to the revenue billing context of the table.

The following Sections outline those tables that shall be considered Metrological, if they are used in a context of revenue metering. Consider the following examples:

**Example 1:** Assume that a meter utilizes Table 23, “Current Register Data” to report register values for revenue metering. The same meter is also equipped with a recorder capability, through the use of Table 64, “Load Profile Data Set 1”. However, Table 64 is not used (or verified) for revenue related operations (say it is used for load profiling). Then Table 23, and associated control tables are Metrological Tables. Table 64 and related control tables are not Metrological tables.

**Example 2:** Assume a meter same as described in example 1. However, Table 64 is used (and verified) for revenue related operations. Then Table 23, and associated control tables are Metrological Tables. Also Table 64 and related control tables are Metrological tables.

**Example 3:** Assume a meter similar to those described in example 1 or 2 above, and in addition this meter implements Table 74, “History Log Data”. It is clear and obvious that the history log does not contain metrological values (by design), therefore all related tables (that only control and govern the operation of the History Log) are not considered Metrological so they can be programmed regardless of the state of the Seal, or whether an event counter or event logger is available in this end-device.

Therefore, in this Section we shall assert that both Table 23, “Current Register Data”, Table 64, “Load Profile Data Set 1” and related configuration, selections and controls are Metrological. Table 74, “History Log Data”, and configuration tables (which only influence the behaviour of Table 74) are not Metrological.

Given the obvious complexity of the Standards, and the less then obvious relationships among Tables, the Measurement Canada “Task Force on Metering Data Communication Protocol for Electronic Metering Devices”, reviewed all Standard Tables and identified a subset of all Standard Tables and Procedures, which it deems Metrological and in accordance with Measurement Canada requirements. In addition, the Task Force identified a select subset of the Metrological Tables that can only be re-programmed in the presence of an Event Logger (Table 76).

This effort will assist both the Manufacturer and Measurement Canada identify the tables used in an end-device as “potentially” Metrological, on a use-case basis in an actual implementation. The Manufacturers shall then have a reasonable positive expectation that an application for re-programming allowance of a Metrological Table shall be granted by Measurement Canada, when it falls within the Task Force identified ANSI C12.19 / IEEE 1377 Metrological Standard Tables and Procedures.

### 2.3.2 The ANSI C12.19 / IEEE 1377 Metrological Tables

The following charts list the end-device Standard tables and characterize them as Metrological or not. The governing operating assumption is that the end-device has been verified, approved and Sealed when these assertions are operative.

Distinction is made whether the end-device is only sealed (and optionally implementing event counters) and whether the end-device is sealed and deploying an event-logger. The deployment of event counters in addition to event logger is non consequential in this context.

Chart 2.1

Symbols used to describe Metrological attributes and Event Log creation requirements

| Heading /Entry | Heading Description  |
|----------------|--|
| P              | Programming and modifications is allowed to this table (or procedure) unconditionally. i.e in the absence of an Event Log or Event Counters.   |
| E              | When an Event Log or Event Counters are available then programming and modifications is allowed and an event log entry shall be created or an Event Counter shall increment according to Measurement Canada Audit Trail implementation requirements. |
| ✓              | Affirmative yes, it is permissible to program/re-program, update or write to this Table or any of its elements.  |

## Metrological Tables

**Chart 2.1**

Symbols used to describe Metrological attributes and Event Log creation requirements

| Heading /Entry | Heading Description (Continued)   |
|----------------|---|
| <b>X</b>       | Affirmative no, it is <b>not</b> permissible to program/re-program, update or write to this Table or any of its elements. <b>Such an action shall result in a re-verification event.</b> When the Table is a Status or Sensor Table it shall reflect actual values or state values, that can change to reflect the true operating state of the end-device. When procedures are provided that can indirectly affect the state of this Table then the Procedure's Metrological attributes shall govern the outcome of the change operation (e.g. Procedure 4, "Reset List Pointers" can be used to change some of the pointer elements found in Table 26, "Self Read Data", despite the fact that Table 26 is marked with an <b>X</b> in the <b>P</b> and <b>E</b> columns. |
| <b>✓</b>       | May be considered Metrological, subject to other conditions. Final resolution is possible only when all the conditions were evaluated (e.g. Table 7, "Procedure Initiate" may invoke Metrological or Non-metrological Procedures. The final determination depends on the type of procedure in question and whether its invocation affects changes to metrological quantities.)  |
|                | Blank (as in space) under column <b>E</b> indicates that an Event Log entry need not be created, when column <b>P</b> is marked with a <b>✓</b> . This signifies that the entry is not considered a metrological table and it can be re-programmed subject to Standard end-device operating principles.   |

**Chart 2.2**

Metrological Characteristics of Decade 0, End Device Configuration Identification and Procedures

| Table ID                                       | P        | E        |
|--|----------|----------|
| Table 00 - General Configuration               | <b>X</b> | <b>X</b> |
| Table 01 - General Manufacturer Identification | <b>X</b> | <b>X</b> |
| Table 02 - Device Nameplate                    | <b>X</b> | <b>✓</b> |
| Table 03 - ED_MODE Status                      | <b>X</b> | <b>X</b> |
| Table 04 - Pending Status                      | <b>X</b> | <b>X</b> |
| Table 05 - Device Identification               | <b>✓</b> |          |
| Table 06 - Utility Information                 | <b>✓</b> |          |
| Table 07 - Procedure Initiate                  | <b>✓</b> | <b>✓</b> |
| Table 08 - Procedure Response                  | <b>X</b> | <b>X</b> |

**Chart 2.3**

Metrological Characteristics of Decade 1, Data Source Tables.

| Table ID                              | P | E |
|---------------------------------------|---|---|
| Table 10 - Dimension Sources Limiting | X | X |
| Table 11 - Actual Sources Limiting    | X | X |
| Table 12 - Unit Of Measure            | X | X |
| Table 13 - Demand Control             | X | ✓ |
| Table 14 - Data Control               | X | X |
| Table 15 - Constants                  | X | ✓ |
| Table 16 - Source Definition          | X | X |

---

## Metrological Tables

---

All sources defined in Decade 1, “Data Source Tables”, shall be verified by Measurement Canada before physically sealing the end device. Only verified, approved and sealed sources can be made available for selection and used in revenue metering.

**Chart 2.4**

Metrological Characteristics of Decade 2, Register Tables.

| Table ID  | P | E |
|---|---|---|
| Table 20 - Dimension Register Limiting <sup>a</sup> | X | ✓ |
| Table 21 - Actual Register <sup>a</sup>             | X | ✓ |
| Table 22 - Data Selection <sup>a</sup>              | X | ✓ |
| Table 23 - Current Register Data                    | X | X |
| Table 24 - Previous Season Data                     | X | X |
| Table 25 - Previous Demand Reset Data               | X | X |
| Table 26 - Self Read Data                           | X | X |
| Table 27 - Present Register Selection <sup>a</sup>  | X | ✓ |
| Table 28 - Present Register Data                    | X | X |

- a. When changes are permitted to this table. The changes shall not result in the introduction of uncertainties (invalid values) into the current, and previous data registers. Also the registers shall not “lose” their values or introduce values that are not derived from a valid measurement. i.e. verified approved and sealed sensors shall result in summations and totalizers that continue to accumulate even when the source is not selected.

**Chart 2.5**

Metrological Characteristics of Decade 3, Local Display Tables.

| Table ID                               | P | E |
|--|---|---|
| Table 30 - Dimension Display Limiting  | X | X |
| Table 31 - Actual Display              | X | X |
| Table 32 - Display Source <sup>a</sup> | X | ✓ |
| Table 33 - Primary Display List        | X | ✓ |
| Table 34 - Secondary Display List      | ✓ | X |

- a. Modifications are limited to formatting fields only not selections. Modifications to selections shall result in a re-verification event.

**Chart 2.6** Metrological Characteristics of Decade 4, Security Tables.

| Table ID                               | P | E |
|--|---|---|
| Table 40 - Dimension Security Limiting | ✓ |   |
| Table 41 - Actual Security Limiting    | ✓ |   |
| Table 42 - Security                    | ✓ |   |
| Table 43 - Default Access Control      | ✓ |   |
| Table 44 - Access Control              | ✓ |   |
| Table 45 - Key                         | ✓ |   |

**Chart 2.7** Metrological Characteristics of Decade 5, Time and Time of Use Tables.

| Table ID   | P | E |
|--|---|---|
| Table 50 - Dimension Limiting Time and Time of Use | ✓ | ✓ |
| Table 51 - Actual Time And TOU Limiting            | ✓ | ✓ |
| Table 52 - Clock <sup>a</sup>                      | X | X |
| Table 53 - Time Offset                             | ✓ | ✓ |
| Table 54 - Calendar                                | ✓ | ✓ |
| Table 55 - Clock State                             | X | X |
| Table 56 - Time Remaining                          | X | X |

- a. The clock shall be set using Standard Procedure 10, “Set Date and/or Time”. Direct programming of Table 52, “Clock”, is strongly discouraged. Also changing the time in such a manner that the modification to the clock represents an appreciable time discontinuity (one that affects a demand or time-of-use measurement).

**Chart 2.8** Metrological Characteristics of Decade 6, Load Profile Tables.

| Table ID                                    | P | E |
|---|---|---|
| Table 60 - Dimension Limiting Load Profile  | X | ✓ |
| Table 61 - Actual Load Profile Limiting     | X | ✓ |
| Table 62 - Load Profile Control             | X | ✓ |
| Table 63 - Load Profile Status              | X | X |
| Tables 64..67 - Load Profile Data Sets 1..4 | X | X |

**Chart 2.9** Metrological Characteristics of Decade 7, History and Event Logs.

| Table ID                                      | P | E |
|---|---|---|
| Table 70 - Limiting Log Dimensions            | X | X |
| Table 71 - Actual Log Dimensions <sup>a</sup> | ✓ |   |
| Table 72 - Event Identification               | X | X |
| Table 73 - History Log Control                | ✓ |   |
| Table 74 - History Log Data                   | X | X |
| Table 75 - Event Log Control <sup>b</sup>     | X | X |
| Table 76 - Event Log Data                     | X | X |

- a. Programming operations to elements that affect the operation of the Event Log (Tables 72, 75 and 76) shall result in a re-verification event (i.e. they are not permitted).
- b. Only the events that are required by Measurement Canada for use in the Event Logger shall be selected.

**Chart 2.10** Metrological Characteristics of Decade 8, User Defined Tables.

| Table ID  | P | E |
|---|---|---|
| Table 80 - Dimension Function Limiting                                      | ✓ |   |
| Table 81 - Actual Function Limiting   | ✓ |   |
| Table 82 - List   | ✓ |   |
| Table 83 - Selection  | ✓ |   |
| Table 84 - First User Defined to Table 89 - Sixth User Defined <sup>a</sup> | ✓ | ✓ |

- a. The write, event and metrological characteristics of these table and contained fields shall be compliant with the associated tables and elements conveyed within.

**Chart 2.11** Metrological Characteristics of Decade 9, Telephone Control Tables.

| Table ID                        | P | E |
|---------------------------------|---|---|
| Table 90 - Dimension Telephone  | ✓ |   |
| Table 91 - Actual Telephone     | ✓ |   |
| Table 92 - Global Parameters    | ✓ |   |
| Table 93 - Originate Parameters | ✓ |   |
| Table 94 - Originate Schedule   | ✓ |   |
| Table 95 - Answer Parameters    | ✓ |   |
| Table 96 - Call Purpose         | ✓ |   |
| Table 97 - Call Status          | X | X |

**Chart 2.12** Metrological Procedures

| Procedure ID  | P | E |
|---|---|---|
| Procedure 0 - Cold Start (factory setup)                          | X | X |
| Procedure 1 - Warm Start (reboot)                                 | X | ✓ |
| Procedure 2 - Save Configuration (to nvmem) <sup>a</sup>          | X | ✓ |
| Procedure 3 - Clear Data (registers only) <sup>b</sup>            | X | X |
| Procedure 4 - Reset List Pointers <sup>b,c</sup>                  | ✓ | ✓ |
| Procedure 5 - Update Last Read Entry <sup>d,c</sup>               | ✓ | ✓ |
| Procedure 6 - Change End Device Mode                              | X | ✓ |
| Procedure 7 - Clear Standard Status Flags                         | ✓ |   |
| Procedure 8 - Clear Manufacturer Status Flags                     | ✓ |   |
| Procedure 9 - Remote Reset (demand, season)                       | ✓ |   |
| Procedure 10 - Set Date and/or Time <sup>e</sup>                  | ✓ | ✓ |
| Procedure 11 - Execute Diagnostics Procedure <sup>f</sup>         | ✓ |   |
| Procedure 12 - Activate All Pending Tables <sup>g</sup>           | ✓ | ✓ |
| Procedure 13 - Activate Specific Pending Tables <sup>g</sup>      | ✓ | ✓ |
| Procedure 14 - Clear All Pending Tables                           | ✓ |   |
| Procedure 15 - Clear Specific Pending Tables                      | ✓ |   |
| Procedure 16 - Start Load Profile (recorder) <sup>h</sup>         | ✓ | ✓ |
| Procedure 17 - Stop Load Profile (recorder) <sup>h</sup>          | ✓ | ✓ |
| Procedure 18 - Log In (establish security/session)                | ✓ |   |
| Procedure 19 - Log out  | ✓ |   |
| Procedure 20 - Initiate An Immediate Telephone Call (ANSI C12.21) | ✓ |   |

- a. When the “Save Configuration” procedure is used, all writes to tables shall be deferred until the “Save Configuration” procedure is executed. If any of the modified set of tables include a metrological table then event log entry (or entries) shall be generated. If the “Save Configuration” is not executed by the end of a session, all modifications to tables during the session shall be lost.
- b. In Canada it is not permitted to clear data registers or metrological quantities.
- c. This procedure can be executed if it does not affect any metrological quantity, and when it is not used to manipulate the pointers of the self-contained event logger.
- d. An event log entry shall be created when updating the event logger list pointers.
- e. An event log entry shall be created when the time correction affects the meter accuracy performance. This implies that when changing the time in such a manner that the modification to the clock represents an appreciable time discontinuity (one that affects a demand or time-of-use measurement).
- f. This procedure is not allowed to execute if it interferes with metering operations, or it shall generate a re-verification event.
- g. An event log entry is created only upon activation of tables which might result in changes to metrological quantities.
- h. Event Log if load profile is used for revenue metering (for trade).

---

# CHAPTER 3

## DECADE 7 - HISTORY AND EVENT LOG TABLES

---

---

### 3.1 Overview

---

This decade contains the tables associated with the maintenance of the History and Event Logs. The tables described within are identical to those describe in ANSI C12.19 / IEEE 1377 Standards and in the “Utility Industry Standard Tables User’s Guide”, published by Measurement Canada in 1998.

History tables are provided in support of general purpose audit trail and process tracking tools. These operate according to the end-device design and application needs of the end-user. The history log is independent and distinct from the event log, whose purpose is to secure the integrity of metrologically sensitive adjustments and sealable parameters, while allowing changes to features and functions. The role of the event logger is to store end-device change-related information so that a record of the device’s metrological integrity and functionality can be made available for the administration of applicable legislation or in case it becomes necessary to reconstruct the device’s programming over time in order to assist in the resolution of a complaint and/or dispute.

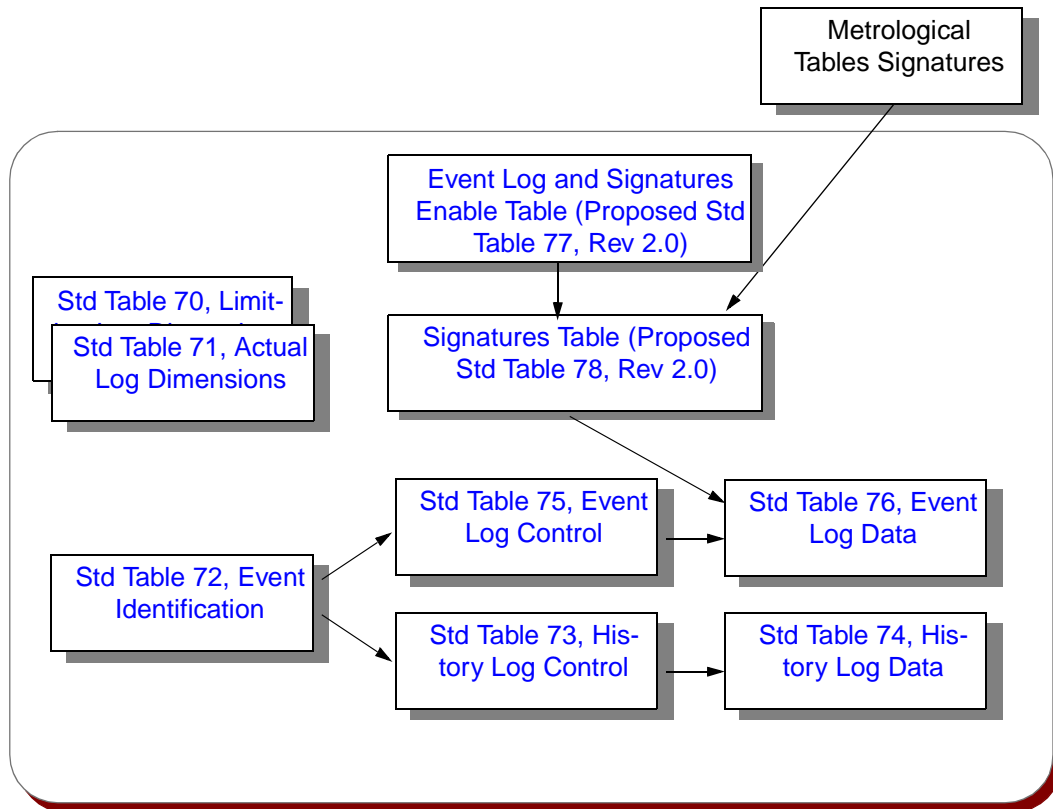
The first edition of the ANSI C12.19 / IEEE 1377 Standard, 1997, provides a complete solution for history and event-logger requirements, except that it falls short of actually describing how [Std Table 76, Event Log Data](#), is to operate in order to provide the necessary protections and audit trail properties required for a uniform engineering of event-loggers for use in Canada.

Chapter 1, “[Audit Trail Implementation Specification for ANSI C12.19 / IEEE 1377 Meters](#)”, identifies the event logger implementation principles. As described it provides minimal requirements for implementing a Measurement Canada compliant event logger. This implementation also describes “[Event Log and Signatures Enable Table \(Proposed Std Table 77, Rev 2.0\)](#)” on page 3-90 and “[Signatures Table \(Proposed Std Table 78, Rev 2.0\)](#)” on page 3-92. The Signatures related tables provides end-device Tables control, identification, state

---

and program status information in support of the event logging process. These tables are new Manufacturer Tables, when implemented in ANSI C12.19 / IEEE 1377 Version 1.0 end-devices. They have been proposed as a contribution as Standard Tables 77/78, to the second revision of ANSI C12.19 / IEEE 1377. Manufacturers of end-devices with event loggers are encouraged to implement the Signatures Tables. When doing so, one will also need to include the extensions to Tables 70 and 71 as described in [Limiting Log Dimensions \(Proposed Table 70, Std Rev 2.0\)](#) and [Actual Log Dimensions \(Proposed Table 71, Std Rev 2.0\)](#).

Figure 3.1 Decade 7 - History and Event Tables



## 3.2 Std Table 70, Limiting Log Dimensions

### 3.2.1 Description

Std Table 70, Limiting Log Dimensions, defines the maximum size and capabilities of the History and Event Log decade.

### 3.2.2 Synopsis

```

TYPE LOG_FLAGS_BFLD = BIT FIELD OF UINT8
    EVENT_NUMBER_FLAG           : BOOL(0);
    HIST_DATE_TIME_FLAG         : BOOL(1);
    HIST_SEQ_NBR_FLAG           : BOOL(2);
    HIST_INHIBIT_OVF_FLAG       : BOOL(3);
    EVENT_INHIBIT_OVF_FLAG      : BOOL(4);
    FILLER                       : FILL(5..7);

END;

TYPE LOG_RCD = PACKED RECORD
    LOG_FLAGS                     : LOG_FLAGS_BFLD;
    NBR_STD_EVENTS                 : UINT8;
    NBR_MFG_EVENTS                 : UINT8;
    HIST_DATA_LENGTH               : UINT8;
    EVENT_DATA_LENGTH              : UINT8;
    NBR_HISTORY_ENTRIES            : UINT16;
    NBR_EVENT_ENTRIES              : UINT16;

END;

TABLE DIM_LOG_TBL = LOG_RCD;

```

### 3.2.3 Data Definition

| Identifier     | Subfield            | Value        | Definition   |
|----------------|---------------------|--------------|--|
| LOG_FLAGS_BFLD | EVENT_NUMBER_FLAG   | <b>FALSE</b> | A common event number can not be maintained in the History & Event Logs. |
|                |                     | <b>TRUE</b>  | A common event number can be maintained in the History & Event Logs.     |
|                | HIST_DATE_TIME_FLAG | <b>FALSE</b> | Date & time can not be maintained in the History Log.                    |
|                |                     | <b>TRUE</b>  | Date & time can be maintained in the History Log.                        |
|                | HIST_SEQ_NBR_FLAG   | <b>FALSE</b> | A sequence number can not be transported through the History Log.        |

### 3.2.3 Data Definition (Continued)

| Identifier | Subfield               | Value        | Definition  |
|------------|------------------------|--------------|---|
|            |                        | <b>TRUE</b>  | A sequence number can be transported through the History Log.   |
|            | HIST_INHIBIT_OVF_FLAG  | <b>FALSE</b> | History Log is not capable of inhibiting overflow.  |
|            |                        | <b>TRUE</b>  | History Log is capable of inhibiting overflow.  |
|            | EVENT_INHIBIT_OVF_FLAG | <b>FALSE</b> | Event Log is not capable of being blocked.  |
|            |                        | <b>TRUE</b>  | Event Log is capable of being blocked.  |
| LOG_RCD    |                        |              |   |
|            | LOG_FLAGS              |              | See LOG_FLAGS_BFLD.   |
|            | NBR_STD_EVENTS         | 0..255       | Maximum number of octets in the set <a href="#">STD_EVENTS_SUPPORTED</a> (See “Std Table 72, Event Identification” on page 3-75). |
|            | NBR_MFG_EVENTS         | 0..255       | Maximum number of octets in the set <a href="#">MFG_EVENTS_SUPPORTED</a> (See “Std Table 72, Event Identification” on page 3-75). |
|            | HIST_DATA_LENGTH       | 0..255       | Maximum number of octets in the <a href="#">HISTORY_ARGUMENT</a> (See “Std Table 74, History Log Data” on page 3-78).             |
|            | EVENT_DATA_LENGTH      | 0..255       | Maximum number of octets in the <a href="#">EVENT_ARGUMENT</a> (See “Std Table 76, Event Log Data” on page 3-83).                 |
|            | NBR_HISTORY_ENTRIES    | 0..65535     | Maximum number of entries in <a href="#">Std Table 74, History Log Data</a> .   |
|            | NBR_EVENT_ENTRIES      | 0..65535     | Maximum number of entries in the event log in <a href="#">Std Table 76, Event Log Data</a> .                                      |

### 3.3 Std Table 71, Actual Log Dimensions

#### 3.3.1 Description

Std Table 71, Actual Log Dimensions, defines the actual size and capabilities of the History and Event Log decade.

#### 3.3.2 Synopsis

**TABLE** ACT\_LOG\_TBL = DIM\_LOG\_TBL.LOG\_RCD;

#### 3.3.3 Data Definition

| Identifier     | Subfield               | Value        | Definition   |
|----------------|------------------------|--------------|--|
| LOG_FLAGS_BFLD |                        |              |  |
|                | EVENT_NUMBER_FLAG      | <b>FALSE</b> | A common event number is not maintained in the History & Event Logs.   |
|                |                        | <b>TRUE</b>  | A common event number is maintained in the History & Event Logs.   |
|                | HIST_DATE_TIME_FLAG    | <b>FALSE</b> | A date & time is not maintained in the History Log.  |
|                |                        | <b>TRUE</b>  | A date & time is maintained in the History Log.  |
|                | HIST_SEQ_NBR_FLAG      | <b>FALSE</b> | A sequence number is not transported through the History Log.  |
|                |                        | <b>TRUE</b>  | A sequence number is transported through the History Log.  |
|                | HIST_INHIBIT_OVF_FLAG  | <b>FALSE</b> | History Log is not inhibiting new entries when an overflow condition exists.   |
|                |                        | <b>TRUE</b>  | History Log is inhibiting new entries when an overflow condition exists.   |
|                | EVENT_INHIBIT_OVF_FLAG | <b>FALSE</b> | Event Log is not inhibiting new entries when an overflow condition exists.   |
|                |                        | <b>TRUE</b>  | Event Log is inhibiting new entries when an overflow condition exists.   |
| LOG_RCD        |                        |              |  |
|                | LOG_FLAGS              |              | See LOG_FLAGS_BFLD.  |
|                | NBR_STD_EVENTS         | 0..255       | Number of octets in the set <code>STD_EVENTS_SUPPORTED</code> (See "Std Table 72, Event Identification" on page 3-75). |

### 3.3.3 Data Definition (Continued)

| Identifier | Subfield            | Value    | Definition   |
|------------|---------------------|----------|--|
|            | NBR_MFG_EVENTS      | 0..255   | Number of octets in the set <code>MFG_EVENTS_SUPPORTED</code> (See “Std Table 72, Event Identification” on page 3-75). |
|            | HIST_DATA_LENGTH    | 0..255   | Number of octets in the <code>HISTORY_ARGUMENT</code> (See “Std Table 74, History Log Data” on page 3-78).             |
|            | EVENT_DATA_LENGTH   | 0..255   | Number of octets in the <code>EVENT_ARGUMENT</code> (See “Std Table 76, Event Log Data” on page 3-83).                 |
|            | NBR_HISTORY_ENTRIES | 0..65535 | Actual maximum number of entries in the History Log.   |
|            | NBR_EVENT_ENTRIES   | 0..65535 | Actual maximum number of entries in the Event Log.   |

## 3.4 Std Table 72, Event Identification

### 3.4.1 Description

Std Table 72, [Event Identification](#), contains the events that are supported by the end device.

### 3.4.2 Synopsis

```
TYPE EVENTS_SUPPORTED_RCD = PACKED RECORD
    STD_EVENTS_SUPPORTED: SET(ACT_LOG_TBL.NBR_STD_EVENTS);
    MFG_EVENTS_SUPPORTED: SET(ACT_LOG_TBL.NBR_MFG_EVENTS);
END;
```

```
TABLE EVENTS_ID_TBL = EVENTS_SUPPORTED_RCD;
```

### 3.4.3 Data Definition

| Identifier           | Subfield             | Value | Definition   |
|----------------------|----------------------|-------|--|
| EVENTS_SUPPORTED_RCD |                      |       |  |
|                      | STD_EVENTS_SUPPORTED |       | This set variable indicates which of the standard events are supported in the Event Log ( <a href="#">Appendix A, "ANSI Standard C12.19 History Log Codes"</a> ). Event codes are represented by bits 0 through $(8 \times \text{NBR\_STD\_EVENTS} - 1)$ , with a one (1) representing a <b>TRUE</b> or implemented condition and a zero (0) representing a <b>FALSE</b> or not implemented condition. |
|                      | MFG_EVENTS_SUPPORTED |       | This set variable indicates which of the manufacturer events are supported in the Event Log. Events are enabled by bits 0 through $(8 \times \text{NBR\_STD\_EVENTS} - 1)$ , with a one (1) representing a <b>TRUE</b> or implemented condition and a zero (0) representing a <b>FALSE</b> or not implemented condition.   |

## 3.5 Std Table 73, History Log Control

### 3.5.1 Description

Std Table 73, History Log Control, defines the History Log codes to be written to the History Log. It also defines which specific procedures and or table writes that are to be acknowledged in the History Log. For a specific procedure or table to be acknowledged, three independent tests shall all be true:

1. The procedure or table shall be used in the end device, per the “Table 00 - General Configuration” on page 4-2.
2. The appropriate History Log code shall be used, per this table.
3. The procedure or table shall be requested to be acknowledged, per this table.

### 3.5.2 Synopsis

```

TYPE HISTORY_CTRL_RCD = PACKED RECORD
  STD_EVENTS_MONITORED_FLAGS: SET(ACT_LOG_TBL.NBR_STD_EVENTS);
  MFG_EVENTS_MONITORED_FLAGS: SET(ACT_LOG_TBL.NBR_MFG_EVENTS);
  STD_TBLS_MONITORED_FLAGS: SET(
    GEN_CONFIG_TBL.DIM_STD_TBLS_USED);
  MFG_TBLS_MONITORED_FLAGS: SET(
    GEN_CONFIG_TBL.DIM_MFG_TBLS_USED);
  STD_PROC_MONITORED_FLAGS: SET(
    GEN_CONFIG_TBL.DIM_STD_PROC_USED);
  MFG_PROC_MONITORED_FLAGS: SET(
    GEN_CONFIG_TBL.DIM_MFG_PROC_USED);

END;

```

```

TABLE HISTORY_LOG_CTRL_TBL = HISTORY_CTRL_RCD;

```

### 3.5.3 Data Definition

| Identifier       | Subfield                   | Value        | Definition  |
|------------------|----------------------------|--------------|---|
| HISTORY_CTRL_RCD | STD_EVENTS_MONITORED_FLAGS |              | Bit position is linearly associated with corresponding standard event code.     |
|                  |                            | <b>FALSE</b> | Turns off event recording for associated event code.                            |
|                  |                            | <b>TRUE</b>  | Turns on event recording for associated event code.                             |
|                  | MFG_EVENTS_MONITORED_FLAGS |              | Bit position is linearly associated with corresponding manufacturer event code. |

## 3.5.3 Data Definition (Continued)

| Identifier | Subfield                 | Value        | Definition  |
|------------|--------------------------|--------------|---|
|            |                          | <b>FALSE</b> | Turns off event recording for associated event code.                                  |
|            |                          | <b>TRUE</b>  | Turns on event recording for associated event code.                                   |
|            | STD_TBLS_MONITORED_FLAGS |              | Bit position is linearly associated with corresponding standard table number.         |
|            |                          | <b>FALSE</b> | Turns off event recording for associated table.                                       |
|            |                          | <b>TRUE</b>  | Turns on event recording for associated table.  |
|            | MFG_TBLS_MONITORED_FLAGS |              | Bit position is linearly associated with corresponding manufacturer table number.     |
|            |                          | <b>FALSE</b> | Turns off event recording for associated table.                                       |
|            |                          | <b>TRUE</b>  | Turns on event recording for associated table.  |
|            | STD_PROC_MONITORED_FLAGS |              | Bit position is linearly associated with corresponding standard procedure number.     |
|            |                          | <b>FALSE</b> | Turns off event recording for associated procedure.                                   |
|            |                          | <b>TRUE</b>  | Turns on event recording for associated procedure.                                    |
|            | MFG_PROC_MONITORED_FLAGS |              | Bit position is linearly associated with corresponding manufacturer procedure number. |
|            |                          | <b>FALSE</b> | Turns off event recording for associated procedure.                                   |
|            |                          | <b>TRUE</b>  | Turns on event recording for associated procedure.                                    |

## 3.6 Std Table 74, History Log Data

---

### 3.6.1 Description

Std Table 74, History Log Data, provides the History Log.

### 3.6.2 Synopsis

```

TYPE LIST_STATUS_BFLD = BIT FIELD OF UINT8
    ORDER                : UINT(0..0);
    OVERFLOW_FLAG        : BOOL(1);
    LIST_TYPE             : UINT(2..2);
    INHIBIT_OVERFLOW_FLAG : BOOL(3);
    FILLER                : FILL(4..7);

END;

TYPE HISTORY_ENTRY_RCD = PACKED RECORD
    IF ACT_LOG_TBL.HIST_DATE_TIME_FLAG THEN
        HISTORY_TIME          : LTIME_DATE;
    END;
    IF ACT_LOG_TBL.EVENT_NUMBER_FLAG THEN
        EVENT_NUMBER          : UINT16;
    END;
    IF ACT_LOG_TBL.HIST_SEQ_NBR_FLAG THEN
        HISTORY_SEQ_NBR       : UINT16;
    END;
    USER_ID                  : UINT16;
    HISTORY_CODE              : TABLE IDB_BFLD;
    HISTORY_ARGUMENT          : ARRAY[ACT_LOG_TBL.HIST_DATA_LENGTH] OF
        UINT8;

END;

TYPE HISTORY_LOG_RCD = PACKED RECORD
    HIST_FLAGS                : LIST_STATUS_BFLD;
    NBR_VALID_ENTRIES         : UINT16;
    LAST_ENTRY_ELEMENT        : UINT16;
    LAST_ENTRY_SEQ_NBR        : UINT32;
    NBR_UNREAD_ENTRIES        : UINT16;
    ENTRIES                   : ARRAY[ACT_LOG_TBL.NBR_HISTORY_ENTRIES]
        OF HISTORY_ENTRY_RCD;

END;

TABLE HISTORY_LOG_DATA_TBL = HISTORY_LOG_RCD;

```

### 3.6.3 Data Definition

| Identifier            | Subfield        | Value    | Definition   |
|-----------------------|-----------------|----------|--|
| LIST_STATUS_BFLD      | ORDER           | 0        | The log is transported in ascending order (element N is older than element N+1).   |
|                       |                 | 1        | The log is transported in descending order (element N is newer than element N+1).  |
|                       | OVERFLOW_FLAG   | FALSE    | Overflow has not occurred.   |
|                       |                 | TRUE     | An attempt was made to enter an event such that the number of un-read entries would have exceeded the actual number of possible entries in the log.  |
|                       | LIST_TYPE       | 0        | FIFO - as placed in log.   |
|                       |                 | 1        | Circular - as placed in log.   |
| INHIBIT_OVERFLOW_FLAG |                 |          | The same value as HIST_INHIBIT_OVF_FLAG (See "Std Table 71, Actual Log Dimensions" on page 3-73).  |
| TABLE_IDB_BFLD        | TBL_PROC_NBR    | 0..2047  | Event number logged.   |
|                       | STD_VS_MFG_FLAG | FALSE    | Event number is standard defined.  |
|                       |                 | TRUE     | Event number is manufacturer defined.  |
|                       | SELECTOR        |          | Not Used.  |
| HISTORY_ENTRY_RCD     | HISTORY_TIME    |          | Date and time of History Log entry.  |
|                       | EVENT_NUMBER    | 0..65535 | Event number common to both the History and Event Logs.  |
|                       | HISTORY_SEQ_NBR | 0..65535 | Sequence number associated with the History Log only.  |
|                       | USER_ID         | 0..65535 | The User ID associated with this History Log entry. It comes from the Log In Procedure or from a communication session initiation sequence. A USER_ID of zero means the end device initiated the event. A USER_ID of one means the event was manually initiated. |
|                       | HISTORY_CODE    |          | Event code logged. See TABLE_IDB_BFLD. above, Table ID "B" Bit Field and Appendix A, "ANSI Standard C12.19 History Log Codes".   |

### 3.6.3 Data Definition (Continued)

| Identifier      | Subfield           | Value            | Definition  |
|-----------------|--------------------|------------------|---|
|                 | HISTORY_ARGUMENT   |                  | Argument associated with a specific entry. For Standard event arguments, refer to, "ANSI Standard C12.19-1997 History Log Codes" and "ANSI Standard C12.21-1999 Extensions to the History Log Codes" found in Appendix A. Also See Appendix B, "Event Log Codes for Use by Canadian Industry" |
| HISTORY_LOG_RCD |                    |                  |   |
|                 | HIST_FLAGS         |                  | See LIST_STATUS_BFLD.   |
|                 | NBR_VALID_ENTRIES  | 0..65535         | Number of valid entries in the log. The range is from zero (meaning the log is empty) to the actual dimension of the log.   |
|                 | LAST_ENTRY_ELEMENT | 0..65535         | The array element number of the newest valid entry in the log.  |
|                 | LAST_ENTRY_SEQ_NBR | 0..4,294,967,295 | The sequence number of the newest valid entry in the log.   |
|                 | NBR_UNREAD_ENTRIES | 0..65535         | The number of entries in the log that have not yet been read. It is only changed through a procedure.   |
|                 | ENTRIES            |                  | Array of History Log entries.   |

## 3.7 Std Table 75, Event Log Control

### 3.7.1 Description

Std Table 75, Event Log Control, defines the Event Log codes to be written to the Event Log. It also defines which specific procedure and or table writes that are to be acknowledged in the Event Log. For a specific procedure or table to be acknowledged, three independent tests shall all be true:

1. The procedure or table shall be used in the end device, per “Table 00 - General Configuration” on page 4-2.
2. The appropriate Event code shall be used, per this table.
3. The procedure or table shall be requested to be acknowledged, per this table.

This data structure is identical to the structure in “Std Table 73, History Log Control” on page 3-76.

### 3.7.2 Synopsis

**TABLE** EVENT\_LOG\_CTRL\_TBL = HISTORY\_LOG\_CTRL\_TBL.HISTORY\_CTRL\_RCD;

### 3.7.3 Data Definition

| Identifier               | Subfield                   | Value   | Definition  |
|--------------------------|----------------------------|---|---|
| EVENT_CTRL_RCD           | STD_EVENTS_MONITORED_FLAGS |   | Bit position is linearly associated with corresponding standard event code.     |
|                          |                            | <b>FALSE</b>  | Turns off event recording for associated event code.                            |
|                          | <b>TRUE</b>                | Turns on event recording for associated event code.                           |   |
|                          | MFG_EVENTS_MONITORED_FLAGS |   | Bit position is linearly associated with corresponding manufacturer event code. |
|                          |                            | <b>FALSE</b>  | Turns off event recording for associated event code.                            |
|                          | <b>TRUE</b>                | Turns on event recording for associated event code.                           |   |
| STD_TBLS_MONITORED_FLAGS |                            | Bit position is linearly associated with corresponding standard table number. |   |
|                          | <b>FALSE</b>               | Turns off event recording for associated table.                               |   |
|                          | <b>TRUE</b>                | Turns on event recording for associated table.                                |   |

### 3.7.3 Data Definition (Continued)

| Identifier               | Subfield | Value        | Definition  |
|--------------------------|----------|--------------|---|
| MFG_TBLS_MONITORED_FLAGS |          |              | Bit position is linearly associated with corresponding manufacturer table number.     |
|                          |          | <b>FALSE</b> | Turns off event recording for associated table.                                       |
|                          |          | <b>TRUE</b>  | Turns on event recording for associated table.  |
| STD_PROC_MONITORED_FLAGS |          |              | Bit position is linearly associated with corresponding standard procedure number.     |
|                          |          | <b>FALSE</b> | Turns off event recording for associated procedure.                                   |
|                          |          | <b>TRUE</b>  | Turns on event recording for associated procedure.                                    |
| MFG_PROC_MONITORED_FLAGS |          |              | Bit position is linearly associated with corresponding manufacturer procedure number. |
|                          |          | <b>FALSE</b> | Turns off event recording for associated procedure.                                   |
|                          |          | <b>TRUE</b>  | Turns on event recording for associated procedure.                                    |

## 3.8 Std Table 76, Event Log Data

### 3.8.1 Description

Std Table 76, Event Log Data, provides the Event Log.

### 3.8.2 Synopsis

```

TYPE LIST_STATUS_BFLD = BIT FIELD OF UINT8
    ORDER                : UINT(0..0);
    OVERFLOW_FLAG        : BOOL(1);
    LIST_TYPE             : UINT(2..2);
    INHIBIT_OVERFLOW_FLAG : BOOL(3);
    FILL                  : FILL(4..7);
END;

TYPE EVENT_ENTRY_RCD = PACKED RECORD
    EVENT_TIME            : LTIME_DATE;
    IF ACT_LOG_TBL.EVENT_NUMBER_FLAG THEN
        EVENT_NUMBER: UINT16;
    END;
    EVENT_SEQ_NBR        : UINT16;
    USER_ID              : UINT16;
    EVENT_CODE           : TABLE_IDB_BFLD;
    EVENT_ARGUMENT       : ARRAY[ACT_LOG_TBL.EVENT_DATA_LENGTH] OF
                          UINT8;
END;

TYPE EVENT_LOG_RCD = PACKED RECORD
    EVENT_FLAGS          : LIST_STATUS_BFLD;
    NBR_VALID_ENTRIES   : UINT16;
    LAST_ENTRY_ELEMENT  : UINT16;
    LAST_ENTRY_SEQ_NBR  : UINT32;
    NBR_UNREAD_ENTRIES  : UINT16;
    ENTRIES              : ARRAY[ACT_LOG_TBL.NBR_EVENT_ENTRIES] OF
                          EVENT_ENTRY_RCD;
END;

TABLE EVENT_LOG_DATA_TBL = EVENT_LOG_RCD;

```

### 3.8.3 Data Definition

| Identifier            | Subfield        | Value        | Definition   |
|-----------------------|-----------------|--------------|--|
| LIST_STATUS_BFLD      | ORDER           | 0            | The log is transported in ascending order (element N is older than element N+1).   |
|                       |                 | 1            | The log is transported in descending order (element N is newer than element N+1).  |
|                       | OVERFLOW_FLAG   | <b>FALSE</b> | Overflow has not occurred.   |
|                       |                 | <b>TRUE</b>  | An attempt was made to enter an event such that the number of un-read entries would have exceeded the actual number of possible entries in the log.  |
|                       | LIST_TYPE       | 0            | FIFO - as placed in log.   |
|                       |                 | 1            | Circular - as placed in log.   |
| INHIBIT_OVERFLOW_FLAG |                 |              | The same value as <a href="#">EVENT_INHIBIT_OVF_FLAG</a> .   |
| TABLE_IDB_BFLD        | TBL_PROC_NBR    | 0..2047      | Event code number logged.  |
|                       | STD_VS_MFG_FLAG | <b>FALSE</b> | Event code is standard defined.  |
|                       |                 | <b>TRUE</b>  | Event code is manufacturer defined.  |
|                       | SELECTOR        |              | Not Used.  |
| EVENT_ENTRY_RCD       | EVENT_TIME      |              | Date and time of Event Log entry.  |
|                       | EVENT_NUMBER    | 0..65535     | Event number common to both the History and Event Logs.  |
|                       | EVENT_SEQ_NBR   | 0..65535     | Sequence number associated with the Event Log only.  |
|                       | USER_ID         | 0..65535     | The User ID associated with this Event Log entry. It comes from the Log In Procedure or from a communication session initiation sequence. A USER_ID of zero means the end device initiated the event. A USER_ID of one means the event was manually initiated. |
|                       | EVENT_CODE      |              | Event code logged. See <a href="#">TABLE_IDB_BFLD</a> , "Table ID "B" Bit Field" on page 3-24, and <a href="#">Appendix B</a> , "Event Log Codes for Use by Canadian Industry".  |
|                       | EVENT_ARGUMENT  |              | Argument associated with a specific entry. Refer to <a href="#">Appendix B</a> , "Event Log Codes".  |

### 3.8.3 Data Definition (Continued)

| Identifier    | Subfield           | Value            | Definition   |
|---------------|--------------------|------------------|--|
| EVENT_LOG_RCD |                    |                  |  |
|               | EVENT_FLAGS        |                  | See <b>LIST_STATUS_BFLD</b> above.   |
|               | NBR_VALID_ENTRIES  | 0..65535         | Number of valid entries in the log. The range is zero (meaning the log is empty) to the actual dimension of the log. |
|               | LAST_ENTRY_ELEMENT | 0..65535         | The array element number of the newest valid entry in the log.   |
|               | LAST_ENTRY_SEQ_NBR | 0..4,294,967,295 | The sequence number of the newest valid entry in the log.  |
|               | NBR_UNREAD_ENTRIES | 0..65535         | The number of entries in the log that have not yet been read. It is only changed through a procedure.                |
|               | ENTRIES            |                  | Array of Event Log entries.  |

## 3.9 Limiting Log Dimensions (Proposed Table 70, Std Rev 2.0)

### 3.9.1 Description

[Limiting Log Dimensions \(Proposed Table 70, Std Rev 2.0\)](#), defines the maximum size and capabilities of the History and Event Log decade. It is identical to [Std Table 70, Limiting Log Dimensions](#), but it extends it to facilitate the support of the Table's signature and event check signatures. The proposed revision 2.0 to the Standard Table 70 are boxed in the Synopsis below.

### 3.9.2 Synopsis

```

TYPE LOG_FLAGS_BFLD = BIT FIELD OF UINT8
    EVENT_NUMBER_FLAG      : BOOL(0);
    HIST_DATE_TIME_FLAG    : BOOL(1);
    HIST_SEQ_NBR_FLAG      : BOOL(2);
    HIST_INHIBIT_OVF_FLAG  : BOOL(3);
    EVENT_INHIBIT_OVF_FLAG : BOOL(4);
    METROLOGICAL_SIG_FLAG  : BOOL(5);
    PROGRAM_SIG_FLAG       : BOOL(6);
    ALTERNATE_SIG_FLAG     : BOOL(7);
END;

```

```

TYPE LOG_2_RCD = PACKED RECORD
    LOG_FLAGS           : LOG_FLAGS_BFLD;
    NBR_STD_EVENTS      : UINT8;
    NBR_MFG_EVENTS      : UINT8;
    HIST_DATA_LENGTH    : UINT8;
    EVENT_DATA_LENGTH   : UINT8;
    NBR_HISTORY_ENTRIES : UINT16;
    NBR_EVENT_ENTRIES   : UINT16;
    NBR_PROGRAM_TABLES  : UINT16;
END;

```

**TABLE** DIM\_LOG\_2\_TBL = LOG\_2\_RCD;

### 3.9.3 Data Definition

| Identifier     | Subfield              | Value        | Definition   |
|----------------|-----------------------|--------------|--|
| LOG_FLAGS_BFLD |                       |              | Identical in structure and values found in “Std Table 70, Limiting Log Dimensions” on page 3-71, plus the following extensions.            |
|                | METROLOGICAL_SIG_FLAG | <b>FALSE</b> | The end device cannot supply metrological signatures in “Signatures Table (Proposed Std Table 78, Rev 2.0)” on page 3-92.                  |
|                |                       | <b>TRUE</b>  | The end device can supply metrological signatures in Signatures Table (Proposed Std Table 78, Rev 2.0).                                    |
|                | PROGRAM_SIG_FLAG      | <b>FALSE</b> | The end device cannot supply program signatures in Signatures Table (Proposed Std Table 78, Rev 2.0).                                      |
|                |                       | <b>TRUE</b>  | The end device can supply program signatures in Signatures Table (Proposed Std Table 78, Rev 2.0).   |
|                | ALTERNATE_SIG_FLAG    | <b>FALSE</b> | The end device cannot supply alternate signatures in Signatures Table (Proposed Std Table 78, Rev 2.0).                                    |
|                |                       | <b>TRUE</b>  | The end device can supply alternate signatures in Signatures Table (Proposed Std Table 78, Rev 2.0).                                       |
| LOG_2_RCD      |                       |              | Identical to LOG_RCD in structure and values found in “Std Table 70, Limiting Log Dimensions” on page 3-71, plus the following extensions: |
|                | LOG_FLAGS             |              | See LOG_FLAGS_BFLD above.  |
|                | NBR_PROGRAM_TABLES    | 0.65535      | Maximum number of tables that can be stored in the TABLE_LIST of the “Signatures Table (Proposed Std Table 78, Rev 2.0)” on page 3-92.     |

## 3.10 Actual Log Dimensions (Proposed Table 71, Std Rev 2.0)

### 3.10.1 Description

[Actual Log Dimensions \(Proposed Table 71, Std Rev 2.0\)](#), defines the actual size and capabilities of the History and Event Log decade. It is identical to [Std Table 71, Actual Log Dimensions](#), but it extends to facilitate the support of the Table's signature and event check signatures.

### 3.10.2 Synopsis

**TABLE** ACT\_LOG\_2\_TBL = LOG\_2\_RCD;

### 3.10.3 Data Definition

| Identifier     | Subfield              | Value        | Definition  |
|----------------|-----------------------|--------------|---|
| LOG_FLAGS_BFLD |                       |              | Identical in structure and values found in <a href="#">"Std Table 71, Actual Log Dimensions" on page 3-73</a> , plus the following extensions.                            |
|                | METROLOGICAL_SIG_FLAG | <b>FALSE</b> | The end device does not supply metrological signatures in <a href="#">"Signatures Table (Proposed Std Table 78, Rev 2.0)" on page 3-92</a> .                              |
|                |                       | <b>TRUE</b>  | The end device supplies metrological signatures in <a href="#">Signatures Table (Proposed Std Table 78, Rev 2.0)</a> .  |
|                | PROGRAM_SIG_FLAG      | <b>FALSE</b> | The end device does not supply program signatures in <a href="#">Signatures Table (Proposed Std Table 78, Rev 2.0)</a> .  |
|                |                       | <b>TRUE</b>  | The end device supplies program signatures in <a href="#">Signatures Table (Proposed Std Table 78, Rev 2.0)</a> .   |
|                | ALTERNATE_SIG_FLAG    | <b>FALSE</b> | The end device does not supply alternate signatures in <a href="#">Signatures Table (Proposed Std Table 78, Rev 2.0)</a> .  |
|                |                       | <b>TRUE</b>  | The end device supplies alternate signatures in <a href="#">Signatures Table (Proposed Std Table 78, Rev 2.0)</a> .   |
| LOG_2_RCD      |                       |              | Identical in structure and values to <a href="#">LOG_RCD</a> found in <a href="#">"Std Table 71, Actual Log Dimensions" on page 3-73</a> , plus the following extensions: |

### 3.10.3 Data Definition (Continued)

| Identifier | Subfield           | Value    | Definition   |
|------------|--------------------|----------|--|
|            | LOG_FLAGS          |          | See <a href="#">LOG_FLAGS_BFLD</a> above.  |
|            | NBR_PROGRAM_TABLES | 0..65535 | Actual number of tables that are available the <a href="#">TABLE_LIST</a> of the “ <a href="#">Signatures Table (Proposed Std Table 78, Rev 2.0)</a> ” on page 3-92. |

## 3.11 Event Log and Signatures Enable Table (Proposed Std Table 77, Rev 2.0)

---

### 3.11.1 Description

The [Event Log and Signatures Enable Table \(Proposed Std Table 77, Rev 2.0\)](#) contains four collections of Standard and Manufacturer Table-set selectors that identify Metrological, Program and Alternate sets of tables whose signatures shall be recorded in the [Signatures Table \(Proposed Std Table 78, Rev 2.0\)](#) and those tables that need to be logged in [Std Table 76, Event Log Data](#). Only tables identified within shall be recorded in the Signatures Table and only those Tables identified in the EVENTLOG\_SEL shall trigger a Table-changed event entry in [Std Table 76, Event Log Data](#).

### 3.11.2 Synopsis

```
TYPE SIGNATURE_SEL_RCD = PACKED_RECORD
    STD_TBLS_SEL : SET(GEN_CONFIG_TBL.DIM_STD_TBLS_USED);
    MFG_TBLS_SEL : SET(GEN_CONFIG_TBL.DIM_MFG_TBLS_USED);
END;
```

```
TYPE SIG_ENABLE_RCD = PACKED_RECORD
    IF METROLOGICAL_SIG_FLAG THEN
        METROLOGICAL_SEL : SIGNATURE_SEL_RCD;
        EVENTLOG_SEL     : SIGNATURE_SEL_RCD;
    END;
    IF PROGRAM_SIG_FLAG THEN
        PROGRAM_SEL      : SIGNATURE_SEL_RCD;
    END;
    IF ALTERNATE_SIG_FLAG THEN
        ALTERNATE_SEL    : SIGNATURE_SEL_RCD;
    END;
END;
```

```
TABLE SIGNATURES_TBL = SIG_ENABLE_RCD;
```

### 3.11.3 Data Definition

| Identifier        | Subfield         | Value | Definition   |
|-------------------|------------------|-------|--|
| SIGNATURE_SEL_RCD |                  |       | Identification of the Standard and Manufacturer Tables that may trigger table change events or may be signed.  |
|                   | STD_TBLS_SEL     |       | This <b>SET</b> indicates which of the Standard Tables can be selected. Tables are represented by bits 0 through $(8 * DIM\_STD\_TBLS\_USED - 1)$ , with a 1 representing a <b>TRUE</b> or a possible implementation and a 0 representing a <b>FALSE</b> or not implemented.     |
|                   | MFG_TBLS_SEL     |       | This <b>SET</b> indicates which of the Manufacturer Tables can be selected. Tables are represented by bits 0 through $(8 * DIM\_STD\_TBLS\_USED - 1)$ , with a 1 representing a <b>TRUE</b> or a possible implementation and a 0 representing a <b>FALSE</b> or not implemented. |
| SIG_ENABLE_RCD    |                  |       | Identification of the Metrological, event loggable and alternate sets of tables that need to be signed and/or event logged. Changes to Metrological tables that are not listed as event loggable, shall trigger a re-verification event.   |
|                   | METROLOGICAL_SEL |       | Identifies the Tables which contain metrological elements and will be included in the computation of the METROLOGICAL_SIG.   |
|                   | EVENTLOG_SEL     |       | Indicates which of the Tables shall trigger a table change event in “Std Table 76, Event Log Data” on page 3-83. Tables that are not selected within, but are identified in METROLOGICAL_SEL, shall trigger a re-verification event upon change.                                 |
|                   | PROGRAM_SEL      |       | Identifies the Tables which contain program elements and will be included in the computation of the PROGRAM_SIG.   |
|                   | ALTERNATE_SEL    |       | Identifies the Tables which contain metrological elements and will be included in the computation of the ALTERNATE_SIG.  |

## 3.12 Signatures Table (Proposed Std Table 78, Rev 2.0)

---

### 3.12.1 Description

The [Signatures Table \(Proposed Std Table 78, Rev 2.0\)](#) contains a list of metrological and program table identifiers and storage allocation for the corresponding signature. This list identifies which tables contain metrological information and which tables need to be logged when any of their elements are changed. It is a Manufacturer Table, when implemented in a Standard Version 1.0 end-device. This table provides information that can be used to validate the integrity of an event logger itself (`METROLOGICAL_FLAG`). This table also lists the tables that were modified hence need to be downloaded when the event logger is downloadable (`MODIFIED_FLAG`). It also provides assist technology for improved field programming and program change detection in real-time (`PROGRAM_SIG` and `ALTERNATE_SIG`).

There are a number of variations of signatures. Global signatures that represent the programming state of a selection of Tables, `METROLOGICAL_SIG`, `PROGRAM_SIG` and `ALTERNATE_SIG`. And audit trail signatures that cover the programming history of the end-device, `EVENT_CHECK_SIG`. When Table 78 is not implemented its conceptual implementation logic shall be used to derive the event check signature, `EVENT_CHECK_SIG`, used in [“Std Table 76, Event Log Data”](#) on page 3-83.

### 3.12.2 Synopsis

```
TYPE TABLE_IDSIG_BFLD = BIT FIELD OF UINT16
    TBL_PROC_NBR      : UINT(0..10);
    STD_VS_MFG_FLAG   : BOOL(11);
    MODIFIED_FLAG     : BOOL(12);
    METROLOGICAL_FLAG : BOOL(13);
    LOGABLE_FLAG      : BOOL(14);
    ALTERNATE_FLAG    : BOOL(15);
```

**END;**

```
TYPE TABLE_LIST_RCD = PACKED RECORD
    TABLE_ID      : TABLE_IDSIG_BFLD;
    TABLE_SIG     : ARRAY[16] OF UINT8;
```

**END;**

```

TYPE SIGNATURES_RCD = PACKED RECORD
  ALGORITHM_ID          : ARRAY[16] OF UINT8;
  IF METROLOGICAL_SIG_FLAG THEN
    METROLOGICAL_SIG    : ARRAY[16] OF UINT8;
    EVENT_CHECK_SIG     : ARRAY[16] OF UINT8;
  END;
  IF PROGRAM_SIG_FLAG THEN
    PROGRAM_SIG         : ARRAY[16] OF UINT8;
  END;
  IF ALTERNATE_SIG_FLAG THEN
    ALTERNATE_SIG      : ARRAY[16] OF UINT8;
  END;
  TABLE_LIST          : ARRAY[ACT_LOG_2_TBL.NBR_PROGRAM_TABLES]
                        TABLE_LIST_RCD;

END;

TABLE SIGNATURES_TBL = SIGNATURES_RCD;

```

### 3.12.3 Data Definition

| Identifier     | Subfield          | Value        | Definition   |
|----------------|-------------------|--------------|--|
| TABLE_LIST_RCD |                   |              | Table identifier and the signature of the table.   |
| TABLE_ID       |                   |              | Table number and signature properties identification bit field.  |
|                | TBL_PROC_NBR      | 0..2039      | Table identifier   |
|                | STD_VS_MFG_FLAG   | <b>FALSE</b> | Standard Table   |
|                |                   | <b>TRUE</b>  | Manufacturer Table   |
|                | MODIFIED_FLAG     | <b>FALSE</b> | This table was not changed since the last invocation of Standard Procedure 5, "Update Last Read Entries". Invocation of Standard Procedure 0, "Cold Start", or Procedure 4, "Reset List Pointers" or Standard Procedure 5, "Update Last Read Entries", shall set this flag to <b>FALSE</b> . |
|                |                   | <b>TRUE</b>  | This table was changed since the last invocation of Standard Procedure 5, "Update Last Read Entries".  |
|                | METROLOGICAL_FLAG | <b>FALSE</b> | The table does not contain metrological elements. It is not included in the computation of the METROLOGICAL_SIG.   |

### 3.12.3 Data Definition (Continued)

| Identifier     | Subfield       | Value   | Definition   |
|----------------|----------------|---|--|
|                |                | <b>TRUE</b>                                   | The table contains metrological elements. It is included in the computation of the METROLOGICAL_SIG.   |
|                | LOGABLE_FLAG   | <b>FALSE</b>                                  | The table is not event logged.   |
|                |                | <b>TRUE</b>                                   | The table is event legible. Changes to this table shall result in the creation of an event log entry in StdTable 76, Event Log Data.   |
|                | ALTERNATE_FLAG | <b>FALSE</b>                                  | The table is not included in the computation of the alternate signature, ALTERNATE_SIG.  |
|                |                | <b>TRUE</b>                                   | The table is included in the computation of the alternate signature, ALTERNATE_SIG.  |
| TABLE_SIG      |                |   | Active table signature. The signature is used to detect changes made to this table. This signature is computed using the ALGORITHM_ID algorithm to process the entire binary content of the table. The resulting signature becomes available as input in the computation of the PROGRAM_SIG, EVENT_CHECK_SIG or the ALTERNATE_SIG. |
| SIGNATURES_RCD |                |   | Container for all computed signatures.   |
|                | ALGORITHM_ID   |   | Universal Identifier representing the algorithm used to produce a signature in this table. The end device must support at least one of the following algorithms:   |
|                |                | 06 08 2A 86 48<br>86 F7 0D 02 05 <sub>H</sub> | MD5 algorithm. The object identifier, using dot notation, is 1.2.840.113549.2.5.   |
|                |                | 06 05 2B 0E 03<br>03 13 <sub>H</sub>          | MDC2 algorithm. The object identifier, using dot notation, is 1.3.14.3.2.19  |

## 3.12.3 Data Definition (Continued)

| Identifier | Subfield         | Value | Definition  |
|------------|------------------|-------|---|
|            | METROLOGICAL_SIG |       | <p>A signature that captures the active metrological state of a end device. It is the digest of all tables having the sub-element FLAG1 of TABLE_LIST[]. TABLE_ID set to <b>TRUE</b>.</p> <p>The algorithm digests each TABLE_SIG in order of increasing index, starting with Standard Tables then Manufacturer Tables.</p> <p>When changing a Metrological Table that is not loggable the end-device shall generate a re-verification event.</p>   |
|            | EVENT_CHECK_SIG  |       | <p>A signature that is used to detect event log discontinuity for the life of the end-device. This signature is created by the application of the algorithm identified by ALGORITHM_ID to process the new EVENT_ENTRY_RCD entry, which is being added to the event log data table (Table 76). The processing logic is as follows:</p> <ol style="list-style-type: none"> <li>1. Update TABLE_SIG to reflect the changes to metrological tables.</li> <li>2. Update the signature element METROLOGICAL_SIG to reflect the changes to metrological tables.</li> <li>3. Create a new event entry in Std Table 76, Event Log Data, and initialize all of its EVENT_ARGUMENT elements to binary 0.</li> <li>4. Fill in the EVENT_TIME, the optional EVENT_NUMBER, the required EVENT_SEQ_NBR, USER_ID and EVENT_CODE then the NEW_VALUES portion of EVENT_ARGUMENT (TABLE_IDAS and NEW_VALUES, are placed after the EVENT_CHECK_SIG).</li> </ol> |

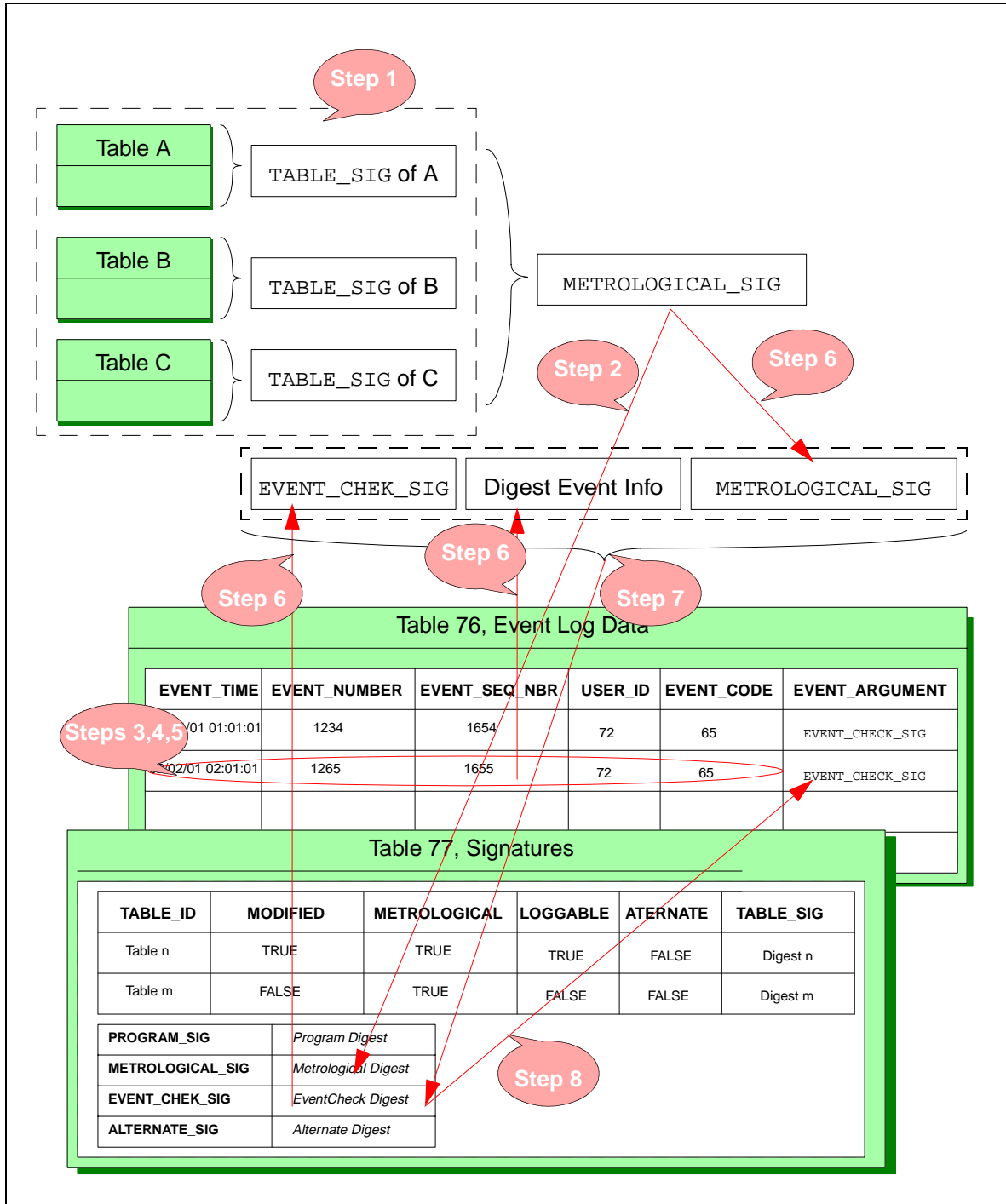
### 3.12.3 Data Definition (Continued)

| Identifier | Subfield | Value | Definition   |
|------------|----------|-------|--|
|            |          |       | <p>5. Using the ALGORITHM_ID algorithm, digest all the new event information that was placed in the EVENT_ENTRY_RCD. This includes all non EVENT_CHECK_SIG elements of the EVENT_ARGUMENT, as applicable, based on the value of the EVENT_CODE (for example: when EVENT_CODE=65, the EVENT_ARGUMENT will not be digested; when EVENT_CODE=64, only the TABLE_IDA field of the EVENT_ARGUMENT will be digested; when EVENT_CODE=70, then all octets starting at offset 16 to offset ACT_LOG_TBL.EVENT_DATA_LENGTH-1 of the EVENT_ARGUMENT will be digested).</p> <p>6. Fetch the present value of the EVENT_CHECK_SIG (or use a signature of binary 0 if this is the first event entry being created), append to it the EVENT_ENTRY_RCD signature, then append the newly computed metrological signature, METROLOGICAL_SIG,</p> <p>7. Digest the concatenated collection using the ALGORITHM_ID algorithm to produce the new event logger entry signature, EVENT_CHECK_SIG.</p> <p>8. Update (replace) the field EVENT_CHECK_SIG with the newly computed EVENT_CHECK_SIG and copy it to octets 0..15 of the EVENT_ARGUMENT in the newly create event log entry.</p> |

## 3.12.3 Data Definition (Continued)

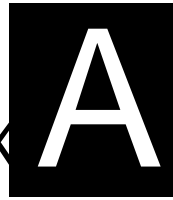
| Identifier | Subfield      | Value | Definition  |
|------------|---------------|-------|---|
|            | PROGRAM_SIG   |       | A signature that captures the active programmed state of an end device. It is the digest of all tables listed in TABLE_LIST using the ALGORITHM_ID algorithm. The algorithm digests each TABLE_SIG in order of increasing index, starting with Standard Tables then Manufacturer Tables. This signature can provide assist values for an external device in the caching of meter configuration state.   |
|            | ALTERNATE_SIG |       | <p>A signature that captures the active state of an end device, subject to (alternate unspecified) application requirements. It is the digest of all tables having the sub-element FLAG3 of TABLE_LIST[ ].TABLE_ID set to <b>TRUE</b>.</p> <p>The algorithm digests each TABLE_SIG in order of increasing index, starting with Standard Tables then Manufacturer Tables.</p> <p>A typical use of this signature is the collection of end-device ID independent programming table, which can be used to compare and validate the program state of a population of deployed end-device.</p> |

Figure 3.2 Event Log and Signatures Tables Update Sequence



---

# APPENDIX



## ANSI STANDARD C12.19 HISTORY LOG CODES

---

---

### A.1 ANSI Standard C12.19-1997 History Log Codes

---

| Event Code | Event Description                   | Argument Type         | Argument Description                    |
|------------|-------------------------------------|-----------------------|---|
| 00         | No Event                            | NA                    | None                                    |
| 01         | Primary Power Down                  | NA                    | None                                    |
| 02         | Primary Power Up                    | NA                    | None                                    |
| 03         | Time Changed (old time)             | NA                    | None, time tag if used equals old time. |
| 04         | Time Changed (new time)             | NA                    | None, time tag if used equals new time. |
| 05         | Time Changed (old time)             | <b>STIME_DATE</b>     | Old time                                |
| 06         | Time Changed (new time)             | <b>STIME_DATE</b>     | New time                                |
| 07         | End Device Accessed for Read        | NA                    | None                                    |
| 08         | End Device Accessed for Write       | NA                    | None                                    |
| 09         | Procedure Invoked                   | <b>TABLE_IDB_BFLD</b> | Procedure Number                        |
| 10         | Table Written To                    | <b>TABLE_IDC_BFLD</b> | Table Number                            |
| 11         | End Device Programmed               | NA                    | None                                    |
| 12         | Communication Terminated Normally   | NA                    | None                                    |
| 13         | Communication Terminated Abnormally | NA                    | None                                    |

---

## A.1 ANSI Standard C12.19-1997 History Log Codes (Continued)

| Event Code | Event Description            | Argument Type          | Argument Description  |
|------------|------------------------------|------------------------|---|
| 14         | Reset List Pointers          | INT8                   | See Procedure 4, "Reset List Pointers".   |
| 15         | Updated List Pointers        | UINT8                  | See Procedure 5, "Update Last Read Entry".  |
| 16         | History Log Cleared          | NA                     | None  |
| 17         | History Log Pointers Updated | UINT16                 | Number of entries advanced. See Procedure 5, "Update Last Read Entry".  |
| 18         | Event Log Cleared            | NA                     | None  |
| 19         | Event Log Pointers Updated   | UINT16                 | Number of entries advanced. See Procedure 5, "Update Last Read Entry".  |
| 20         | Demand Reset Occurred        | NA                     | None  |
| 21         | Self Read Occurred           | NA                     | None  |
| 22         | Daylight Savings Time On     | NA                     | None  |
| 23         | Daylight Savings Time Off    | NA                     | None  |
| 24         | Season Changed               | UINT8                  | New Season Number. See Procedure 9, "Remote Reset", Table 54, "Calendar" and Table 55, "Clock State".                       |
| 25         | Rate Change                  | UINT8                  | New Rate <sup>a</sup>   |
| 26         | Special Schedule Activated   | UINT8                  | See Table 54, "Calendar".   |
| 27         | Tier Switch Change           | ARRAY[ 2 ] OF<br>UINT8 | New Current Tier followed by New Demand Tier <sup>b</sup> , See Table 54, "Calendar" and Table 23, "Current Register Data". |
| 28         | Pending Table Activation     | TABLE_IDA_BFLD         | Table number activated  |
| 29         | Pending Table Clear          | TABLE_IDA_BFLD         | Table number removed from end device prior to activation.   |

a. The exact interpretation of the term "rate" is implementation specific.

b. Although it is possible to switch summations tiers independently from demand tiers. The Standard does not provide independent state information in support of this capability in Table 23, "Current Register Data".

## A.2 ANSI Standard C12.21-1999 Extensions to the History Log Codes

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



---

# APPENDIX

# B

## EVENT LOG CODES

---

In Version 1.0 of the Standard these event codes are be used exclusively by Canadian Event loggers. It is proposed that these shall be incorporated into Version 2.0 of the the ANSI C12.19 Standard for general use in regulated environmnets or where a stonger audit trail is desired.

---

### B.1 Event Log Codes for Use by Canadian Industry

---

#### B.1.1 Event Codes for Self-contained Event Loggers

| Event Code | Event Description                                 | Argument Type              | Argument Description  |
|------------|---|----------------------------|---|
| 58         | A single metrological procedure invocation event. | TABLE_IDA_BFLD, NEW_VALUES | TABLE_IDA_BFLD identifies the metrological Procedure that triggered this event. NEW_VALUES represent the collection of all changes that occurred as a result of the invocation of the metrological procedure. These are encoded as PSEM write service sequence. |
| 59         | A single metrological Table modification event.   | TABLE_IDA_BFLD, NEW_VALUES | TABLE_IDA_BFLD identifies the metrological Table change that triggered this event. All the fields are identical to that of event code 58, except that TBL_PROC_NBR sub-element is the table number.   |
| 60         | Metrological tables modified or programmed event. | NEW_VALUES                 | NEW_VALUES represent the collection of all changes that occurred in more than one metrological table. These are encoded as PSEM write service sequence.   |

---

### B.1.1 Event Codes for Self-contained Event Loggers (Continued)

| Event Code | Event Description      | Argument Type | Argument Description   |
|------------|------------------------|---------------|--|
| 61         | Verification Event.    | NEW_VALUES    | <p>NEW_VALUES representing the content of all metrological tables that were configured prior to sealing the end-device. These are encoded as PSEM write service sequence.</p> <p>If the EVENT_ARGUMENT is too small to hold a copy of all metrological tables at the time of the verification, then it is expected that this event code shall be preceded by one or more instances of codes 58, 59 or 60 as needed to capture the state of all metrological tables as verified, approved and sealed.</p> |
| 62         | Re-verification event. | NEW_VALUES    | <p>NEW_VALUES represent the collection of all changes that occurred to cause the re-verification event. These are encoded as PSEM write service sequence.</p> <p>If the EVENT_ARGUMENT is too small to hold a copy of all changes that triggered this event, this code may be preceded by one or more instances of codes 58, 59 or 60 as needed to capture the changes to the end-device metrological tables elements.</p>   |

### B.1.2 Event Codes for Signed Event Logs with no New Values

| Event Code | Event Description                                 | Argument Type                      | Argument Description  |
|------------|---|------------------------------------|---|
| 63         | A single metrological procedure invocation event. | EVENT_CHECK_SIG,<br>TABLE_IDA_BFLD | EVENT_CHECK_SIG is the event check signature for this event entry. TABLE_IDA_BFLD identifies the metrological Procedure that triggered this event.    |
| 64         | A single metrological Table modification event.   | EVENT_CHECK_SIG,<br>TABLE_IDA_BFLD | EVENT_CHECK_SIG is the event check signature for this event entry. TABLE_IDA_BFLD identifies the metrological Table change that triggered this event. |
| 65         | Metrological tables modified or programmed event. | EVENT_CHECK_SIG                    | EVENT_CHECK_SIG is the event check signature.   |

### B.1.2 Event Codes for Signed Event Logs with no New Values (Continued)

| Event Code | Event Description      | Argument Type   | Argument Description                          |
|------------|------------------------|-----------------|---|
| 66         | Verification Event.    | EVENT_CHECK_SIG | EVENT_CHECK_SIG is the event check signature. |
| 67         | Re-verification event. | EVENT_CHECK_SIG | EVENT_CHECK_SIG is the event check signature. |

### B.1.3 Event Codes for Signed Event Logs with New Values

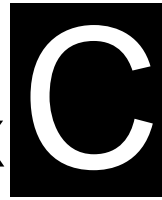
| Event Code | Event Description                                 | Argument Type                                     | Argument Description   |
|------------|---|---|--|
| 68         | A single metrological procedure invocation event. | EVENT_CHECK_SIG,<br>TABLE_IDA_BFLD,<br>NEW_VALUES | EVENT_CHECK_SIG is the event check signature for this event entry. TABLE_IDA_BFLD identifies the metrological Procedure that triggered this event. NEW_VALUES represent the collection of all changes that occurred as a result of the invocation of the metrological procedure. These are encoded as PSEM write service sequence. |
| 69         | A single metrological Table modification event.   | EVENT_CHECK_SIG,<br>TABLE_IDA_BFLD,<br>NEW_VALUES | EVENT_CHECK_SIG is the event check signature for this event entry. TABLE_IDA_BFLD identifies the metrological Table change that triggered this event. NEW_VALUES represent the collection of all changes that occurred as a result of the invocation of the metrological Table. These are encoded as PSEM write service sequence.  |
| 70         | Metrological tables modified or programmed event. | EVENT_CHECK_SIG,<br>NEW_VALUES                    | EVENT_CHECK_SIG is the event check signature. NEW_VALUES represent the collection of all changes that occurred as a result of the modification to the metrological tables. These are encoded as PSEM write service sequence.   |

### B.1.3 Event Codes for Signed Event Logs with New Values (Continued)

| Event Code | Event Description      | Argument Type                  | Argument Description  |
|------------|------------------------|--------------------------------|---|
| 71         | Verification Event.    | EVENT_CHECK_SIG,<br>NEW_VALUES | <p>EVENT_CHECK_SIG is the event check signature. NEW_VALUES represent the content of all metrological tables that were configured prior to sealing the end-device. These are encoded as PSEM write service sequence.</p> <p>If the EVENT_ARGUMENT is too small to hold a copy of all metrological tables at the time of the verification, then it is expected that this event code shall be preceded by one or more instances of codes 68, 69 or 70 as needed to capture the state of all metrological tables as verified, approved and sealed.</p> |
| 72         | Re-verification event. | EVENT_CHECK_SIG,<br>NEW_VALUES | <p>EVENT_CHECK_SIG is the event check signature. NEW_VALUES represent the collection of all changes that occurred to cause the re-verification event. These are encoded as PSEM write service sequence.</p> <p>If the EVENT_ARGUMENT is too small to hold a copy of all changes that triggered this event, this code may be preceded by one or more instances of codes 68, 69 or 70 as needed to capture the changes to the end-device metrological tables elements.</p>  |

---

# APPENDIX



## GLOSSARY

---

---

|                |   |
|----------------|---|
| <b>Acrobat</b> | Acrobat is a program from Adobe that lets you capture a document and view it in its original format and appearance. Acrobat is ideal for making documents or brochures that were designed for the print medium viewable electronically and capable of being shared with others on the Internet. To view an Acrobat document, which is called a Portable Document Format (PDF) file, you need Acrobat Reader. The Reader is free and can be downloaded from Adobe ( <a href="http://www.adobe.com">www.adobe.com</a> ). You can use it as a standalone reader or as a plug-in on a Web browser. This book is published using the Portable Document Format. |
| <b>ACT</b>     | Abbreviation for "ACTUAL", indicating the programmed functional capabilities of an end device. <sup>a</sup>   |
| <b>Address</b> | Those inputs whose states select a particular cell or group of cells. <sup>a</sup>  |
| <b>ADSL</b>    | Asymmetric Digital Subscriber Line.<br>ADSL is a technology for transmitting digital information at high speeds on existing copper phone lines to homes and businesses. ADSL is asymmetric in that it uses most of the channel to transmit downstream to the user and only a small part to receive information from the user. ADSL can transmit data at speeds ranging from 1.544 Mbps to 8 Mbps.   |

---

---



---

|                          |   |
|--------------------------|---|
| <b>Algorithm</b>         | The term algorithm (pronounced “AL-go-rith-um”) is a procedure or formula for solving a problem. The word derives from the name of the Arab mathematician, Al-Khowarizmi (825 AD). A computer program can be viewed as an elaborate algorithm. In mathematics and computer science, an algorithm usually means a small procedure that solves a recurrent problem. <sup>b</sup>  |
| <b>AMRA</b>              | Automated Meter Reading Association. Sponsor of the “Table Fests”.  |
| <b>ANSI</b>              | American National Standards Institute. The primary organization for fostering the development of technology standards in the United States. ANSI works with industry groups and is the U.S. member of the International Standards Organization (ISO) and the International Electrotechnical Commission (IEC).   |
| <b>Application layer</b> | The application layer (OSI layer 7) . The layer at which a user application interfaces with a communication network.  |
| <b>ASCII</b>             | American Standard Code for Information Interchange. ASCII is the most common format for text files in computers and on the Internet. In an ASCII file, each alphabetic, numeric, or special character is represented with a 7-bit binary number (a string of seven 0s or 1s). 128 possible characters are defined. ASCII was developed by the American National Standards Institute (ANSI).   |
| <b>Average demand</b>    | The consumption ( <i>e.g.</i> energy, volume) recorded during the integration period divided by the integration time period. <sup>a</sup>   |
| <b>Bandwidth</b>         | <p>The bandwidth of a transmitted communications signal is a measure of the range of frequencies the signal occupies. The term is also used in reference to the frequency-response characteristics of a communications receiving system. All transmitted signals, whether analog or digital, have a certain finite, non-zero bandwidth. The same is true of receiving systems.</p> <p>Generally speaking, bandwidth is directly proportional to the amount of data transmitted or received per unit time.</p>   |
| <b>Baud</b>              | <p>Baud was the prevalent measure for data transmission speed until replaced by a more accurate term, bps (bits per second). One baud is one electronic state change per second. Since a single state change can involve more than a single bit of data, the bps unit of measurement has replaced it as a better expression of data transmission speed.</p> <p>The measure was named after a French engineer, Jean-Maurice-Emile Baudot. It was first used to measure the speed of telegraph transmissions.</p> |
| <b>Baudrate</b>          | See Baud.   |

| <b>Binary</b>                | <p>Binary is the base two number system that computers use to represent data. It consists of only two numbers: “0” and “1”. In the table below, binary numbers are shown with their decimal equivalents:</p> <table> <thead> <tr> <th><u>Binary</u></th> <th><u>Decimal</u></th> </tr> </thead> <tbody> <tr><td>0000</td><td>0</td></tr> <tr><td>0001</td><td>1</td></tr> <tr><td>0010</td><td>2</td></tr> <tr><td>0011</td><td>3</td></tr> <tr><td>0100</td><td>4</td></tr> <tr><td>0101</td><td>5</td></tr> <tr><td>0110</td><td>6</td></tr> <tr><td>0111</td><td>7</td></tr> <tr><td>1000</td><td>8</td></tr> <tr><td>1001</td><td>9</td></tr> <tr><td>1010</td><td>10</td></tr> </tbody> </table> | <u>Binary</u> | <u>Decimal</u> | 0000 | 0 | 0001 | 1 | 0010 | 2 | 0011 | 3 | 0100 | 4 | 0101 | 5 | 0110 | 6 | 0111 | 7 | 1000 | 8 | 1001 | 9 | 1010 | 10 |
|------------------------------|---|---------------|----------------|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|----|
| <u>Binary</u>                | <u>Decimal</u>  |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| 0000                         | 0   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| 0001                         | 1   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| 0010                         | 2   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| 0011                         | 3   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| 0100                         | 4   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| 0101                         | 5   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| 0110                         | 6   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| 0111                         | 7   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| 1000                         | 8   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| 1001                         | 9   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| 1010                         | 10  |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| <b>Bit</b>                   | <p>Binary digit. A bit is the smallest unit of information in a computer. A bit has a single binary value, either 0 or 1. Although computers usually provide instructions that can test and manipulate bits, they generally are designed to store data in memory and execute instructions in bit multiples called bytes. In most computer systems, there are eight bits in a byte.</p>  |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| <b>Block average demand</b>  | <p>An average value occurring over a demand period specified by the end device. (e.g. Watthours/hours). The value may be saved by the end device for maximum or minimum registration.<sup>a</sup></p>   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| <b>BNF</b>                   | <p>In order to describe the tables, their syntax, contents and definitions, a formal table description language was developed. This language is described through a series of formal grammar rules written in a form called BNF or Backus Naur Format.</p>  |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| <b>Boolean logic</b>         | <p>The term “boolean logic” refers to a system of logical thought developed by the English mathematician, George Boole (1815-64). In computer operation with binary values, boolean logic can be used to describe electromagnetically charged memory locations or circuit states that are either charged (1 for true) or not charged (0 for false). The computer can use an AND gate or an OR gate operation to obtain a result that can be used for further processing.</p>  |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| <b>BPS</b>                   | <p>In data communications, bits per second (abbreviated bps) is a common measure of data speed for computer modems and transmission carriers. As the term implies, the speed in bps is equal to the number of bits transmitted or received each second (also See Baud).</p>   |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| <b>Call event</b>            | <p>An event instance, which is caused by a receipt of an explicit synchronous request (e.g. a request to write a table by a communication protocol).</p>  |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |
| <b>Category 1 end-device</b> | <p>An end-device that does not have remote configuration capabilities for its sealable parameters.</p>  |               |                |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |   |      |    |

---



---

|                              |   |
|------------------------------|---|
| <b>Category 2 end-device</b> | An end-device offering remote configuration capability for its sealable parameters and providing enabling/disabling hardware to control remote configuration.   |
| <b>Category 3 end-device</b> | An end-device that provides remote access to its sealable parameters and providing an audit trail to monitor changes to its sealable parameters.  |
| <b>Change event</b>          | An event instance, which is caused by change in state (e.g. Tamper detect, metering mode).  |
| <b>CCAC</b>                  | Consumer and Corporate Affairs Canada.<br>Now known as Measurement Canada of Industry Canada.   |
| <b>CCITT</b>                 | Comite Consultatif Internationale de Telegraphique et Telephonique; or Consultative Committee on International Telephone and Telegraphy. The CCITT, now known as the ITU-T (for Telecommunication Standardization Sector of the International Telecommunications Union), is the primary international body for fostering cooperative standards for telecommunications equipment and systems. It is located in Geneva, Switzerland.  |
| <b>Checksum</b>              | A checksum is a count of the number of bits in a transmission unit that is included with the unit so that the receiver can check to see whether the same number of bits arrived. If the counts match, it's assumed that the complete transmission was received. Both ANSI C12.18-1996 and ANSI C12.21-19xx provide a checksum count and verification as one of their services.  |
| <b>Clock</b>                 | A device that generates periodic signals used for synchronization. A device that measures and indicates time. A register whose content changes at regular intervals in such a way as to measure time. <sup>a</sup>  |
| <b>Collision</b>             | A collision is the result of two devices on the same communication interface transmitting packets at the same time. The collision fragments are discarded.  |
| <b>CRC</b>                   | Cyclic redundancy check. Cyclic redundancy checking is a method of checking for errors in data that has been transmitted on a communications link. A sending device applies a 16- or 32-bit polynomial to a block of data that is to be transmitted and appends the resulting cyclic redundancy code (CRC) to the block. The receiving end applies the same polynomial to the data and compares its result with the result appended by the sender. If they agree, the data has been received successfully. If not, the sender can be notified to re-send the block of data. |
| <b>Cumulative demand</b>     | An indicating demand meter in which the accumulated total of maximum demands during the preceding periods is indicated during the period after the meter has been reset and before it is reset again. Note: The maximum demand for any one period is equal or proportional to the difference between the accumulated readings before and after reset. <sup>a</sup>  |

|                        |  |
|------------------------|--|
| <b>Customer</b>        | The purchaser and/or user of a product or service supplied by a service provider or utility. <sup>a</sup>  |
| <b>Data encryption</b> | The changing of the form of a data stream such that only the intended recipient can read or alter the information and detect unauthorized messages. <sup>a</sup>   |
| <b>Data-link layer</b> | The data-link layer (OSI layer 2). This layer provides error control and synchronization for the physical level and does bit-stuffing for strings of 1's in excess of 5.   |
| <b>Datagram</b>        | A datagram is a self-contained, independent entity of data carrying sufficient information to be routed from the source to the destination computer without reliance on earlier exchanges between this source and destination computer and the transporting network. The term has been generally replaced by the term packet. <sup>c</sup> |
| <b>DCE</b>             | Data Communication Equipment. The RS-232C interface that a MODEM or other serial device uses in exchanging data with the computer (DTE).   |
| <b>Download</b>        | The act of downloading is the confirmed and validated process of transferring a copy of the content of end-device Table elements into the end-device-agent's record-keeping system. This process shall be reliable, secure, and demonstratively verifiable.  |
| <b>Decade</b>          | A functional grouping of tables by application into groups of ten. The tables are numbered "X0" through "X9", with "X" representing the decade number. <sup>a</sup>  |
| <b>Element</b>         | As in ANSI C12.19 / IEEE 1377 Table Element, is the union of all of the terminal-elements, which share the same index prefix. An element can be a simple type, a set, an array or a selection from an array.   |
| <b>Demand</b>          | The rate of consumption ( <i>e.g.</i> power, volume/hour). <sup>a</sup>  |
| <b>DES</b>             | Data Encryption Standard. A widely-used method of data encryption using a private (secret) key.  |
| <b>DIM</b>             | Abbreviation for "DIMENSION", indicating the maximum functional capability designed into an end device. <sup>a</sup>   |
| <b>DTE</b>             | Data Terminal Equipment. In computer data transmission, DTE is the RS-232C interface that a computer uses to exchange data with a MODEM or other DCE serial device.  |
| <b>EIA</b>             | Electronic Industries Association.   |
| <b>End-device</b>      | The closest device to the sensor or control point within a metering application communication system which is compliant with the Utility Industry End Device Data Tables (ANSI C12.19-1997). <sup>a</sup>  |

---



---

|                              |  |
|------------------------------|--|
| <b>End-device agent</b>      | That who is responsible for managing the end-device operations and programs. It is capable of affecting change to the end-device program. It is also responsible for keeping a record of all end-device programming parameters, constants, change history and event logger records. The End-Device Agent is also responsible for maintaining and demonstrating continuity between metrological data records and end-device programs, exclusively through the use of these tables.  |
| <b>Event-log Downloading</b> | The process of downloading newly created entries or all entries from Event-Log Table (Table 76, ENTRIES), and all related end-device tables, and end-device Identification Table, in the correct order and the preservation of 32 bit sequence numbers using the 32 bit LAST_ENTRY_SEQ_NBR.  |
| <b>Event</b>                 | An event is a noteworthy occurrence that has a location in time and space. It occurs at a point in time; it does not have duration. Implement (model) something as an event if its occurrence has consequences. When we use the term event by itself, we actually mean the event logger data - that is, a description of all related individual event occurrences that have the same general form.   |
| <b>Event instance</b>        | <p>A specific occurrence of an event. Events have parameters that characterize each individual event instance. The event instance parameters include information about the state of the end-device before the event, after the event, the event time and related event-identifying information. The event parameters can be made available for inspection directly or indirectly through a derivative calculable. There are four kinds of event instances: call event, change event, signal event and time event.</p> <p>EVENT LOGGER DATA (N): This is the entire content of all tables designated as programming tables, associated pointers and signatures that are expressed by or implied by event logger record. The event logger data originates initially inside the End-Device, but it may extend into the End-Device agent's record keeping system</p> |
| <b>FLC table</b>             | Function Limiting Control table. The first table in each decade specifies the limits designed into an end device with respect to variables used within the decade. <sup>a</sup>  |
| <b>Full-duplex</b>           | Full-duplex data transmission means that data can be transmitted in both directions on a signal carrier at the same time. One workstation can be sending data on the line while another workstation is receiving data. Full-duplex transmission necessarily implies a bidirectional line (one that can move data in both directions).  |

---

---

|                           |   |
|---------------------------|---|
| <b>GPS</b>                | <p>Global Positioning System</p> <p>The Global Positioning System is a “constellation” of 24 well-spaced satellites that orbit the Earth and make it possible for people with ground receivers to pinpoint their geographic location. The location accuracy is anywhere from 100 to 10 meters for most equipment. Accuracy can be pinpointed to within one (1) meter with special military-approved equipment. GPS equipment is widely used in science and has now become sufficiently low-cost for use in commercial applications.</p> |
| <b>Gregorian calendar</b> | <p>The Gregorian calendar is the calendar in current use in the Western world, both as the civil and Christian ecclesiastical calendar. Instituted by Pope Gregory XIII in 1582, the calendar has 365 days with an extra day every four years (leap year) except in years divisible by 100 but not divisible by 400.</p>  |
| <b>Half-duplex</b>        | <p>Half-duplex data transmission means that data can be transmitted in both directions on a signal carrier, but not at the same time. One workstation can send data on the line and then immediately receive data on the line from the same direction in which data was just transmitted. Like full-duplex transmission, half-duplex transmission implies a bidirectional line (one that can carry data in both directions).</p>  |
| <b>HDLC</b>               | <p>High-Level Data Link Control. A group of protocols or rules for transmitting data between network points (sometimes called nodes). In HDLC, data is organized into a unit (called a frame) and sent across a network to a destination that verifies its successful arrival. The HDLC protocol also manages the flow or pacing at which data is sent. HDLC is one of the most commonly-used protocols Layer 2 of OSI.</p>   |

---

---

**Hexadecimal**

Hexadecimal describes a base-16 number system. That is, it describes a numbering system containing 16 sequential numbers as base units (including 0) before adding a new position for the next number. The hexadecimal numbers are 0-9 and then use the letters from A-F (to represent the decimal range 10 to 15).

Hexadecimal is a convenient way to express binary numbers in modern computers in which a byte is almost always defined as containing eight bits. One hexadecimal digit can represent four binary digits. Two hexadecimal digits can represent eight binary digits, or an octet.

The equivalence of binary, decimal, and hexadecimal numbers is shown below:

| <u>Binary</u> | <u>Decimal</u> | <u>Hexadecimal</u> |
|---------------|----------------|--------------------|
| 0             | 0              | 0                  |
| 1             | 1              | 1                  |
| 10            | 2              | 2                  |
| 11            | 3              | 3                  |
| 100           | 4              | 4                  |
| 101           | 5              | 5                  |
| 110           | 6              | 6                  |
| 111           | 7              | 7                  |
| 1000          | 8              | 8                  |
| 1001          | 9              | 9                  |
| 1010          | 10             | A                  |
| 1011          | 11             | B                  |
| 1100          | 12             | C                  |
| 1101          | 13             | D                  |
| 1110          | 14             | E                  |
| 1111          | 15             | F                  |
| 10000         | 16             | 10                 |

**IC**

[Industry Canada](#), The sponsor of this book and a key contributor to the realization of ANSI C12.18/19/21/22, has a mandate to make Canada more competitive by fostering the growth of Canadian business, by promoting a fair and efficient marketplace for businesses and consumers, and by encouraging scientific research and technology diffusion. Industry Canada serves a diverse client base. Clients range from firms in such sectors as telecommunications, aerospace, forestry, manufacturing and services to small businesses in all sectors of the economy. Clients also include the scientific and academic communities, consumer organizations and professional groups.

**IEC**

International Electrotechnical Commission.

---

---

|                           |  |
|---------------------------|--|
| <b>IEEE</b>               | Institute of Electrical and Electronics Engineers. The IEEE describes itself as “the world’s largest technical professional society -- promoting the development and application of electrotechnology and allied sciences for the benefit of humanity, the advancement of the profession, and the well-being of our members”. The IEEE acts as the developer of standards that often become national and international standards.  |
| <b>Interchangeability</b> | Any two end devices are interchangeable if one can be replaced by another and be expected to operate identically. In this context the end devices are identical if they measure the same sources, have identical data registers, use the same device programming methods and use the same communication interfaces.  |
| <b>Interface</b>          | Interface has a number of meanings. A programming interface is the set of statements, functions, options, and other ways of expressing program instructions and data provided by a program or language for a programmer to use. The physical and logical arrangement supporting the attachment of any device to a connector or another device.   |
| <b>Interoperability</b>   | Any two end devices are interoperable if one can be substituted by another. Such devices must have equivalent capabilities with respect to the functions they are capable of performing. Interoperability may be achieved either at the end device level, through device configuration, or at the master station level through the use of communication and data conversion facilities.  |
| <b>ISO</b>                | International Standards Organization. ISO, founded in 1946, is the leading international standards organization. Among its developed standards is Open Systems Interconnection (OSI), a suite of communication protocols used widely in Europe. Many countries have national standards organizations such as the American National Standards Institute (ANSI) that participate in and contribute to ISO standards making.  |
| <b>ISO date format</b>    | The ISO date format is a standard way to express a numeric calendar date that eliminates ambiguity using printable characters (e.g. ASCII). ISO 8601 provides a standard cross-national approach that says: Year first, followed by month, then day with each separated by a hyphen “-”. Numbers less than 10 preceded by a leading zero. Years expressed as “0” prior to year 1 and as “-1” for the year prior to year 0 (and so forth). Thus, March 31, 1998 would be: 1998-03-31. To express whether the date reflects the Julian calendar or the Gregorian calendar, the date can be followed with a “J” or a “G”. |
| <b>ITU-T</b>              | International Telecommunications Union - Telecommunication Standardization Sector. The ITU-T is the primary international body for fostering cooperative standards for telecommunications equipment and systems. It was formerly known as the CCITT. It is located in Geneva, Switzerland.   |

---



---

|                               |   |
|-------------------------------|---|
| <b>Load profile</b>           | The recording, storage and analysis of consumption data over a period of time for a particular installation. <sup>a</sup>   |
| <b>Manufacturer Tables</b>    | Manufacturer tables are those structures specified by individual end device vendors. They can be used to allow introduction of new innovations or to provide customer requested data structures. They also provide a path for new functions to evolve into a standard table. ANSI Std. C12.19-1997 provides access for a total of 2048 manufacturer tables. <sup>a</sup>  |
| <b>Maximum demand</b>         | The highest demand measured over a selected period of time, such as one month. <sup>a</sup>   |
| <b>MC</b>                     | <p>Measurement Canada. <a href="#">Industry Canada</a>, through Measurement Canada, has the power in Canada to issue official certificates of calibration and inspection for measuring instruments used in trade. Device types inspected under the authority of the Electricity and Gas Inspection Act include Gas and Electricity measuring meters.</p> <p>The mandate of <a href="#">Measurement Canada</a> is to create and maintain a fair and efficient marketplace for the buying and selling of electricity, uniform units of measurement have been established and are displayed by all in-service meters. The units for energy measurement are contained in Section 3 (1) (a) of the Electricity and Gas Inspection Act and the units of demand measurement are specified in Sections 5 (1) and (2) of the Electricity and Gas Inspection Regulations.</p> |
| <b>MD5 algorithm</b>          | <p>The MD5 algorithm takes as input a message of arbitrary length and produces as output a 128-bit “fingerprint” or “message digest” of the input. The MD5 algorithm is intended for digital signature applications, where a large file must be “compressed” in a secure manner before being encrypted with a private (secret) key under a public-key crypto-system such as RSA.</p> <p>In essence, MD5 is a way to verify data integrity, and is much more reliable than checksum and many other commonly used methods.</p>  |
| <b>Meter</b>                  | A device which measures and records the consumption or usage of the product/service. <sup>a</sup>   |
| <b>Metrological Parameter</b> | A metrological parameter is any constant, factor or algorithm used by metering device to produce results for the purpose revenue metering.  |
| <b>Minimum demand</b>         | The lowest demand measured over a selected period of time, such as one month. <sup>a</sup>  |

|                              |   |
|------------------------------|---|
| <b>MODEM</b>                 | <p>Modulator/demodulator. A MODEM modulates digital signals from a computer or other digital device to analog signals for a conventional copper twisted-pair telephone line and demodulates the analog signal and converts it to a digital signal for the digital device.</p> <p>In recent years, the 2400 bps MODEM has become obsolete. Today's 14.4 Kbps and 28.8 Kbps MODEMS are themselves a temporary stepping stone to the higher bandwidth devices and carriers of tomorrow. In 1997, MODEMS with 56 Kbps downstream capability have become common. Asymmetric Digital Subscriber Line (ADSL) offers bandwidth in the megabyte range.</p>   |
| <b>MODEM V.xx standards</b>  | <p>The most commonly used modem transfer rates are embodied in the series of CCITT/ITU standards, including: V.22, V.22bis, V.32, 32bis, V.32terbo, V.34, V.34bis, V.42 and V.90. In general, when modems "handshake," they agree on the highest standard transfer rate that both can achieve. Beginning with V.22bis, CCITT/ITU transfer rates increase in 2400 bps multiples. Bis stands for "second version". Terbo stands for "third version".</p>  |
| <b>MOD, Modulus operator</b> | <p>The modulus, or the remainder, operator can only be applied to integral operands. It returns the remainder of the left operand divided by the right operand. e.g. <math>10 \text{ MOD } 3 = 1</math>.</p>  |
| <b>NEMA</b>                  | <p>National Electrical Manufacturers Association. Sponsor and publisher of ANSI C12.18-1996, ANSI C12.19-1997, ANSI C12.21-19xx and ANSI C12.22-19xx.</p>   |
| <b>Network</b>               | <p>A series of points or nodes interconnected by communication paths. Networks can interconnect with other networks and contain sub-networks. The most common configurations of networks include the bus, star and ring topologies. Networks can be characterized in terms of spatial distance as local area networks (LANs), metropolitan area networks (MANs) and wide area networks (WANs). Networks can also be characterized by the type of data transmission technology in use on it; by the type of information it carries (a combination of voice, video or data); the nature of its connections (dial-up, switched, dedicated, virtual, peer-to-peer, one-to-many or many-to-many connections); and by the type of the physical links (for example optical fibre, coaxial cable or copper wire).</p> |
| <b>Network layer</b>         | <p>The network layer (OSI layer 3). This layer handles the routing of the data (sending it in the right direction to the right destination on outgoing transmissions and receiving incoming transmissions at the packet level).</p>   |
| <b>NIST</b>                  | <p>National Institute of Standards and Technology. NIST, formerly the National Bureau of Standards, promotes and maintains measurement standards. It also has active programs for encouraging and assisting industry and science to develop and use these standards.</p>  |

---

---

|               |   |
|---------------|---|
| <b>Octet</b>  | <p>In a computer or in a network, an octet (from the Latin octo or “eight”) is a sequence of eight bits. A octet is an eight-bit byte, which in most, but not all, computer systems is eight sequential bits. Since “byte” does not in itself always mean eight bits, a non-ambiguous term was needed.</p>  |
| <b>OOP</b>    | <p>Object-oriented programming. Object-oriented programming is a way of looking at computer programming. Historically, programs have been viewed as procedures (or we may think of these as “verbs”) that operate on data. OOP takes the view that programs should start by thinking about the data (or “nouns”) first.</p> <p>By using data modelling concepts and techniques, a programmer can identify data objects and their relationships. A generalization of a data object along with its possible data variables and methods (what to do with the variables) is a class of data objects. A real instance of a class is an object. (It’s what you run in the computer.)</p>                                      |
| <b>OSI</b>    | <p>Open Systems Interconnection. OSI is a standard model for the functional layering of a communications system. Currently, it is Recommendation X.200 of the CCITT (the international telecommunications standards-making body). Although many existing hardware and software products have been developed on a slightly different model, the OSI model is generally respected as the standard to aspire to in building future products.</p> <p>OSI has a seven layer model: The application layer (layer 7), the presentation layer (layer 6), the session layer (layer 5), the transport layer (layer 4), the network layer (layer 3), the link (or data-link) layer (layer 2) and the physical layer (layer 1).</p> |
| <b>Packet</b> | <p>A packet is the unit of data that is routed between an origin and a destination on a packet-switched network. The individual packets for a given record may travel different routes through the network. When they have all arrived, they are reassembled into the original record at the receiving end.</p> <p>A packet-switching scheme is an efficient way to handle transmissions on a connectionless network such as the Internet. “Packet” and “datagram” in general mean the same thing.</p>  |

|                       |  |
|-----------------------|--|
| <b>Parity</b>         | <p>Parity (from the Latin <i>paritas</i>: equal or equivalent) refers to a technique of checking whether data has been lost or written over when it's moved from one place in storage to another or when transmitted between computers.</p> <p>The parity is an additional bit, the parity bit, which is added to a group of bits that are moved together. This bit is used only for the purpose of identifying whether the bits being moved arrived successfully.</p> <p>Before the bits are sent, they are counted and if the total number of data bits is even, the parity bit is set to one so that the total number of bits transmitted will form an odd number (assuming that the system operates with an odd parity settings). If the total number of data bits is already an odd number, the parity bit remains or is set to 0 (unless the system operates using even parity settings). At the receiving end, each group of incoming bits is checked to see if the group totals to an odd (assuming odd parity settings) number. If the total is even (odd for even parity settings), a transmission error has occurred and the transmission is either retried or aborted.</p> |
| <b>Parser</b>         | <p>In computer technology, a parser is a program, usually part of a compiler, that receives input in the form of sequential source program instructions, interactive on-line commands, mark-up tags, or some other defined interface and breaks them up into parts (for example, the nouns (objects), verbs (methods), and their attributes or options) that can then be managed by other programming (for example, other components in a compiler). A parser may also check to see that all necessary input has been provided.</p>  |
| <b>Pascal</b>         | <p>Pascal is a strongly-typed third-generation language (3GL) with a one-pass compiler. Designed for instructional purposes about 1967-68 by Nicholas Wirth, Pascal requires a programmer to define all routines and variables fully, including the nature of their use, before using them. Pascal is the language on which many programmers first learn how to write structured, compiled programs. While commercial versions of Pascal have been made available, it has had limited success in the business world.</p> <p>The Utility Industry Standard Tables language is derived from Pascal.</p>  |
| <b>PBX</b>            | <p>Private Branch Exchange. A PBX is a telephone system within an enterprise that switches calls between enterprise users on local lines while allowing all users to share a certain number of external phone lines. The main purpose of a PBX is to save the cost of requiring a line for each user to the telephone company's central office. The PBX is owned and operated by the enterprise rather than the telephone company.</p>   |
| <b>Peak demand</b>    | <p>References the maximum demand.<sup>a</sup></p>  |
| <b>Physical layer</b> | <p>The physical layer (OSI layer 1). This layer conveys the bit stream through the network at the electrical and mechanical level.</p>   |

---



---

|   |   |
|---|---|
| <b>Point-to-point</b>                   | Point-to-point communications is defined as communication between two devices through a single optical interface. <sup>d</sup>  |
| <b>Present average demand</b>           | An average value occurring during a current demand interval or sub-interval. (e.g. Watts or VA). <sup>a</sup>   |
| <b>Presentation layer</b>               | The presentation layer (OSI layer 6). This is a layer, usually part of an operating system, that converts incoming and outgoing data from one presentation format to another. It identifies the syntax of the user data being transmitted and provides user service functions such as encryption, file data conversion and terminal emulation.  |
| <b>Program table</b>                    | An end-device table that contains setup, configuration or general product or record keeping information about the end-device. Programming tables shall not contain sensory information, end-device state information or end-device logs that are volatile and change in the normal course of operation of the end-device.   |
| <b>PSEM</b>                             | Protocol Specification for Electricity Metering as referenced in C12.18-1996.   |
| <b>Remotely configurable end-device</b> | An end-device that permits metrological adjustments or sealable parameters to be deleted, appended to, modified, or substituted in whole or in part by any type of communications link from another device, such as a geographically local or remote console or computer, whether or not the secondary apparatus is part of the network connecting the devices.   |
| <b>RS-232C</b>                          | Recommended Standard 232 Revision C. RS-232C is a long-established standard ("C" is the current version) that describes the physical interface and protocol for relatively low-speed serial data communication between computers and related devices. It was defined by EIA.  |
| <b>RSA</b>                              | Rivest-Shamir-Adleman. RSA is an Internet encryption and authentication system that uses an algorithm developed in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman. The RSA algorithm is the most commonly used encryption and authentication algorithm and is included as part of the Web browsers from Netscape and Microsoft. It's also part of Lotus Notes, Intuit's Quicken, and many other products. The encryption system is owned by RSA Data Security, now a subsidiary of Security Dynamics. The company licenses the algorithm technologies and also sells development kits. The technologies are part of existing or proposed Web, Internet, and computing standards. |
| <b>Season</b>                           | A calendar specified period used for activation of rate schedules. <sup>a</sup>   |
| <b>Serial</b>                           | In data transmission, the technique of time division is used, where time separates the transmission of individual bits of information sent serially.  |
| <b>Session layer</b>                    | The session layer (OSI layer 5). This layer manages the establishment of a continuing series of requests and responses between the user applications at each end.   |

|                              |   |
|------------------------------|---|
| <b>Signal event</b>          | An event instance that is caused by explicitly named asynchronous communication among objects (e.g. advancing list pointers as a result of execution of an advance list pointers procedure).  |
| <b>Signature</b>             | The signature, often referred to as a hash-code, is an n-bit sequence that is generated using an algorithm on the content of an end-device table or a group of end-device tables. The resulting sequence of bits is truncated to a small number of bits (e.g. 128 bits). A good hash function will enable (a) detection of file (or table) content corruption (unauthorized change); and (b) authenticate the End-Device Agent who created the file (or table). |
| <b>Sliding window demand</b> | Sliding window (rolling-interval) demand. The block demand calculated over an integration period which includes sub-intervals of previous demand calculations. <sup>a</sup>   |
| <b>Standard Tables</b>       | Standard tables are those whose structures are specified by ANSI Std. C12.19-1997. These should be used for both end device programming and reading. The tables provide control tables as well as data tables for a wide variety of functions. ANSI Std. C12.19-1997 provides for a total of 2048 standard tables. <sup>a</sup>   |
| <b>Sub element</b>           | A sub-element represents a subset of a <b>terminal-element</b> , that is a single bit of a <b>SET</b> , or a member of a <b>BIT FIELD</b> ,   |
| <b>Table</b>                 | Functionally related utility application data <b>elements</b> , grouped together into a single data structure for transport. <sup>a</sup>   |
| <b>Time event</b>            | An event instance that is caused by the arrival of an absolute time or the passage of relative amount of time (e.g. activation of pending tables, performance of a self-read based on a time of use schedule).  |
| <b>Tariff</b>                | A published list of rate schedules and terms and conditions. <sup>a</sup>   |
| <b>Terminal element</b>      | A terminal-element is a restricted subset of an <b>element</b> that is the smallest component of an element that can be transmitted as an integral number of <b>octets</b> without loss of its meaning or interpretation during transmission, in accordance with octet ordering and bit packing.  |
| <b>Time-of-use metering</b>  | See TOU metering.   |
| <b>TOU metering</b>          | Metering equipment that separately records metered or measured quantities according to a time schedule. <sup>a</sup>  |
| <b>Transport layer</b>       | The transport layer (OSI layer 4). This layer manages the end-to-end control (for example, determining whether all packets have arrived) and error-checking.  |
| <b>UOM</b>                   | Unit of Measure. <a href="#">Table 12 - Unit Of Measure</a> , provides a method for describing or selecting data source attributes. Some of the attributes include the physical quantity measured, the time base used for averaging, the scaling constants, direction of flow, method of measurement and harmonic components.   |

---

|                   |  |
|-------------------|--|
| <b>Utility</b>    | A provider of electricity, gas, water, telecommunications or related services to a community. <sup>a</sup>   |
| <b>V.22</b>       | V.22 provides 1200 bits per second at 600 baud.  |
| <b>V.32</b>       | V.32 Provides 4800 and 9600 bits per second at 2400 baud.  |
| <b>V.32bis</b>    | V.32bis Provides 14,400 bits per second or fall-back to 12,000, 9600, 7200, and 4800 bits per second.  |
| <b>V.32terbo</b>  | V.32terbo Provides 19,200 bits per second or fall-back to 12,000, 9600, 7200, and 4800 bits per second; can operate at higher data rates with compression.   |
| <b>V.34</b>       | V.34 Provides 28,800 bits per second or fall-back to 24,000 and 19,200 bits per second and backwards compatibly with V.32 and V.32bis.   |
| <b>V.34bis</b>    | V.34bis Provides up to 33,600 bits per second or fall-back to 31,200 or V.34 transfer rates.   |
| <b>V.42</b>       | V.42 has the same transfer rate as V.32, V.32bis, and other standards but with better error correction and therefore more reliable.  |
| <b>V.90</b>       | V.90 Provides up to 56,000 bits per second downstream (but in practice somewhat less). Derived from US Robotics's x2 technology and Rockwell's K56flex technology.   |
| <b>whatis.com</b> | <a href="http://whatis.com">whatis.com</a> is a knowledge exploration tool pertaining to information technology, in particular the Internet and computers. It contains over 1,200 individual definition/topics and a number of quick-reference pages. The topics contain over 5,000 hyper-linked cross-references and over 3,000 links to others sites for further information. Many glossary definitions have been acquired from this site.   |
| <b>X.25</b>       | The X.25 protocol, adopted as a standard by CCITT, is a commonly-used network protocol. The X.25 protocol allows computers on different public networks to communicate through an intermediary computer at the network layer level. X.25's protocols correspond closely to the data-link and physical-layer protocols defined in the OSI communication model.  |
| <b>Year 2000</b>  | The year 2000 raises problems for anyone who depends on a program in which the year is represented by a two-digit number, such as "97" for 1997. The problem is that when the two-digit space allocated for "99" rolls over to 2000, the next number will be "00". Frequently, program logic assumes that the year number gets larger, not smaller, so "00" may wreak havoc in a program that hasn't been designed to account for the millennium.<br><br>ANSI C12.19-1997 and supporting communication protocols are immune to this problem. |

a. ANSI Std. C12.19-1997.

b. The Oxford Dictionary of Philosophy, Oxford University Press, (1994).

c. Internet's Request for Comments (RFC) 1594

d. ANSI Std. C12.18-1996.

---

# APPENDIX **D**

## REFERENCES

---

- 
1. *ANSI C12.19 over ANSI C12.18 Standards Conformance Tests*, Nertec Design Inc, Contribution to ANSI C12.23 working group, June 2001.
  2. ANSI Std C12.18-1996, *Protocol Specification for ANSI Type 2 Optical Port*, NEMA, 1996.
  3. ANSI Std C12.19-1997, *Utility Industry End Device Data Tables*, NEMA, 1997.
  4. ANSI Std C12.21-1999, *Protocol Specification for Telephone Modem Communication*, NEMA
  5. ANSI Std X3.92-1981, *Data Encryption Algorithm*, 1981.
  6. ANSI/IEEE Std 1012-1986, *Software Verification and Validation Plans*.
  7. ANSI/IEEE Std 829-1983, *Software Test Documentation*. revised 1991.
  8. CAN/CSA ISO/IEC-10118-1:2000(E), Information Technology - Security techniques - Hash-functions - Part 1 *General*.
  9. CAN/CSA ISO/IEC-10118-2:2000(E), Information Technology - Security techniques - Hash-functions - Part 2 *Hash-functions using an n-bit block cipher algorithm*.
  10. *Flowchart Security Of Trade Devices*, Measurement Canada, 2001
  11. ISO 7498/1, *OSI 7 Layer Reference Model*.
  12. ISO/IEC 646: 1991, *Information technology - ISO 7-bit coded character set for information interchange*.
  13. *Measurement Canada Principles for Sealing Meters and Trade Devices*, Measurement Canada, 2001
  14. *Physical Seal And Event Logger*, Measurement Canada, 2001
-

- 
- 
15. *Proposed Event Logger Specifications*, Measurement Canada, 2000.
  16. *Specifications Relating to Metrological Audit Trails* , Measurement Canada Provisional Specification, PS-EGMV-XX-E, 2001
  17. *The Oxford Dictionary of Philosophy*, Oxford University Press, (1994).
  18. *The Unified Modeling Language Reference Manual*, Addison Wesley, 1998. ISBN 0-201-30998-X
  19. US Standard X9.31, Part 1: *RSA signature standard*.
  20. US Standard X9.31, Part 2: *MD2, MD5, SHA, MDC-2*.
  21. US Standard X9.31, Part 3: *Certificate management*.