LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY
- LIGO -
CALIFORNIA INSTITUTE OF TECHNOLOGY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Technical Note **LIGO-T990111-B - D** 5/19/00

Calibration Record Proposal

Daniel Sigg

Distribution of this draft:

all

This is an internal working note
of the LIGO Project.

LIGO Hanford Observatory
P.O. Box 1970 S9-02
Richland, WA 99352
Phone (509) 372-8106
FAX (509) 372-8137
E-mail: info@ligo.caltech.edu

LIGO Livingston Observatory
19100 LIGO Lane
Livingston, LA 70754
Phone (504) 686-3100
FAX (504) 686-7189
E-mail: info@ligo.caltech.edu

California Institute of Technology
LIGO Project - MS 51-33
Pasadena CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project - MS NW17-161
Cambridge, MA 01239
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

www: http://www.ligo.caltech.edu/

1 INTERFACE REQUIREMENTS

There shall be a calibration database which stores calibration data and can be queried by clients through a common C/C++ API. Calibration data is returned to the client in the LIGO light weight data format where it can be transformed back into a list of calibration records. Calibration data can be added to the database either through a graphical user interface or by supplying a file (LIGO lightweight format). In general calibration records are only added. However, there shall be a way to remove records which were found to be incorrect.

2 RECORD DEFINITION

A calibration record contains the following elements:

- CHANNEL: channel name, e.g., “H1:LSC-GW”.
- TIME: time when calibration tales effect (in GPS sec).
- DURATION: Duration while calibration is valid (in sec).
- REFERENCE: reference point, e.g., “ADC input” or “coil driver”.
- UNIT: unit string, e.g., “V” or “m/s”.
- TYPE: calibration type, an or’ed list of the following:
  CALAMPLITUDE (1): amplitude conversion supported,
  CALOFFSET (2): offset correction supported,
  CALTIMEDELAY (4): time delay correction supported,
  CALTRANSFERFUNCTION (8): transfer function supported in array form,
  CALPOLEZERO (16): transfer function supported in pole/zero notation.
- CONVERSION: amplitude correction coefficient.
- OFFSET: offset correction.
- TIMEDELAY: phase (in rad) or time delay (in sec)
- TRANSFERFUNCTION: array with elements in format {Frequency (Hz), Amplitude ratio, Phase shift (rad)}. A positive phase represents a lead.
- POLEZERO structure with
  GAIN: gain factor of transfer function,
  POLENUM: number of poles,
  ZERONUM: number of zeros,
  POLEZEROS: complex array of poles and zeros in format {fPole_1, fZero_1, fPole_2, ...}.
- DEFAULT: indication if this is the default calibration.
- PREFERREDMAG: preferred order of magnitude, clients are encouraged to recognize at least:
  –15 (femto) –12 (pico), –9 (nano), –6 (micro), –3 (milli), 0, 3 (kilo), 6 (Mega), 9 (Giga),
  12 (Terra) and 15 (Peta).
- PREFERREDD: preferred order of derivative.
- COMMENT: arbitrary text.

Comments:

a. The record elements CHANNEL, TIME, REFERENCE and UNIT must be unique, e.g., a calibration of a single channel can be done at different reference points with several different unit conversions.
b. A time of zero indicates a record which is valid from the beginning. A duration of zero indicates a calibration which has no end date.
c. The type argument is a bit-encoded number describing the calibration corrections supported by this record. Currently, non-linear amplitude corrections are not supported. A missing CALOFFSET or CALTIMEDELAY flag defaults the corresponding calibration correction values to zero, but a missing CALAMPLITUDE or CALTRANSFERFUNCTION/ CALPOLEZERO flag indicates that there is no simple time domain or frequency domain calibration transformation, available, respectively.
d. Transfer functions should supply frequency points covering the full signal range, i.e. from zero to the Nyquist frequency.
e. Calibration corrections are in general referred to the time series as stored in the Frame. There is no automatic serialization of calibrations which are performed in a “piece-by-piece” measurement.

A representation in C could look like:

```
#define CALCHANNELSIZE    40
#define CALREFERENCESIZE  40
#define CALUNITSIZE       40

#define CALAMPLITUDE          1
#define CALOFFSET             2
#define CALTIMEDELAY          4
#define CALTRANSFERFUNCTION   8
#define CALPOLEZERO          16

struct polezero_t {
   double         fGain;
   int            fPoleNum;
   int            fZeroNum;
   float*        fPoleZeros;
};
typedef struct polezero_t polezero_t;

struct calrec_t {
  char          fChannel[CALCHANNELSIZE];
  unsigned long fTime;
  unsigned long fDuration;
  char          fReference[CALREFERENCESIZE];
  char          fUnit[CALUNITSIZE];
  int           fType;
  double        fConversion;
  double        fOffset;
  double        fTimeDelay;
  float*        fTransferFunction;
  int           fTransferFunctionLen;
```

```
    polezero_t     fPoleZero;
    int            fDefault;
    int            fPreferredMag;
    int            fPreferredD;
    char*          fComment;
};
typedef struct calrec_t calrec_t;
```

3 NAMING CONVENTION

Channel names follow the guidelines outlined in T990033-A. There are a few predefined reference points:

```
sensor   represents the quantity measured by a sensor
ADC      represents the signal as measured by an ADC input
memory   represents a digital signal
```

In general, the units associated with an ADC input are “V” (Volts) and “#” (counts), whereas physical units are associated with a sensor reference point.

Recognized fundamental units are:

```
s       Time (second)
m       Length (meter)
m/s     Velocity (meter per second)
m/s^{2}Acceleration (meter per second square)
Hz      Frequency (Hertz)
#       Counts, Number
N       Force (Newton)
Pa      Pressure (Pascal)
l       Volume (liter)
W       Power, Intensity (Watts)
C       Temperature (Celius)
V       Voltage (volts)
A       Current (Ampere)
T       Magnetic field strength (Tesla)
Ohm     Resistance (Ohm)
rad     Angle (radians)
```

It is preferable to use the above units in the calibration database and make scaling adjustments, e.g., milli or micro, together with simple conversions, such as degree/rad, in the client interface. These are then called transformed units (see last section).

4 CALIBRATION CORRECTIONS

A simple calibration transformation of a raw time series signal can use the following formula

$$y_i = \mathrm{CONVERSION} \times (x_i - \mathrm{OFFSET}) \tag{1}$$

where x_i is the originally sampled data point and y_i is the transformed data point which represent the data value with the physical units specified in the UNIT string. If a time delay is specified the time axis is transformed by

$$t_i = T_i - \text{TIMEDELAY} \tag{2}$$

If the time delay is larger than a sampling period, the client will have to decide whether it wants to shift the time series points (e.g., useful for plots displaying multiple time series) or just modify the time axis.

If a transfer function is specified, an advanced client may decide to construct a time domain filter which approximates the transfer function and filter the data prior of further processing. If the final result is defined in the frequency domain, the calibration transformation is best applied there, i.e.,

$$\tilde{y}_i = c_i \times \tilde{x}_i \tag{3}$$

where $\tilde{x}_i$ is the data value describing the i-th frequency point, $\tilde{y}_i$ is the transformed frequency point representing physical units, and c_i is the transfer coefficient which corresponds to the i-th frequency point (written as a complex number). If the transfer function does not specify a coefficient at the exact point i, the coefficient has to be interpolated from the neighboring points. An offset correction is preferably applied to the original time series before making a Fourier transformation.

Sometimes there is no good way to specify a simple calibration transformation for a time series (e.g., if the calibration transformation has a $1/f$ dependency); at the same time the calibration transformation in the frequency domain might be well defined. In this case the calibration record should specify a transfer function and optionally an offset and time delay, but must not set the CALAMPLITUDE flag.

A transfer function can either be specified as a list of coefficients or a list of poles and zeros. The transfer function is deduced from the poles and zeros as follows:

$$C(f) = c_{gain} \prod_i \text{pole}(f, p_i) \prod_i \text{zero}(f, z_i) \tag{4}$$

The pole and zero functions are defined as follwos:

$$\text{pole}(f,p) = \begin{cases} \dfrac{1}{1 + if/p} & \text{when } p > 0 \\ \dfrac{1}{if} & \text{when } p = 0 \end{cases} \quad \text{and} \quad \text{zero}(f,z) = \begin{cases} 1 + if/z & \text{when } z \neq 0 \\ if & \text{when } z = 0 \end{cases} \tag{5}$$

For both the transfer function and the pole-zero notation the units for frequency is Hz rather than rad/s (angular frequency). In particuar, poles and zeros are specified by their location in (nromal) frequency space.

5 XML Representation

The representation of a calibration record in the LIGO-LW format looks like follows:

```
<LIGO_LW Name="Calibration">
  <Param Name="Channel" Type="string">H0:PEM-LVEA_SEISX</Param>
  <Time  Type="GPS">615445949</Time>
  <Param Name="Duration" Type="int">0</Param>
  <Param Name="Reference" Type="string">ADC</Param>
  <Param Name="Unit" Type="string">m/s</Param>
  <Param Name="Conversion" Type="double">0.000061035</Param>
  <Param Name="Offset" Type="double">-950</Param>
  <Param Name="TimeDelay" Type="double">0.00097</Param>
  <Param Name="TransferFunction" Type="double" Dim="24">
    0    0.000061035 0
    10   0.000061033-0.061040
    100  0.000061003-0.610497
    300  0.000060669-1.87035
    800  0.000060919-6.32689
    850  0.000061052-7.2299
    900  0.000038326-8.5596
    1024 0.000002865-10.079
  </Param>
  <Param Name="Gain" Type="double">1.0</Param>
  <Param Name="Poles" Type="doubleComplex" Dim="2">
    0.2 0.7
    0.2 -0.7
  </Param>
  <Param Name="Zeros" Type="doubleComplex" Dim="1">
    0.0 0.0
  </Param>
  <Param Name="Default" Type="boolean">0</Param>
  <Param Name="PreferredMag" Type="int">-6</Param>
  <Param Name="PreferredD" Type="int">-1</Param>
  <Param Name="Comment" Type="string">done by Ski</Param>
</LIGO_LW>
```

The type argument is encoded implicitly, i.e., only specified calibration corrections are set valid.

When storing a set of calibraton records in a file or when sending it over the network, a well-formed XML doument has to be built. It follows the LIGO-LW defintion and may look like:

```
<?xml version="1.0"?>
<!DOCTYPE LIGO_LW SYSTEM "http://www.cacr.caltech.edu/projects/ligo_lw.dtd">
<LIGO_LW>
  <LIGO_LW Name="Calibration">
    <Param Name="Channel" Type="string">H0:PEM-LVEA_SEISX</Param>
    ...
  </LIGO_LW>
  <LIGO_LW Name="Calibration">
    <Param Name="Channel" Type="string">H0:PEM-LVEA_SEISY</Param>
    ...
  </LIGO_LW>
  ...
```

```
</LIGO_LW>
```

Note that all calibration records are enclosed within a single LIGO_LW structure. Also, the first two lines should be copied exactly the way above.

Sometimes it is useful to be able to distinguish calibration records within a set. This is supported by an optional index which can be added to the name attribute. For example:

```
<?xml version="1.0"?>
<!DOCTYPE LIGO_LW SYSTEM "http://www.cacr.caltech.edu/projects/ligo_lw.dtd">
<LIGO_LW>
  <LIGO_LW Name="Calibration[0]">
    ...
  </LIGO_LW>
  <LIGO_LW Name="Calibration[1]">
    ...
  </LIGO_LW>
  ...
</LIGO_LW>
```

Clients which read calibration records from a file but do not need the index information must simply ignore them. To support future extensions clients also have to ignore any parameters within a calibration record or attributes thereof which they do not recognize.

6 CLIENT-SERVER PROTOCOL

Clients communicate with the calibration server by exchanging LIGO-LW text files through, for example, a socket interface. The client indicates the requested service by adding a type attribute to the calibration record. Supported services are:

```
<LIGO_LW Name="Calibration" Type="Add">    Adding a calibration record
<LIGO_LW Name="Calibration" Type="Delete"> Delete a calibration record
<LIGO_LW Name="Calibration" Type="Query">  Query the database
```

A typical query might look like:

```
<?xml version="1.0"?>
<!DOCTYPE LIGO_LW SYSTEM "http://www.cacr.caltech.edu/projects/ligo_lw.dtd">
<LIGO_LW>
  <LIGO_LW Name="Calibration" Type="Query">
    <Param Name="Channel" Type="string">H0:PEM-LVEA_SEISX</Param>
  </LIGO_LW>
</LIGO_LW>
```

Query parameters can be supplied for CHANNEL, TIME, DURATION, REFERENCE and UNIT. If no CHANNEL, REFERENCE or UNIT is supplied, the server will ignore the parameter in the matching and return all calibration records which match the supplied parameters. If the both TIME and DURATION are missing or set to zero, the server will return the currently valid calibration. Queries support the wildcard character '*' as part of the channel, reference or unit specification. The wildcard character must be the last character in the specified string and is interpreted as anything which matches the specified string up to the wildcard character. For example:

```
<?xml version=”1.0”?>
<!DOCTYPE LIGO_LW SYSTEM “http://www.cacr.caltech.edu/projects/ligo_lw.dtd”>
<LIGO_LW>
  <LIGO_LW Name=”Calibration” Type=”Query”>
    <Param Name=”Channel” Type=”string”>H0:PEM-*</Param>
    <Param Name=”Reference” Type=”string”>ADC</Param>
    <Param Name=”Unit” Type=”string”>*</Param>
  </LIGO_LW>
</LIGO_LW>
```

This query will return all channels beginning with “H0:PEM-” which have their reference set to “ADC”. The comparison is not case sensitive. In the above example the unit specification is a wildcard only which means the query will not discriminate against unit strings (this is equivalent to omitting the unit specification all together).

When specifying a time or a time interval for a query, a few special cases are treated separately:

```
<?xml version=”1.0”?>
<!DOCTYPE LIGO_LW SYSTEM “http://www.cacr.caltech.edu/projects/ligo_lw.dtd”>
<LIGO_LW>
  <LIGO_LW Name=”Calibration” Type=”Query”>
    <Param Name=”Channel” Type=”string”>H0:PEM-LVEA_SEISX</Param>
    <Param Name=”Duration” Type=”int”>0</Param>
  </LIGO_LW>
</LIGO_LW>
```

An omitted duration or one that is set to zero indicates that only the most recent calibration should be returned, if multiple calibration records are found which match in channel, reference and unit. If a non-zero time argument is specified only calibration records which are more recent than the specified time will be considered in the query. An omitted or zero time specification always indicates that the query should not restrict the start time of the search. When a query is intended to return all calibrations before a certain time, the time argument should be set to zero (or omitted) and the duration should be set to this time. When a query is intended to return all calibrations after a certain time, the time argument should be set to this time and the duration must contain a very large number which covers at least the time between the start time and the current time. If for some reasons the desired query result can not be specified with the above arguments, the client can always ask for more than it needs and implement a more restrictive search on its own.

The return of a query is another LIGO LW file consisting of all calibration records which match the query request. The end of the returned answer is again indicated by an empty LIGO_LW element.

A calibration record scheduled to be added or deleted must supply CHANNEL, TIME, REFERENCE and UNIT. (The server will compute the DURATION field automatically.) In order to add or delete a calibration record the client will also need to provide a valid authorization code. The authorization element has to be supplied before any add or delete operations are requested:

```
<LIGO_LW Name=”Authorization”>
  <Param Name=”User” Type=”string”>me</Param>
  <Param Name=”Password” Type=”string”>why?not</Param>
</LIGO_LW>
```

The return of and add or delete operation is an empty LIGO LW file upon success. If an operation failed the retuned LIGO LW file contains a lists of all records which couldn't be added or deleted, respectively. The type of these failed record is set to "Error". Additionally, an error parameter is added to indicate what went wrong. For example:

```
<?xml version="1.0"?>
<!DOCTYPE LIGO_LW SYSTEM "http://www.cacr.caltech.edu/projects/ligo_lw.dtd">
<LIGO_LW>
  <LIGO_LW Name="Calibration" Type="Error">
    <Param Name="Error" Type="string">Unknown channel name</Param>
    <Param Name="Channel" Type="string">H0:LVEA_SEISX</Param>
    <Time  Type="GPS">615445949</Time>
    <Param Name="Duration" Type="int">0</Param>
    <Param Name="Reference" Type="string">ADC</Param>
    <Param Name="Unit" Type="string">V</Param>
    <Param Name="Conversion" Type="double">0.000061035</Param>
    <Param Name="Offset" Type="double">-950</Param>
    <Param Name="TimeDelay" Type="double">0.00097</Param>
  </LIGO_LW>
</LIGO_LW>
```

7 Deduced, Transformed and Composite Units

A typical data client may present the user with the following choices for selecting a unit:

- *none* which displays the data as is,
- *list* which presents the user with a list of units to choose from, and
- *user* which lets the user specify the unit string, the conversion factor and the offset correction.

All three cases seem straight forward to implement, in practice however, there are several subtleties which have to be addressed:

- The choices in the list—if they are obtained directly from the calibration database—could be non-ideal, e.g., the user would like to see the temperature in Kelvin rather than Celsius, or a unit such as μV would be more appropriate than just Volts. It is impractical to store each of these possible variants in the calibration database.
- When calculating a power spectral density, more than one possibility exists to deduce a unit from the original time series, i.e., V, $V/\sqrt{Hz}$, V^2,V^2/Hz.
- When displaying data in the frequnecy domain it is straight forward to account for derivatives with respect to time, e.g, if the units are $m/s^2/\sqrt{Hz}$ the user may want to display $m/\sqrt{Hz}$.
- How to deal with quantities whose calculation involved multiple channels, e.g., transfer functions or cross-power spectra.
- How to deal with complex quantities such as transfer coefficients.

The following paragraphs describe the solution adopted by the diagnostics test tool to handle the following plot types:

- normal and down-converted time series,
- power spectral density estimations,
- coherence and cross-power spectra,
- transfer functions and coherence functions,
- transfer, coherence, harmonic and two-tone-intermodulation coefficients.

In short, the diagnostics calibration involves the following steps:

- deduce the appropriate units from the calibration records which are obtained from the calibration database,
- let the user select a unit,
- apply the calibration correction to the data, and
- for complex data let the user select whether to display the real part, the imaginary part, the magnitude, the dB magnitude, the phase or the continuous phase.

Findinf related units involves the following steps:

- Finding related units: rad $\rightarrow$ deg, C $\rightarrow$ K, Pa $\rightarrow$ mBar, m/s $\rightarrow$ km/h, etc.
- Taking into account the unit transformation introduced by the analysis.
- Adding a magnitude qualifier if required: fm (femto), p (pico), n (nano), u (micro), m (milli), k (kilo), M (Mega), G (Giga), T (Terra) and P (Peta).

Trivial cases of analysis transformations are coherence estimates which are unit less and time traces which just carry over the original units. The remaining cases can be divided into three categories: spectral densities, ratios (channel B over channel A) and cross-spectral densities (channel A times channel B). Spectral density units are deduced from their original unit as follows: “unit” for amplitude, “(unit)/rtHz” for amplitude density, “(unit)^{2}” for power, and “(unit)^{2}/Hz for power density. For simple units the brackets are omitted. Ratios are formed by taking all possible combinations of “(unitB)/(unitA)”. For simple units the brackets may be omitted. If both units are identical the unit string becomes empty. Similarly, cross-spectral densities are formed by taking all possible combinations of “(UnitA)(UnitB)/Hz”.

The calibration records contains three fields which can be used by a client for determining default display units:

- If more than one calibration records are defined for a given channel, the first one which has the default flag set should be used,
- If a preferred magnitue is specified it should be used, and
- If the client support derivative units, the preferred D value should be considered.

If spectral densities are plotted, the default unit should be an amplitude spectral density.