ANSI C12.19-2008 Load Profile Implementation Guide

Written by Avygdor Moise

Future DOS R&D Inc.

1 References

1.1 ANSI C12.19-2008 Utility Industry End Device Data Tables

1.2 Algorithms for the Conversion of Table Element Values to Engineering Units

Algorithms for the Conversion of Table Element Values to Engineering Units, A. Moise and R. Tucker, May 16, 2008. Also See ANNEX L of ANSI C12.19-2008.

2 Terms and Definitions

2.1 API

Throughout this document, the terms “API” refers to the Application Programming Interface used the retrieval of load profile interval data from ANSI C12.19 End Devices.

2.2 Clock

Throughout this document, the terms “clock” refers to an ANSI C12.19 compliant End Device’s real-time clock calendar function that can be retrieved strictly using ST52 or ST55 and set using SP10 or SP32. The Element-type representation of the clock is retrieved from ST0, GEN_CONFIG_TBL.FORMAT_CONTROL_2.TM_FORMAT, and the allowable values that this Element may attain are defined by TDL.TM_FORMAT_ENUM.

2.3 End Device

See Meter.

2.4 Load Profile, Load Profile Data, Interval Data

Throughout this document, the terms “load profile”, “load profile data” or “interval data”, regardless of singular or plural form, refers to an ANSI C12.19 compliant End Device that can be configured to record interval data strictly using ANSI C12.19 Standard Tables from Decade 6, “Load Profile Tables”.

2.5 Manufacturer

Throughout this document, the term “Manufacturer”, regardless of singular or plural form, refers to the individual (or plurality of) meter vendor (or vendors).

2.6 Manufacturer Tables

Throughout this document, the term “Manufacturer Tables”, regardless of singular or plural form, refers to table structures specified by individual meter vendor.

2.7 Meter

Throughout this document, the term “meter”, regardless of singular or plural form, refers to an ANSI C12.19 compliant End Device that is capable of recording interval data using load profile tables.

2.8 MP

Throughout this document, the terms “MPn”, regardless of the numeric value assumed by the letter “n” refers to a Manufacturer proprietary Procedure. *e.g.* General Electric Procedure 1 may be referenced as MP1 (some manufacturers number their procedures 2048 through 4087; these are equivalent to manufacturer tables MP0 through MP2039).

2.9 MT

Throughout this document, the terms “MTn”, regardless of the numeric value assumed by the letter “n” refers to a Manufacturer proprietary Table. *e.g.* General Electric Table 1 may be referenced as MT1 (some manufacturers number their

tables 2048 through 4087; these are equivalent to manufacturer tables MT0 through MT2039).

2.10 SP

Throughout this document, the terms “SPn”, regardless of the numeric value assumed by the letter “n” refers to an ANSI C12.19 compliant Standard Procedure. *e.g.* ANSI C12.19 Standard Procedure 10, “Set Date and/or Time” may be referenced as ST10.

2.11 ST

Throughout this document, the terms “STn”, regardless of the numeric value assumed by the letter “n” refers to an ANSI C12.19 compliant Standard Table. *e.g.* ANSI C12.19 Standard Table 05, “Device Identification Table” may be referenced as ST5.

2.12 TDL

Table Definition Language is the syntax used for detailing the ANSI C12.19 Standard and/or an End Device Tables’ structure, defined types and constraints.

2.13 Time Stamp

Throughout this document, the terms “time stamp”, “time-stamp” or “timestamp” refer to an ANSI C12.19 compliant clock value that was captured in an ANSI C12.19 Table Element and formatted strictly as described in ANSI C12.19 data types for the corresponding tables and ST0 of the meter. Also See CLOCK.

3 ANSI C12.19-2008 Load Profile Overview

ANSI C12.19-2008, “Utility Industry End Device Data Tables”, defines a collection of tables supporting the configuration of meters and the collection of load profile data. These tables are referred collectively as “Load Profile Tables”, defined as follows:

Table 60, “Load Profile Dimension Limits”,
Table 61, “Actual Load Profile Limiting”,
Table 62, “Load Profile Control”,
Table 63, “Load Profile Status”, and
Table 64, “Load Profile Data Set One” through Table 67, “Load Profile Data Set Four”.

In addition ANSI C12.19-2008 defines a number of procedures that were designed for the management of the meter's load profilers. These are defined as follows:

Procedure 04, "Reset List Pointers",
Procedure 05, "Update Last Read Entry",
Procedure 10, "Set Date and/or Time"
Procedure 16 "Start Load Profile", and
Procedure 17 "Stop Load Profile",
Procedure[1] 30, "Start Secured Registers",
Procedure 31, "Stop Secured Registers" and
Procedure[2] 32, "Set Precision Date and/or Time".

The above Tables and Procedure may not be reliably interpreted without supporting information about the meter's operating state and its configuration that is available from other tables. The following ANSI C12.19 Tables may be of importance:

Table 00, "General Configuration Table",
Table 01, "General Manufacturer Identification Table",
Table 03, "End Device Mode Status Table",
Table 05, "Device Identification Table",
Table 06, "Utility Information Table", [3]
Table 10, "Data Source Dimension Limits Table",
Table 11, "Actual Data Sources Limiting Table",
Table 12, "Units of Measure Entry Table",
Table 13, "Demand Control Table",
Table 14, "Data Control Table",
Table 15, "Constants Table",
Table 16, "Source Definition Table",
Table 17, "Transformer Loss Compensation Table",
Table 52, "Clock Table"
Table 53, "Time Offset Table",
Table 55, "Clock State Table"
Table 78, "End Device Program State Table"

---

[1] This Table or Procedure is new in ANSI C12.19-2008 is unlikely to be present in any technology that predates 2009. Procedure 30 should be used, when available, to snap-shot (freeze) the load profile data values.

[2] The "high-precision" alternative to SP10.

[3] This is an alternative place holder to ST5 IDENTIFICATION Element. When it is desired to obtain the meter's IDENTIFICATION Element then use ST5. When ST5 is not available in the meter and ST6 is available then use ST6.DEVICE_ID, since both represent the same thing.

Tables 10 through 17 may need to be interrogated only when it is desired to report load profile data values in terms of their fundamental engineering units rather than raw Table values.

ANSI C12.19-2008 provides for great flexibility in End Device programming, which leads at times to an increased complexity for the back-end system (an ANSI C12.19-2008 Standard design objective). The standard's flexibility is also exploited by some manufacturers through the use of Manufacturer defined load profile Tables (a non-compliant solution).

4 Location of Load-profile Descriptors in a Compliant ANSI C12.19 Meter

Load-profile dimensions, block boundary and status is derived from ANSI C12.19 ST61 (ACT_LP_TBL), ST62 (LP_CTRL_TBL), ST63 (LP_STATUS_TBL) and ST64 (LP_DATA_SET1_TBL) as shown in **Table 1: Status information available from ANSI C12.19 Table 61, "Actual Load Profile Limiting Table"**, **Table 2: Status information available from ANSI C12.19 Table 62, "Load Profile Control Table"**, **Table 3: Status information available from ANSI C12.19 Table 63, "Load Profile Status Table"** and **Table 4: Status information available in ANSI C12.19 Table 64, "Load Profile Data Set One Table"**.[4]

[4] The table elements described here assume that the API Load Profile Data Set parameter issued was one (1), thus requesting data to be retrieved from ST64. When other data set selection parameter values are supplied (e.g. set 2, 3 or 4) they represent selection of ST65, ST66 or ST67. Therefore, the correct element name should be selected so that it corresponds to the indicated data set. For example when data set selection =2 then LP_STATUS_TBL.LP_STATUS_SET2.LP_SET_STATUS_FLAGS should be used and load profile data retrieved will be from ST65. The same logic applies to all load profile element selector that contain the term "SETn", where n = 1, 2, 3 or 4.

These are graphically illustrated below:

Figure 1: The structure of Standard Table 64, Load Profile Data Set One.

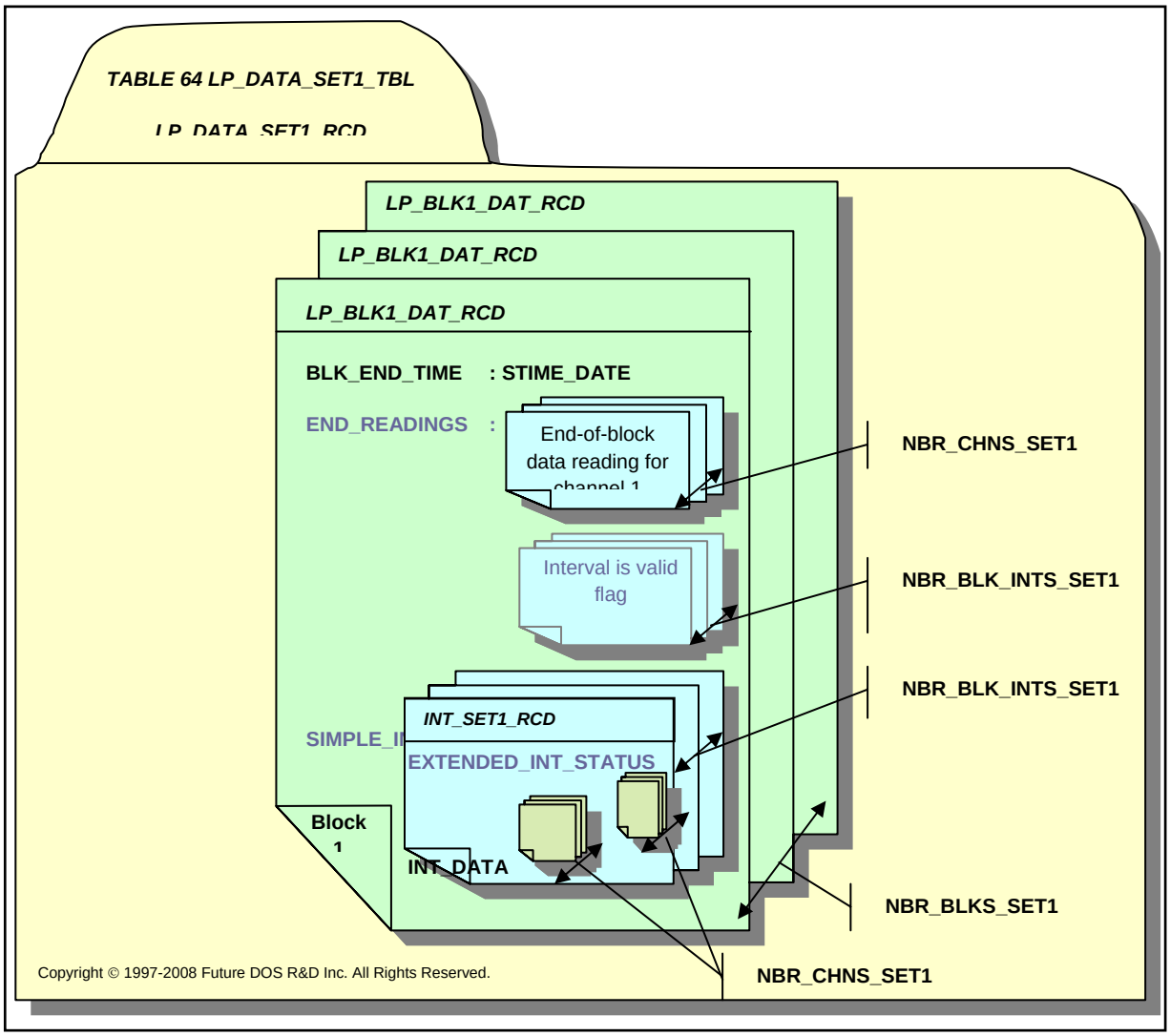
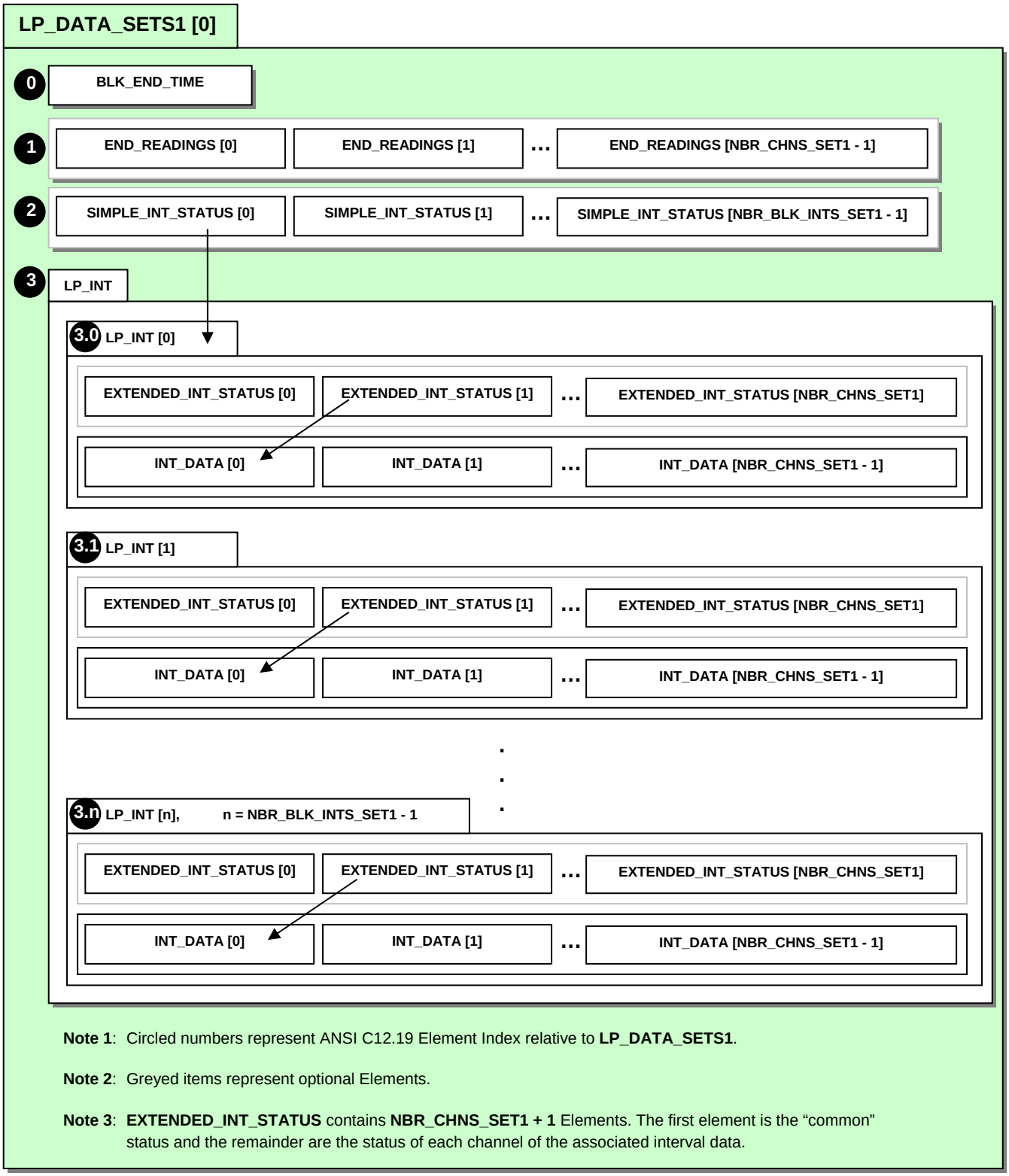


Figure 2: The structure of a single load profile block in Table 64, Load Profile Data Set One.



Note 1: Circled numbers represent ANSI C12.19 Element Index relative to **LP_DATA_SETS1**.

Note 2: Greyed items represent optional Elements.

Note 3: **EXTENDED_INT_STATUS** contains **NBR_CHNS_SET1 + 1** Elements. The first element is the "common" status and the remainder are the status of each channel of the associated interval data.

Table 1: Status information available from ANSI C12.19 Table 61, "Actual Load Profile Limiting Table"

Number of blocks in load profiler	=	ACT_LP_TBL.NBR_BLKS_SET1
Number of intervals per block	=	ACT_LP_TBL.NBR_BLK_INTS_SET1
Number of channels	=	ACT_LP_TBL.NBR_CHNS_SET1
Interval length in minutes	=	ACT_LP_TBL.MAX_INT_TIME_SET1

Table 1 above does not contain a definitive list of all status Elements. Other status Elements are defined in ST61. Elements such as LP_SET1_INHIBIT_OVF_FLAG, BLK_END_READ_FLAG, SCALAR_DIVISOR_FLAG_SET$_N$, *etc*., are useful in testing capabilities, data availability and the computation of offsets into block data. As such they are not covered in this section. They will be covered in the context where needed.

Table 2: Status information available from ANSI C12.19 Table 62, "Load Profile Control Table"

Interval data source for each channel	=	LP_CTRL_TBL.LP_SEL_SET1[ACT_LP_TBL.NBR_CHNS_SET1] A collection of {CHNL_FLAG, LP_SOURCE_SELECT, END_BLK_RDG_SOURCE_SELECT} Elements used to determine whether ST64 contains block end readings (when CHNL_FLAG.END_RDG_FLAG = TRUE), the channel's input source for intervals data and channel's source for block end reading.
Data format of all interval data in ST64	=	LP_CTRL_TBL.INT_FMT_CDE1 This element describes the data type used in each interval. All intervals in all channels in all blocks that are common to one profiler (*e.g.* ST64) share the same data type. Given that most profilers use integral data types when storing interval data, it may be necessary to scale these values to proper floating point value in the TMS. The scaling operation is determined by the appropriate SCALARS_SET1 and DIVISOR_SET1 on a per-channel basis.

Scalars applied to interval data on a per-channel basis[5]	=	LP_CTRL_TBL.SCALARS_SET1[ACT_LP_TBL.NBR_CHNS_SET1] In order to scale the interval data values, the TMS software needs to divide the interval data by SCALARS_SET1[n], where n is the channel number n = {0, 1, ... , NBR_CHNS_SET1-1}.
Divisors applied to interval data on a per-channel basis	=	LP_CTRL_TBL.DIVISOR_SET1 [ACT_LP_TBL.NBR_CHNS_SET1] In order to scale the interval data values, the TMS software needs to multiply the interval data by DIVISOR_SET1[n], where n is the channel number n = {0, 1, ... , NBR_CHNS_SET1-1}.

[5] This conversion does not yield engineering units. If it is desired to produce engineering units then it is necessary to use the correct conversions per INT_SOURCE_SELECT found in ST62.

Table 3: Status information available from ANSI C12.19 Table 63, "Load Profile Status Table"

Load profile status	=	LP_STATUS_TBL.LP_STATUS_SET1.LP_SET_STATUS_FLAGS Information about the nature and operation mode of the End Device load-profile 1 (ST64) as follows: Bit 0: BLOCK_ORDER, 0 = Blocks are ordered in ascending order (block N is older than block N+1), 1 = descending order. Bit 1: OVERFLOW_FLAG, TRUE = block data overflow has occurred. Bit 2: LIST_TYPE, 0 = Load-profile blocks are stored in a FIFO (First In First Out), 1 = Circular list. Bit 3: BLOCK_INHIBIT_OVERFLOW_FLAG , FALSE = End Device is not inhibiting Load-profile overflow (roll over), TRUE = block roll-over is inhibited. Bit 4: INTERVAL_ORDER, 0 = Intervals in each block are in ascending order (N is older than N+1), 1 = descending order. Bit 5: ACTIVE_MODE_FLAG, FALSE = load-profiler is disabled, TRUE = load-profiler is enabled and running. Bit 6: TEST_MODE, 0 = End Device is operating in Metering Mode data, 1 = operating in test mode.
Number of valid blocks	=	LP_STATUS_TBL.LP_STATUS_SET1.NBR_VALID_BLOCKS The number of valid blocks in load-profile 1 (ST64). The value zero implies that there are no data blocks available in ST64. A load-profile data block is considered valid when it contains at least one complete interval.
Last block element	=	LP_STATUS_TBL.LP_STATUS_SET1.LAST_BLOCK_ELEMENT The index of the newest valid data block in ST64. The value of LAST_BLOCK_ELEMENT is valid only when NBR_VALID_BLOCKS is greater than zero.

Last block sequence number.	=	LP_STATUS_TBL.LP_STATUS_SET1.LAST_BLOCK_SEQ_NBR Sequence number of the newest valid data block (per LAST_BLOCK_ELEMENT). This sequence number corresponds to the first valid interval entry in the block.
Number or unread blocks	=	LP_STATUS_TBL. LP_STATUS_SET1.NBR_UNREAD_BLOCKS Number of Load-profile blocks that have not been read. This value may be updated through the invocation of C12.19 Procedure 4 “Reset List Pointers” or Procedure 5, “Update Last Read Entry”.[6]
Valid intervals in active block	=	LP_STATUS_TBL. LP_STATUS_SET1.NBR_VALID_INT Number of valid intervals stored in the newest Load-profile block whose array index is LAST_BLOCK_ELEMENT.

[6] Many implementations do not update the number of unread blocks to allow the utility to manage this quantity manually. This does not pose any problem as long as LP_STATUS_TBL.LP_STATUS_SET1.LP_SET_STATUS_FLAGS. BLOCK_INHIBIT_OVERFLOW_FLAG is set to FALSE.

Table 4: Status information available in ANSI C12.19 Table 64, "Load Profile Data Set One Table"

End block time	=	LP_DATA_SET1_TBL.LP_DATA_SETS1[n].BLK_END_TIME This entry contains the ending date and time of the last interval data recorded in this data block. This is the end-of-interval value (for example, if a one-hour interval started at 10:00 and terminated prematurely at 10:15, the BLK_END_TIME shall indicate 11:00). Only when the block is closed, regardless of reason, the BLK_END_TIME is the block's actual starting time + NBR_BLK_INTS_SET1 * MAX_INT_TIME_SET1. This way it is possible to compute the closed block's boundary start-time, being (BLK_END_TIME - NBR_BLK_INTS_SET1 * MAX_INT_TIME_SET1).[7] Otherwise we need to use the NBR_VALID_INT to determine start-time of the block (BLK_END_TIME - NBR_VALID_INT * MAX_INT_TIME_SET1).
Simple interval status	=	LP_DATA_SET1_TBL.LP_DATA_SETS1 [n]. SIMPLE_INT_STATUS [i] The simple interval status is a collection of single status bits, one per interval, that simply indicate whether the corresponding interval is valid (TRUE) or invalid (FALSE). This element is present only when ACT_LP_TBL.SIMPLE_INT_STATUS_FLAG = TRUE.

[7] When a block is closed it is necessary to compute the block's start-time and block's end-time. If the block's start-time is less then the API requested interval-time and the block's end-time is greater or equal to the TMS requested interval-time then the block may contain the desired interval data. In order to be certain, it is necessary to read (and compute) the start-time of the next block to ensure that it is later than the end-time of the current block. Otherwise it is necessary to check the status of all intervals (simple or extended) of the current block to establish whether the block actually contains the desired interval or not (a possible reason for not having the desired interval data is early closure of the block due to power failure or time adjustments).

Extended interval status	=	LP_DATA_SET1_TBL.LP_DATA_SETS1 [n]. LP_INT[i]. EXTENDED_INT_STATUS[chNbr + 1] These are additional status flags that provide interval common status and per-channel status information. The presence of this field is contingent on the Element ACT_LP_TBL.EXTENDED_INT_STATUS_FLAG = TRUE. The EXTENDED_INT_STATUS is an array of nibbles (4-bits), where the first nibble represents the interval common status and the remainder is a collection of one interval status per channel. The first byte holds in its MSN (most significant nibble, bits 4..7) the interval's common status and interval's 1st channel status in the LSN (Least Significant Nibble, bits 0..3). The rest of the bytes (at indices 1, 2, …, ACT_LP_TBL.NBR_CHNS_SET1/2) contain channels 2nd, 4th, 6th, …, status values in the MSN and channels 3rd, 5th, 7th, …, in the LSN.
Interval data for channel chNbr	=	LP_DATA_SET1_TBL.LP_DATA_SETS1 [n]. LP_INT[i]. INT_DATA [chNbr] The interval data value for the channel selected. The data is encoded according to the value indicated by LP_CTRL_TBL.INT_FMT_CDE1 as follows:

INT_FMT_CDE1	Data Type
1	UINT8
2	UINT16
4	UINT32
8	INT8
16	INT16
32	INT32
64	NI_FMAT1
128	NI_FMAT2