NOAH

Storing Audiological Measurements

Loudness Scaling Standard

DataFmtCodeStd 110
Version 1.1

HIMSA II K/S

Preface

This document describes the Extended Loudness Scaling standard with DataFmtCodeStd 110. The coexistence of this Extended version and the DataFmtCodeStd 100 is described at the end of this preface.

The Hearing Instrument Manufacturers' Software Association A/S (HIMSA A/S) was founded at the beginning of 1993 by a group of hearing instrument manufacturers. It has been HIMSA A/S's mission to develop and market the NOAH software, and to make it a de facto standard for integrated hearing care software within the entire hearing industry.

The NOAH Fitting Framework is a software application that enables fitting and measurement software to share data on a common platform (NOAH). The fitting and measurement applications are provided by manufacturers who have signed a know-how licence agreement with HIMSA and thereby obtained the right to distribute the NOAH software, and to develop NOAH-compatible software applications, also referred to as modules.

Data format standards are a natural prerequisite for the ability to share data. Therefore, in co-operation with its licensees, HIMSA has prepared data format standards for audiograms, REM/HIT, Loudness Scaling, Impedance, Oto Acoustic Emissions and Evoked Response Audiometry measurement types.

The documentation for these standards is available in so-called header files. These files are part of the 'Software Development Kit', which HIMSA automatically distributes to its licensees.

Unfortunately, it is our experience that the header files are too easily misinterpreted. It has thus been decided that HIMSA must prepare a comprehensive standard document for each of the aforementioned measurement types. These documents will provide a detailed presentation of the data structure of the measurement formats as well as describe the application of the various types of, e.g. 'specific audiograms'.

The various data standards are subject to revision twice a year by a committee consisting of manufacturers of Audiological Measurement Equipment (AEMs). Based on input prepared by HIMSA, it will be the responsibility of this committee to approve both new standard documents and updates of existing standards. The AEM Committee will meet on the Saturday following the end of the UHA Convention in Germany, i.e. in October, and on the Saturday following the end of the AAA Convention in the US, i.e. in April.

HIMSA also invites non-licensees to take part in the process of preparing and maintaining measurement data standards.

Figure 1 presents the principles by which NOAH administrates the measurement formats. Each block of stored data must be equipped with a header. This header uniquely identifies, e.g. the manufacturer who created the measurement, the type of measurement data contained in the data block and the measurement data format's revision number.

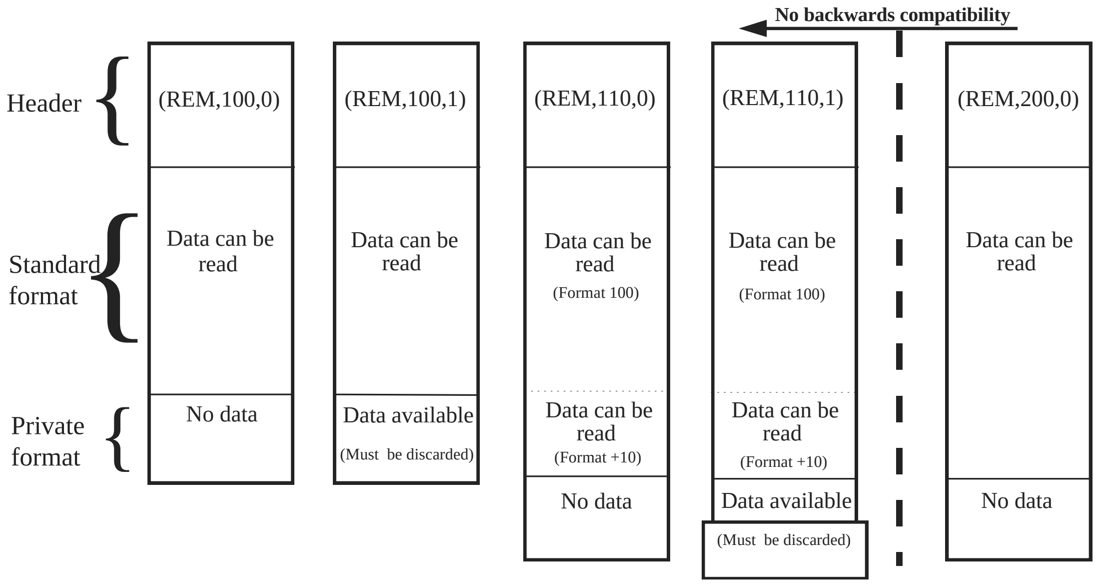


Figure 1: The handling of measurement data by NOAH

The basic revision number for a data format is 100. A data format with the revision number 110 is a direct extension of the basic 100 format. It is therefore possible for a revision 100 module to still read and understand a data block generated by a revision 110 module as it will simply discard the '+10' extension. A data format with the version number 200 would constitute a totally new revision thus making it impossible for revision 1xx modules to read revision 2xx data formats.

It is possible for a manufacturer to add non-standardised measurements to the public data block.

Document History

ver.	0.1	97-09-24	First draft
ver.	0.9	97-09-24	Second draft of the Extended LSdef.h with DataFmtCodeStd 110. (Then called Rexton Danplex Extended version)
ver.	0.95	97-11-30	Third draft. This draft was prepared after the AEM Meeting in Germany in October 1997.
ver.	1.00	97-12-05	First approved version.
ver.	1.1	09-05-01	Note on extra byte for alignment – Section 2.1.2.

1 Introduction

1.1 A few words about programming with LSDEF.H

This document intends to explain the use of the NOAH ver 2.0 standard for storing Loudness Scaling Measurements according to the Extended Loudness Scaling standard with DataFmtCodeStd 110, the LSDEF.H header file. This header file written in the programming language "C" defines an LSSession structure (An LSSession consists of 2 LSMeasurements: HL calibrated and SPL calibrated, plus an extension, consisting of some additional data). The LSMeasurement structure contains the result of a Loudness Scaling Measurement done at up to 11 different frequencies:

The LSMeasurement structure Measuring Conditions	
Parameter	*Values*
Signal Output	❑ Phone ❑ Insertphone ❑ Soundfield (free field)
Signal Type	❑ Pure Tone ❑ Warble Tone ❑ Narrow Band Noise
dB Weighting	❑ HL ❑ SPL
Loudness Scaling Data:	
Component	*Explanation*
Frequency	Measuring Frequency No. [0..10]
Scaling Result	Three points on a Loudness / level curve allowing one knee point
Normal Curve	Three points on a Loudness / level curve allowing one knee point
Measured Raw Data	Up to 50 observed Curve Points on a Loudness / Level curve
	(… continue with the next frequencies, up to 11 structures can be saved)

The aim of this document is to explain the correct use of the LSMeasurement structure. This is done by reading the header file LSDEF.H, Extended Loudness Scaling standard with DataFmtCodeStd 110, “upside down” starting with the “outer” definition of LSSession. Then comes the type definition of the measurement, the Measuring Conditions and associated curve points, ending with the definition of all “inner” types, all defined as integers.

This document is written as the third part of documentation for software developers of the NOAH Framework Programming Interface:

NOAH: Storing Audiological Measurements Document series		
Document Title	*Header File explained*	*Status*
Audiogram Standard	formats\audiogrm\AUDdef.h	Ver. 1.0 of November 1997 available.
REM/HIT Standard	formats\remhit\REMHIT.h	Ver. 1.0 of July 1997 is available.
Loudness Scaling Standard DatasFmtCodeStd 100	formats\loudness\LSdef.h	Approved in October 1997
Extended Loudness Scaling Standard DatasFmtCodeStd 110	formats\loudness\Extended LSdef.h	Approved in October 1997. (This document)
Impedance Measurement Standard	formats\impedan\IMPdef.h	Approved in October 1997.
Oto Acoustic Emissions Standard	formats\oae\OAEdef.h	Planned
Evoked Response Audiometry Standard	formats\era\ERAdef.h	Planned

Data can be exchanged across these interfaces among the NOAH modules. In this way data can be shared among different Hearing Instrument- and Audiological Equipment-manufacturers.

This document describes the Loudness Scaling measurement format and can be read independently of other NOAH documentation. It is intended as a starting point for interested, prospective licensees.

1.2 Contents

1.3 References

[Framework] NOAH Framework ver. 0.85. System Architecture Specification. Pallas Informatik A/S.

[HOCA-4] Handbook of Clinical Audiology, edited by Jack Katz.
Williams & Wilkins, 1994, 4. Edition.

[IEC 645-1] Audiometers - Part 1: Pure-tone Audiometers, November 1993. From 1997 renumbered to IEC 60645-1.

2 The NOAH standard for Loudness Scaling

2.1 Data Structure

In order to describe the data structure as it is defined in Extended LSdef.H with DataFmtCodeStd 110, an extended version of the language Abstract Syntax Notation No. 1 (ASN.1) is used [1]. This is done for the following reasons:

1. Explanation of the data structure in Extended LSdef.H with DataFmtCodeStd 110 starting with “the basic LSSession format" and continuing with the structures defining it. From this “outer”, all-embracing type all constituent types are defined as we go by. (In effect, the header file ‘upside down’). The definition in ASN.1 ends in the case of this header file by defining all the fundamental types as integers.

2. ASN.1 contains a few useful distinctions, used in this chapter to explain important places in Extended LSdef.H with DataFmtCodeStd 110 , where the order of variables matters, and where it does not. Note, that variables are called ‘components’ when in an outer structure:

SEQUENCE	Ordered collection of component types.
SEQUENCE OF	Ordered collection of variables of the component type.
SET	Unordered collection of component types, all distinct.
SET OF	Unordered collection of variables of the component type

[1] ASN.1 is defined by ISO and the International Telecommunication Union (ITU) (see ISO 8824) with a set of so-called Basic Encoding Rules which we shall NOT use here. Instead, a “Direct Encoding Rule” can be formulated: Data are encoded exactly as they are shown, down to the definition of the INTEGER as consisting of two byte, low-order transmitted first (placed at lower address).

2.1.1 The Integer type used in Extended LSdef.H

minInt	-32768 #8000 hex	Lowest negative value represented in two byte using standard “2’s complement” representation. According to [Framework] , this value is **illegal** for the integer types defined in Extended LSdef.H with DataFmtCodeStd 110.
undefInt	-32767 #8001 hex	Used to indicate that the value is **undefined,** a value which is assigned to the constant undefInt. Ref. [Framework]
minParmInt	-32766 #8002 hex	Lowest negative value **legal** in parameters defined as integer types in Extended LSdef.H according to [Framework].
Unknown	0 #0000 hex	*In Parameters:* The parameter is **defined,** however to an **unknown** value**.** *In Curve points:* Use logic here! For the types TdB10, TPct100, the value 0 is of course **defined** and **valid,** however for the THertz type, the value means **undefined**.
NoParam	1 #0001 hex	*In Parameters:* The parameter is in other header files **defined** Not Used (channel, parameter…), see AUDdef.h or REMHIT.h. However, in Extended LSdef.H according to the explanation for the different types: TLSSignalOutput, TLSSignalType, TLSdB the value 1 is defined and in use as a legal parameter.
MaxInt	32767 #7FFF hex	Highest positive value. Ref. [Framework].

2.1.2 Definition of the Extended LSDef Standard

NOTE: When adding a rule name, a single byte needs to be used for alignment. For example, if a field is defined to have 51 characters, where each character is 1 byte, then an extra byte needs to be added for alignment purposes. This is an empty byte, set aside to serve as a placeholder.

LoudnessScaling DEFINITIONS ::=
--
-- DataFmtCodeStd 110

BEGIN

IMPORTS ALL FROM Noahdef -- noahdef.h

2.1.2.1 TLSSession

A Loudness Scaling Session in the non-extended version

TLSSession Not used	The Extended version of LSDef extends this definiton by adding a new data structure to the two Loudness Scaling Measurements, refer para. 2.1.3.2: TLSSession110 on page 14. This non-extended version of the structure consists of two Loudness Scaling Measurement Structures: One for saving results measured in dB HL and one for results measured dB SPL. TLSSession was defined for use in LSdef.h DataFmtCodeStd 100.

TLSSession ::= SET OF 2 TLSMeasurement

2.1.2.2 TLSMeasurement

Loudness Scaling Measurement structure	
TLSMeasurement	The result of a Loudness Scaling measurement can be saved for up to 11 frequencies. Use the component named frequency to indicate the actual number of used Data structures according to the definition below. The use of frequency = UndefInt to indicate endCurve is explained in chapter 2.1.4: Reading and writing Loudness Scaling curve points.

```
TLSMeasurement ::= {
    signalOutput        TLSSignalOutput,
    signal              TLSSignalType,
    dB                  TLSdB,
    data                SET OF 11{
                            frequency       THertz,
                            scalingResult   TScalingResult,
                            norm            TNorm,
                            rawData         TRawData
                        }
}
```

2.1.2.3 RawData

RawData	
rawData	The maximum number of Loudness Scaling Measuring Points that can be saved for each of the 11 Measuring Frequencies. By this value definition, up to 50 Measuring Points can be saved per Measuring Frequency.
TRawdata	SET OF rawData (i.e. 50) Loudness Scaling Measuring Points.
Measuring Method	The Person under Test will be asked to scale the loudness of sample sounds presented in at variety of orders. A combination of randomization and "window function" is often used. From an estimated MCL value, a window from e.g. 40 to 60 dB limits the choice of legal start values. Inside this window a random value for presentation loudness is chosen. The window is then moved either towards higher allowed values or towards lower allowed values. The result is a swarm of curvepoints around the average Loudness Scaling curve for the Person under Test.

TRawData ::= SET OF rawData TLSPoint -- The number 50 defined by the identifier rawData

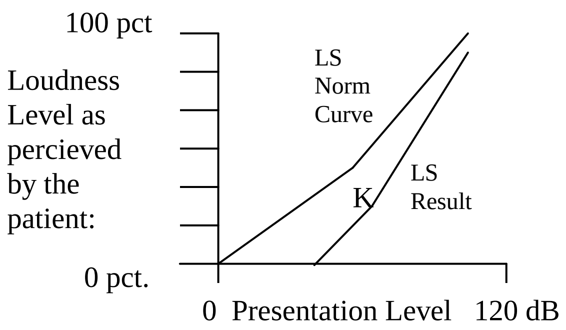
rawData INTEGER ::= 50

2.1.2.4 Loudness Scaling Curves

Loudness Scaling Curves

The Loudness Scaling for a hearing-impaired person is pictured against a curve for normal hearing persons, the so-called LS Norm graph. The three curve points in this graph allow for one knee point.

The measuring conditions influence the norm curve, and this fact is the reason for saving it together with the actual Loudness Scaling data. The Loudness is expressed as a percentage indicated on a patient-answering device. This device can have many possible arrangements of buttons but the 0-6 level type is generally accepted:



The TNorm curve contains the three curvepoints in the LS Norm Curve

The TScalingResult contains three calculated averages of the RawData swarm of curvepoints.

Note the kneepoints (K) in the middle indicating the middle curve points.

LS	*Client's Subjective Rating:*	*Subjective percentage rating*
6	Uncomfortably loud (UCL) (Too Loud)	100
5	Very loud	90
4	Loud	70
3	Most Comfortable Level (MCL) (Medium/ Comfortable)	50
2	Low (Soft)	30
1	Very low (Very Soft)	10
0	Not Heard	0

TNorm ::= SET OF 3 TLSPoint

TScalingResult ::= SET OF 3 TLSPoint

2.1.2.5 TLSPoint

Loudness Scaling point

	Generic point definition in the Loudness Scaling Plane: [Presentation level = x, Loudness Scaling = y] , as shown above. A point is invalid if Level == UndefInt, see chapter 2.1.4 : Reading and writing Loudness Scaling curve points.

```
TLSPoint ::= SEQUENCE{
    level               TdB10,              -- Presentation Level
    loudness            TPct100             -- Perceived Loudness
}
```

```
undefInt INTEGER ::=        -32767                    -- Ref. [Framework]
```

2.1.2.6 THertz

```
THertz ::=                  INTEGER                   -- Frequency in Hz
```

2.1.2.7 TdB10

```
TdB10 ::=                   INTEGER                   -- Sound Pressure in centiBell
```

2.1.2.8 TPct100

```
TPct100 ::=                 INTEGER                   -- Hundredths of Percent
```

2.1.2.9 TLSSignalOutput

```
TLSSignalOutput ::= INTEGER                           -- Signal output
{
   lsso_Phone               1,                        -- Transducer is AC Phone
   lsso_InsertPhone         2,                        -- Transducer is Insert Phone
   lsso_SoundField          3,                        -- Transducer is Free Field
-- additional define by Extension:
   lsso_BoneConduction      4                         -- Transducer is Bone Conductor
   lsso_user_1              5                         -- User defined 1
   lsso_user_2              6                         -- User defined 2
   lsso_user_3              7                         -- User defined 3
}
```

2.1.2.10 TLSSignalType

```
TLSSignalType ::= INTEGER                             -- Signal type
{
   lsst_Tone                1,                        -- Pure Tone
   lsst_Warble              2,                        -- Warble Tone
   lsst_NBN                 3,                        -- Narrow Band Noise (not specified)
-- additional defines by Extension:
   lsst_NBN13               4,                        -- 1/3 Octave Narrow Band Noise
   lsst_NBN12               5,                        -- 1/2 Octave Narrow Band Noise
   lsst_SpeechNoise         6,                        -- Speech Noise (IEC _____)
   lsst_WhiteNoise          7,                        -- White Noise (IEC 645-1)
   lsst_Naturesound         8                         -- Nature Sound (IEC _____)
   lsst_SpeechSignals       9,                        -- Speech Signals (IEC_____)
   lsst_user_1              10,                       -- User defined 1
   lsst_user_2              11,                       -- User defined 2
   lsst_user_3              12                        -- User defined 3
}
```

2.1.2.11 TLSdB

```
TLSdB ::= INTEGER                                     -- dB weighting used
{
   lsdbHL                   1,                        -- Hearing Level
   lsdbSPL                  2                         -- Sound Pressure Level
}
END
```

2.1.3 Extension to LSDEF.H with DataFmtCodeStd 110

LSExtendedMeasCond DEFINITIONS ::=

BEGIN
-- Loudness Scaling Extended Version
-- Uses DataFmtCodeStd 110

2.1.3.1 TLSExtendedMeasCond

Loudness Scaling Data: Additional Data by Extension	
TLSExtended MeasCond	This Sequence summarizes the extended Loudness Scaling Measuring Conditions that are added to the DataFmtCodeStd 100 data format to form the DataFmtCodeStd 110 format.

```
TLSExtendedMeasCond ::= SEQUENCE
{
    lsAided             TLSAided,             -- Ref. 2.1.3.4
    lsAnswerCategory    TLSAnswerCategory,    -- Ref. 2.1.3.5
    lsMaxAnswer         TLSMaxAnswer,         -- Ref. 2.1.3.6
    lsMethod            TLSMethod,            -- Ref. 2.1.3.7
    lsBinaural          TLSBinaural           -- Ref. 2.1.3.8
}
```

2.1.3.2 TLSSession110

A complete LS Session - Extended	
TLSSession110	The outer structure for Loudness Scaling DataFmtCodeStd 110 , Loudness Scaling Session 110

```
TLSSession110 ::= SEQUENCE
{
    lsMeasurement        SET OF 2 TLSMeasurement,  -- As DataFmtCodeStd 100
    lsExtendedMeasCond   TLSExtendedMeasCond       -- Extended Measuring Conditions
}
```

2.1.3.3 dfcs_110

A Format Code for the extended version	
	This is in ASN.1 a value definition for the Integer type.

```
dfcs_110 INTEGER ::= 110 -- Value for Extended version DataFmtCodeStd
```

2.1.3.4 TLSAided

Loudness Scaling while using hearing aids	
TLSAided	The Loudness Scaling Aided expresses whether the client was using a hearing instrument during the Loudness Scaling:
Not Aided:	The client was not using hearing aids
Left aided:	Only the Left ear was aided.
Right aided:	Only the Right ear was aided.
Both Aided:	Both Left and Right Ear was aided.

```
TLSAided ::= INTEGER
{
    lsa_NotAided        1,      -- Not Aided
    lsa_LeftAided       2,      -- Left Ear aided
    lsa_RightAided      3,      -- Right Ear aided
    lsa_BothAided       4       -- Left and Right Ear aided
}
```

2.1.3.5 TLSAnswerCategory

Answer Categories, i.e. Loudness Weightings	
	Count of the different Loudness Weightings available, i.e. answer categories, choices on the patient answering device, including the "not heard" or "0" option. Range [0..99]. In para. 2.1.2.4: Loudness Scaling Curves is used 7 answer categories incl. the "0" option. If undefInt is inserted, the number of categories is not available. This number can instead be extracted from the raw data points.

TLSAnswerCategory ::= INTEGER

2.1.3.6 TLSMaxAnswer

Maximum Number of answers (Raw data points)	
TLSMaxAnswer	The actual number of filled Raw data points may be lower than indicated with this variable. If undefInt is inserted, no Maximum Number of answers is available. The range for MaxAnswer: [1..50] - Up to 50 raw data points can be saved in the LSdef.h Extended data structure.

TLSMaxAnswer ::= INTEGER

2.1.3.7 TLSMethod

Identification Number and Name for the Loudness Scaling Method

Id	Loudness Scaling Method Identification Number.If this Integer is given the value undefInt it means that no LSMethod.id is available
Name	Loudness Scaling Method Name. A string of 32 ASCII characters that describes the Loudness Scaling Method identified with LSMethod.Id. If this string is filled with ASCII " " (spaces, h20) it means that no valid LS Method.Name is available.

```
TLSMethod ::= SEQUENCE
{
   Id        INTEGER {
                lsid_Unknown      0,        -- The LS Method is unknown
                lsid_Not_Used     1,        -- None of the listed LS methods were used
                lsid_IHAFF_1      2,        -- Independent Hearing Aid Fitting Forum Method 1
                lsid_IHAFF_2      3,        -- Independent Hearing Aid Fitting Forum Method 2
                lsid_LGOB_1       4,        -- Loudness Growth in Half-Octave Bands Method 1
                lsid_LGOB_2       5,        -- Loudness Growth in Half-Octave Bands Method 2
                lsid_JRP          6,        -- German Joint Research Project
                lsid_RELM         7,        -- Real Ear Loudness Mapping
                lsid_WHF          8,        -- Würzburger Hörfeld Method
                lsid_MD_1         9,        -- Madsen/Danavox Method 1
                lsid_MD_2         10,       -- Madsen/Danavox Method 2
                lsid_USER1        11,       -- For future use
                lsid_USER2        12,       -- For future use
                lsid_USER3        13,       -- For future use
                lsid_USER4        14        -- For future use
             },
   Name      CHARACTER STRING (Length 32)
}
```

2.1.3.8 TLSBinaural

Test Type	
TLSBinaural	The Binaural variable indicates if stimulus was applied to both ears during the loudness scaling. The client answers would in this case be based on a binaural loudness scaling measurement. In the binaural case, the measuring results have to be saved in two TLSSessions, one for each ear, the reason being that the patients' answers would be related to a Loudness Scaling related on both ears. The two TLSSessions should be identical, and it is thus only necessary to read one of the TLSSessions.

```
TLSBinaural ::=
{
   lsb_Monaural          0,                    -- Stimulus to one ear
   lsb_Binaural          1                     --Stimulus to both ears
}
```

END -- Of module LSExtendedMeasCond

2.1.4 Reading and writing Loudness Scaling curve points

Extended LSDEF.H defines the following curves:

Loudness Scaling Curves / Measured swarm of curvepoints (raw data) Curve / Measured Data points		
Curve Identifier	*Curve Type*	*Type of the curve points*
scalingResult	TScalingResult	TLSPoint
norm	TNorm	TLSPoint
rawData	TRawData	TLSPoint

Added to this list of curves comes the structure called data: For a maximum of 11 frequencies, the following can be stored:
- One ScalingResult curve (3 TLSPoints)
- One Norm curve (3 TLSPoints)
- Up to 50 raw data points in the [Presentation Level = x, Loudness Scaling = y] plane.

In this paragraph, the reading and writing of Loudness Scaling curves will be explained. In the following paragraph, the reading and writing of the structure data is explained.

The reading of curve points in a Loudness Scaling measurement from NOAH ver 2.0 is per definition done in the following way:

The level component is read first. The curve points might be ordered, but since they are defined as a set, they also might be *unordered* with respect to level. Read the curve points while checking level. There is a maximum of three curvepoints per set.

Curve points are read until the namedValue endCurve occurs:

```
-- Do not overlook this end of curve marker !!!

endCurve TLSPoint ::=

{
    undefInt,                -- level= undefInt defines the endCurve
    undefInt                 -- loudness can be undefInt or any other value
```

In general: Curve points with level= undefInt (-32 767) are discarded.

When writing curve points, place them sorted with level in ascending order ending with endCurve. Unused curvepoints are filled with endCurve markers (undefInts). This filling is not mandatory but is considerate to fellow programmers.

2.1.5 Reading and writing the structure: data

The reading of the structure called data in a Loudness Scaling measurement from NOAH ver 2.0 is per definition done in the following way:

The frequency is read first. Data might be ordered, but since they are defined as a set, they also might be *unordered* with respect to frequency. Read the curve points while checking Frequency.

Curve points are read until the namedValue endCurve occurs:

```
-- Do not overlook this end of curve marker !!!

endCurve Data::=


{
    undefInt,          -- frequency= undefInt defines the endCurve
    undefInt           -- scalingResult.level can be undefInt or any other value
    undefInt           -- scalingResult.loudness can be undefInt or any other value
    undefInt           -- norm.level can be undefInt or any other value
    undefInt           -- norm.loudness can be undefInt or any other value
    undefInt           -- rawData can be undefint or any other value
    …                  -- undefInt should be inserted for all rawData
}
```

After endCurve, Curve points with frequency= 0 or frequency = undefInt (-32 767) are discarded. Curve points with such unreasonable frequency should be discarded at any time during the reading.

When writing curve points, place them sorted with the frequency in ascending order ending with endCurve and fill the rest of the array with endCurve markers (undefInts). This filling is not mandatory but is considerate to fellow programmers.

2.1.6 Loudness Scaling Measuring Conditions

The measurement conditions for Loudness Scaling consist of the components TLSSignalOutput, TLSSignalType and TLSdB. These types are found in the structured type TLSMeasurement.

It is recommended that when your module defines a complete TLSSession according to this NOAH standard, these three parameters are initialised to undefInt. The actual values are later inserted as selected by the user. The value undefInt is considered the "legal" filling of an empty structure. Empty structures may be saved together with e.g. private dumps.

2.2 Reading and writing Loudness Scaling Measurements

In the previous chapter, the Loudness Scaling data structure was explained. This chapter will give some hints to the actual reading and writing of the structure as defined in the NOAH standard version 2.0.

The basic principle is that a whole structure has to be saved although perhaps only part of the structure is actually used. Unfortunately, this means that maybe only a small fraction of a TLSSession is filled by usable data. The NOAH database caters for this by compressing data before adding it to its database / expanding it before supplying the data to an external software module. The price paid in other words is slowed down communication, the gain is a uniform structure of data.

2.2.1 Reading the Loudness Scaling Measurements

In order to find the measurements that contain useful data when reading a TLSSession structure, your program should read the Measuring Conditions attached to each measurement, i.e. the fields TLSSignalOutput, TLSSignalType and TLSdB.

In this chapter a namedValue[2] called lsInitialCond is introduced. Most of the measurement conditions will be equal to this namedValue: lsInitialCond. Subsequent chapters describe the minimum changes in lsInitialCond that make the measuring conditions valid for each of the measurements that constitute a complete TLSSession.

Note 1: If the Loudness Scaling Measuring Conditions for a measurement are completely identical to lsInitialCond, this means that the associated measurement is empty.

Note 2: The definitions for Integer values written in the beginning of this chapter apply. However, the value zero can be found in empty measurements where the correct value should have been undefInt.

2.2.2 Writing the Loudness Scaling Measurements

When writing a Loudness Scaling Measurement, use the following method:

1) Initialise the two measurements in the structure by setting the Measuring Conditions to the initial conditions lsInitialCond (see below). The codepoints should be initialised with endCurve. Refer to paragraph 2.1.4 Reading and writing Loudness Scaling curve points.

2) Insert the appropriate values in the actual Loudness Scaling Measuring Conditions for the measurements that you want to save.

3) The curvepoints are then inserted. Their insertion follows the directions mentioned in paragraph 2.1.4 Reading and writing Loudness Scaling curve points.

Loudness Scaling Initial Measurement Conditions

Data Type	*Field*	*Value*
TLSSignalOutput	SignalOutput	undefInt
TLSSignalType	Signal	undefInt
TLSdB	dB	undefInt

[2] ASN.1 defines namedValues as structures of an indicated type with a defined content.

Loudness Scaling Initial Measurement Conditions - Extension

Data Type	*Field*	*Value*
TLSAided	lsAided	undefInt
TLSAnswerCategory	lsAnswerCategory	undefInt
TLSMaxAnswer	lsMaxAnswer	undefInt
TLSMethod	id	undefInt
	name	32 ASCII characters " " (spaces = h20)
TLSBinaural	lsBinaural	undefInt

2.2.3 Minimum settings for Loudness Scaling

Minimum settings for Loudness Scaling Measuring Conditions

Component of TLSMeasurement	*Legal Values*	*Explanation*
TLSSignalOutput	Phone, InsertPhone or SoundField (Free field)	Insert the Signal Output device used during the measurement
TLSSignalType	Tone, Warble tone or Narrow Band Noise	Insert the Signal Type used during the measurement
TLSdB	HL or SPL	Insert the standard calibration used for measuring the level of the Signal Type

Minimum settings for Loudness Scaling Measuring Conditions -Extension

Component of TLSData	*Legal Values*	*Explanation*
TLSAided	Not Aided, left Aided, Right Aided or Both aided.	Insert the Hearing aid situation at the measurement. The default is Not aided, if the variable is not filled. (Not mandatory).
TLSAnswerCategory	[1..99]	Insert the number of Answer Categories on the patient answering device used during the measurement. According to the selected Method, a loudness scaling percentage is saved for each raw data point. The number of answer categories can be calculated from this. (Not mandatory).
TLSMaxAnswer	[1..50]	Insert the max. number of answers used per frequency. The number of frequencies can be deducted from the raw data points. (Not mandatory).
TLSMethod	Id: [0..17] refer 2.1.3.7	Insert the correct Id for the LS Method (Mandatory)
	Name:	Insert a text describing the used LS Method . (Not mandatory)
TLSBinaural	Monaural, Binaural.	Insert the situationStimulus Side relative to the actual measurement DataTypeCode. Refer 2.1.3.8. The default value is Monaural. (Not Mandatory).

As the table shows, all three fields in the Loudness Scaling Measuring Conditions are mandatory. If one or more fields are set to undefInt, the whole TLSSession should be discarded. Of the extended Measuring conditions, only the TLSMethod.Id is mandatory.

Appendix A: Vocabulary and Abbreviations

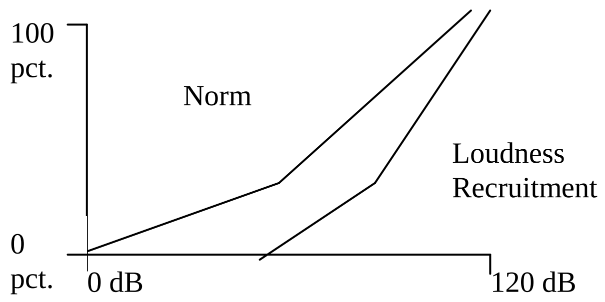
A

Aided or Unaided Loudness Scaling
: The extended version of LSdef.h includes a set of extended Measuring Conditions, TLSExtendedMeasCond. The component lsAided indicates whether hearing aids were used during the measurement. Ref. Para. 2.1.3.4: TLSAided.

ASN.1
: Abstract Syntax Notation No. 1. ITU and OSI defined language for specification of protocol message content.

C

components
: Used in ASN.1 for the fields in a structured type (a “C” structure). The components are given Identifiers, i.e. a field name, in “C” referred to as the member.



curve
: The scaling result graph and the norm graph consists of three curvepoints each. The graphs can be plotted as curves with one so-called knee-point each:

D

data
: The term "data" is used for a set of 11 components of the structure TLSMeasurement. See para. 2.1.2: Definition of the Extended LSDef Standard on page 9.

data structure
: Extended LSDEF.H describes the data structure for interchange of data with the NOAH ver. 2.0 database. The extension uses DataFmtCodeStd 110.

dB — The term "dB" is used for a component of the TLSMeasurement structure. It is part of the Loudness Scaling Measuring Conditions and it can take the values HL or SPL.

DevTypeCode — Defined as Integer in Noahdef.h. Identifies a particular device or instrument type to a NOAH module. Defined individually by NOAH modules. Ref. [Framework].

dfcs_110 — In this extension of the Loudness Scaling standard the Data Format Code Standard is set to the integer 110.

E

endCurve — The set of curve points in a TLSSession structure is not necessarily filled with data. It is recommended to save an endCurve marker after the curve points with actual data. The unused curve points can be endCurve or null-filled. Refer 2.1.4 Reading and writing Loudness Scaling curve points.

F

Frequency — Component of the structure Data. Up to 50 TLSPoints can be recorded per Frequency. Measurements can be saved for up to 11 Frequencies.

I

Id — The Identification Number of the Loudness Scaling Method. Refer 2.1.3.7: TLSMethod on page 16.

Identification Number — See Id above.

L

Level — A component of the structure TLSPoint of type TdB10 (dB x 10 or centiBel).

Loudness — A component of the structure TLSPoint of type TPct100. The loudness is represented on a Percentage Scale from 0 to 100 representing the Threshold of Hearing (0) and the

	Uncomfortably Loud (UCL) level (100).
Loudness Scaling point See TLSPoint	For up to 11 frequencies, a ScalingResult can be recorded and saved together with a so-called Norm curve. Both the ScalingResult and the Norm curve consist of Loudness Scaling Points, i.e. a structure containing a Level in dB plus a Loudness Scaling in pct.

Loudness Answer Categories	Count of the different Loudness Weightings available, i.e. answer categories, choices on the patient answering device, including the "not heard" or "0" option. Range [0..99]. In para. 2.1.2.4: Loudness Scaling Curves is used 7 answer categories incl. the "0" option. If undefInt is inserted, the number of categories is not available. This number can instead be extracted from the raw data points.
lsAnswerCategory	Count of the different Loudness Weightings available, i.e. answer categories, choices on the patient answering device, including the "not heard" or "0" option. Range [0..99]. In para. 2.1.2.4: Loudness Scaling Curves is used 7 answer categories incl. the "0" option. If undefInt is inserted, the number of categories is not available. This number can instead be extracted from the raw data pointsRef. 2.1.3.5: TLSAnswerCategory on page 15.
lsb_Binaural see TLSBinaural	The loudness scaling is covering both ears.
lsb_Monaural see TLSBinaural	The loudness scaling is covering one ear.
lsBinaural	This variable is referenced in the structure TLSExtendedMeasCond, reference para. 2.1.3.1 on page 14. For an explanation of its use, see Para. 2.1.3.8: TLSBinaural on page 17.
lsdbHL See TLSdB	The Loudness Scaling Measurement can be saved with measurements represented in Hearing Level (HL) or it can be saved in Sound Pressure Levels (SPL). The structure TLSSession leaves room for saving both representations. Recommended is HL at index 0, SPL at index 1.
lsdbSPL. See TLSdB	(See explanation for lsdbHL).
LSExtendedMeasCond	Loudness Scaling Extended Measuring Conditions. Refer 2.1.3.1: TLSExtendedMeasCond on page 14.
lsa_BothAided see TLSAided	Both ears were aided during the loudness scaling measurement.
lsa_LeftAided see TLSAided	The left ear was aided during the loudness scaling measurement.
lsa_NotAided see TLSAided	No hearing aids were used during the Loudness Scaling.

lsa_RightAided see TLSAided	The right ear was aided during the loudness scaling measurement.
lsid_IHAFF_1 See TLSMethod	Independent Hearing Aid Fitting Forum Method 1.
lsid_IHAFF_2 See TLSMethod	Independent Hearing Aid Fitting Forum Method 2.
lsid_JRP See TLSMethod	German Joint Research Project.
lsid_LGOB_1 See TLSMethod	Loudness Growth in Half-Octave Bands Method 1.
lsid_LGOB_2 See TLSMethod	Loudness Growth in Half-Octave Bands Method 2.
lsid_MD_1 See TLSMethod	Madsen / Danavox Method 1.
lsid_MD_2 See TLSMethod	Madsen / Danavox Method 2.
lsid_RELM See TLSMethod	Real Ear Loudness Mapping.
lsid_USER1 See TLSMethod	User defined Loudness Scaling Method 1.
lsid_USER2 See TLSMethod	User defined Loudness Scaling Method 2.
lsid_USER3 See TLSMethod	User defined Loudness Scaling Method 3.
lsid_USER4 See TLSMethod	User defined Loudness Scaling Method 4.
lsid_WHF See TLSMethod	Würzburger Hörfeld Method.
lsMeasurement	The LSMeasurement structure contains the result of a Loudness Scaling Measurement done at up to 11 different frequencies.
lsMethod	Loudness Scaling Method. Ref. 2.1.3.7. A component of the type TLSExtended MeasCond.
lsMaxAnswer	The actual number of filled Raw data points may be lower than

	indicated with this variable. If undefInt is inserted, no Maximum Number of answers is available. The range for MaxAnswer: [1..50] - Up to 50 raw data points can be saved in the LSdef.h Extended data structure. Reference: Ref. 2.1.3.6: TLSMaxAnswer on page 15.
lsso_BoneConduction See TLSSignalOutput	The stimulus transducer used during the measurement is a Bone Conductor. Ref. 2.1.2.9: TLSSignalOutput on page 13.
lsso_InsertPhone. See TLSSignalOutput	The stimulus is presented via an Insertphone.
lsso_Phone. See TLSSignalOutput	The stimulus is presented via earphones.
lsso_SoundField. See TLSSignalOutput	The stimulus is presented via a free field in a sound chamber.
lsst_NatureSound See TLSSignalType	The signal type used during the measurement is Nature Sound. (Ref IEC __________).
lsst_NBN. See TLSSignalType	The signal type used during the measurement is Narrow Band Noise (unspecified bandwidth).
lsst_NBN12 See TLSSignalType	The signal type used during the measurement is 1/2 octave Narrow Band Noise. The Middle Frequency should be saved as a Frequency component in the Loudness Scaling Data. Reference: [IEC 645-1] para 6.3.1.
lsst_NBN13 See TLSSignalType	The signal type used during the measurement is 1/3 octave Narrow Band Noise. The Middle Frequency should be saved as a Frequency component in the Loudness Scaling Data.
lsst_Speech See TLSSignalType	The signal type used during the measurement is speech weighted noise refer IEC ________.
lsst_Tone. See TLSSignalType	The signal type used during the measurement is a pure tone. Refer IEC________.
lsst_Warble. See TLSSignalType	The signal type used during the measurement is a warble tone. Refer IEC________.

M

Maximum Number of possible answers	The actual number of filled Raw data points may be lower than indicated with this variable. If undefInt is inserted, no Maximum Number of answers is available. The range for MaxAnswer: [1..50] - Up to 50 raw data points can be saved in the LSdef.h Extended data structure. Reference: 2.1.3.6: TLSMaxAnswer on page 15.
maxInt	Integers are stored using 2’s complement in a two-byte store. This means that maxInt =32767 (#7FFF hex). [Framework] confirms that this is the highest positive value used in NOAH. See chapter 2.1.1:The Integer type used in Extended LSdef.H.
MeasCond	Measuring Conditions. In this Extended LSDEF.H the measuring conditions consists of the fields TLSSignalOutput, TLSSignalType and TLSdB.
Minimum Settings	The recommended minimum of Measurement Conditions that must be saved with a measurement in order to make it valuable when retrieved at a later stage.
minInt	Integers are stored using 2’s complement in a two-byte store. This means that minInt = -32768 or #8000 hex. See chapter 2.1.1: The Integer type used in .

N

Name	Loudness Scaling Method Name. A string of 32 ASCII characters that describes the Loudness Scaling Method identified with LSMethod.Id. If this string is filled with ASCII " " (spaces) it means that no valid LS Method.Name is available.
New DataformatCode for extended data	Up to the AEM meeting in Germany in October 1997 existed LSdef.h DataFmtCodeStd 100. This header has been extended to form LSdef.h DataFmtCodeStd 110 in October 1997. Both formats are valid as Noah Loudness Scaling Standard.
norm	Up to three points on a Loudness Scaling / Loudness Level curve allowing for maximum one knee point. See para. 2.1.2.4: Loudness Scaling Curves on page 11.
Norm graph. See Tnorm.	A normal hearing person's Loudness Scaling data can be represented for each frequency by up to three Loudness Scaling Points (Type TLSPoint), i.e. max. one knee point is allowed per frequency. See para. 2.1.2.4 Loudness Scaling Curves on page 11.

R

RawData. See TrawData	The nature of the Loudness Scaling Measurement makes it necessary to present a large number of samples for the Hearing Instrument User, who in turn will indicate the perceived loudness on a scale from 0 to 100. Up to 50 TLSPoints can be stored as RawData per frequency. The various measurements results in Raw Data points that later can be interpreted using statistical methods. The outcome, the Loudness Scaling Result can be saved in 3 TLSPoints allowing a graph with max. one knee point per frequency.

S

Scaling result graph. See TScalingResult	(See explanation for RawData).
ScalingResult	A component of the structure Data. The type is TScalingResult, i.e. 3 TLSPoints. See para. 2.1.2.4: Loudness Scaling Curves on page 11.
SignalOutput	The channel/media used to present the signal/stimulus. This can be headphones, insertphones or a free field representation, indicated by #defines of type Integer.
SignalType	The signal type used during the measurement. It can be Pure tone, Warble Tone or Narrow Band Noise. Type Integer.
SPL	Sound Pressure Level relative to the reference level 20 microPascal. (µPa).

T

TdB10	A Level (often Sound Pressure Level) expressed in dB x 10 or CentiBel.
TDevTypeCode	Device Type Code, defined as Integer in noahdef.h.
THertz	The frequency of the stimulus signal is represented in Hertz. Range [20..20 000].
TLSAided	The Loudness Scaling Aided expresses whether the client was using a hearing instrument during the Loudness Scaling:
TLSExtended-MeasCond	Extended Measuring Conditions. See para. 2.1.3.1: TLSExtendedMeasCond on page 14.

TLSMeasurement	Loudness Scaling Measurement Structure. See para.1.1: A few words about programming with LSDEF.H on page 5 and also the definition in para. 2.1.2.2 TLSMeasurement on page 10.
TLSMethod	Loudness Scaling Method. See para. 2.1.3.7: TLSMethod on page 16.
TLSdB	The Weighting of the measured Loudness Scaling can be SPL or HL. Type Integer.
TLSMeasurement	The central structure in LSDEF.H. Contains Measuring Conditions plus measurement data for up to 11 measuring frequencies.
TLSPoint	The TLSMeasurement Structure contains a Loudness Scaling Result, a Norm Graph plus a set of RawData. All of these components are of the type TLSPoint.
TLSSession	The complete data structure Loudness Scaling Session consists of two Loudness Scaling Measurements with the result saved in dB SPL and dB HL respectively.
TLSSignalOutput	The Signal Output can be defined as Phone, insertphones or Sound Field (Free field). The type represents the presentation of stimulus. Type Integer.
TLSSignalType	The Signal Type can be defined as Tone, Warble Tone or Narrow Band Noise (NBN). The type represents the type of stimulus used. Type Integer.
TNorm	The type TNorm is defined as three TLSPoints (i.e. Presentation level in dB, Perceived Loudness in percent). The three points represent a normal hearing person's Loudness Scaling function. One knee point is thus allowed.
TPct100	Percentage x 100. (or 1/10 000).
TRawData	The type TRawData is defined as 50 TLSPoints (i.e. Presentation level in dB, Perceived Loudness in percent). The raw data points are saved so alternative algorithms for finding Loudness Scaling curve can be applied at a later stage by retrieval of all the measurements done. The actual resulting three curvepoints are calculated on basis of the raw data and saved in the structure TScalingResult. See para. 2.1.2.3: RawData on page 10.
TScalingResult	(see explanation above)

U

undefInt	The Integer value -32767. (#8001 hex). Used to indicate that a value is undefined; a value which is assigned to the constant undefInt Ref. [Framework].
unknown.	The Integer value 0. (#0000 hex). When used as a parameter value it means that the parameter is **defined,** however to an unknown value.

Appendix B: The header file Extended LSDEF.H

```
//////////////////////////////////////////////////////////////////////
// Extended LSDEF.H
//////////////////////////////////////////////////////////////////////
// Version 1.90
//////////////////////////////////////////////////////////////////////


//////////////////////////////////////////////////////////////////////
// Definition of Aurical Loudness Scaling public data
//////////////////////////////////////////////////////////////////////
//
// Document History:
//
// 950720: Supplied by Madsen Electronics
//
// 950720: 950720,BBJ   Enums changed to #defines
//
// 960730: Modified by Carsten Schmidt, Rexton Danplex A/S
//
// 971127: Changes after AEM Meeting in October 1997 by SOK
//
// Format: DataFmtCodeStd = 110
//          DataTypeCode = dtc_LS_L (17) / dtc_LR_R (18)
//

#ifndef __LSDEF_H
#define __LSDEF_H


////////////////////////////////////////////////////////////////////////////
// From NOAHdef.h & AUDdef.h:
////////////////////////////////////////////////////////////////////////////

#define          UndefInt    (-32767)
typedef int      THertz;     // Frequency in Hertz
typedef int      TdB10;      // Level in centiBell
typedef int      TPct100;    // Hundredths of percent or 10EE-4

////////////////////////////////////////////////////////////////////////////
// Extended Loudness Scaling definition:
////////////////////////////////////////////////////////////////////////////

// Signal output
typedef int TLSSignalOutput
#define lsso_Unknown          0     // The Signal Output is unknown
#define lsso_Phone            1     // Phone
#define lsso_InsertPhone      2     // Insert Phone
#define lsso_SoundField       3     // Free Field
// Additional defines by extension (0 also added)
#define lsso_BoneConduction   4     // Bone Conductor
#define user_1                5     // User defined 1
#define user_2                6     // User defined 2
#define user_3                7     // User defined 3
```

```
// Signal type
typedef int TLSSignalType;
#define lsst_Unknown        0       // The Signal Type is unknown
#define lsst_Tone           1       // Pure Tone
#define lsst_Warble         2       // WarbleTone
#define lsst_NBN            3       // Narrow Band Noise (not specified)
// Additional defines by extension (0 also added):
#define lsst_NBN13          4       // Selected signal is 1/3 NarrowBandNoise
#define lsst_NBN12          5       // Selected signal is 1/2 NarrowBandNoise
#define lsst_Speech         6       // Selected signal is Speech Noise
#define lsst_Naturesound    7       // Selected signal is nature sound

// dB weighting
// Additional define by extension: 0
typedef int TLSdB;
#define lsdb_Unknown        0       // The reference level is unknown
#define lsdb_HL             1       // Levels saved in Hearing Level
#define lsdb_SPL            2       // Levels saved in SPL


// Loudness Scaling point (A point is invalid if Level == UndefInt )
typedef struct {
    TdB10           Level;
    TPct100         Loudness;
} TLSPoint;

// Scaling result graph (3 points)
typedef TLSPoint    TScalingResult[3];

// Norm graph (3 points)
typedef TLSPoint    TNorm[3];

// Raw data (RAWDATA points)
#define RAWDATA     50

typedef TLSPoint    TRawData[RAWDATA];


// A Loudness Scaling struct (up to 11 frequencies)
// ( If Data[x].Frequency == UndefInt -> Data[0]..Data[x-1] is valid )

typedef struct {
    TLSSignalOutput     SignalOutput;
    TLSSignalType       Signal;
    TLSdB               dB;
    struct {
                        THertz              Frequency;
                        TScalingResult      ScalingResult;
                        TNorm               Norm;
                        TRawData            RawData;
    } Data[11];
} TLSMeasurement;
```

```
// Not used in the Extended LSDef.h:
// A complete Loudness Scaling session (2 Loudness Scaling struct)
// typedef            TLSMeasurement     TLSSession[2];



// Additional definitions by extension
// Changing DataFmtCodeStd from 100 to 110

// The Loudness Scaling Aided expresses whether the client was using
// a hearing instrument during the Loudness Scaling:
typedef int TLSAided;
#define lsa_Unknown        0        // Unknown
#define lsa_NotAided       1        // Not aided
#define lsa_LeftAided      2        // Left Ear aided
#define lsa_RightAided     3        // Right Ear aided
#define lsa_BothAided      4        // Left and Right Ear aided

// Loudness Answer Categories
// Count of different answer categories, i.e. loudness weightings
// i.e. 5, 7, …  keys on LS keyboard
// The value undefInt means Answer Categories are Not available
typedef int TLSAnswerCategory;

// Maximum number of answers
// (i.e. 50 means [0..50] answers)
// The value undefInt means No Maximum number of answers available
// The value 0 means that the Maximum number of answers is unknown

typedef int TLSMaxAnswer;

// Identification Structure
// Identification Number and Name of the LS Method
// If the Id element is undefInt it means No LSMethod is available
// The ASCII character "      " (space) is recommended in this case for Name

typedef struct
{
    int Id;
    char Name[32];
}
TLSMethod;
```

```
// The Identification number is part of the structure TLSMethod
// The value undefInt means No valid Identification

#define lsid_Unknown        0     // The LS Method is unknown
#define lsid_Not_Used       1     // None of the listed LS methods were used
#define lsid_IHAFF_1        2     // Independent Hearing Aid Fitting Forum Method 1
#define lsid_IHAFF_2        3     // Independent Hearing Aid Fitting Forum Method 2
#define lsid_LGOB_1         4     // Loudness Growth in Half-Octave Bands Method 1
#define lsid_LGOB_2         5     // Loudness Growth in Half-Octave Bands Method 2
#define lsid_JRP            6     // German Joint Research Project
#define lsid_RELM           7     // Real Ear Loudness Mapping
#define lsid_WHF            8     // Würzburger Hörfeld
#define lsid_MD_1           9     // Madsen/Danavox Method 1
#define lsid_MD_2           10    // Madsen/Danavox Method 2
#define lsid_User1          11    // For future use
#define lsid_User2          12    // For future use
#define lsid_User3          13    // For future use
#define lsid_User4          14    // For future use



// The Binaural variable indicates if stimulus was applied to both ears during the loudness scaling.
// The client answers would in this case be based on a binaural loudness scaling measurement.
//
// In the binaural case, the measuring results have to be saved in two TLSSessions, one for each ear, the
// reason being that the patients' answers would be related to a Loudness Scaling related on both ears.
// The two TLSSessions should be identical, and it is thus only necessary to read one of the TLSSessions.

typedef int TLSBinaural;

#define lsb_Monaural        0     // Stimulus to one ear
#define lsb_Binaural        1     // Stimulus to both ears



// Extended Measuring Conditions
typedef struct
{
    TLSAided            LSAided;            // The use of hearing aids during LS
    TLSAnswerCategory   LSAnswerCategory;   // Answer categories on patient answering device
    TLSMaxAnswer        LSMaxAnswer;        // Maximum number of answers per frequency
    TLSMethod           LSMethod;           // Identification Number and name of LS method
    TLSBinaural         LSBinaural;         //
}
TLSExtendedMeasCond;
```

```
//A complete LS session: Extended version DataFmtCodeStd 110
// 2 LSMeasurement + LSData

typedef struct
{
    TLSMeasurement          LSMeasurement[2];
    TLSExtendedMeasCond  LSExtendedMeasCond;
}
TLSSession110;

// New DataFormatCodeStandard for extended data
#define dfcs_110          110

#endif
```

Index

U