SiWIM-E Mk.III Technical Reference Manual
v5.27.66

Written by Jan Kalin <jan.kalin@zag.si>

21. December 2015

Contents

Chapter 1

Components, Cores and Configuration Files

1.1 Introduction and General Information

This document:

- describes the SiWIM-E Components,
- lists the so-called cores that perform weighing calculations and various other operations on the signals
- describes all the options and modes of operations of these cores and
- describes the contents of configuration files that control the behaviour of the system and contain options for cores.

It is not intended as substitute for a user manual for end-users and it does not attempt to guide towards a successful setup of a working SiWIM system.

1.1.1 File Names

Most configuration files belong to one of the cores used in the SiWIM system, Since some standard module names do not correspond to core names and, since a core can be used in an arbitrarily named module, this documentation assumes that the module name is the same as the core name, for the sake of simplicity.

E.g., the fftf core (Fast Fourier Transform Filter) is described as using the fftf.conf configuration file. The actual configuration file name used for the standard module "filter", that uses the fftf core, would be filter.conf.

Some other files, e.g., vehicle_classes.conf, are used by more than one core or by the main components of a SiWIM-E system. Names of those files do not change.

N.B.: All references to directory names in the configuration files and throughout this documentation contain forward slashes (/) and not backslashes (\), as is usual on Windows OS. The reason is that the software also runs on linux, where only forward slashes are accepted, whereas on Windows both are acceptable.

1.1.2 Names, Types and Default Values

Each documented value includes in its description:

- The type of value
- The system default value
- The template default value

1.1.2.1 Names of Values

The names of values are mostly very descriptive, so it's relatively easy to edit the configuration files by hand, if necessary. The convention is that the names are written in lowercase with underscores as separators between words, e.g., `system_location`.

If a value is a physical quantity, the units are separated from the name with two underscores, e.g., `sampling_rate__Hz` and `max_speed__m/s`

In some cases the value is a percentage of some other value defined in a configuration file or a value obtained during data processing. The percent sign is, similarly to the units, separated by two underscores, e.g., `adjust_W1__%` and `ats_start__%`.

1.1.2.2 Types of Values

The values within the configuration files are generally scalars of one of five types:

- integer
- unsigned integer,
- floating point,
- boolean (which can be `true` or `false`) and
- string.

Some values are lists of scalars (e.g., a list of floats), where the values in a list are separated by commas.

In other cases the formats of values are more complicated and are described where they occur, but generally the lists of non-scalars are separated by semicolons (`;`) and the elements within such lists are separated by commas.

1.1.2.3 System Default Values

Some of the values have defaults that are, in some cases, sufficient for basic operation and, in other cases, serve merly to get the SiWIM-E running so that the proper values can be determined.

For many values there can be no sensible default. If the SiWIM-E attempts to read such a value, but it has not been defined, it will:

- write a line to log file, explaining which value is missing from which configuration file, and
- the core that had attempted to read this value will dump all incoming data to disk.

The same will happen if a value is present, but is of the wrong type or, in some cases, if it is outside sane boundaries or incompatible with other values.

N.B.: Not all accepted values are also sensible, e.g., a trigger level can be set at 12V, but the signal levels never rise above 10V, so the trigger will never go off.

It is impossible for SiWIM-E to determine exactly what the user intended and whether a combination of the values fulfills the user's expecations, without building a large expert system for checking the values.

This is currently far beyond the capabilies of the SiWIM-E, so SiWIM-F is expected to provide some guidance towards choosing sensible values, in concert with a user manual, that should be available Real Soon Now.

1.1.2.4 Template Default Values

SiWIM-F has the capability of using predefined templates to generate configuration files. Some of the values depend on the template used, some are constant throughout the templates.

Since the templates have not been examined during the writing of this documentation, the defaults are currently unknown and marked as such.

The authors of templates are welcome to contribute to this documentation!

1.1.3 Coordinate System

All the positions in the SiWIM-E configuration files are defined in a Cartesian coordinate system.

The x-axis is parallel to the length of the bridge and placed on the edge of the bridge. This means that generally all the y-coordinates are positive, except possibly in the case of bridges on a curved section of the road.

The y-axis is perpendicular to the length of the bridge and placed so, that the x-coordinates of the majority of weighing sensors are 0. If the bridge is skewed and the sensors not mounted in a line perpendicular to bridge edge, the y axis should contain the intersection of the bridge's centreline and the line through the weighing sensors.

This placement of the y-axis means that x-coordinates can be negative or positive and implies that the peak of the influence line will be at x=0. Any deviation from this is usually due to imprecise AD/ADMP location measurement or bridge skewness.

Usually the orientation of the x-axis is such that the traffic flows in the positive direction (from left to right) on lane 1, the vehicles' speeds are positive. On a bi-directional bridge the traffic on lane 2 usually flows in the negative direction (from right to left) and the velocities are negative (although the speeds saved in the `.nswd` files are always positive).

N.B.: The negative velocities sometimes have the effect of reversing directions on those graphs in SiWIM-F and `siwim_eventview.exe`, that have time on the abscissa. E.g., when determining the boundary of influence lines on lane 2, the right boundary (the boundary with positive x-coordinate) is the first boundary that the vehicle encounters and the left boundary (the boundary with negative x-coordinate) is encountered last by the vehicle. Thus the time-dependent graph of signals displays the right influence line boundary to the left and the left influence line boundary to the right.

1.2 SiWIM-E Components

SiWIM-E is a collection of separate executable files, each of which has a specfic role. Each of the executables can accept command-line arguments. A list of command-line arguments can be listed by running the executable with the argument `-h`, e.g., `siwim_mcp.exe -h`.

All of the executables check to make sure that only a single instance is running, the second instance will refuse to run.

For advanced purposes it is possible to run multiple instances by making copies of executables and adding a numerical suffix to their names, e.g., copy `siwim_mcp.exe` to `siwim_mcp_1.exe`, `siwim_engine.exe` to `siwim_engine_1.exe`,...

In this case the SiWIM-F will need to contact the alternative executables on different TCP ports. By default the ports are 9000 and 9001. For the `_1` alternatives the ports are 10000 and 10001, for the `_2` alternatives the ports are 11000 and 11001,...

SiWIM-E is also capable of running on a linux system. It has been tested and used on both 32- and 64-bit debian distributions running on Intel-based computers, as well as on an ARM-based computer. Note that on linux computers the executables' names are without extension, e.g., `siwim_mcp` as opposed to `siwim_mcp.exe`.

1.2.1 siwim_mcp.exe

The `siwim_mcp.exe` is the executable that is started when one wishes to run a complete SiWIM-E system, either on-location or at the office.

Its job is to start the other executables and make sure that they are constantly running.

If an executable spawned by the `siwim_mcp.exe` exits unexpectedly, i.e., is not shut down by `siwim_mcp.exe`, the return code is examined.

Return code 2 (runtime error) implies that, after initialisation, some non-recoverable error had occured within the executable while the executable was running. In this case the executable is restarted.

Return code 1 (fatal startup error), however, implies that the executable had tried to start, but has ecountered an error that had prevented it from starting up, e.g., a required configuration file was missing, a TCP server could not be opened,... In this case the executable is *not* restarted, as this would result in continuous restarts and is not solvable without human intervention.

The following command-line arguments are accepted by this executable:

```
usage: siwim_mcp [options] [<file1.list> ...]
opts.: -r         Replay mode (engine and fserver get passed this switch)
       -l         Use external logger (don't spawn siwim_logger.exe)
       -f         Use external fserver (don't spawn siwim_fserver.exe)
       -e         Use external engine (don't spawn siwim_engine.exe)
       -u         Use external engine modules (don't spawn any module.exe)
       -m args    Use remote engine module(s) (engine gets passed this switch)
       -d         Debug - replay is always allowed (engine gets passed this switch)
       -h         Print this help and exit
notes:  If you supply it with a list of files, it will process the list.
rtrn.:  0         Normal termination
        1         Fatal startup error
        2         Runtime error
```

This executable does not require any configuration files, but the other executables must be located in the same directory as this one.

1.2.2 siwim_logger.exe

The `siwim_logger.exe` is the executable that's responsible for writing all log messages from other executables. There are two locations containing log files.

The global log file is located in `../log` relative to the executable's location. This log file contains messages from logger itself and fatal startup error logs. This directory also contains temperatures and voltages from the acq_ctu core.

The site log files are located in `./log` relative to the site directory. All log messages are written into files named `YYYY-MM-DD.log`. Additionally each module's log messages are written into separate log files named `<MODULE>-YYYY-MM-DD.log`.

The following command-line arguments are accepted by this executable:

```
usage: siwim_logger [options]
opts.: -s            Spawned by MCP (Ctrl-C is ignored)
       -h            Print this help and exit
rtrn.:  0            Normal termination
        1            Fatal startup error
        2            Runtime error
```

This executable requires the siwim.conf configuration file.

1.2.3 siwim_fserver.exe

The `siwim_fserver.exe` is used to isolate the rest of the SiWIM-E from the so-called SiWIM Frontend (SiWIM-F), the application used for initial setup and control of a running SiWIM-E.

The following command-line arguments are accepted by this executable:

```
usage: siwim_fserver [options]
opts.: -s           Spawned by MCP (Ctrl-C is ignored)
       -f <N>       Fill queue with N dummy messages
       -r           Replay mode - allows generation of list files
       -h           Print this help and exit
rtrn.:  0           Normal termination
        1           Fatal startup error
        2           Runtime error
```

This executable requires the siwim.conf configuration file. In replay mode it also reads rply.conf.

1.2.4 siwim_engine.exe

The `siwim_engine.exe` is the "heart" of the SiWIM-E. It is responsible for:

- loading modules defined in modules.conf,
- maintaining connections between modules and passing messages from one module to another,
- checking the correct function of modules and restarting any non-responsive modules,
- saving the processed data and
- reading saved data for reprocessing.

The following command-line arguments are accepted by this executable:

```
usage: siwim_engine [options] [<file1.list> ...]
opts.: -r           Replay mode (doesn't load any ACQ module)
       -s           Spawned by MCP (Ctrl-C is ignored)
       -m args      Use remote module(s)
       -d           Debug - replay is always allowed
       -c           Force internal (C)hain
       -h           Print this help and exit
notes:  If you supply a list of files, it will process the list.
        Arguments for option -m are 'module:ip:port[,module:ip:port]'. For each
        module listed the Engine just opens a TCP client to ip:port, primarily
        used for acq_bmc.
rtrn.:  0           Normal termination
        1           Fatal startup error
        2           Runtime error
```

This executable requires the siwim.conf configuration file. In the replay mode it reads rply.conf. And, of course, the file modules.conf is also quite useful, if anything other than being pretty is to be expected from SiWIM-E ⌣̈

1.2.4.1 Copy Protection

The release versions of `siwim_engine.exe` are protected against copying by relying on a hardware USB key. The current protection is relatively simple date-based - the expiry date of the executable is hard-coded into the executable itself. This is due to change in the next release.

On startup and each minute therafter the executable checks for the presence of the harware key. If the key is missing, or if certain data is not present on the key, the executable exits with a fatal startup error (meaning it will not get restarted by `siwim_mcp.exe`).

If the key is present and a real-time clock (RTC) is found on the key, the time from the RTC is checked agains the expiry date. If the RTC is past the expiry date, the executable exits. If the expiry date is within 1 month of the RTC, a warning is written to a log file.

If the key is present and a RTC is not found on the key, it is assumed that the key is the SiWIM-F protection key. In this case the `siwim_engine.exe` still functions, but any cores containg the string `acq_` are not loaded, meaning that the SiWIM-E can only be used for reprocessing and not for data acquisition.

1.3 Modular Design

The old MkII SiWIM was a monolithic piece of software, where an error in one part of the processing sequence would frequently (and unnecessarily) bring down the whole program.

The processing also proceeded strictly sequentially - until one piece of data was processed even if it took a long time, the system was unresponsive and unable to process further data (apart from being able to buffer the incoming raw data from the acquisition card).

The MkIII SiWIM is designed to be modular and enables parallel processing.

1.3.1 Modules

Each of the modules is responsible for a very specific part of the processing sequence. This forces one to make clear and logical boundaries between subprocesses, as the output of one subprocess must contain exactly the information necessary for the next subprocess to do its work.

The modules themselves are viewed as black boxes that take one piece of input data and one or more configuration files and produce one piece of output data.

The main part of the program is agnostic about contents of the data passed between modules and is responsible only for making sure that the output data is collected from a module, possibly saved to disk and passed along to the next module.

This division has also proven to be of great use in designing the internals of the programs, i.e., C++ classes, logical flow of data, TCP connections between various executables,... and to produce reliable and fault-tolerant software.

1.3.1.1 External Chain

Due to reliability problems with the MkII software, the MkIII software was designed so that each module would run as a separate executable. In this mode even a catastrophic error within a module would not cause problems throughout the software, but would be localised. The data flow handling part of the software can detect these failures, save the piece of data that could have potentially caused problems and restart the module.

However, there is some overhead, since the modules are connected to the main part of the software with TCP connections and each time a piece of data is passed along these connections it needs to be packed and unpacked. On the current field PCs used to perform measurements this takes around 10-15% of CPU time

1.3.1.2 Internal Chain

The reliability of the software has by now increased to such an extent, that the sorts of errors where a module would "lock-up" are, for all practical purposes, non-existent.

Thus the preferred (and, in the newest versions, the default) mode is to start modules not as an external chain of executables, but as internal chain within the main software. In the reprocessing mode this can decrease the reprocessing times by almost an order of magnitude as compared to using an external chain, since only pointers to data can be passed between modules.

The choice between the modes is made in the siwim.conf configuration file.

1.3.2 Parallel Processing

In the modular scheme, it is easy to create more than one instance of a module. The number of instances is determined by values in the parallel section of the siwim.conf file.

Each of the instances waits for the data from the preceeding module and the first idle instance can process data immediately. This has advantages in both live and replay modes (see rply.conf).

In live mode a lenghty weighing calculation, e.g., when a long train of vehicles needs to be weighed, will not prevent other, single, vehicles to be weighed quickly. In case of a module failure, SiWIM-E will wait for a certain ammount of time before restarting that module. During the waiting period the non-failed instance can process the data until the failed instance is restarted. On a typical live system the setup is something like:

```
[parallel]
camera=3
cf=2
evcam=3
il=2
speed=2
vehicle_fad=2
vspeed=2
weigh=2
```

In the replay mode the parallelism takes the advantage of multi-core PCs to speed up reprocessing significantly. Speeds of over 250 processed events per second have been achieved on an Intel i7 system with 8 cores, running 64-bit versions of linux and SiWIM-E and with the following setup:

```
[parallel]
REPLAY=16
bogey=8
carriage=8
cf=8
daf=1
espeed=8
fad_filter=8
filter2=8
il=8
speed=8
stats=8
strains=8
vehicle=8
vehicle_fad=16
vspeed=8
weigh=32
```

N.B.: One should *never* run more than one instance of modules that handle one-second long pieces of data, i.e., modules up to and including tsplit. Such a setup would cause the one-second pieces of data to get jumbled up and the resulting data would be useless (and original order unrecoverable in the reprocessing phase). There are also some special cores that need to be run as single instances, but they are not a part of the standard processing chain.

1.4 Standard Cores

These cores are present in almost every processing chain.

1.4.1 acq_bmc

This core sets up BMC data acquisition card, starts acquisition and reads one-second long blocks of data. It also performs the so-called zeroing, where each channel is checked periodically whether it has drifted and the drift corrected, if it has.

The zeroing algorithm inspects one channel per acquired block and determimes whether that channel can be zeroed in the next second. The inspected channel is rotated among the acquired signals.

The algorithm calculates the Δ and the average value for that channel. If the Δ is below a specified value and the average is outside the specified bounds, it applies a zeroing signal for a specified length of time to the digital port of the BMC data acquisition card.

If a channel is zeroed, its `invalid` flag is set for the second in which it is zeroed and for the next second.

The core also intercepts temperature and voltage data from acq_ctu, saves it, and adds it to the acquisition data. If there has been no temperature and voltage data in the last 2 minutes, nothing is added to the acquisition data.

This core uses acq_bmc.conf file.

1.4.2 acq_ctu

This core connects to a CTU module over a COM port. It reads temperatures and voltages and sends the data along the chain, where it is intercepted by the acq_bmc core.

The core also intercepts confirmation from acq_vpn core that the VPN server is reachable and turns on the VPN LED on the CTU, if it is.

This core uses acq_ctu.conf file.

1.4.3 acq_vpn

Generally a SiWIM sistem is connected to a VPN server. The VPN connection is used to control the system remotely and to occasionaly download some data from the system.

The core periodically checks the reachability of the VPN server and sends a confirmation along the chain, where it is intercepted by the acq_ctu core.

This core uses acq_vpn.conf file.

1.4.4 camera

This core is used to capture photos using an external camera. It has two operation modes. The selection of mode is made automatically by the presence of a list of detected vehicles in the event that is passed to this core.

- If such a list exists, the core saves photos of individual vehicles. A module running core in this mode must be positioned after either vehicle_ad or vehicle_fad cores.

- If the list does not exists, the core saves photos of an entire event. In this case the module containing running this core must be positioned after tsplit core and before either vehicle_ad or vehicle_fad cores.

This core uses camera.conf file.

1.4.5 cf

This core reads *bare* weights, produced by the m1weigh core and multiplies them by a calibration factor. Each lane has its own calibration factor and one can specify separate calibration factors for different subclasses of vehicles.

After multiplication with a "static" calibration factor, various compensations and tweaks can be performed to account for dynamic effects. The compensations are listed in the order of application to calibrated weights:

1.4.5.1 Individual Axle Factors

Each axle's calibrated weight can be multiplied by a fixed factor.

1.4.5.2 GVW Speed Compensation

Vehicle speed can be used to compensate the GVW. The compensation function is defined by at least two pairs (v_i, f_i), $i = 1 \dots N$. The function is:

- linearly interpolated for $v_1 \leq v \leq v_N$ and
- extended with f_1 for $v < v_1$ and with f_N for $v > v_N$.

The GVW is divided by the value of the function.

1.4.5.3 GVW Temperature Compensation

One of the 6 temperatures read by the SiWIM-E can be used to compensate the GVW. The compensation function is defined by at least two pairs (T_i, f_i), $i = 1 \dots N$. The function is:

- linearly interpolated for $T_1 \leq T \leq T_N$ and
- extended with f_1 for $T < T_1$ and with f_N for $T > T_N$.

The GVW is divided by the value of the function.

If the temperature chosen for the compensation is not present in the event, this compensation is not performed and a warning flag is written to the `.nswd` file.

1.4.5.4 W1 Redistribution

A portion of the weight of the first axle can be redistributed to either the second axle or all of the remaining axles. In the latter case, the weight is distributed proportionally to the remaining axles' weights, e.g., if 1/3 of the weight of the first axle were to be distributed in a truck with axle loads 9 kN, 10 kN and 5 kN, the resulting axle loads would be 6 kN, 12 kN and 6 kN.

1.4.5.4.1 Fixed W1 Redistribution

A fixed portion, e.g., 10%, can be redistributed among the axles.

1.4.5.4.2 Speed-Based W1 Redistribution

In addition to this, the vehicle speed can be used to determine the portion to be distributed. The compensation function is defined by at least two pairs (v_i, f_i), $i = 1 \dots N$. The function is:

- linearly interpolated for $v_1 \leq v \leq v_N$ and
- extended with f_1 for $v < v_1$ and with f_N for $v > v_N$.

The distribution portion is is divided by the value of the function.

1.4.5.5 Negative Weights

If one of the calibrated weights becomes negative as a result of these compensations, all the compensations are thrown away and a warning flag is written to the `.nswd` file.

If any of the bare weights has been negative, all the calibrated axle weights (and thus the GVW) are set to 0 and a warning flag written to the `.nswd` file. This behaviour can be overriden for special applications.

This core uses cf.conf, cf.filter and vehicle_classes.conf files.

1.4.6 fftf

This core is used for Fast Fourier Transform filtering. The global section of the configuration file specifies the combination of one or more of the three filters to be used:

- Low-pass filter, that passes only frequencies below some specified value
- High-pass filter, that passes only frequencies above some specified value
- Notch filter, that cuts out a range of frequencies

This core can also operate in the Power Spectrum Density mode, which can be used to determine the eigenfrequencies of a bridge. It operates on blocks of data supplied by the PSDP mode of the tsplit core, calculates PSD of each block and calculates either an average of N PSDs (in which case the data is output once after N blocks has been processed) or a moving average (in which case the data is output for each block after N blocks have been accumulated).

The files are output in directories `fftf/[ma|av]-<res>-<N>`, where `[ma|av]` depends on whether the moving average or ordinary average was used, `<res>` is the inverse resolution of PSDs (this depends on the length of processed blocks) and `<N>` is the number of averaged PSDs.

The names of files have the format `YYYY-MM-DD-HH-MM-SS-mmm.psd`, where `YYYY-MM-DD-HH-MM-SS-mmm` is the timestamp of the block.

The files are ASCII tab-delimited files suitable for opening in Excel. The first line contains data labels for easy graphing. If using the acq_mni, then the channel labels can be defined in the ACQ_MNI_CONF_ROLES_SECTION, whereas for acq_bmc and acq_ni the channel labels are generic `ch<N>`.

The columns are:

- 1: Freqency in Hz (labelled `f [Hz]`)
- 2: First PSD (labelled, e.g., `ch1`)
- ...

Note that PSDs can also calculated from sums of channels. In this case the labels for those columns will be, e.g., `ch1,ch2,ch3`.

This core uses fftf.conf file.

1.4.7 kmil

This core is used to calculate influence lines. In order to do so, the same procedure is used as for weighing where the influence lines, multiplied by the axle weights, are fit to the measured signals. This is done by minimising

$$\chi^2 = \sum_j \left[\frac{s(t_j)}{N} - \sum_i w_i I\left(v_i(t_j - t_i)\right) \right]^2 ,$$

where $s(t_j)$ are the sums of values of signal from N sensors at times t_j, w_i are the axle weights, $I(x)$ is the influence line, and position, x, is calculated from the vehicle velocity, v_i, and arrival times of individual axles, t_i.

The difference is that, when weighing, the influence line is known, whereas when calculating it, the influence line is also an unknown and is calculated as a part of the process. The modelled influence line is defined with a cubic spline for which some of the points are fixed (supports), some are "forced" to describe the peak of the IL, whose radius depends on the bridge thickness, and some are varied to obtain the best fit. Since the system does not depend linearly on the unknowns, the calculation method is inherently non-linear. SiWIM uses `Powell's minimisation` to solve the problem. A number of parameters are required to describe the influence line and have to be provided before the calculation:

- Coordinates of the start and end of the influence line.
- Coordinates of the bridge supports where the value of influence line should be equal to zero.
- Thickness of the superstructure that dictates how the peak of the IL is rounded.
- Number and location of the optimisation points, in which the algorithm searches for the match; typically on optimisation point per bridge span is defined.
- Channels from which to calculate the IL.
- Vehicle classes from which to calculate the IL; usually vehicles with long axle spacings are selected;
- Some additional parameters that specify whether for example, the positions of the supports are optimised, position and location of the peak are optimised, etc.

Even though the m1weigh core can use one IL per strip, the kmil core cannot yet calculate IL per strip automatically. Instead, separate runs, one for each channel list, i.e., one per strip, are needed.

In order to compensate for the signal drift and/or ST remnant deformation, global and local linear trends can be subtracted from the signals prior to the summation.

This core uses kmil.conf, mp_factors.conf, mp_time_factors.conf and vehicle_classes.conf files.

1.4.8 m1weigh

This core weighs vehicles by minimising

$$\sum_j \left[\frac{s(t_j)}{N} - \sum_i w_i I\left(v_i(t_j - t_i)\right) \right]^2 ,$$

where $s(t_j)$ are the sums of values of signal from N sensors at times t_j, w_i are the axle weights, $I(x)$ is the influence line, and position, x, is calculated from the vehicle velocity, v_i, and arrival times of individual axles, t_i.

Since the fitted function depends linearly on the fitted weights w_i, a relatively simple linear algorithm, using `Singular Value Decomposition`, can be used.

In order to compensate for the signal drift and/or ST remnant deformation, global and local linear trends can be subtracted from the signals prior to the summation.

1.4.8.1 Automatic Timestep

By default the fitted function is evaluated at each measured sample. Optionally one can choose to evaluate it only at points separated by a specified length. The quotient between this length and the maximum speed in the event is used as the timestep between function evaluations. This can have an enormous influence on calculation speed in RailWIM systems.

This core uses m1weigh.conf, .dists, .il, mp_factors.conf, mp_time_factors.conf, reconstruct.conf, strips.conf and vehicle_classes.conf files.

1.4.9 offset

The signals' zero levels, i.e., levels when there is no traffic on the bridge, slowly (or quickly) drift with time due to several influences:

- temperature changes and different thermal expansion coefficients of materials used in sensor construction and bridge materials,
- current leakage in the amplifiers.

To correct for this effect, this core uses some predefined value of maximum signal Δ to determine whether the bridge is truly devoid of traffic and calculates the average signal level for each channel, if it is. The data for average values is then added to the one-second blocks of data and is retained when the blocks are joined into events. The data is used by practically all cores downstream from this one.

See also the math core.

This core uses offset.conf file.

1.4.10 speed

This core calculates the vehicle's speed by using signals from pairs of Speed Measurement Point (SMP) sensors. Cross-correlation betwen the two signals is calculated to determine the difference between arrival times and the speed is calculated based on the known distance between the SMPs.

Speeds on each lane can be calculated from several pairs of SMPs (where one SMP can appear in more than one pair). In this case the speeds from pairs of SMPs are averaged to obtain the final speed estimate for that lane.

1.4.10.1 Envelope Subtraction

SMPs mounted on beam webs typically exhibit a marked negative overall response, which can cause a large contribution to the cross-correlation, masking the peak that corresponds to the correct speed. To remove this unwanted part of the response, each SMP can have an approximation to the negative envelope subtracted from it.

The signal is scanned for the location of the minimum value and, once found, a triangle is constructed through the minimum location and the endpoints. The triangle is then subtracted from the signal.

1.4.10.2 Event Mode

By default the speed core calculates one speed for each lane using signal from the entire event, so all vehicles in one lane have the same speed. This is a valid assumption on short bridges and short vehicles.

1.4.10.3 Vehicle Mode

For RailWIM this assumption has been shown to be invalid. Railway trains can be several hundred metres long. The speed difference between the individual carriages can be several tens of percents, decreasing weighing accuracy by a similar ammount. Thus the core can be directed to calculate speeds of individual vehicles.

Vehicle speeds are calculated by using only those intervals of the SMP signals in which the vehicle is on the bridge. Currently only one pair of SMPs can be used for vehicle speed calculation. The vehicles must have already been identified, so in this mode the core must be placed in a separate module that comes after the vehicle_ad or vehicle_fad cores in the procesing chain.

1.4.10.4 Acceleration Mode

In the vehicle mode it is also possible to take into account vehicle acceleration by calculating speeds for individual axles, fitting a line through the calculated speeds and reading the average speed and acceleration from the fitted line. This has only recently been introduced and not yet fully tested.

This core uses speed.conf file.

1.4.11 tsplit

This core has the job of concatenating one-second blocks of data and splitting the concatenated data into so-called events that contain signals from one or more vehicles whose influence on the bridge could possibly overlap.

In each of the three modes of operation this core checks for data that is above or below certain predefined leveles.

- By default the core identifies the triggered intervals and determines the time between them. If the time interval is longer than a predefined minimum idle time, the triggered sections are considered as not influencing one another and are split into separate events. If the interval is shorter, the data is considered as one event. Each event contains at least minimum idle time interval of signal before and after the triggered part.
- In the complement trigger mode, the core passess along all data outside the triggered intervals. This can be useful for determining the signal noise levels. The events in this case can be very long - as long as the interval between passages of two vechicles on the bridge.
- The power spectrum density preprocessor mode operates similarly to the complement trigger mode, except that the events are always of a certain predefined length.
- The block mode looks at each one-second block of data and determines whether each block has triggered. If so, it constructs a event of a certain minimum length with a certain number of prepended blocks. If the trigger is repeated:
  - If it's within the event length, nothing changes,
  - If it's beyond the event lenght, the event lenght is increased
- Block mode also has an option to make blocks of certain lenght unconditionally

In all cases it is possible to save a text representation of the event, by using the option TSPLIT_CONF_GLOBAL_SAVE_TXT_FILES_KEY. The names of files have the format `YYYY-MM-DD-HH-MM-SS-mmm.txt`, where `YYYY-MM-DD-HH-MM-SS-mmm` is the timestamp of the event.

This core uses tsplit.conf file.

1.4.12 vehicle_ad

This core is used to detect vehicles based on axle detectors (ADs) mounted on the pavement. These detectors are expected to provide digital signals.

This core uses vehicle_ad.conf and vehicle_classes.conf file.

1.4.13 vehicle_fad

This core is used to detect vehicles based on axle detector measurement point (ADMP) sensors mounted on the structure, not on the pavement. This core is by far the hardest to set up, with many of the parameters interacting in non-predictable and hard to understand ways. In future a probabilistic approach should be used to lighten the operator load.

Axle positions are constructed from peaks in the ADMP strain signal. Axle distances are calculated using times between axles and the event speed for that lane. After axles are found, they are used to construct vehicles.

Ideally the axle detector channel would produce sharp peaks of equal height for each passing axle. This would enable the usage of a single threshold to determine axle positions as those regions where the signal rises above that threshold. This the case for signals from axle detectors mounted on the pavement.

However, when instrumenting a bridge for FAD installation, the signals, even after extensive conditioning and filtering, rarely produce peaks which would enable the simple threshold method. Several enhancements to the basic method are available, presented roughly in the order of application to the acquired ADMP signals.

1.4.13.1 Multiple ADMPs

On shorter slab bridges, where individual axles can be seen in the weighing sensors even if they are installed at mid-span, several sensors can be addressed as axle detectors. The software detects axles from the signal with the highest peak-to peak value, assuming that the sensor under the wheels of the specific vehicle shall provide the highest peaks.

1.4.13.2 Multiple Channels per ADMP

Alternatively, both speed measurement sensors, spaced a few meters apart (or even additional channels) can be combined, after being multiplied with predefined factors, into a joint axle detection signal.

In the basic mode the sensors used in the sum must have the same longitudinal position, otherwise the average response will show multiple peaks for each axle (one from each sensor).

To circumvent this, it is possible to use the auto-shift method. In this case case the first channel in the list of channels is taken as a reference. After multiplying the values and before summing up the values into an average, a cross-correlation between the reference channel and each of the other channels is calculated and the other channels are shifted in time so the correlation with the reference channel is the greatest, hopefully aligning the peaks.

1.4.13.3 Envelope Subtraction

ADMPs mounted on beam webs typically exhibit a marked negative overall response. To remove this unwanted part of the response, envelope subtraction can be applied.

The signal is scanned from left and right to find points L and R, where it drops below a certain predefined level. Within the interval $[L, R]$ the location of the minimum of the signal is found and a trangle constructed, with verteces at points L, R and at the location of the minimum. This triangle is then subtracted from the signal, effectively raising the signal and increasing the axle peaks' amplitudes.

If the procedure finds more than one such interval, and the intervals are more than a certain distance (comparable with the maximum axle distance) apart, a triangle is constructed in each of the intervals. If the intervals are less than this distance apart, they are joined into one.

1.4.13.4 Signal Conditioning

The signals are conditioned by averaging them over two different typical lengths. The first average is on the order of a metre or less and has the effect of reducing high-frequency noise while still retaining axle peaks. The second average is with a longer length, which makes it follow the general shape of the signal. The difference between the first and the second averages usually exhibits pronounced peaks at axle positions.

1.4.13.4.1 Averaging

For each of the averages, three different methods of averaging are available.

By default the averaging is performed using a square window, where a certain number of samples to the left and right of the central point are averaged and this procedure is performed for each point of the signal.

Instead of considering all points equally, the triangular window assigns the points closer to the central point a higher weight and points further away a lower weigh, with the weights dropping off linearly with the distance from the central point. This has the effect of enhancing peaks at the expense of admitting more high-frequency noise. The triangular window has proven to be particularly useful for the second average, as it typically raises the negative part of the signal and thus pronounces the peaks.

And, finally, one can choose the `Savitzky-Golay` smoothing filter, which replaces the averaging with a polynomial fit over a certain number of points to the left and right of the central point. Unlike averaging this preserves the heights of peaks. The axle peaks shall remain more pronounced, while the noise still gets suppressed. Savitzky-Golay smoothing has shown to be useful for the first average.

1.4.13.4.2 Alternate Difference

It is possible to use a different method of obtaining the difference. The shorter average is first subtracted from the original signal and this difference is then averaged over a longer length. This averaged difference is then subtracted from the shorter average.

The advantage is that the shorter average follows the signal quite closely, except in the regions of axles. By subtracting the shorter average from the signal, the axles are subtracted from the signal. In the original averaging, the axles, especially if they are very pronounced, tend to make the longer average deviate from the general shape of the signal and generally lower the axle peaks.

1.4.13.5 Two-Phase Threshold

The values of thresholds used in locating axles are based on the maximum of the signal. If the lower value is set too low, spurious axles may appear in the heavier vehicles, whereas if it set too high, it may miss light vehicles.

The two-phase threshold, can be used to distinguish heavy and light vehicles in the same events. The signal is first split into regions based on a simple threshold, which can be set relatively low - just above the noise. If two regions are separated by more than the maximum axle distance, they are considered as containing two separate vehicles and each of the regions is processed separately.

1.4.13.6 Locating axles

Once the signal has been conditioned and possibly split into independent regions, it is searched for peaks representing axles. There are two methods available.

1.4.13.6.1 ATS Algorithm

The algorithm first uses a predefined starting threshold V_s. Tentative axle positions are defined at the midpoints of the intervals where the signal rises above that threshold. The next threshold is obtained by incrementing the threshold by some value ΔV and new tentative axle positions obtained. This procedure is repeated until the threshold sweep reaches V_e.

Once the sweep is finished, the tentative axle positions are examined. If an axle appears in less than a certain cutoff number of cases, it is discarded, otherwise it is promoted to a definite axle.

1.4.13.6.2 Qualified Maximum/Minimum

QMM (Qualified Maximum/Minimum) algorithm may be used as an alternative to the ATS algorithm.

Instead of performing a threshold sweep, the conditioned signal is checked for so-called qualified maxima. For a maximum M to be considered as qualified there must exist one point to the left of the maximum and one point to the right of the maximum, where both points are at least some Δ below the maximum. In other words, the signal must drop to below $M - \Delta$ on both sides of the maximum. This has the effect of selecting only pronounced peaks while discarding noise.

Once the qualified maxima are identified, only those are retained as axles, that are above a certain percentage of the maximum of the conditioned signal.

1.4.13.7 Axle Substitution

If the bridge superstructure is thick and stiff, as is it usually is for railway bridges with the ballast between the rails and the bridge, the peaks of the closely spaced axles are smeared. Even with axles in a bogie spaced over 2m apart it is impossible to distinguish them in the signal. This effect is even more pronounced as the span increases in length.

It is possible for each identified axle to be unconditionally replaced with two axles at a certain predetermined distance apart. This method has a limitation that the double axle spacing needs to be known in advance and that this spacing needs to be entered as a part of the procedure.

This option is not meant for general use. Instead it is used as a prelude to calculating an influence line from, e.g., passenger trains with known axle spacings. Once the influence line is obtained, this option is turned off and the reconstuction available in the m1weigh core turned on to construct axles from smeared peaks.

1.4.13.8 Carriage Mode

In RailWIM the axle spacings have a completely different pattern than those in road applications. The distances between bogies within a single carriage are longer than distances between bogies from two consecutive cariages. The standard vehicle construction algorithm developed for road usage cannot cope with this.

To circumvent this, the carriage mode can be used. In this mode the axles are joined into vehicles in two passes. First they are joined into 1-4 axle bogies, then consecutive bogies are joined into carraiges.

N.B.: This procedure will fail in the case of passenger trains, where consecutive carriages can share a bogie and where there can be an odd number of bogies. The general algorithm for joining axles into carriages is non-trivial and will require further study.

See also fix_spacings core for further RailWIM-related enhancements.

This core uses vehicle_fad.conf and vehicle_classes.conf file.

1.5 Extra Cores

Some cores are used only for special purposes or have been used in development phase.

1.5.1 acq_bmc_alive

This core is used when copy protection is active, but the senselock key has not been found. In this case the SiWIM-E discards all the information in modules.conf file and loads just one module with this core.

The only job of this core is to toggle the alive LED so that the SiWIM system does not reboot itself automatically.

This core does not use any conf file.

1.5.2 acq_mni

This core can use Multiple National Instruments devices for data acquisition. This core neither toggles alive LED, nor does it perform zeroing, so it's not useful for SiWIM systems.

This core uses acq_mni.conf file.

1.5.3 acq_ni

This core is almost identical to the acq_bmc core, except that it uses National Instruments instead of BMC data acquisition device. The internal logic is the same and only a few parameters are added for this core as compared to the acq_bmc core.

This core uses acq_ni.conf file.

1.5.4 autosplit

The core is an experimental core that implements a "histogram" method of splitting data and that could possibly replace the tsplit core.

The idea is that histograms of signal amplitudes should have a trimodal distribution. The first peak is at very low amplitudes, which correspond to noise, The second peak is at higher amplitudes and corresponds to cars. The last peak at even higher amplitudes corresponds to trucks, even though this peak is not very pronounced.

By identifying the trough between the first and second peaks one could automatically determine the value above which there is something on the bridge and use that value to produce an event.

Currently all the parameters are hard-coded into the source, as the algorithm is in a very early stage of testing. They will be added to this file if/when as the algorithm stabilises and is actually used.

This core uses autosplit.conf file.

1.5.5 cclear

This core is used to clear some channels from data, A typical usage is in the so-called double-chain setups, where on system is used to acquire data for two physically separated bridges (e.g., on a highway).

In this case it is deleterious to weigh vehicles with combined signals from all weighing sensors. Instead the processing chain is split and each bridge processed in its own branch of the chain. To remove duplicate data, this core is used to clear irrelevant data for each of the subchains.

See also csplit.

This core uses cclear.conf file.

1.5.6 csplit

This core used to pass only some channels from data, A typical usage is in the so-called double-chain setups, where on system is used to acquire data for two physically separated bridges (e.g., on a highway).

In this case it is deleterious to weigh vehicles with combined signals from all weighing sensors. Instead the processing chain is split and each bridge processed in its own branch of the chain. To remove duplicate data, this core is used to pass only the relevant data to each of the subchains.

See also cclear.

This core uses csplit.conf file.

1.5.7 daf

This core is used to calculate the so-called Dynamic Amplification Factor (DAF), which is a measure of the dynamic response of the bridge - a ratio between the dynamic and static responses. The core produces an ASCII tab-delimited file `DAF_YYYY-MM-DD-HH-MM-SS-mmm.txt`, where the timestamp is the timestamp of the start of the calculation. One line is written for each interval in which there is something on the bridge. The line contains the following fields:

1. Timestamp of the vehicle in one of the formats:
    - `YYYY-MM-DD-HH-MM-SS-mmm`,
    - `YYYY,MM,DD,HH,MM,SS,mmm` or
    - `YYYY-MM-DDTHH:MM:SS.mmmZ`,
2. 1 if this was an MP event, 0 otherwise
3. Vehicle class
4. Calibrated GVW of the truck in kN
5. Maximum signal in V

6. DAF1, which is the ratio of maximum measured signal to maximum reconstructed signal (i.e., ratio of "sig" and "sum" channels in "weigh" diagnostics
7. DAF2, which is the ratio of maximum measured signal to maximum signal that has been filtered with a low-pass FFT filter with zero width
8. DAF3, which is the ratio of maximum measured signal to maximum signal that has been filtered with a low-pass FFT filter with a non-zero width

This core can also work in the DAF3 optimisation mode, in which the cutoff frequency and width are determined with Powell's minimisation of the sum of square of difference between the signal and the sum of ILs. In this case another ASCII file is produced, named `OPT_YYYY-MM-DD-HH-MM-SS-mmm.txt`, where the timestamp is the timestamp of the start of the calculation. Each line of the file contains:

1. Timestamp of the vehicle in one of the formats:
    - `YYYY-MM-DD-HH-MM-SS-mmm`,
    - `YYYY,MM,DD,HH,MM,SS,mmm` or
    - `YYYY-MM-DDTHH:MM:SS.mmmZ`,
2. 1 if this was an MP event, 0 otherwise
3. Vehicle class
4. Maximum signal in V
5. Calibrated GVW of the vehicle in kN
6. Optimised frequency in Hz
7. Optimised width in Hz

This core uses daf.conf file.

1.5.8 exp

The experimental core is a core used for various experiments and not usually used in production systems. The main section of the exp.conf file selects the operation mode and each mode has its own section.

1.5.8.1 Weigh Amplitude Mode

This mode was used for TRIMM project to try and correlate weights of passing vehicles with acoustic emmisions within the concrete. In this mode the core calculates the sum of selected channels and produces several output files.

- `<PREFIX>_max_ch.list`, an ASCII tab-delimited file with the fields:
    - 1: Name of event in the format `YYYY-MM-DD-HH-MM-SS-mmm.event`
    - 2: Maximum value of the first channel in volts
    -
    - N+1: Maximum value of the N-th channel in volts
- `<PREFIX>_sorted_by_amplitude.html`, a list of events sorted in descending order by the weighing amplitude. This is an HTML file containing a table where first column is the amplitude in volts and the second column a link to the event.
- `<PREFIX>_sorted_by_amplitude.list`, a list of events sorted in descending order by the weighing amplitude. This is an tab-delimited text file, where the first column is the amplitude in volts and the second column is the event timestamp.

- `<PREFIX>_sorted_by_event.html`, a list of events sorted in ascending order by event timestamp. This is an HTML file containing a table where first column is the amplitude in volts and the second column a link to the event.
- `<PREFIX>_sorted_by_event.list`, a list of events sorted in ascending order by event timestamp. This is an tab-delimited text file, where the first column is the amplitude in volts and the second column is the event timestamp.

These files are all written to directory `/usr/weigh_amplitude` as opposed to the normal directory within the data hierarchy.

1.5.8.2 Extract Temp and Volt Mode

This mode is used to extract the temperatures and voltage from events in cases where `ctu_*` files are not available. The core extracts data and writes one line for each event into an ASCII tab-delimited file `temp_and_volt.txt`. The fields in the file are:

1. Timestamp of the event in one of the formats:
   - `YYYY-MM-DD-HH-MM-SS-mmm`,
   - `YYYY-MM-DDTHH:MM:SS.mmmZ`,
2. T_1
3. T_2
4. T_3
5. T_4
6. T_5
7. T_i
8. U_b

1.5.9 fix_spacings

This core is used for RailWIM to adjust the measured axle spacings to the known axle spacings of railway locomotives and carraiges. There are two adjustements availabe. In both cases the axle distances are first checked against known spacings to find the closest match.

When spacing correction is enabled and, if the measured and predefined spacings differ by less than some specified ammount, the measured spacings are changed to correspond exactly to the predefined ones.

When speed correction is enabled, a speed which would minimise the differences between the measured and predefined spacings is calculated. If the change in speed is less than some specified ammount, the speed of the carriage is changed.

If both corrections are enabled, the speed correction is performed first, then the spacing correction is applied.

This core uses fix_spacings.conf file.

1.5.10 gpsrecv

This core is used for reception of GPS data for automatic calibration procedure. with a GPS receiver in a calibration vehicle. The GPS coordinates are continuously sent to the system, which can then automatically mark certain vehicles as being calibration vehicles. This core is still in the testing phase, so it is not documented.

This core uses gpsrecv.conf file.

1.5.11 kmweigh

This core implements a non-linear weighing algorithm. Currently it is not used and not documented because:

- it has only been partially tested,
- it contains known bugs, which were fixed in the linear weighing core m1weigh and
- the linear and non-linear algorithms are due to be integrated in a common core, where the user will have the option of either manually selecting the linear or non-linear mode, or the mode will be automatically be chosen based on other parameters.

1.5.12 m1il

This core was used to calculate influence line (IL). It is an obsolete core that is due to be removed.

1.5.13 math

This core can perform various mathematical operations on data and can generate signal-based statistics.

By setting the MATH_CONF_GLOBAL_SAVE_TSV_KEY option in the global section of the math.conf file, this core can also save ASCII tab-delimited files of acquisition data after is has finished processing data (and, of course, if no processing has been done) and it can downsample the data before saving it

The types of operations and the sequence is determned by the sections of the configuration file. E.g., if sections `1_invert` and `2_multiply_all` were present (with the requisite parameters inside each section) then the first operation would be to invert one of more channels and the second operation would multiply all channels with a factor.

The operations currently implemented are:

- invert one or more channels, which is useful if some of sensors have been wired in reverse, causing the voltages to drop instead of rise when a vehicle passes
- multiply one or more channels with individual factors
- multiply all channels with a factor
- relabel channels, which can be useful if the sensor roles have been changed during the measurement or for special setups
- math_conf_rms that calculates RMS values for event data
- write signal statistics that analyses event signals and writes several files, depending on the options.

All files produced by the write signal statistics are ASCII tab/comma-separated files. Files that begin with `ALL_` contain the data from all events, while those that begin with `VALID_` contain only data from events that contain no invalid channels. The following files are produced by this option.

`invalid_channels.txt` contains a list of invalid channels in the event. The fields are:

- 1..7: Event timestamp in the format `YYYY,MM,DD,HH,MM,SS,mmm` or `YYYY-MM-DDTHH:MM:S↩ S.mmmZ`
- 8, ...: List of invalid channels

`ALL_delta.txt` and `VALID_delta.txt` contain the channel Δ values. The fields are:

- 1..7: Event timestamp in the format `YYYY,MM,DD,HH,MM,SS,mmm` or `YYYY-MM-DDTHH:MM:SS.mmmZ`
- 8: Number of channels in the event
- 9: Channel with the highest Δ
- 10: The highest Δ
- 11 ... 11+N: Channel Δ values for each of the channels
- 12+N ... 17+N: Temperatures T1...T5 and Ti

`ALL_maxminavg.txt` and `VALID_maxminavg.txt` contain the channel's maximum, minimum and average values. The fields are:

- 1..7: Event timestamp in the format `YYYY,MM,DD,HH,MM,SS,mmm` or `YYYY-MM-DDTHH:MM:SS.mmmZ`
- 8: Number of channels in the event
- 9, 10, 11: The first channel's maximum, minimum and average
- 12, 13, 14: The second channel's maximum, minimum and average
- ...

`ALL_offset.txt` and `VALID_offset.txt` contain the channel's offset. The fields are:

- 1..7: Event timestamp in the format YYYY,MM,DD,HH,MM,SS,mmm
- 8: Number of channels in the event
- 9, ...: Offset for each of the channels

This core uses math.conf file.

1.5.14 smi

This core is an obsolete core used as an interface between SiWIM and MATLAB. It was developed in the early stages of the BridgeMon project, but never used and is due to be removed.

1.5.15 test

This core was used for development to test module recovery. It has the ability to swallow data and to throw an exception, die, or cause a timeout at a certain probability. Used without any parameters it simply passes the data unchanged.

1.6 Extra Configuration and Other Files

Some files are not core-specific, but are instead used by more cores and/or SiWIM-F. Others are core-specific, but deserve a more detailed explanation.

1.6.1 cf.filter

The purpose of the `cf.filter` file is to allow only specified vehicles to be written to `.nswd` files by the cf core. This is useful when processing a set of calibration events and one wishes to write only calibration vehicles, not any other vehicles.

The file is simply a list of vehicle timestamps in the format `YYYY-MM-DD-HH-MM-SS-mmm`, against which the vehicle timestamp is checked before it is written to the `.nswd` file.

The lines need be only as specific so as to select the specfied vehicle. For example, if there's only one vehicle within a certain second, the `-mmm` part can be omitted, if the selected vehicle is the only one within one tenth of a second, the `-mmm` part can be shortened to `-m`, etc.

N.B.: If the vehicle_fad parameters are changed for reprocessing, the vehicles' timestamps may change slightly and may no longer be selected by the filter. It is thus beneficial to use as short a filter line as possible.

1.6.2 .dists

The transverse distributions are written to binary and text files, depending on the needs.

Currently the transverse distributions are calculated by integrating absolute values of signals, divided by the lenght of the interval in seconds. For binary files the scaling with length is irrelevant, since the values are always normalised so that the sum is 1. But when using text files, this scaling allows comparison between values for short and long intervals.

The binary files are used to collect data that helps determine whether one or more signals read from strain transducers (STs) needs to be multiplied by a constant factor to obtain the expected transverse distribution of signals.

For example, on an infinitely long bridge with an infinitely high torsional stiffness, the responses from all the STs should be exactly the same, i.e., the tranverse distribution would be a straight line. If one or more STs were not proprely mounted, or there were some local effect (e.g., the presence of a crack in the concrete), the responses would not be all the same. In this case the factors, with which the signals would have to be multiplied to obtain the expected response, would be straightforward to determine.

Generally the transverse response is not straight line. To determine the actual response, the system gathers statistics of responses from each sensor and writes them to the `m1weigh.dists` file. Separate statistics are gathered for each lane and the system only adds data to statistics when there is no cross-lane multiple presence and no axles have negative weights.

After a suitable number of samples have been gathered, the `siwim_distsview.exe` program can be used to manually determine the correct factors and write them to the mp_factors.conf file. There is also a MATLAB method that can automatically determine the optimum mp_factors.conf and calculate the vehicle transverse position based on the sensor response, but it has not yet been implemented in SiWIM-E.

The `m1weigh.dists` file is necessary when the so-called strips are used, to define the influence of a vehicle on signals from groups of sensors.

1.6.2.1 Binary Files

`.dists` files are binary files, viewable with `siwim_distsview.exe` program, which is included as one of the utility programs. The name of the file depends on its provenance/use:

- When calculating average distributions, the m1weigh core writes them to file `average.dists`
- Transverse distributions for calculation interval can also be written to a separate file named `YY↩ YY-MM-DD-HH-MM-SS-mmm.dists`, where the name reflects the timestamp of the initial sample of the interval.
- If the m1weigh core finds a file named `m1weigh.dists` in the site configuration directory, it read and uses it, if the use of strips is enabled.

1.6.2.2 Text Files

The individial distributions can be written to text files The name is based on the timestamp of the initial sample of the calculation interval, so there can be more than one output file per event. The distributions are written to one or two files:

- dists.txt for all distributions, regardless of multiple presence and
- dists_lane<L>.txt for no cross-lane multiple presence distribution on lane L.

The ASCII tab/comma-delimited files contain the fields:

- 1..7: Vehicle timestamp in the format YYYY,MM,DD,HH,MM,SS,mmm,
- 8: 1 for a multiple presence event and 0 otherwise,
- 9: lane
- 10: sum of all contributions
- 11..: contribution of each sensor

1.6.3 .il

The .il files are binary files containing the influence lines (ILs), which can be read by siwim_ilview.exe program included as one of the utility programs, or by SiWIM-F. Within the file the ILs are represented by ordinates spaced 1cm apart. The names of files vary according to their provenance/use.

The files generated by the kmil core are named YYYY-MM-DD-HH-MM-SS-mmm_I_L.il, where YYYY-MM-DD-HH-MM-SS-mmm is the timestamp of the event from which the IL was calculated, I is the index of the interval within the event and L is the lane for which the IL was calculated.

The naming scheme of the generated ILs is due to be changed with the addition of per-strip (see strips.conf) and/or per-sensor IL generation. An additional field S will be added and the complete name will be YYYY-MM-DD-HH-MM-SS-mmm_I_L_S.il

The IL files are used by the m1weigh core. The core chooses the most specific IL available. The names of IL files, ordered from the most to the least specific IL, are:

- m1weigh_L_S.il for the IL for strip S for the vehicle in the L-th lane
- m1weigh_L.il for the IL for a vehicle in the L-th lane
- m1weigh.il for the default IL for all lanes and strips

If none of these exist, the m1weigh core reads parameters from the default_il section of the m1weigh.conf file and constructs a simple triangular IL with the boundaries defined by the bridge length and the peak at x=0.

1.6.4 modules.conf

SiWIM-E sets up an arborescence of modules, described in the modules.conf file, located in directory ./conf relative to the site directory. If this file is not present, SiWIM-E starts, but remains dormant until restarted with a valid file.

1.6.5 mp_factors.conf

The mp_factors.conf file is used to define factors with which the values produced by individual strain transducers are multiplied to correct for:

- improper mounting
- local effects

See also .dists.

1.6.6 mp_time_factors.conf

The mp_time_factors.conf file is used to define factors with which the values produced by individual strain transducer's are corrected for the amplification factor drift with time. The corrections are defined by "datestamps", `YYYY-MM-DD`, of correction points and by factors at those datestamps.

N.B.: The values are *divided* by the "factors", so as to make the generation of the file a little bit easier and analogous to the time corrections in the cf.

This is the only place in SiWIM-E where extrapolation beyond the last datestamp is not simply a continuation of the last value, but is actually linearly extrapolated using the last pair of values. The assumption is that the amplification factor drift is constant and, once calculated using pairs of values, say a half a year apart, this drift will continue into the future at the same rate.

1.6.7 .nswd

The `.nswd` files are the final output from SiWIM. They are ASCII tab-delimited files. Names of files are in the format `YYYY-MM-DD.nswd`.

1.6.7.1 Fields

The fields and formats in a `.nswd` file are described in table 1.1.

The reduced chi-square measures the goodness-of-estimate of the weighing results. For each weighing interval the value of χ^2 (defined in sections 1.4.8 and 1.6.11) is calculated. This value is then distributed among the weighed axles in inverse proportion to their speeds.

The lower the value the better the fit. The absolute values of RCS are not indicative, instead one should use values relative to average RCS. As a rule-of-thumb, results with RCS above the 95th percentile (relative value of roughly 3) should be examined further. Note that this cutoff value is not necessarily true for sites with high dynamic amplification factor.

1.6.7.2 Warning flags

Warning flags are represented with a 8-character hexadecimal representation of a long word (4 bytes), from “00000000” to “FFFFFFFF”. Individual bits are set when warnings as specified in table 1.2 occur.

N.B.: Even though all of the warning flags positions are listed, not all of them are used. This is to ensure compatibility with MkII `.nswd` files.

Field	Data	Comments and units
1	Timestamp	Format is YYYY-DD-MM-HH-MM-SS-mmm
2	Offset	[s], measured from start of corresponding `.event` file
3	Site ID	
4	Stage trace	A list of processing stages that the event passed through
5	Warning flags	See table 1.2 for meaning of these
6	Lane	
7	v	[m/s], speed
8	N	Number of axles
9	Subclass ID	
10	Axle groups	E.g., 113 for a semitrailer
11	W_{GV}	[kN], gross vehicle weight
12	W_1	[kN], axle 1 load
$\vdots$	$\vdots$	$\vdots$
$11+N$	W_N	[kN], axle N load
$12+N$	$\sum_{i=1}^{N-1} A_i$	[m], total axle distance
$13+N$	A_1	[m], distance between axles 1 and 2
$\vdots$	$\vdots$	$\vdots$
$11+2N$	A_{N-1}	[m], distance between axles $N-1$ and N
$12+2N$	T	[^{0}C], temperature used for compensation
$13+2N$	Impact Factor	Legacy value, always 1
$14+2N$	χ^2_R	Reduced chi-square
$15+2N$	M	The number of strips used (see section 1.6.11 for details)
$16+2N$	$\sum_j s_1(t_j)$	Sum of signals for strip 1
$\vdots$	$\vdots$	$\vdots$
$15+M+2N$	$\sum_j s_M(t_j)$	Sum of signals for strip M

Table 1.1: Fields and formats in `.nswd` files

Bit	Character	Warning
31	1 (`X-------`), `0x80000000`	
30	1 (`X-------`), `0x40000000`	
29	1 (`X-------`), `0x20000000`	
28	1 (`X-------`), `0x10000000`	
27	2 (`-X------`), `0x08000000`	
26	2 (`-X------`), `0x04000000`	
25	2 (`-X------`), `0x02000000`	
24	2 (`-X------`), `0x01000000`	Incomplete MP signals
23	3 (`--X-----`), `0x00800000`	Vehicle overloaded
22	3 (`--X-----`), `0x00400000`	Axle overloaded
21	3 (`--X-----`), `0x00200000`	Classification manually changed
20	3 (`--X-----`), `0x00100000`	First axle position manually adjusted
19	4 (`---X----`), `0x00080000`	Axles and loads manually adjusted
18	4 (`---X----`), `0x00040000`	Reset negative loads after adjustment
17	4 (`---X----`), `0x00020000`	Missing temperature for compensation
16	4 (`---X----`), `0x00010000`	Autogroup disabled for reconstruction
15	5 (`----X---`), `0x00008000`	Vehicle reconstructed
14	5 (`----X---`), `0x00004000`	
13	5 (`----X---`), `0x00002000`	
12	5 (`----X---`), `0x00001000`	
11	6 (`-----X--`), `0x00000800`	Missing MP signals compensated
10	6 (`-----X--`), `0x00000400`	
9	6 (`-----X--`), `0x00000200`	Vehicle reclassified
8	6 (`-----X--`), `0x00000100`	
7	7 (`------X-`), `0x00000080`	Negative axle loads or W_{GV}
6	7 (`------X-`), `0x00000040`	
5	7 (`------X-`), `0x00000020`	
4	7 (`------X-`), `0x00000010`	
3	8 (`-------X`), `0x00000008`	
2	8 (`-------X`), `0x00000004`	
1	8 (`-------X`), `0x00000002`	
0	8 (`-------X`), `0x00000001`	Multiple truck presence

Table 1.2: Warning flags in `.nswd` files

1.6.8 reconstruct.conf

Once a vehicle is weighed, the axle spacings are examined to see if they fit a vehicle on which one or more axles may not have been detected, e.g., an axle in a semitrailer's tridem is potentially missing. For the predefined rules, specified in the reconstruct.conf file, the missing axles are added and a second weighing is performed. If the fit with the measured signal is better by some predefined amount, typically by at least 2%, the reconstructed axles are retained in the vehicle.

For example, the following rule checks if a 5-axle semi-trailer is missing the first axle in its tridem (class 112 was detected instead of 113), by investigating the configuration with another axle placed 1.35 m before the detected third axle:

```
[item_1]
comment=Missing first axle in class 113 semitrailer’s tridem: 112 -> 113
max_axle_distance__m=5.0,10.4,1.75
min_axle_distance__m=2.2,3.25,0.9
rule_1=3,-1.35
```

It is also possible to limit the axle weights ratios for the rule to be applied. For example, adding the lines:

```
ratio_reference_axle=2
ratio_ratios=1,0,0.25
```

would only apply the rules if the first axle is within 0% and 25% of the weight of the second axle.

1.6.9 rply.conf

SiWIM-E operates in two distinct modes. The default is the so-called live mode.

In this mode the data is acquired by data acquisition hardware and processed in real time. The data saved in this mode is saved to directory `./live`, relative to the site directory. Any dumped files are saved to directory `./live_dump`.

In the replay (or reprocessing) mode, that is switched on by command-line option `-r`, the previously saved files are read from the input stage, reprocessed and written to `./rp<NN>` and `./rp<NN>_dump` directories, where `<NN>` is the current output stage.

In this case SiWIM-E attempts to read the rply.conf file, located in the `./usr/rply` directory relative to the site directory. If the file is not found, the input stage is assumed to be 0, i.e., live data, and output stage is assumed to be 1.

There can be at most 99 output stages and output stage number must be higher than the input stage number.

1.6.10 siwim.conf

Global, non site-specific options of the SiWIM-E are located in the siwim.conf file, located in directory `../conf` relative to the executable directory. If this file is not present, all the executables, except for `siwim_mcp.exe`, will exit with a fatal startup error.

1.6.11 strips.conf

The strips.conf file is used to split the measured signals into so-called strips.

Usually the signals from all strain transducers are summed up into the measured signal, which is used for calculating axle weights (see m1weigh core). However, in a case of a multiple-presence event, the two axles on neighbouring lanes can cause a peak at the same time in the measured signal. This causes the system of equations to become ill-conditioned and the combined weight of the two axles is distributed more or less arbitrarily between the two axles.

To mitigate this, the sensors are gropued into strips. The usual method is to assign sensors under lane 1 to strip 1 and sensors under lane 2 to strip 2, but this is not necesary and the `siwim_distsview.exe` program can be used as help in determining the channels to be used for each strip. The expression to be minimised then changes from

$$\chi^2 = \sum_j \left[\frac{s(t_j)}{N} - \sum_i w_i I\left(v_i(t_j - t_i)\right) \right]^2$$

to

$$\chi^2 = \sum_k \left[\sum_j \left[\frac{s_k(t_j)}{N_k} - \sum_i \alpha_{k,l(i)} w_i I_k\left(v_i(t_j - t_i)\right) \right]^2 \right],$$

where k is the index of the strip. The number of equations is thus multiplied by the number of strips.

In the modified expression, the $s_k(t_j)$ is the sum of N_k signals in the k-th strip, as opposed to $s(t_j)$, which is the sum of all N signals. $I_k(x)$ the influence line for the k-th strip and $\alpha_{k,l(i)}$ is the contribution of i-th axle on lane l to k-th strip.

After a suitable sample of transverse distibutions has been gathered, and the strip channels have been defined in the strips.conf file, the system automatically calculates the matrix $\alpha_{k,l}$.

The channels defined in the strips.conf file must match the channels in the .dists file and the channels in the data, otherwise the strip calculations are automatically disabled.

1.6.12 vehicle_classes.conf

The vehicle_classes.conf file is used to classify a vehicle based on axle spacings.

Some subclasses also contain reclassification information that is used to reclassify vehicles based on GVW. For example, if a vehicle classified as subclass 30 has a GVW higher than specified, it is reclassified to subclass 40.

The algorithm allows sequential reclassification, but has a check that prevents inifinite reclassification loops (defined as more than 8 reclassifications) that could arise from misspecified parameters.

1.7 SiWIM-F Configuration Files

SiWIM-F uses three configuration files, which are not used by SiWIM-E, although two of them are stored in the site directory.

1.7.1 site.conf

The file site.conf, located in directory `./conf` relative to the site directory, contains sensor positions and their roles (weigh, smp, admp).

N.B.: Never edit this file outside SiWIM-F since the changes to it will not be reflected in other configuration files. This is done automagically by SiWIM-F.

1.7.2 siwim_f.conf

The file siwim_f.conf, located in directory `../conf` relative to the executable directory, is SiWIM-F's main configuration file.

It contains user's preferences such as units for displaying data (weight, speed, distance, temperature), selected language, etc. as well as systems and their IP addresses.

1.7.3 cf_calibration_runs.conf

The file cf_calibration_runs.conf, located in directory `./conf` relative to the site directory, contains information about calibration runs, used by SiWIM-F to calculate the calibration factor.

1.8 Revision History

- v5.27.66 (21. December 2015)
  - Added .nswd section
  - Added RECONSTRUCT_CONF_ITEM_RATIO_RATIOS_KEY to reconstruct.conf
- v5.25.65 (22. June 2015)
  - Version downgrade
- v5.26.64 (19. June 2015)
  - Some updates in the fftf core, including the FFTF_CONF_PSD_CHOP_SIGNAL_KEY and FFTF_CONF_PSD_RMS_MODE_KEY modes
  - Added TSPLIT_CONF_GLOBAL_SAVE_TXT_FILES_KEY for tsplit core to write text files
  - Numerous changes in daf core
  - Added MATH_CONF_GLOBAL_SAVE_TSV_KEY and math_conf_rms for math core
- v5.26.63 (16. February 2015)
  - Added VEHICLE_CLASSES_SUBCLASS_MAX_AXLE_WEIGHT_KEY and VEHICL↩ E_CLASSES_GLOBAL_COUNTRY_CODE_KEY
  - Some defintions for SiWIM-D (not yet complete)
  - Added M1WEIGH_CONF_GLOBAL_REWEIGH_VEHICLES_KEY
  - Added block mode for tsplit core
  - Added OFFSET_CONF_GLOBAL_HOLD_KEY to offset_conf_global
- v5.25.62 (10. February 2015)
  - Added core acq_mni and corresponding file acq_mni.conf
  - Some changes in acq_ni.conf
  - Field added in M1WEIGH_CONF_GLOBAL_WRITE_DISTS_TXT_KEY
  - Renamed all ∗_REGEXP to ∗_REGEX
- v5.25.60
  - Added some text for FFTF_CONF_PSD_SUMS_KEY and MODULES_CONF_MODU↩ LE_CAN_DROP_SWU_DATA_KEY
- v5.25.59
  - Initial version.

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

File Documentation

3.1 conf_sections_and_keys.h File Reference

```
#include <string>
```

Macros

- #define str const std::string

Variables

"acq_bmc.conf", acquisition section

- str ACQ_BMC_CONF_ACQUISITION_SECTION = "acquisition"
- str ACQ_BMC_CONF_ACQUISITION_SAMPLING_RATE_KEY = "sampling_rate___Hz"
- str ACQ_BMC_CONF_ACQUISITION_CHANNELS_KEY = "ch"
- str ACQ_BMC_CONF_ACQUISITION_NUMBER_OF_CHANNELS_KEY = "number_↩ of_channels"
- str ACQ_BMC_CONF_ACQUISITION_DIGITAL_CHANNELS_KEY = "digital_ch"
- str ACQ_BMC_CONF_ACQUISITION_NUMBER_OF_DIGITAL_CHANNELS_KEY = "number_of_digital_channels"
- str ACQ_BMC_CONF_ACQUISITION_RANGES_KEY = "ranges___V"

"acq_bmc.conf", zeroing section

- str ACQ_BMC_CONF_ZEROING_SECTION = "zeroing"
- str ACQ_BMC_CONF_ZEROING_HOLD_VALUE_ON_DIGITAL_PORT_KEY = "hold_value_on_digital_port_for___msec"
- str ACQ_BMC_CONF_ZEROING_MKII_HW_KEY = "mkii_hw"
- str ACQ_BMC_CONF_ZEROING_ONLY_WHEN_DELTA_BELOW_KEY = "only_↩ when_delta_below___V"
- str ACQ_BMC_CONF_ZEROING_WHEN_AVERAGE_ABOVE_KEY = "when_average↩ _above___V"
- str ACQ_BMC_CONF_ZEROING_WHEN_AVERAGE_BELOW_KEY = "when_↩ average_below___V"

"acq_ctu.conf", global section

- str ACQ_CTU_CONF_GLOBAL_SECTION = "global"
- str ACQ_CTU_CONF_GLOBAL_COM_PORT_KEY = "com_port_number"
- str ACQ_CTU_CONF_GLOBAL_DUMP_TEMP_AND_VOLT_KEY = "dump_temp_↩ and_volt"

- str ACQ_CTU_CONF_GLOBAL_LOG_TEMP_AND_VOLT_INTERVAL_KEY = "log↩
_temp_and_volt_interval___s"
- str ACQ_CTU_CONF_GLOBAL_MOVING_AVERAGE_POINTS_KEY = "moving_↩
average_points"

"acq_mni.conf", acquisition section

- str ACQ_MNI_CONF_ACQUISITION_SECTION = "acquisition"
- str ACQ_MNI_CONF_ACQUISITION_SAMPLING_RATE_KEY = "sampling_rate___Hz"

"acq_mni.conf", device section

- str ACQ_MNI_CONF_DEVICE_SECTION_REGEX = "^device(\\d+)$"
- str ACQ_MNI_CONF_DEVICE_CHASSIS_NAME_KEY = "chassis_name"
- str ACQ_MNI_CONF_DEVICE_DEVICE_NAME_KEY = "device_name"
- str ACQ_MNI_CONF_DEVICE_NUMBER_OF_CHANNELS_KEY = "number_of_↩
channels"
- str ACQ_MNI_CONF_DEVICE_RANGES_KEY = "ranges___V"

"acq_mni.conf", roles section

- str ACQ_MNI_CONF_ROLES_SECTION = "roles"
- str ACQ_MNI_CONF_DEVICE_CHANNEL_ROLE_PATTERN = "ch%d"

"acq_ni.conf", acquisition section

- str ACQ_NI_CONF_ACQUISITION_SECTION = "acquisition"
- str ACQ_NI_CONF_ACQUISITION_DISABLE_TOGGLE_ALIVE_KEY = "disable_↩
toggle_alive"
- str ACQ_NI_CONF_ACQUISITION_SAMPLING_RATE_KEY = "sampling_rate___Hz"
- str ACQ_NI_CONF_ACQUISITION_CHANNELS_KEY = "ch"
- str ACQ_NI_CONF_ACQUISITION_NUMBER_OF_CHANNELS_KEY = "number_of_↩
channels"
- str ACQ_NI_CONF_ACQUISITION_DIGITAL_CHANNELS_KEY = "digital_ch"
- str ACQ_NI_CONF_ACQUISITION_NUMBER_OF_DIGITAL_CHANNELS_KEY =
"number_of_digital_channels"
- str ACQ_NI_CONF_ACQUISITION_RANGES_KEY = "ranges___V"
- str ACQ_NI_CONF_ACQUISITION_CHASSIS_NAME_KEY = "chassis_name"
- str ACQ_NI_CONF_ACQUISITION_DEVICE_NAME_KEY = "device_name"
- str ACQ_NI_CONF_ACQUISITION_DIG_PORT_KEY = "dig_port"

"acq_ni.conf", zeroing section

- str ACQ_NI_CONF_ZEROING_SECTION = "zeroing"
- str ACQ_NI_CONF_ZEROING_DISABLE_KEY = "disable"
- str ACQ_NI_CONF_ZEROING_HOLD_VALUE_ON_DIGITAL_PORT_KEY = "hold_↩
value_on_digital_port_for___msec"
- str ACQ_NI_CONF_ZEROING_MKII_HW_KEY = "mkii_hw"
- str ACQ_NI_CONF_ZEROING_ONLY_WHEN_DELTA_BELOW_KEY = "only_when↩
_delta_below___V"
- str ACQ_NI_CONF_ZEROING_WHEN_AVERAGE_ABOVE_KEY = "when_average_↩
above___V"
- str ACQ_NI_CONF_ZEROING_WHEN_AVERAGE_BELOW_KEY = "when_average_↩
below___V"

"acq_vpn.conf", global section

- str ACQ_VPN_CONF_GLOBAL_SECTION = "global"
- str ACQ_VPN_CONF_GLOBAL_IP_ADDRESS_KEY = "ip_address"
- str ACQ_VPN_CONF_GLOBAL_PING_TIMEOUT_KEY = "ping_timeout___s"
- str ACQ_VPN_CONF_GLOBAL_PING_INTERVAL_KEY = "ping_interval___s"

"autosplit.conf", global section

- str AUTOSPLIT_CONF_GLOBAL_SECTION = "global"

"camera.conf", global section

- str CAMERA_CONF_GLOBAL_SECTION = "global"
- str CAMERA_CONF_GLOBAL_MAX_WAIT_TIME_KEY = "max_wait_time___s"

"camera.conf", camera section

- str CAMERA_CONF_CAMERA_SECTION_BASE = "camera"
- str CAMERA_CONF_CAMERA_ADMIN_PASSWORD_KEY = "admin_password"
- str CAMERA_CONF_CAMERA_CAM_FOCUS_TO_AD_PATTERN = "lane%d_cam_↩ focus_to_ad___m"
- str CAMERA_CONF_CAMERA_FIXED_OFFSET_PATTERN = "lane%d_fixed_offset_↩ _ms"
- str CAMERA_CONF_CAMERA_VEHICLE_CLASS_ABOVE_PATTERN = "lane%d_↩ vehicle_class_above"
- str CAMERA_CONF_CAMERA_ENABLED_KEY = "enabled"
- str CAMERA_CONF_CAMERA_EVENT_PHOTOS_TIMEDELTA_KEY = "time_↩ between_event_photos___s"
- str CAMERA_CONF_CAMERA_EVENT_PREPEND_KEY = "event_prepend___s"
- str CAMERA_CONF_CAMERA_EVENT_APPEND_KEY = "event_append___s"
- str CAMERA_CONF_CAMERA_IP_ADDRESS_KEY = "ip_address"
- str CAMERA_CONF_CAMERA_MAX_TIME_DIFFERENCE = "max_time_difference_↩ _ms"
- str CAMERA_CONF_CAMERA_REQUESTED_EVENTLIST_SIZE = "requested_↩ eventlist_size"
- str CAMERA_CONF_CAMERA_SHOOTING_LANES_KEY = "shooting_lanes"
- str CAMERA_CONF_CAMERA_STORY_SIZE_KEY = "story_size"

"cclear.conf", global section

- str CCLEAR_CONF_GLOBAL_SECTION = "global"
- str CCLEAR_CONF_GLOBAL_CLEAR_ANALOGUE_CHANNELS_KEY = "clear_↩ analogue_channels"
- str CCLEAR_CONF_GLOBAL_DIGITAL_CHANNELS_KEY = "digital_channels"

"cf.conf", global section

- str CF_CONF_GLOBAL_SECTION = "global"
- str CF_CONF_GLOBAL_ALLOW_NEGATIVE_WEIGHTS_KEY = "allow_negative_↩ weights"
- str CF_CONF_GLOBAL_SITE_KEY = "site"
- str CF_CONF_GLOBAL_USE_TEMPERATURE_KEY = "use_temperature"
- str CF_CONF_GLOBAL_FILTER_KEY = "use_filter"
- str CF_CONF_GLOBAL_WRITE_CARS_KEY = "write_cars"

"cf.conf", section for default calibration factor for one lane

- str CF_CONF_DCF_SECTION_REGEX = "^lane(\\d)_dcf$"
- str CF_CONF_DCF_SECTION_PATTERN = "lane%d_dcf"

"cf.conf", section for specific calibration factor for one lane

- str CF_CONF_SCF_SECTION_REGEX = "^lane(\\d)_cf(\\d+)$"
- str CF_CONF_SCF_SECTION_PATTERN = "lane%d_cf%d"
- str CF_CONF_SCF_SECTION_SEPARATORS = "acefln_"
- str CF_CONF_SCF_SUBCLASSES_KEY = "subclasses"

"cf.conf", values for both defautt and specific calibration factors

- str CF_CONF_CF_CF_KEY = "cf"
- str CF_CONF_CF_CF_SPEED_COMP_ENABLED_KEY = "cf_speed_comp_enabled"
- str CF_CONF_CF_CF_SPEED_COMP_SPEED_KEY = "cf_speed_comp_speed___m/s"
- str CF_CONF_CF_CF_SPEED_COMP_F_KEY = "cf_speed_comp_f"
- str CF_CONF_CF_CF_TEMP_COMP_ENABLED_KEY = "cf_temp_comp_enabled"
- str CF_CONF_CF_CF_TEMP_COMP_TEMP_KEY = "cf_temp_comp_temp___C"
- str CF_CONF_CF_CF_TEMP_COMP_F_KEY = "cf_temp_comp_f"
- str CF_CONF_CF_ADJUST_W1_KEY = "adjust_W1"
- str CF_CONF_CF_ADJUST_W1_ACROSS_ALL_AXLES_KEY = "adjust_W1_across↩ _all_axles"
- str CF_CONF_CF_ADJUST_W1_PERCENT_KEY = "adjust_W1___%"
- str CF_CONF_CF_ADJUST_W1_SPEED_COMP_ENABLED_KEY = "adjust_W1_↩ speed_comp_enabled"
- str CF_CONF_CF_ADJUST_W1_SPEED_COMP_SPEED_KEY = "adjust_W1_speed↩ _comp_speed___m/s"
- str CF_CONF_CF_ADJUST_W1_SPEED_COMP_F_KEY = "adjust_W1_speed_comp↩ _f"
- str CF_CONF_CF_AXLE_FACTOR_KEY = "axle_factor"

"csplit.conf", global section

- str CSPLIT_CONF_GLOBAL_SECTION = "global"
- str CSPLIT_CONF_GLOBAL_ANALOGUE_CHANNELS_KEY = "analogue_channels"
- str CSPLIT_CONF_GLOBAL_DIGITAL_CHANNELS_KEY = "digital_channels"

"daf.conf", global section

- str DAF_CONF_GLOBAL_SECTION = "global"
- str DAF_CONF_GLOBAL_DAF4_ENABLED_KEY = "daf4_enabled"
- str DAF_CONF_GLOBAL_ISO_TIMESTAMP_KEY = "iso_timestamp"
- str DAF_CONF_GLOBAL_SHORT_NAME_KEY = "short_name"
- str DAF_CONF_GLOBAL_USE_COMMAS_FOR_TIMESTAMP_KEY = "use_commas↩ _for_timestamp"
- str DAF_CONF_GLOBAL_WRITE_SIG_SUM_KEY = "write_sig_sum"

"daf.conf", DAF2 section

- str DAF_CONF_DAF2_SECTION = "daf2"
- str DAF_CONF_DAF2_CUTOFF_FREQUENCY_KEY = "cutoff_frequency___Hz"

"daf.conf", DAF3 section

- str DAF_CONF_DAF3_SECTION = "daf3"
- str DAF_CONF_DAF3_CHISQR_FTOL_KEY = "chisqr_ftol"
- str DAF_CONF_DAF3_CUTOFF_FREQUENCY_KEY = "cutoff_frequency___Hz"
- str DAF_CONF_DAF3_HEIGHT_KEY = "height"
- str DAF_CONF_DAF3_LINMIN_FTOL_KEY = "linmin_ftol"
- str DAF_CONF_DAF3_OPTIMISE_KEY = "optimise"
- str DAF_CONF_DAF3_OPTIMISED_VARIABLES_KEY = "optimised_variables"
- str DAF_CONF_DAF3_WIDTH_KEY = "width___Hz"

"daf.conf", DAF4 section

- str DAF_CONF_DAF4_SECTION = "daf4"
- str DAF_CONF_DAF4_INVERSE_RESOLUTION_KEY = "inverse_resolution___1/Hz"
- str DAF_CONF_DAF4_COEFFICIENTS_KEY = "coefficients"

"exp.conf", global section

- str EXP_CONF_GLOBAL_SECTION = "global"
- str EXP_CONF_GLOBAL_MODE_KEY = "mode"

"exp.conf", weigh_amplitude section

- str EXP_CONF_WEIGH_AMPLITUDE_SECTION = "weigh_amplitude"
- str EXP_CONF_WEIGH_AMPLITUDE_CHANNELS_KEY = "channels"
- str EXP_CONF_WEIGH_AMPLITUDE_PREFIX_KEY = "prefix"

"exp.conf", extract_temp_and_volt section

- str EXP_CONF_EXTRACT_TEMP_AND_VOLT_SECTION = "extract_temp_and_volt"
- str EXP_CONF_EXTRACT_TEMP_AND_VOLT_ISO_TIMESTAMP_KEY = "iso_↩ timestamp"

"fftf.conf", global section

- str FFTF_CONF_GLOBAL_SECTION = "global"
- str FFTF_CONF_GLOBAL_FILTERS_KEY = "filters"
- str FFTF_CONF_GLOBAL_CHANNELS_KEY = "channels"
- str FFTF_CONF_GLOBAL_ADD_FILTERED_CHANNELS_KEY = "add_filtered_↩ channels"
- str FFTF_CONF_GLOBAL_PSD_MODE_KEY = "psd_mode"

"fftf.conf", lpf section

- str FFTF_CONF_LPF_SECTION = "lpf"
- str FFTF_CONF_LPF_CUTOFF_FREQUENCY_KEY = "cutoff_frequency___Hz"

"fftf.conf", hpf section

- str FFTF_CONF_HPF_SECTION = "hpf"
- str FFTF_CONF_HPF_CUTOFF_FREQUENCY_KEY = "cutoff_frequency___Hz"

"fftf.conf", notch section

- str FFTF_CONF_NOTCH_SECTION = "notch"
- str FFTF_CONF_NOTCH_INTERVAL_KEY = "interval___Hz"

"fftf.conf", psd section

- str FFTF_CONF_PSD_SECTION = "psd"
- str FFTF_CONF_PSD_AVERAGE_POINTS_KEY = "average_points"
- str FFTF_CONF_PSD_CHOP_SIGNAL_KEY = "chop_signal___s"
- str FFTF_CONF_PSD_MAX_FREQUENCY_KEY = "max_frequency___Hz"
- str FFTF_CONF_PSD_MIN_FREQUENCY_KEY = "min_frequency___Hz"
- str FFTF_CONF_PSD_MOVING_AVERAGE_KEY = "moving_average"
- str FFTF_CONF_PSD_RMS_MODE_KEY = "rms_mode"
- str FFTF_CONF_PSD_SUMS_KEY = "sums"

"fix_spacings.conf", global section

- str FIX_SPACINGS_CONF_GLOBAL_SECTION = "global"
- str FIX_SPACINGS_CONF_GLOBAL_MAX_SPACING_CORRECTION_KEY = "max↩ _spacing_correction___m"
- str FIX_SPACINGS_CONF_GLOBAL_MAX_SPEED_CORRECTION_KEY = "max_↩ speed_correction___%"
- str FIX_SPACINGS_CONF_GLOBAL_SPACING_CORRECTION_ENABLED_KEY = "spacing_correction_enabled"

- str FIX_SPACINGS_CONF_GLOBAL_SPEED_CORRECTION_ENABLED_KEY = "speed_correction_enabled"
- str FIX_SPACINGS_CONF_WRITE_DIAGS_KEY = "write_diags"

"fix_spacings.conf", spacings section

- str FIX_SPACINGS_CONF_SPACINGS_SECTION = "spacings"
- str FIX_SPACINGS_CONF_SPACINGS_SPACING_REGEX = "spacing(\\d+)___m"

"gpsrecv.conf", global section

- str **GPSRECV_CONF_SYSTEM_LOCATION_SECTION** = "system_location"
- str **GPSRECV_CONF_SYSTEM_LOCATION_LAT_KEY** = "lat"
- str **GPSRECV_CONF_SYSTEM_LOCATION_LON_KEY** = "lon"

"kmil.conf", bridge section

- str KMIL_CONF_BRIDGE_SECTION = "bridge"
- str KMIL_CONF_BRIDGE_THICKNESS = "thickness___m"

"kmil.conf", global section

- str KMIL_CONF_GLOBAL_SECTION = "global"
- str KMIL_CONF_GLOBAL_ALLOW_MP_KEY = "allow_MP"
- str KMIL_CONF_GLOBAL_ALLOW_CARS_KEY = "allow_cars"
- str KMIL_CONF_GLOBAL_AUTOGROUP_KEY = "autogroup___m"
- str KMIL_CONF_GLOBAL_AXLE_GROUPS_KEY = "axle_groups"
- str KMIL_CONF_GLOBAL_AXLE_WEIGHT_RATIO_KEY = "axle_weight_ratio"
- str KMIL_CONF_GLOBAL_CHANNELS_KEY = "channels"
- str KMIL_CONF_GLOBAL_DUMP_UNUSED_EVENTS_KEY = "dump_unused_events"
- str KMIL_CONF_GLOBAL_FIXED_AXLE_WEIGHT_RATIO_KEY = "fixed_axle_weight_ratio"
- str KMIL_CONF_GLOBAL_FLATTEN_LENGTH_KEY = "flatten_length___s"
- str KMIL_CONF_GLOBAL_LANE_KEY = "lane"
- str KMIL_CONF_GLOBAL_MIN_AMPLITUDE_KEY = "min_amplitude___V"
- str KMIL_CONF_GLOBAL_NUMBER_OF_SAMPLES_FOR_SMOOTHING_KEY = "number_of_samples_for_smoothing"
- str KMIL_CONF_GLOBAL_SUBTRACT_LINEAR_TREND_MAX_DELTA_KEY = "subtract_linear_trend_max_delta___%"
- str KMIL_CONF_GLOBAL_SUBTRACT_LOCAL_LINEAR_TREND_ENABLED_KEY = "subtract_local_linear_trend_enabled"
- str KMIL_CONF_GLOBAL_SUBTRACT_LOCAL_LINEAR_TREND_SECONDS_KEY = "subtract_local_linear_trend___s"
- str KMIL_CONF_GLOBAL_SUBTRACT_GLOBAL_LINEAR_TREND_ENABLED_KEY = "subtract_global_linear_trend_enabled"
- str KMIL_CONF_GLOBAL_SUBTRACT_GLOBAL_LINEAR_TREND_SECONDS_KEY = "subtract_global_linear_trend___s"
- str KMIL_CONF_GLOBAL_WRITE_TEXT_FILE_KEY = "write_text_file"

"kmil.conf", numerical section

- str KMIL_CONF_NUMERICAL_SECTION = "numerical"
- str KMIL_CONF_NUMERICAL_AUTO_INTERVALS = "auto_intervals"
- str KMIL_CONF_NUMERICAL_CALC_METHOD_KEY = "calc_method"
- str KMIL_CONF_NUMERICAL_CHISQR_FTOL_KEY = "chisqr_ftol"
- str KMIL_CONF_NUMERICAL_INTERP_METHOD_KEY = "interp_method"
- str KMIL_CONF_NUMERICAL_ITERATION_DIAGS_KEY = "iteration_diags"
- str KMIL_CONF_NUMERICAL_LINMIN_FTOL_KEY = "linmin_ftol"
- str KMIL_CONF_NUMERICAL_MODEL_PEAK = "model_peak"
- str KMIL_CONF_NUMERICAL_MODEL_PEAK_EDGE = "model_peak_edge"
- str KMIL_CONF_NUMERICAL_MODEL_SUPPORT_WIDTH = "model_support_width"

- str KMIL_CONF_NUMERICAL_MULTIPLE_SPLINES = "multiple_splines"
- str KMIL_CONF_NUMERICAL_NUMBER_OF_OPT_POINTS_KEY = "number_of_↩ opt_points"
- str KMIL_CONF_NUMERICAL_OPTIMISE_AD = "optimise_axle_distance"
- str KMIL_CONF_NUMERICAL_OPTIMISE_LEFT_BOUNDARY_POSITION = "optimise↩ _left_boundary_position"
- str KMIL_CONF_NUMERICAL_OPTIMISE_LEFT_SUPPORT_POSITIONS = "optimise↩ _left_support_positions"
- str KMIL_CONF_NUMERICAL_OPTIMISE_PEAK_HEIGHT = "optimise_peak_height"
- str KMIL_CONF_NUMERICAL_OPTIMISE_PEAK_POSITION = "optimise_peak_↩ position"
- str KMIL_CONF_NUMERICAL_OPTIMISE_PEAK_WIDTH = "optimise_peak_width"
- str KMIL_CONF_NUMERICAL_OPTIMISE_RIGHT_BOUNDARY_POSITION = "optimise_right_boundary_position"
- str KMIL_CONF_NUMERICAL_OPTIMISE_RIGHT_SUPPORT_POSITIONS = "optimise↩ _right_support_positions"

"kmil.conf", default IL section

- str KMIL_CONF_DEFAULT_IL_SECTION_REGEX = "^default_il_lane\\d+$"
- str KMIL_CONF_DEFAULT_IL_SECTION_DELIMITERS = "_adefilntu"
- str KMIL_CONF_DEFAULT_IL_LEFT_BOUNDARY_KEY = "left_boundary___m"
- str KMIL_CONF_DEFAULT_IL_LEFT_DERIVATIVE_KEY = "left_derivative"
- str KMIL_CONF_DEFAULT_IL_LEFT_SUPPORT_POSITIONS_KEY = "left_support↩ _positions___m"
- str KMIL_CONF_DEFAULT_IL_PEAK_POSITION_KEY = "peak_position___m"
- str KMIL_CONF_DEFAULT_IL_RIGHT_BOUNDARY_KEY = "right_boundary___m"
- str KMIL_CONF_DEFAULT_IL_RIGHT_DERIVATIVE_KEY = "right_derivative"
- str KMIL_CONF_DEFAULT_IL_RIGHT_SUPPORT_POSITIONS_KEY = "right_↩ support_positions___m"
- str KMIL_CONF_DEFAULT_IL_OPT_POINTS_KEY = "opt_points"

"kmweigh.conf", global section

- str **KMWEIGH_CONF_GLOBAL_SECTION** = "global"
- str **KMWEIGH_CONF_GLOBAL_CHANNELS_KEY** = "channels"
- str **KMWEIGH_CONF_GLOBAL_AUTOGROUP_KEY** = "autogroup___m"
- str **KMWEIGH_CONF_GLOBAL_ITERATION_DIAGS_KEY** = "iteration_diags"
- str **KMWEIGH_CONF_GLOBAL_ITERATION_LOG_KEY** = "iteration_log"
- str **KMWEIGH_CONF_GLOBAL_RECONSTRUCTION_ENABLED_KEY** = "reconstruction_enabled"
- str **KMWEIGH_CONF_GLOBAL_RECONSTRUCTION_DIAGS_KEY** = "reconstruction_diags"
- str **KMWEIGH_CONF_GLOBAL_RECONSTRUCTION_DISABLES_AUTO↩ GROUP_KEY** = "reconstruction_disables_autogroup"
- str **KMWEIGH_CONF_GLOBAL_RECONSTRUCTION_MIN_CS_IMPROV↩ EMENT_KEY** = "reconstruction_min_cs_improvement___%"
- str **KMWEIGH_CONF_GLOBAL_SUBTRACT_LINEAR_TREND_MAX_D↩ ELTA_KEY** = "subtract_linear_trend_max_delta___%"
- str **KMWEIGH_CONF_GLOBAL_SUBTRACT_LOCAL_LINEAR_TREND_↩ ENABLED_KEY** = "subtract_local_linear_trend_enabled"
- str **KMWEIGH_CONF_GLOBAL_SUBTRACT_LOCAL_LINEAR_TREND_↩ SECONDS_KEY** = "subtract_local_linear_trend___s"
- str **KMWEIGH_CONF_GLOBAL_SUBTRACT_GLOBAL_LINEAR_TREN↩ D_ENABLED_KEY** = "subtract_global_linear_trend_enabled"
- str **KMWEIGH_CONF_GLOBAL_SUBTRACT_GLOBAL_LINEAR_TREN↩ D_SECONDS_KEY** = "subtract_global_linear_trend___s"
- str **KMWEIGH_CONF_GLOBAL_CHISQR_FTOL_KEY** = "chisqr_ftol"
- str **KMWEIGH_CONF_GLOBAL_LINMIN_FTOL_KEY** = "linmin_ftol"
- str **KMWEIGH_CONF_GLOBAL_OPTIMISE_AD** = "optimise_axle_distance"

"kmweigh.conf", default_il section

- str **KMWEIGH_CONF_DEFAULT_IL_SECTION** = "default_il"
- str **KMWEIGH_CONF_DEFAULT_IL_XL_KEY** = "xL___m"
- str **KMWEIGH_CONF_DEFAULT_IL_XR_KEY** = "xR___m"

"kmweigh.conf", strips section

- str **KMWEIGH_CONF_STRIPS_SECTION** = "strips"
- str **KMWEIGH_CONF_AUTOSTRIPS_KEY** = "autostrips"
- str **KMWEIGH_CONF_STRIPS_ENABLED_KEY** = "enabled"
- str **KMWEIGH_CONF_STRIPS_USE_ONLY_FOR_MP_KEY** = "use_only_↩ for_MP"

"m1weigh.conf", global section

- str M1WEIGH_CONF_GLOBAL_SECTION = "global"
- str M1WEIGH_CONF_GLOBAL_AUTOGROUP_KEY = "autogroup___m"
- str M1WEIGH_CONF_GLOBAL_AUTO_TIMESTEP_ENABLED_KEY = "auto_↩ timestep_enabled"
- str M1WEIGH_CONF_GLOBAL_AUTO_TIMESTEP_LENGTH_KEY = "auto_timestep↩ _length___m"
- str M1WEIGH_CONF_GLOBAL_CHANNELS_KEY = "channels"
- str M1WEIGH_CONF_GLOBAL_DIAGS_ONLY_KEY = "diags_only"
- str M1WEIGH_CONF_GLOBAL_ISO_TIMESTAMP_KEY = "iso_timestamp"
- str M1WEIGH_CONF_GLOBAL_NO_DIAGS_KEY = "no_diags"
- str M1WEIGH_CONF_GLOBAL_RECONSTRUCTION_ENABLED_KEY = "reconstruction↩ _enabled"
- str M1WEIGH_CONF_GLOBAL_RECONSTRUCTION_DIAGS_KEY = "reconstruction↩ _diags"
- str M1WEIGH_CONF_GLOBAL_RECONSTRUCTION_DISABLES_AUTOGROUP_K↩ EY = "reconstruction_disables_autogroup"
- str M1WEIGH_CONF_GLOBAL_RECONSTRUCTION_MIN_CS_IMPROVEMENT_K↩ EY = "reconstruction_min_cs_improvement___%"
- str M1WEIGH_CONF_GLOBAL_REWEIGH_VEHICLES_KEY = "reweigh_vehicles"
- str M1WEIGH_CONF_GLOBAL_SUBTRACT_LINEAR_TREND_MAX_DELTA_KEY = "subtract_linear_trend_max_delta___%"
- str M1WEIGH_CONF_GLOBAL_SUBTRACT_LOCAL_LINEAR_TREND_ENABLED↩ _KEY = "subtract_local_linear_trend_enabled"
- str M1WEIGH_CONF_GLOBAL_SUBTRACT_LOCAL_LINEAR_TREND_SECONDS↩ _KEY = "subtract_local_linear_trend___s"
- str M1WEIGH_CONF_GLOBAL_SUBTRACT_GLOBAL_LINEAR_TREND_ENABLE↩ D_KEY = "subtract_global_linear_trend_enabled"
- str M1WEIGH_CONF_GLOBAL_SUBTRACT_GLOBAL_LINEAR_TREND_SECOND↩ S_KEY = "subtract_global_linear_trend___s"
- str M1WEIGH_CONF_GLOBAL_WRITE_AVERAGE_DISTS_KEY = "write_average_↩ dists"
- str M1WEIGH_CONF_GLOBAL_WRITE_DISTS_KEY = "write_dists"
- str M1WEIGH_CONF_GLOBAL_WRITE_DISTS_TXT_KEY = "write_dists_txt"

"m1weigh.conf", default_il section

- str M1WEIGH_CONF_DEFAULT_IL_SECTION = "default_il"
- str M1WEIGH_CONF_DEFAULT_IL_XL_KEY = "xL___m"
- str M1WEIGH_CONF_DEFAULT_IL_XR_KEY = "xR___m"

"m1weigh.conf", strips section

- str M1WEIGH_CONF_STRIPS_SECTION = "strips"
- str M1WEIGH_CONF_STRIPS_ENABLED_KEY = "enabled"
- str M1WEIGH_CONF_STRIPS_USE_ONLY_FOR_MP_KEY = "use_only_for_MP"

"math.conf", global section

- str MATH_CONF_GLOBAL_SECTION = "global"
- str MATH_CONF_GLOBAL_SAVE_TSV_KEY = "save_tsv"
- str MATH_CONF_GLOBAL_TSV_DOWNSAMPLE_KEY = "tsv_downsample"

"math.conf", invert section

- str MATH_CONF_INVERT_SECTION_REGEX = "^\\d+_invert$"
- str MATH_CONF_INVERT_SECTION_CHANNELS_KEY = "channels"

"math.conf", multiply section

- str MATH_CONF_MULTIPLY_SECTION_REGEX = "^\\d+_multiply$"
- str MATH_CONF_MULTIPLY_CHANNEL_FACTOR_REGEX = "^ch(\\d+)_factor$"
- str MATH_CONF_MULTIPLY_REMOVE_OFFSET_KEY = "remove_offset"

"math.conf", multiply_all section

- str MATH_CONF_MULTIPLY_ALL_SECTION_REGEX = "^\\d+_multiply_all$"
- str MATH_CONF_MULTIPLY_ALL_FACTOR_KEY = "factor"
- str MATH_CONF_MULTIPLY_ALL_REMOVE_OFFSET_KEY = "remove_offset"

"math.conf", relabel section

- str MATH_CONF_RELABEL_SECTION_REGEX = "^\\d+_relabel$"
- str MATH_CONF_RELABEL_CLEAR_ALL_KEY = "clear_all"
- str MATH_CONF_RELABEL_CHANNEL_LABEL_REGEX = "^ch(\\d+)_label$"
- str MATH_CONF_RMS_SECTION_REGEX = "^\\d+_rms$"
- str MATH_CONF_RMS_ENABLED_KEY = "enabled"

"math.conf", write_stats section

- str MATH_CONF_STATS_SECTION_REGEX = "^\\d+_write_stats$"
- str MATH_CONF_STATS_ISO_TIMESTAMP_KEY = "iso_timestamp"
- str MATH_CONF_STATS_INVALID_CHANNELS_KEY = "invalid_channels"
- str MATH_CONF_STATS_MAXMINAVG_KEY = "maxminavg"
- str MATH_CONF_STATS_OFFSET_KEY = "offset"
- str MATH_CONF_STATS_DELTA_KEY = "delta"

"modules.conf", global section

- str MODULES_CONF_GLOBAL_SECTION = "global"
- str MODULES_CONF_GLOBAL_MODULES_KEY = "modules"

"modules.conf", module section

- str MODULES_CONF_MODULE_ARGS_KEY = "args"
- str MODULES_CONF_MODULE_DOWNSTREAM_MODULES_KEY = "downstream_↩ modules"
- str MODULES_CONF_MODULE_CAN_DROP_SWU_DATA_KEY = "can_drop_swu_↩ data"
- str MODULES_CONF_MODULE_CORE_KEY = "core"
- str MODULES_CONF_MODULE_ENABLED_KEY = "enabled"
- str MODULES_CONF_MODULE_SAVE_SWU_DATA_KEY = "save_swu_data"

"mp_factors.conf", global section

- str MP_FACTORS_CONF_GLOBAL_SECTION = "global"
- str MP_FACTORS_CONF_GLOBAL_NUMBER_OF_CHANNELS_KEY = "number_of↩ _channels"

"mp_factors.conf", factors section

- str MP_FACTORS_CONF_FACTORS_SECTION = "factors"
- str MP_FACTORS_CONF_FACTORS_FACTOR_REGEX = "ch(\\d+)_factor"

"mp_time_factors.conf", global section

- str MP_TIME_FACTORS_CONF_GLOBAL_SECTION = "global"
- str MP_TIME_FACTORS_CONF_GLOBAL_DATESTAMPS = "datestamps"

"mp_time_factors.conf", factors section

- str MP_TIME_FACTORS_CONF_FACTORS_SECTON = "factors"
- str MP_TIME_FACTORS_CONF_FACTORS_FACTOR_REGEX = "ch(\\d+)_f"

"offset.conf", global section

- str OFFSET_CONF_GLOBAL_SECTION = "global"
- str OFFSET_CONF_GLOBAL_HOLD_KEY = "hold"
- str OFFSET_CONF_GLOBAL_MAX_DELTA_KEY = "max_delta___V"
- str OFFSET_CONF_GLOBAL_NACQ_MODE_KEY = "nacq_mode"
- str OFFSET_CONF_GLOBAL_SUBINTERVAL_LENGTH_KEY = "subinterval_length_↩
__s"
- str OFFSET_CONF_GLOBAL_MAX_SUBINTERVALS_KEY = "max_subintervals"

"reconstruct.conf", item section

- str RECONSTRUCT_CONF_ITEM_SECTION_REGEX = "^item_\\d+$"
- str RECONSTRUCT_CONF_ITEM_SECTION_ROOT = "item_"
- str RECONSTRUCT_CONF_ITEM_SECTION_SEPARATORS = "eimt_"
- str RECONSTRUCT_CONF_ITEM_COMMENT = "comment"
- str RECONSTRUCT_CONF_ITEM_MAX_AXLE_DISTANCE_KEY = "max_axle_↩
distance___m"
- str RECONSTRUCT_CONF_ITEM_MIN_AXLE_DISTANCE_KEY = "min_axle_↩
distance___m"
- str RECONSTRUCT_CONF_ITEM_FORCE_KEY = "force"
- str RECONSTRUCT_CONF_ITEM_RATIO_REFERENCE_AXLE_KEY = "ratio_↩
reference_axle"
- str RECONSTRUCT_CONF_ITEM_RATIO_RATIOS_KEY = "ratio_ratios"
- str RECONSTRUCT_CONF_ITEM_RULE_REGEX = "^rule_\\d+$"
- str RECONSTRUCT_CONF_ITEM_RULE_ROOT = "rule_"
- str RECONSTRUCT_CONF_ITEM_RULE_SEPARATORS = "elru_"

"rply.conf", global section

- str REPLAY_CONF_GLOBAL_SECTION = "global"
- str REPLAY_CONF_GLOBAL_INPUT_STAGE_KEY = "input_stage"
- str REPLAY_CONF_GLOBAL_OUTPUT_STAGE_KEY = "output_stage"

"siwim.conf", global section

- str SIWIM_CONF_GLOBAL_SECTION = "global"
- str SIWIM_CONF_GLOBAL_SITE_KEY = "site"
- str SIWIM_CONF_GLOBAL_RESTART_COMMAND_KEY = "restart_command"

"siwim.conf", engine section

- str SIWIM_CONF_ENGINE_SECTION = "engine"
- str SIWIM_CONF_ENGINE_DUMP_MEMORY_STATE_KEY = "dump_memory_state"
- str SIWIM_CONF_ENGINE_INTERNAL_CHAIN_KEY = "internal_chain"
- str SIWIM_CONF_ENGINE_DEFAULT_TIMEOUT_KEY = "default_timeout___s"

- str SIWIM_CONF_ENGINE_DATATAX_SERVER = "datatax_server"
- str SIWIM_CONF_ENGINE_GPS_SERVER = "gps_server"

"siwim.conf", fs section

- str SIWIM_CONF_FS_SECTION = "fs"
- str SIWIM_CONF_FS_DATA_ROOT_KEY = "data_root"
- str SIWIM_CONF_FS_BACKUP_ROOTS_KEY = "backup_roots"

"siwim.conf", logger section

- str SIWIM_CONF_LOGGER_SECTION = "logger"
- str SIWIM_CONF_LOGGER_DISABLE_INFO_ECHO_KEY = "disable_info_echo"

"siwim.conf", parallel section

- str SIWIM_CONF_PARALLEL_SECTION = "parallel"
- str SIWIM_CONF_PARALLEL_REPLAY_KEY = "REPLAY"
- str SIWIM_CONF_PARALLEL_ENABLED_KEY = "ENABLED"

"siwim_d.conf", forms section

- str SIWIM_D_CONF_FORMS_SECTION = "forms"
- str SIWIM_D_CONF_FORMS_FRM_MAIN_HEIGHT = "frm_main_height"
- str SIWIM_D_CONF_FORMS_FRM_MAIN_LIST_COLUMN_WIDHTS = "frm_main_↩ list_column_widths"
- str SIWIM_D_CONF_FORMS_FRM_MAIN_LIST_HEIGHT = "frm_main_list_height"
- str SIWIM_D_CONF_FORMS_FRM_MAIN_WIDTH = "frm_main_width"

"siwim_d.conf", global section

- str SIWIM_D_CONF_GLOBAL_SECTION = "global"
- str SIWIM_D_CONF_GLOBAL_SITES_DIR_KEY = "sites_dir"

"siwim_f.conf", global section

- str SIWIM_F_CONF_GLOBAL_SECTION = "global"
- str SIWIM_F_CONF_GLOBAL_AUTOCACHE_KEY = "autocache"
- str SIWIM_F_CONF_GLOBAL_DISPLAY_MODULE_QUEUE_SIZE_KEY = "display_↩ module_queue_size"
- str SIWIM_F_CONF_GLOBAL_FORCE_UPDATE_KEY = "force_update"
- str SIWIM_F_CONF_GLOBAL_LANGUAGE_KEY = "language"
- str SIWIM_F_CONF_GLOBAL_LOCAL_E_PARAMETERS_KEY = "local_e_↩ parameters"
- str SIWIM_F_CONF_GLOBAL_MDI_KEY = "mdi"
- str SIWIM_F_CONF_GLOBAL_RXTXLOG_VISIBLE_KEY = "rxtxlog_visible"
- str SIWIM_F_CONF_GLOBAL_RECONNECT_BUTTON_KEY = "reconnect_button"
- str SIWIM_F_CONF_GLOBAL_SITES_DIR_KEY = "sites_dir"
- str SIWIM_F_CONF_GLOBAL_VEHICLE_CLASSES_KEY = "vehicle_classes"
- str SIWIM_F_CONF_GLOBAL_UPDATE_CHANNEL_KEY = "update_channel"

"siwim_f.conf", systems section

- str SIWIM_F_CONF_SYSTEMS_SECTION = "systems"
- str SIWIM_F_CONF_SYSTEMS_SYSTEM_REGEX = "^system(\\d)$"

"smi.conf", global section

- str **SMI_WEIGH_CONF_GLOBAL_SECTION** = "global"

- str **SMI_WEIGH_CONF_GLOBAL_NUMBER_OF_LANES_KEY** = "number_↩ of_lanes"

"smi.conf", default_il section

- str **SMI_WEIGH_CONF_DEFAULT_IL_SECTION** = "default_il"
- str **SMI_WEIGH_CONF_DEFAULT_IL_XL_KEY** = "xL___m"
- str **SMI_WEIGH_CONF_DEFAULT_IL_XR_KEY** = "xR___m"

"speed.conf", global section

- str SPEED_CONF_GLOBAL_SECTION = "global"
- str SPEED_CONF_GLOBAL_MAX_SPEED_KEY = "max_speed___m/s"
- str SPEED_CONF_GLOBAL_MIN_SPEED_KEY = "min_speed___m/s"
- str SPEED_CONF_GLOBAL_NO_DIAGS_KEY = "no_diags"
- str SPEED_CONF_GLOBAL_ACCELERATION_MODE_KEY = "acceleration_mode_↩ enabled"
- str SPEED_CONF_GLOBAL_VEHICLE_MODE_KEY = "vehicle_mode_enabled"
- str SPEED_CONF_GLOBAL_PREVENT_NEGATIVE_SPEEDS_KEY = "prevent_↩ negative_speeds"

"speed.conf", lane_speed section

- str SPEED_CONF_SPEED_SECTION_REGEX = "^lane\\d_speed\\d+$"
- str SPEED_CONF_SPEED_SECTION_DELIMITERS = "_adelnps"
- str SPEED_CONF_SPEED_DIGITAL_KEY = "digital"
- str SPEED_CONF_SPEED_ENABLED_KEY = "enabled"
- str SPEED_CONF_SPEED_FACTOR_KEY = "factor"
- str SPEED_CONF_SPEED_CH1_KEY = "smp1_ch"
- str SPEED_CONF_SPEED_CH2_KEY = "smp2_ch"
- str SPEED_CONF_SPEED_X1_KEY = "smp1_x___m"
- str SPEED_CONF_SPEED_X2_KEY = "smp2_x___m"
- str SPEED_CONF_SPEED_MA1_KEY = "smp1_moving_average___s"
- str SPEED_CONF_SPEED_MA2_KEY = "smp2_moving_average___s"
- str SPEED_CONF_SPEED_TRIGGER1_ABOVE_KEY = "smp1_trigger_above___V"
- str SPEED_CONF_SPEED_TRIGGER2_ABOVE_KEY = "smp2_trigger_above___V"
- str SPEED_CONF_SPEED_TRIGGER1_BELOW_KEY = "smp1_trigger_below___V"
- str SPEED_CONF_SPEED_TRIGGER2_BELOW_KEY = "smp2_trigger_below___V"
- str SPEED_CONF_SPEED_SUBTRACT_ENVELOPE1_KEY = "smp1_subtract_envelope"
- str SPEED_CONF_SPEED_SUBTRACT_ENVELOPE2_KEY = "smp2_subtract_envelope"
- str SPEED_CONF_SPEED_USE_IN_VEHICLE_MODE = "use_in_vehicle_mode"

"speed.conf", vehicle_mode section

- str SPEED_CONF_VEHICLE_MODE_SECTION = "vehicle_mode"
- str SPEED_CONF_VEHICLE_MODE_MOVING_AVERAGE_POINTS_KEY = "moving↩ _average_points"
- str SPEED_CONF_VEHICLE_MODE_SPEED_ERROR_CORRECTION = "speed_error↩ _correction_enabled"
- str SPEED_CONF_VEHICLE_MODE_SPEED_ERROR_AVERAGE_POINTS_KEY = "speed_error_average_points"
- str SPEED_CONF_VEHICLE_MODE_SPEED_ERROR_MAX_DEVIATION_KEY = "speed_error_max_deviation___%"
- str SPEED_CONF_VEHICLE_MODE_WRITE_DIAGS_KEY = "write_diags"
- str SPEED_CONF_VEHICLE_MODE_XL_KEY = "xL___m"
- str SPEED_CONF_VEHICLE_MODE_XR_KEY = "xR___m"

"strips.conf", global section

- str STRIPS_CONF_GLOBAL_SECTION = "global"
- str STRIPS_CONF_GLOBAL_NUMBER_OF_STRIPS_KEY = "number_of_strips"

- str STRIPS_CONF_GLOBAL_STRIP_CHANNELS_ROOT = "channels_strip(\\d+)$"

"test.conf", global section

- str **TEST_CONF_GLOBAL_SECTION** = "global"
- str **TEST_CONF_GLOBAL_BLACK_HOLE_KEY** = "black_hole"
- str **TEST_CONF_GLOBAL_DELAY_KEY** = "delay___s"
- str **TEST_CONF_GLOBAL_EXCEPTION_PROBABILITY_KEY** = "exception↩ _probability___%"
- str **TEST_CONF_GLOBAL_SUICIDE_PROBABILITY_KEY** = "suicide_↩ probability___%"
- str **TEST_CONF_GLOBAL_TIMEOUT_PROBABILITY_KEY** = "timeout_↩ probability___%"
- str **TEST_CONF_GLOBAL_RANDOMISE_KEY** = "randomise"

"tsplit.conf", global section

- str TSPLIT_CONF_GLOBAL_SECTION = "global"
- str TSPLIT_CONF_GLOBAL_COMPLEMENT_TRIGGER_KEY = "complement_trigger"
- str TSPLIT_CONF_GLOBAL_BLOCK_MODE_KEY = "block_mode"
- str TSPLIT_CONF_GLOBAL_CHOP_MODE_KEY = "chop_mode"
- str TSPLIT_CONF_GLOBAL_MAX_EVENT_LENGTH_KEY = "max_event_length___s"
- str TSPLIT_CONF_GLOBAL_MIN_IDLE_TIME_KEY = "min_idle_time___s"
- str TSPLIT_CONF_GLOBAL_NO_DIAGS_KEY = "no_diags"
- str TSPLIT_CONF_GLOBAL_PSDP_MODE_KEY = "psdp_mode"
- str TSPLIT_CONF_GLOBAL_SAVE_TXT_FILES_KEY = "save_txt_files"

"tsplit.conf", block section

- str TSPLIT_CONF_BLOCK_SECTION = "block"
- str TSPLIT_CONF_BLOCK_BLOCK_LENGTH_KEY = "block_length___s"
- str TSPLIT_CONF_BLOCK_PREPEND_KEY = "prepend___s"
- str TSPLIT_CONF_BLOCK_UNCONDITIONAL_KEY = "unconditional"

"tsplit.conf", chop section

- str TSPLIT_CONF_CHOP_SECTION = "chop"
- str TSPLIT_CONF_CHOP_BLOCK_LENGTH_KEY = "block_length___s"

"tsplit.conf", psdp section

- str TSPLIT_CONF_PSDP_SECTION = "psdp"
- str TSPLIT_CONF_PSDP_BLOCK_LENGTH_KEY = "block_length___s"
- str TSPLIT_CONF_PSDP_MIN_LENGTH_KEY = "min_length___%"
- str TSPLIT_CONF_PSDP_DUMP_UNUSED_DATA_KEY = "dump_unused_data"

"tsplit.conf", trigger_above section

- str TSPLIT_CONF_TRIGGER_ABOVE_SECTION = "trigger_above___V"

"tsplit.conf", trigger_below section

- str TSPLIT_CONF_TRIGGER_BELOW_SECTION = "trigger_below___V"

"tsplit.conf", trigger_channel key

- str TSPLIT_CONF_TRIGGER_CHANNEL_KEY_BASE = "ch"

"tsplit.conf", conditions section

- str TSPLIT_CONF_CONDITIONS_SECTION = "conditions"
- str TSPLIT_CONF_CONDITIONS_CONDITION_KEY_BASE = "condition"

"vehicle_ad.conf", global section

- str VEHICLE_AD_CONF_GLOBAL_SECTION = "global"
- str VEHICLE_AD_CONF_GLOBAL_MAX_AXLES_KEY = "max_axles"

"vehicle_ad.conf", lane_ad section

- str VEHICLE_AD_CONF_LANE_AD_SECTION_REGEX = "^lane\\d_ad\\d+$"
- str VEHICLE_AD_CONF_LANE_AD_SECTION_DELIMITERS = "_adeln"
- str VEHICLE_AD_CONF_LANE_AD_CH_KEY = "ch"
- str VEHICLE_AD_CONF_LANE_AD_CORRECTION_KEY = "correction___m"
- str VEHICLE_AD_CONF_LANE_TUBE_LENGTH_KEY = "tube_length___m"
- str VEHICLE_AD_CONF_LANE_AD_X_KEY = "x___m"

"vehicle_fad.conf", global section

- str VEHICLE_FAD_CONF_GLOBAL_SECTION = "global"
- str VEHICLE_FAD_CONF_GLOBAL_MAX_AXLES_KEY = "max_axles"
- str VEHICLE_FAD_CONF_GLOBAL_NO_DIAGS_KEY = "no_diags"
- str VEHICLE_FAD_CONF_GLOBAL_CARRIAGE_MODE_KEY = "carriage_mode"
- str VEHICLE_FAD_CONF_GLOBAL_FORCE_SPLIT_AFTER_AXLE_KEY = "force_↩ split_after_axle"
- str VEHICLE_FAD_CONF_GLOBAL_SPLIT_SINGLE_AXLE_KEY = "split_single_axle"
- str VEHICLE_FAD_CONF_GLOBAL_SPLIT_SINGLE_AXLE_DISTANCE_KEY = "split_single_axle_distance___m"

"vehicle_fad.conf", lane_admp section

- str VEHICLE_FAD_CONF_LANE_ADMP_SECTION_REGEX = "^lane\\d_admp\\d+$"
- str VEHICLE_FAD_CONF_LANE_ADMP_SECTION_DELIMITERS = "_adelmnp"
- str VEHICLE_FAD_CONF_LANE_ADMP_AUTOSHIFT_KEY = "autoshift"
- str VEHICLE_FAD_CONF_LANE_ADMP_CHANNEL_FACTORS_KEY = "channel_↩ factors"
- str VEHICLE_FAD_CONF_LANE_ADMP_CH_KEY = "ch"
- str VEHICLE_FAD_CONF_LANE_ADMP_COMPARISON_FACTOR_KEY = "comparison↩ _factor"
- str VEHICLE_FAD_CONF_LANE_ADMP_CORRECTION_KEY = "correction___m"
- str VEHICLE_FAD_CONF_LANE_ADMP_X_KEY = "x___m"
- str VEHICLE_FAD_CONF_LANE_ADMP_ATS_CUTOFF_KEY = "ats_cutoff"
- str VEHICLE_FAD_CONF_LANE_ADMP_ATS_START_KEY = "ats_start___%"
- str VEHICLE_FAD_CONF_LANE_ADMP_ATS_STEP_KEY = "ats_step___%"
- str VEHICLE_FAD_CONF_LANE_ADMP_ATS_STOP_KEY = "ats_stop___%"
- str VEHICLE_FAD_CONF_LANE_ADMP_AVERAGE1_KEY = "average1___m"
- str VEHICLE_FAD_CONF_LANE_ADMP_AVERAGE1_TYPE_KEY = "average1_type"
- str VEHICLE_FAD_CONF_LANE_ADMP_AVERAGE2_KEY = "average2___m"
- str VEHICLE_FAD_CONF_LANE_ADMP_AVERAGE2_TYPE_KEY = "average2_type"
- str VEHICLE_FAD_CONF_LANE_ADMP_DIFFERENCE_TYPE_KEY = "difference_↩ type"
- str VEHICLE_FAD_CONF_LANE_ADMP_QMM_ENABLED_KEY = "qmm_enabled"
- str VEHICLE_FAD_CONF_LANE_ADMP_QMM_MIN_VALUE_KEY = "qmm_min_↩ value___%"
- str VEHICLE_FAD_CONF_LANE_ADMP_QMM_DELTA_KEY = "qmm_delta___%"
- str VEHICLE_FAD_CONF_LANE_ADMP_SG1_ORDER_KEY = "sg1_order"
- str VEHICLE_FAD_CONF_LANE_ADMP_SG2_ORDER_KEY = "sg2_order"
- str VEHICLE_FAD_CONF_LANE_ADMP_SUBTRACT_ENVELOPE_ENABLED_KE↩ Y = "subtract_envelope_enabled"
- str VEHICLE_FAD_CONF_LANE_ADMP_SUBTRACT_ENVELOPE_BELOW_KEY = "subtract_envelope_below___%"

- str VEHICLE_FAD_CONF_LANE_ADMP_SUBTRACT_ENVELOPE_SPLIT_KEY = "subtract_envelope_split___m"
- str VEHICLE_FAD_CONF_LANE_ADMP_SUBTRACT_GLOBAL_LINEAR_TREND↩_ENABLED_KEY = "subtract_global_linear_trend_enabled"
- str VEHICLE_FAD_CONF_LANE_ADMP_SUBTRACT_GLOBAL_LINEAR_TREND↩_SECONDS_KEY = "subtract_global_linear_trend___s"
- str VEHICLE_FAD_CONF_LANE_ADMP_TPT_ENABLED_KEY = "tpt_enabled"
- str VEHICLE_FAD_CONF_LANE_ADMP_TPT_THRESHOLD_KEY = "tpt_threshold↩___%"
- str VEHICLE_FAD_CONF_LANE_ADMP_TRIGGER_ABOVE_KEY = "trigger_above↩___V"
- str VEHICLE_FAD_CONF_LANE_ADMP_TRIGGER_BELOW_KEY = "trigger_below↩___V"
- str VEHICLE_FAD_CONF_LANE_ADMP_USE_MAX_DIFF_KEY = "use_max_diff"

"vehicle_classes.conf", classes section

- str VEHICLE_CLASSES_GLOBAL_SECTION = "classes"
- str VEHICLE_CLASSES_GLOBAL_COUNTRY_CODE_KEY = "country_code"
- str VEHICLE_CLASSES_GLOBAL_GROUP_MAX_DISTANCE_KEY = "group_max_↩distance___m"
- str VEHICLE_CLASSES_GLOBAL_MAX_AXLE_DISTANCE_KEY = "max_axle_↩distance___m"
- str VEHICLE_CLASSES_GLOBAL_MIN_2AXLE_TRUCK_DISTANCE_KEY = "min_↩2axle_truck_distance___m"
- str VEHICLE_CLASSES_GLOBAL_MIN_AXLE_DISTANCE_KEY = "min_axle_↩distance___m"
- str VEHICLE_CLASSES_GLOBAL_MIN_AVG_AXLE_WEIGHT_KEY = "min_avg_↩axle_weight___kN"
- str VEHICLE_CLASSES_GLOBAL_NO_CLASS_KEY = "unclassified_vehicle_subclass"

"vehicle_classes.conf", subclass section

- str VEHICLE_CLASSES_SUBCLASS_SECTION_REGEX = "^subclass_\\d+$"
- str VEHICLE_CLASSES_SUBCLASS_SECTION_ROOT = "subclass_"
- str VEHICLE_CLASSES_SUBCLASS_SECTION_SEPARATORS = "abclsu_"
- str VEHICLE_CLASSES_SUBCLASS_NUMBER_OF_AXLES_KEY = "number_of_axles"
- str VEHICLE_CLASSES_SUBCLASS_CATEGORY_KEY = "category"
- str VEHICLE_CLASSES_SUBCLASS_LOWER_GVW_LIMIT_KEY = "lower_GVW_↩limit___kN"
- str VEHICLE_CLASSES_SUBCLASS_MAX_AXLE_WEIGHT_KEY = "max_axle_↩weight___kN"
- str VEHICLE_CLASSES_SUBCLASS_MAX_GVW_KEY = "max_GVW___kN"
- str VEHICLE_CLASSES_SUBCLASS_MAX_AXLE_DISTANCE_KEY = "max_axle_↩distance___m"
- str VEHICLE_CLASSES_SUBCLASS_MIN_AXLE_DISTANCE_KEY = "min_axle_↩distance___m"
- str VEHICLE_CLASSES_SUBCLASS_GROUP_WEIGHT_RATIO_KEY_ROOT = "group"
- str VEHICLE_CLASSES_SUBCLASS_RECLASSIFY_DOWN_TO_SUBCLASS_KEY = "reclassify_down_to_subclass"
- str VEHICLE_CLASSES_SUBCLASS_RECLASSIFY_UP_TO_SUBCLASS_KEY = "reclassify_up_to_subclass"
- str VEHICLE_CLASSES_SUBCLASS_UPPER_GVW_LIMIT_KEY = "upper_GVW_↩limit___kN"
- str VEHICLE_CLASSES_SUBCLASS_TYRE_TYPE_KEY = "tyre_type"

"site.conf", global section

- str **SITE_CONF_GLOBAL_SECTION** = "global"
- str **SITE_CONF_GLOBAL_SITE_NAME_KEY** = "site_name"

- str **SITE_CONF_GLOBAL_SITE_GPS_KEY** = "gps_location"
- str **SITE_CONF_GLOBAL_SITE_ROAD_SECTION_KEY** = "road_section"

"site.conf", bridge section

- str **SITE_CONF_BRIDGE_SECTION** = "bridge"
- str **SITE_CONF_BRIDGE_WIDTH_KEY** = "width___m"
- str **SITE_CONF_BRIDGE_LENGTH_KEY** = "length___m"
- str **SITE_CONF_BRIDGE_LENGTH1_KEY** = "length1___m"
- str **SITE_CONF_BRIDGE_LENGTH2_KEY** = "length2___m"
- str **SITE_CONF_BRIDGE_THICKNESS_KEY** = "thickness___m"
- str **SITE_CONF_BRIDGE_SPANS_KEY** = "number_of_spans"
- str **SITE_CONF_BRIDGE_TYPE_KEY** = "type"
- str **SITE_CONF_BRIDGE_DESIGNATION_KEY** = "designation"

"site.conf", direction section

- str **SITE_CONF_DIRECTION1_SECTION** = "direction1"
- str **SITE_CONF_DIRECTION2_SECTION** = "direction2"
- str **SITE_CONF_DIRECTION_ENABLED_KEY** = "enabled"
- str **SITE_CONF_DIRECTION_NAME_KEY** = "name"
- str **SITE_CONF_DIRECTION_NUM_OF_LANES_KEY** = "number_of_lanes"
- str **SITE_CONF_DIRECTION_LANE_WIDTH_KEY** = "lane_width___m"
- str **SITE_CONF_DIRECTION_PAVEMENT_WIDTH_KEY** = "pavement_↩width___m"
- str **SITE_CONF_DIRECTION_EMERGENCY_LANE_WIDTH_KEY** = "emergency↩_lane_width___m"

"site.conf", common section

- str **SITE_CONF_COMMON_SECTION** = "common"
- str **SITE_CONF_COMMON_DIVIDER_WIDTH_KEY** = "divider_width___m"
- str **SITE_CONF_COMMON_SENSOR_Y_MIN_OFFSET** = "sensor_y_min_↩offset___m"
- str **SITE_CONF_COMMON_SENSOR_Y_MAX_OFFSET** = "sensor_y_max_↩offset___m"
- str **SITE_CONF_COMMON_AD_TYPE_KEY** = "ad_type"

"site.conf", sensors section

- str **SITE_CONF_SENSORS_SECTION** = "sensors"
- str **SITE_CONF_DIGITAL_SENSORS_SECTION** = "digital_sensors"
- str **SITE_CONF_SENSORS_SECTION_DISABLED** = "disabled"
- str **SITE_CONF_SENSORS_SECTION_X_PATTERN** = "ch%d_x___m"
- str **SITE_CONF_SENSORS_SECTION_Y_PATTERN** = "ch%d_y___m"
- str **SITE_CONF_SENSORS_SECTION_ROLE_PATTERN** = "ch%d_role"
- str **SITE_CONF_SENSOR_ROLE_NAME_REGEX** = "name_(.*)$"
- str **SITE_CONF_SENSOR_ROLE_VOLTAGE_REGEX** = "voltage(\\d+)"
- str **SITE_CONF_SENSOR_ROLE_WEIGH_PATTERN** = "weigh%d_%d"
- str **SITE_CONF_SENSOR_ROLE_WEIGH_REGEX** = "weigh(\\d+)_(\\d+)"
- str **SITE_CONF_SENSOR_ROLE_SMP_PATTERN** = "lane%d_speed%d_smp%d"
- str **SITE_CONF_SENSOR_ROLE_ADMP_PATTERN** = "lane%d_admp%d"
- str **SITE_CONF_SENSOR_ROLE_SMP_REGEX** = "lane(\\d+)_speed(\\d+)_↩smp(\\d+)"
- str **SITE_CONF_SENSOR_ROLE_ADMP_REGEX** = "lane(\\d+)_admp(\\d+)"
- str **SITE_CONF_SENSOR_ROLE_ANY_REGEX**
- str **SITE_CONF_SENSOR_LABEL_WMP_REGEX** = "w(\\d+)(_(\\d+))?"
- str **SITE_CONF_SENSOR_LABEL_SMP_REGEX** = "s(\\d)(\\d)(\\d)?"
- str **SITE_CONF_SENSOR_LABEL_ADMP_REGEX** = "a(\\d)(\\d)?"
- str **SITE_CONF_SENSOR_LABEL_ANY_REGEX**

"cf_calibration_runs.conf", vehicles section

- str **CALIBRATION_VEHICLE_CONF_VEHICLES_SECTION** = "vehicles"
- str **CALIBRATION_VEHICLE_CONF_VEHICLES_SECTION_IDS_KEY** = "ids"
- str **CALIBRATION_VEHICLE_CONF_RUNS_SECTION** = "vehicle_runs"
- str **CALIBRATION_VEHICLE_CONF_RUNS_SECTION_COUNT_KEY** = "count"
- str **CALIBRATION_VEHICLE_CONF_RUNS_SECTION_ITEM_PATTERN↩_KEY** = "item%d"
- str **CALIBRATION_VEHICLE_CONF_SECTION_AXLE_DIST** = "axle_dist"
- str **CALIBRATION_VEHICLE_CONF_SECTION_AXLE_WEIGHT** = "axle_↩weight"
- str **CALIBRATION_VEHICLE_CONF_SECTION_LICENSE_PLATE** = "license↩_plate"
- str **CALIBRATION_VEHICLE_CONF_SECTION_TYPE** = "type"

3.1.1 Variable Documentation

3.1.1.1 str ACQ_BMC_CONF_ACQUISITION_SECTION = "acquisition"

Section name.

This section controls the acquisition options for the acq_bmc core

3.1.1.2 str ACQ_BMC_CONF_ACQUISITION_SAMPLING_RATE_KEY = "sampling_rate__Hz"

Sampling rate of the acquired signal.

Must be a power of 2

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.3 str ACQ_BMC_CONF_ACQUISITION_CHANNELS_KEY = "ch"

List of acquired analogue channels.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.4 str ACQ_BMC_CONF_ACQUISITION_NUMBER_OF_CHANNELS_KEY = "number_of_channels"

Number of acquired analogue channels.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.5 str ACQ_BMC_CONF_ACQUISITION_DIGITAL_CHANNELS_KEY = "digital_ch"

List of acquired digital channels.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.6 str ACQ_BMC_CONF_ACQUISITION_NUMBER_OF_DIGITAL_CHAN↩ NELS_KEY = "number_of_digital_channels"

Number of acquired digital channels.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.7 str ACQ_BMC_CONF_ACQUISITION_RANGES_KEY = "ranges___V"

Analogue channel ranges.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.8 str ACQ_BMC_CONF_ZEROING_SECTION = "zeroing"

Section name.

This section controls the zeroing options for the acq_bmc core

3.1.1.9 str ACQ_BMC_CONF_ZEROING_HOLD_VALUE_ON_DIGITAL_POR↩ T_KEY = "hold_value_on_digital_port_for___msec"

The zeroing signal is held on digital port for this many milliseconds.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.10 str ACQ_BMC_CONF_ZEROING_MKII_HW_KEY = "mkii_hw"

On MkII hardware the signals on the digital port must be inverted.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.11 str ACQ_BMC_CONF_ZEROING_ONLY_WHEN_DELTA_BELOW_K↩ EY = "only_when_delta_below___V"

Zero the channel if delta is below this value.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.12 str ACQ_BMC_CONF_ZEROING_WHEN_AVERAGE_ABOVE_KEY = "when_average_above___V"

Zero the channel if average is above this value.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.13 str ACQ_BMC_CONF_ZEROING_WHEN_AVERAGE_BELOW_KEY = "when_average_below___V"

Zero the channel if average is below this value.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.14 str ACQ_CTU_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the acq_ctu core

3.1.1.15 str ACQ_CTU_CONF_GLOBAL_COM_PORT_KEY = "com_port_number"

COM port on which the CTU is connected.

Type: string
System default: COM1
Template default: UNKNOWN

3.1.1.16 str ACQ_CTU_CONF_GLOBAL_DUMP_TEMP_AND_VOLT_KEY = "dump_temp_and_volt"

Dumps temperatures and voltages as they are read from CTU to file `/usr/dbg/temp_and_volt.txt`

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.17 str ACQ_CTU_CONF_GLOBAL_LOG_TEMP_AND_VOLT_INTERVA↩L_KEY = "log_temp_and_volt_interval___s"

Interval in seconds between writing averaged temperatures and voltages to a log file.

This information is written twice, once to the site log directory and once to the SiWIM-E log directory. One file per day is produced and the filename is `ctu_YYYY_MM_DD.log`.

Type: unsigned
System default: 60
Template default: UNKNOWN

3.1.1.18 str ACQ_CTU_CONF_GLOBAL_MOVING_AVERAGE_POINTS_KEY = "moving_average_points"

Number of values for temperature and voltage averaging.

Type: unsigned
System default: 60
Template default: UNKNOWN

3.1.1.19 str ACQ_MNI_CONF_ACQUISITION_SECTION = "acquisition"

Section name.

This section controls the acquisition options for the acq_ni core

3.1.1.20 str ACQ_MNI_CONF_ACQUISITION_SAMPLING_RATE_KEY = "sampling_rate___Hz"

Sampling rate of the acquired signal.

Must be a power of 2

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.21 str ACQ_MNI_CONF_DEVICE_SECTION_REGEX = "^device(\\d+)$"

Section name.

A regular expression. An example of a match is `device1`.

3.1.1.22 str ACQ_MNI_CONF_DEVICE_CHASSIS_NAME_KEY = "chassis_name"

Name of National Instruments cDAQ chassis.

Type: string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.23 str ACQ_MNI_CONF_DEVICE_DEVICE_NAME_KEY = "device_name"

Name of National Instruments AD device.

Type: string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.24 str ACQ_MNI_CONF_DEVICE_NUMBER_OF_CHANNELS_KEY = "number_of_channels"

Number of acquired analogue channels.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.25 str ACQ_MNI_CONF_DEVICE_RANGES_KEY = "ranges___V"

Analogue channel ranges.

This can contain either a list of values or a single value. A single value is applied to each channel. The list of values must contain ACQ_MNI_CONF_DEVICE_NUMBER_OF_CHANNELS_KEY values and each channel can then have a separate range.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.26 str ACQ_MNI_CONF_ROLES_SECTION = "roles"

Section name.

This section lists the channels' roles.

3.1.1.27 str ACQ_MNI_CONF_DEVICE_CHANNEL_ROLE_PATTERN = "ch%d"

Channel role.

A pattern. An example of a match is `ch1`, which would read the role for channel 1 and thusly label this channel.

If a role for a channel is not defined, the channel is labeled as `ch<N>`, where<N>} is the serial number of the channel.

Note that channels are labeled across all devices. E.g., if the first device acquires 4 channels and the second device 3 channels, channel 6 is the second channel on the second device.

Type: string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.28 str ACQ_NI_CONF_ACQUISITION_SECTION = "acquisition"

Section name.

This section controls the acquisition options for the acq_ni core

3.1.1.29 str ACQ_NI_CONF_ACQUISITION_DISABLE_TOGGLE_ALIVE_KEY = "disable_toggle_alive"

Disable toggle alive.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.30 str ACQ_NI_CONF_ACQUISITION_SAMPLING_RATE_KEY = "sampling_rate__Hz"

Sampling rate of the acquired signal.

Must be a power of 2

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.31 str ACQ_NI_CONF_ACQUISITION_CHANNELS_KEY = "ch"

List of acquired analogue channels.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.32 str ACQ_NI_CONF_ACQUISITION_NUMBER_OF_CHANNELS_KEY = "number_of_channels"

Number of acquired analogue channels.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.33 str ACQ_NI_CONF_ACQUISITION_DIGITAL_CHANNELS_KEY = "digital_ch"

List of acquired digital channels.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.34 str ACQ_NI_CONF_ACQUISITION_NUMBER_OF_DIGITAL_CHANN↩ ELS_KEY = "number_of_digital_channels"

Number of acquired digital channels.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.35 str ACQ_NI_CONF_ACQUISITION_RANGES_KEY = "ranges___V"

Analogue channel ranges.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.36 str ACQ_NI_CONF_ACQUISITION_CHASSIS_NAME_KEY = "chassis_name"

Name of National Instruments cDAQ chassis.

If this is empty, the first available chassis will be used

Type: string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.37 str ACQ_NI_CONF_ACQUISITION_DEVICE_NAME_KEY = "device_name"

Name of National Instruments AD device.

If this is empty, <ACQ_NI_CONF_ACQUISITION_CHASSIS_NAME_KEY>Mod1 will be used

Type: string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.38 str ACQ_NI_CONF_ACQUISITION_DIG_PORT_KEY = "dig_port"

Name of National Instruments NiDAQ digital port.

Type: string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.39 str ACQ_NI_CONF_ZEROING_SECTION = "zeroing"

Section name.

This section controls the zeroing options for the acq_ni core

3.1.1.40 str ACQ_NI_CONF_ZEROING_DISABLE_KEY = "disable"

Disable zeroing.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.41 str ACQ_NI_CONF_ZEROING_HOLD_VALUE_ON_DIGITAL_PORT↩ _KEY = "hold_value_on_digital_port_for___msec"

The zeroing signal is held on digital port for this many milliseconds.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.42 str ACQ_NI_CONF_ZEROING_MKII_HW_KEY = "mkii_hw"

On MkII hardware the signals on the digital port must be inverted.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.43 str ACQ_NI_CONF_ZEROING_ONLY_WHEN_DELTA_BELOW_KEY = "only_when_delta_below___V"

Zero the channel if delta is below this value.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.44 str ACQ_NI_CONF_ZEROING_WHEN_AVERAGE_ABOVE_KEY = "when_average_above___V"

Zero the channel if average is above this value.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.45 str ACQ_NI_CONF_ZEROING_WHEN_AVERAGE_BELOW_KEY = "when_average_below___V"

Zero the channel if average is below this value.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.46 str ACQ_VPN_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the acq_vpn core

3.1.1.47 str ACQ_VPN_CONF_GLOBAL_IP_ADDRESS_KEY = "ip_address"

IP Address of the VPN server.

Type: string
System default: 172.16.0.1
Template default: UNKNOWN

3.1.1.48 str ACQ_VPN_CONF_GLOBAL_PING_TIMEOUT_KEY = "ping_timeout___s"

Ping timeout.

Type: unsigned
System default: 15
Template default: UNKNOWN

3.1.1.49 str ACQ_VPN_CONF_GLOBAL_PING_INTERVAL_KEY = "ping_interval___s"

Ping interval.

Type: unsigned
System default: 30
Template default: UNKNOWN

3.1.1.50 str AUTOSPLIT_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the autosplit core

3.1.1.51 str CAMERA_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the camera core

3.1.1.52 str CAMERA_CONF_GLOBAL_MAX_WAIT_TIME_KEY = "max_wait_time___s"

Maximum wait time.

Core will wait at most this number of seconds for events in the future.

Type: unsigned
System default: 10
Template default: UNKNOWN

3.1.1.53 str CAMERA_CONF_CAMERA_SECTION_BASE = "camera"

Section name.

There are at most four such sections, from `camera1` to `camera4`, containing options for each of the (at most) four controlled cameras.

3.1.1.54 str CAMERA_CONF_CAMERA_ADMIN_PASSWORD_KEY = "admin_password"

Password for camera's web interface.

Type: string
System default: meinsm
Template default: UNKNOWN

3.1.1.55 str CAMERA_CONF_CAMERA_CAM_FOCUS_TO_AD_PATTERN = "lane%d_cam_focus_to_ad__m"

Camera focus to AD in metres.

Pattern for sprintf, producing, e.g., `lane1_cam_focus_to_ad__m`.

For every lane specified in CAMERA_CONF_CAMERA_SHOOTING_LANES_KEY a value like this should exist. It specifies where the vehicle should get photographed relative to where it is detected by SiWIM.

Type: integer
System default: 0
Template default: UNKNOWN

3.1.1.56 str CAMERA_CONF_CAMERA_FIXED_OFFSET_PATTERN = "lane%d_fixed_offset__ms"

Fixed offset in milliseconds.

Pattern for sprintf, producing, e.g., `lane1_fixed_offset__ms`.

A fixed value that is added to vehicle's timestamp to obtain photo timestamp.

Type: integer
System default: 0
Template default: UNKNOWN

3.1.1.57 str CAMERA_CONF_CAMERA_VEHICLE_CLASS_ABOVE_PATTERN = "lane%d_vehicle_class_above"

Vehicle class above.

Pattern for sprintf, producing, e.g., `lane1_vehicle_class_above`.

Only take photos of vehicles whose subclass ids are greater than this value.

Type: integer
System default: 0
Template default: UNKNOWN

3.1.1.58 str CAMERA_CONF_CAMERA_ENABLED_KEY = "enabled"

Enable this camera.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.59 str CAMERA_CONF_CAMERA_EVENT_PHOTOS_TIMEDELTA_KEY = "time_between_event_photos___s"

Time between event photos in seconds.

Photos for event in the event mode get taken every this many seconds between the first and last photos.

See also CAMERA_CONF_CAMERA_EVENT_PREPEND_KEY and CAMERA_CONF_CAMERA_EVENT_APPEND_KEY.

Type: float
System default: 0.5
Template default: UNKNOWN

3.1.1.60 str CAMERA_CONF_CAMERA_EVENT_PREPEND_KEY = "event_prepend___s"

Prepend in seconds.

This value is subtracted from the timestamp of the first sample in the event to determine the timestamp of the first photo.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.61 str CAMERA_CONF_CAMERA_EVENT_APPEND_KEY = "event_append___s"

Append in seconds.

This value is added to the timestamp of the last sample in the event to determine the timestamp of the last photo.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.62 str CAMERA_CONF_CAMERA_IP_ADDRESS_KEY = "ip_address"

Camera's IP address.

Type: string
System default: NONE
Template default: UNKNOWN

3.1.1.63 str CAMERA_CONF_CAMERA_MAX_TIME_DIFFERENCE = "max_time_difference___ms"

Maximum time difference in milliseconds.

Type: unsigned
System default: 3000
Template default: UNKNOWN

3.1.1.64 str CAMERA_CONF_CAMERA_REQUESTED_EVENTLIST_SIZE = "requested_eventlist_size"

Requested event list size.

Type: unsigned
System default: 10
Template default: UNKNOWN

3.1.1.65 str CAMERA_CONF_CAMERA_SHOOTING_LANES_KEY = "shooting_lanes"

Shooting lanes.

Comma separated list of lane numbers this camera is shooting.

N.B.: In the event mode, photos for each event get taken by every enabled camera, i.e. there are no lane-camera relationships for event cameras, and this value is ignored.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.66 str CAMERA_CONF_CAMERA_STORY_SIZE_KEY = "story_size"

Story size.

How many before and after photos of a vehicle get taken. Story size of 0 means only the central photo is taken. Story size of 1 means one photo before and one after the central photo, i.e. 3 photos altogether, etc.

Type: unsigned
System default: 0
Template default: UNKNOWN

3.1.1.67 str CCLEAR_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the cclear core

3.1.1.68 str CCLEAR_CONF_GLOBAL_CLEAR_ANALOGUE_CHANNELS_KEY = "clear_analogue_channels"

A list of analogue channels to be cleared.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.69 str CCLEAR_CONF_GLOBAL_DIGITAL_CHANNELS_KEY = "digital_channels"

A list of digital channels to be cleared.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.70 str CF_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the cf core

3.1.1.71 str CF_CONF_GLOBAL_ALLOW_NEGATIVE_WEIGHTS_KEY = "allow_negative_weights"

Allow negative weighs in `.nswd` files.

See Negative Weights

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.72 str CF_CONF_GLOBAL_SITE_KEY = "site"

Site name - gets written as the third field in a `.nswd` file.

Type: string
System default: NONE
Template default: UNKNOWN

3.1.1.73 str CF_CONF_GLOBAL_USE_TEMPERATURE_KEY = "use_temperature"

Determines which temerature to use for temperature compensation.

Legal values are T1 through T5 and Ti.

Type: string
System default: Ti
Template default: Ti

3.1.1.74 str CF_CONF_GLOBAL_FILTER_KEY = "use_filter"

Enables the use of cf.filter file.

If this is true, cf core outputs only those vehicles whose timestamp that matches one of the lines in the filter file.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.75 str CF_CONF_GLOBAL_WRITE_CARS_KEY = "write_cars"

Controls whether the cars are written to `.nswd` files.

Legal values and their meanings are:

- 0: Never write cars
- 1: Write cars only when they appear in multiple-presence events with trucks
- 2: Always write cars

Type: unsigned
System default: 1
Template default: UNKNOWN

3.1.1.76 str CF_CONF_DCF_SECTION_REGEX = "^lane(\\d)_dcf$"

Section name.

A regular expression. An example of a match is `lane1_dcf`.

3.1.1.77 str CF_CONF_SCF_SECTION_REGEX = "^lane(\\d)_cf(\\d+)$"

Section name.

A regular expression. An example of a match is `lane1_cf3`.

3.1.1.78 str CF_CONF_SCF_SUBCLASSES_KEY = "subclasses"

A list of subclasses to which this specific calibration factor applies.

Type: list of unsigned
System default: 1
Template default: UNKNOWN

3.1.1.79 str CF_CONF_CF_CF_SPEED_COMP_ENABLED_KEY = "cf_speed_comp_enabled"

Enables the speed compensation of calibration factor.

See GVW Speed Compensation

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.80 str CF_CONF_CF_CF_SPEED_COMP_SPEED_KEY = "cf_speed_comp_speed___m/s"

A list of abscissas in metres per second for the speed compensation function.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.81 str CF_CONF_CF_CF_SPEED_COMP_F_KEY = "cf_speed_comp_f"

A list of ordinates for the speed compensation function.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.82 str CF_CONF_CF_CF_TEMP_COMP_ENABLED_KEY = "cf_temp_comp_enabled"

Enables the temperature compensation of calibration factor.

See GVW Temperature Compensation

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.83 str CF_CONF_CF_CF_TEMP_COMP_TEMP_KEY = "cf_temp_comp_temp___C"

A list of abscissas in degrees centigrade for the temperature compensation function.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.84 str CF_CONF_CF_CF_TEMP_COMP_F_KEY = "cf_temp_comp_f"

A list of ordinates for the temperature compensation function.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.85 str CF_CONF_CF_ADJUST_W1_KEY = "adjust_W1"

Enables the fixed adjustment of the first axle load.

See Fixed W1 Redistribution

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.86 str CF_CONF_CF_ADJUST_W1_ACROSS_ALL_AXLES_KEY = "adjust_W1_across_all_axles"

Enables the adjustment of the first axle load over all axles.

See W1 Redistribution

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.87 str CF_CONF_CF_ADJUST_W1_PERCENT_KEY = "adjust_W1___%"

The ammount of adjustment of the first axle in percent of the axle load.

The value can be negative (reduce first axle load) or positive (increase first axle load)

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.88 str CF_CONF_CF_ADJUST_W1_SPEED_COMP_ENABLED_KEY = "adjust_W1_speed_comp_enabled"

Enables the adjustment of the first axle load over all axles, depending on the vehicle speed.

See Speed-Based W1 Redistribution

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.89 str CF_CONF_CF_ADJUST_W1_SPEED_COMP_SPEED_KEY = "adjust_W1_speed_comp_speed___m/s"

A list of abscissas in metres per second for the first axle load speed compensation function.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.90 str CF_CONF_CF_ADJUST_W1_SPEED_COMP_F_KEY = "adjust_W1_speed_comp_f"

A list of ordinates for the first axle load speed compensation function.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.91 str CF_CONF_CF_AXLE_FACTOR_KEY = "axle_factor"

A list of individual axle factors.

It is possible to adjust individual axle loads using this value, The value is a semicolon separated list of pairs, that determine the axle number and adjustment factor.

For example, if the second axle is to be adjusted by +10% (a factor of 1.1) and the fourth axle by -5% (a factor of 0.95) the value would be `2,1.1;4,0.95`.

Type: list of pair<unsigned, float>
System default: NONE
Template default: UNKNOWN

3.1.1.92 str CSPLIT_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the csplit core

3.1.1.93 str CSPLIT_CONF_GLOBAL_ANALOGUE_CHANNELS_KEY = "analogue_channels"

A list of analogue channels to be passed through.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.94 str CSPLIT_CONF_GLOBAL_DIGITAL_CHANNELS_KEY = "digital_channels"

A list of digital channels to be passed through.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.95 str DAF_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the daf core

3.1.1.96 str DAF_CONF_GLOBAL_DAF4_ENABLED_KEY = "daf4_enabled"

Enables DAF4 calculation.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.97 str DAF_CONF_GLOBAL_ISO_TIMESTAMP_KEY = "iso_timestamp"

Enables ISO 8601 timestamp format suitable for use with numpy.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.98 str DAF_CONF_GLOBAL_SHORT_NAME_KEY = "short_name"

Disables appended date to filename.

Allows parallel processing without the need to manually combine output files.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.99 str DAF_CONF_GLOBAL_USE_COMMAS_FOR_TIMESTAMP_KEY = "use_commas_for_timestamp"

Enables alternate timestamp format.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.100 str DAF_CONF_GLOBAL_WRITE_SIG_SUM_KEY = "write_sig_sum"

Writes sig and sum values to text files.

For each interval writes signal and sum values for later processing

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.101 str DAF_CONF_DAF2_SECTION = "daf2"

Section name.

This section controls the global options for the daf core

3.1.1.102 str DAF_CONF_DAF2_CUTOFF_FREQUENCY_KEY = "cutoff_frequency___Hz"

The low-pass frequency for DAF2.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.103 str DAF_CONF_DAF3_SECTION = "daf3"

Section name.

This section controls the global options for the daf core

3.1.1.104 str DAF_CONF_DAF3_CHISQR_FTOL_KEY = "chisqr_ftol"

χ^2 tolerance for iteration termination

DAF3 optimisation is a non-linear optimisation process. When the relative change of χ^2 changes by less than this ammount between iterations, the approximation is deemed acceptable and iteration stopped.

Type: float
System default: 1e-2
Template default: UNKNOWN

3.1.1.105 str DAF_CONF_DAF3_CUTOFF_FREQUENCY_KEY = "cutoff_frequency___Hz"

The low-pass frequency for DAF3.

In the DAF3 optimisation mode this is used as the initial frequency

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.106 str DAF_CONF_DAF3_HEIGHT_KEY = "height"

The DAF3 height.

In the DAF3 optimisation mode this is used as the initial height

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.107 str DAF_CONF_DAF3_LINMIN_FTOL_KEY = "linmin_ftol"

Numerical tolerance for the interal linear minimisation step of optimisation.

Type: float
System default: 1e-2
Template default: UNKNOWN

3.1.1.108 str DAF_CONF_DAF3_OPTIMISE_KEY = "optimise"

DAF3 optimisation mode.

In this mode the cutoff frequency and width are determined with Powell's minimisation of the sum of square of difference between the signal and the sum of ILs.

N.B.: This can be calculated more efficiently using the `optdaf.py` script.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.109 str DAF_CONF_DAF3_OPTIMISED_VARIABLES_KEY = "optimised_variables"

Number of optimised variables.

1. 1 means optimise just the width (frequency is swept and height is set to value from conf)
2. 2 means optimise the frequency and width (height is set to value from conf)
3. 3 means optimise all 3 variables

Type: unsigned
System default: 1
Template default: UNKNOWN

3.1.1.110 str DAF_CONF_DAF3_WIDTH_KEY = "width___Hz"

The DAF3 width.

In the DAF3 optimisation mode this is used as the initial width

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.111 str DAF_CONF_DAF4_SECTION = "daf4"

Section name.

This section controls the global options for the daf core

3.1.1.112 str DAF_CONF_DAF4_INVERSE_RESOLUTION_KEY = "inverse_resolution___1/Hz"

The inverse resolution of the table.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.113 str DAF_CONF_DAF4_COEFFICIENTS_KEY = "coefficients"

Filter coefficients.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.114 str EXP_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the exp core

3.1.1.115 str EXP_CONF_GLOBAL_MODE_KEY = "mode"

Defines the operation mode.

For each operation mode the section named after that mode must be present and must define parameters for that mode. Currently only the `weigh_amplitude` mode is supported.

Type: string
System default: NONE
Template default: UNKNOWN

3.1.1.116 str EXP_CONF_WEIGH_AMPLITUDE_SECTION = "weigh_amplitude"

Section name.

This section controls the weigh_amplitude options for the exp core

3.1.1.117 str EXP_CONF_WEIGH_AMPLITUDE_CHANNELS_KEY = "channels"

List of summed analogue channels.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.118 str EXP_CONF_EXTRACT_TEMP_AND_VOLT_SECTION = "extract_temp_and_volt"

Section name.

This section controls the extract_temp_and_volt options for the exp core

3.1.1.119 str EXP_CONF_EXTRACT_TEMP_AND_VOLT_ISO_TIMESTAMP_↩ KEY = "iso_timestamp"

Enables ISO 8601 timestamp format suitable for use with numpy.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.120 str FFTF_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the fftf core

3.1.1.121 str FFTF_CONF_GLOBAL_FILTERS_KEY = "filters"

A list of filters to be used.

Type: list of string
System default: NONE
Template default: UNKNOWN

3.1.1.122 str FFTF_CONF_GLOBAL_CHANNELS_KEY = "channels"

A list of channels to be filtered.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.123 str FFTF_CONF_GLOBAL_ADD_FILTERED_CHANNELS_KEY = "add_filtered_channels"

If this is enabled, filtered channels are added to the input channels, otherwise the original channels are overwritten.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.124 str FFTF_CONF_GLOBAL_PSD_MODE_KEY = "psd_mode"

Turns on the PSD mode.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.125 str FFTF_CONF_LPF_SECTION = "lpf"

Section name.

This section controls the low-pass filter options for the fftf core

3.1.1.126 str FFTF_CONF_LPF_CUTOFF_FREQUENCY_KEY = "cutoff_frequency___Hz"

Cutoff frequency for the low-pass filter.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.127 str FFTF_CONF_HPF_SECTION = "hpf"

Section name.

This section controls the high-pass filter options for the fftf core

3.1.1.128 str FFTF_CONF_HPF_CUTOFF_FREQUENCY_KEY = "cutoff_frequency___Hz"

Cutoff frequency for the high-pass filter.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.129 str FFTF_CONF_NOTCH_SECTION = "notch"

Section name.

This section controls the notch filter options for the fftf core

3.1.1.130 str FFTF_CONF_NOTCH_INTERVAL_KEY = "interval___Hz"

Frequency interval for the notch filter.

Components with frequencies in this interval are removed from the signal.

Type: pair<float, float>
System default: NONE
Template default: UNKNOWN

3.1.1.131 str FFTF_CONF_PSD_SECTION = "psd"

Section name.

This section controls the PSD mode of the fftf core

3.1.1.132 str FFTF_CONF_PSD_AVERAGE_POINTS_KEY = "average_points"

Number of PSDs to be averaged.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.133 str FFTF_CONF_PSD_CHOP_SIGNAL_KEY = "chop_signal___s"

Chop the signal.

This will smooth out the spectra at the expense of reduced resolution. The parameter value is the length of the block in seconds. The default value of 0 means do not chop

Type: unsigned
System default: 0
Template default: UNKNOWN

3.1.1.134 str FFTF_CONF_PSD_MAX_FREQUENCY_KEY = "max_frequency___Hz"

Maximum output frequency.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.135 str FFTF_CONF_PSD_MIN_FREQUENCY_KEY = "min_frequency___Hz"

Minimum output frequency.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.136 str FFTF_CONF_PSD_MOVING_AVERAGE_KEY = "moving_average"

Calculate moving average.

If true, then a moving average is calculated and a file is output for each incoming block after FFTF_C↩
ONF_PSD_AVERAGE_POINTS_KEY blocks have been accumulated, otherwise a file is output after every FFTF_CONF_PSD_AVERAGE_POINTS_KEY block.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.137 str FFTF_CONF_PSD_RMS_MODE_KEY = "rms_mode"

RMS mode.

In this mode the spectra will be divided by the square root of the record length

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.138 str FFTF_CONF_PSD_SUMS_KEY = "sums"

Determines which channels are to be summed.

This is a semicolon (`;`) separated list of channels. Each element can contain a comma-separated list of channels to be summed up. E.g., `sums=1;2;3;4` would output 4 PSDs, one for each channel, whereas `sums=1,2;3,4` would output 2 PSDs, one for the sum of channels 1 and 2 and the other for the sum of channels 3 and 4.

This option is available as an addition to the individual channel PSD calculations, for which the FFT↩
F_CONF_GLOBAL_CHANNELS_KEY is used.

Type: list of list of unsigned
System default: false
Template default: UNKNOWN

3.1.1.139 str FIX_SPACINGS_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the fix_spacings core

3.1.1.140 str FIX_SPACINGS_CONF_GLOBAL_MAX_SPACING_CORRECTIO↩ N_KEY = "max_spacing_correction___m"

Maximum spacing correction in metres.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.141 str FIX_SPACINGS_CONF_GLOBAL_MAX_SPEED_CORRECTION_↩ KEY = "max_speed_correction___%"

Maximum speed correction in percent of original speed.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.142 str FIX__SPACINGS__CONF__GLOBAL__SPACING__CORRECTION__ENA↩ BLED__KEY = "spacing__correction__enabled"

Enables spacing correction.

Type: bool
System default: NONE
Template default: UNKNOWN

3.1.1.143 str FIX__SPACINGS__CONF__GLOBAL__SPEED__CORRECTION__ENABL↩ ED__KEY = "speed__correction__enabled"

Enables spacing correction.

Type: bool
System default: NONE
Template default: UNKNOWN

3.1.1.144 str FIX__SPACINGS__CONF__WRITE__DIAGS__KEY = "write__diags"

Enables output of a diagnostic file.

The file is an ASCII tab-delimited file containing the fields:

- 1: Vehicle index in train
- 2: Vehicle timestamp
- 3: Original speed
- 4: Corrected speed
- 5: Speed change relative to original speed
- 6..14: Original axle distances
- 15..23: Corrected axle distances
- 24..32: Axle distance change relative to original axle distance

Note that the number of axles is fixed to 8 (the maximum on a railway carriage). For carraiges with less axles, the irrelevant fields are left empty.

Type: bool
System default: NONE
Template default: UNKNOWN

3.1.1.145 str FIX__SPACINGS__CONF__SPACINGS__SECTION = "spacings"

Section name.

This section controls the axle spacings options for the fix__spacings core

3.1.1.146 str FIX__SPACINGS__CONF__SPACINGS__SPACING__REGEX = "spacing(\\d+)___m"

N-th predefined spacing.

These values define spacings. The values are axle spacings in metres.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.147 str KMIL_CONF_BRIDGE_SECTION = "bridge"

Section name.

This section controls the bridge options for the kmil core

3.1.1.148 str KMIL_CONF_BRIDGE_THICKNESS = "thickness___m"

Defines the bridge thickness.

Used for modelling the IL peak

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.149 str KMIL_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the kmil core

3.1.1.150 str KMIL_CONF_GLOBAL_ALLOW_MP_KEY = "allow_MP"

Allows IL calculation from MP events.

This is incompatible with both KMIL_CONF_GLOBAL_FIXED_AXLE_WEIGHT_RATIO_KEY and KMIL_CONF_NUMERICAL_OPTIMISE_AD, and will be switched off if any of the other options are enabled.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.151 str KMIL_CONF_GLOBAL_ALLOW_CARS_KEY = "allow_cars"

Allows IL calculation from events containing cars.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.152 str KMIL_CONF_GLOBAL_AUTOGROUP_KEY = "autogroup___m"

Axle autogroup distance in metres.

Automatically groups axles if they are less than this distance apart. All the axles in a group will have the same weight.

Type: float
System default: 0 (disabled)
Template default: UNKNOWN

3.1.1.153 str KMIL_CONF_GLOBAL_AXLE_GROUPS_KEY = "axle_groups"

Limits the vehicle configurations from which the IL is calculated.

An example would be `11,113`, which would enable calculation only from two-axle vehicles and semi-trailers.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.154 str KMIL_CONF_GLOBAL_AXLE_WEIGHT_RATIO_KEY = "axle_weight_ratio"

Defines ratio between axle weights.

This is useful when calculcating IL from a vehicle with known axle weights. In this case the algorithm does not need to fit individual axle weights and fits only the GVW, resulting in a better IL.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.155 str KMIL_CONF_GLOBAL_CHANNELS_KEY = "channels"

Defines the channels used for IL calculation.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.156 str KMIL_CONF_GLOBAL_DUMP_UNUSED_EVENTS_KEY = "dump_unused_events"

Determines whether the unused events are dumped.

If false, the events from which IL was not calculated are just ignored.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.157 str KMIL_CONF_GLOBAL_FIXED_AXLE_WEIGHT_RATIO_KEY = "fixed_axle_weight_ratio"

Enables fixed axle weight ratio.

This is incompatible with KMIL_CONF_GLOBAL_ALLOW_MP_KEY

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.158 str KMIL_CONF_GLOBAL_FLATTEN_LENGTH_KEY = "flatten_length__s"

Specifies the "flattening" interval in seconds.

If this is non-zero, then both ends of signal used to calculate the IL will be "flattened" - multiplied by a linear function that is 0 at both ends of the IL and 1 at a distance of KMIL_CONF_GLOBAL_FLATTEN_LENGTH_KEY towards the peak of the IL.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.159 str KMIL_CONF_GLOBAL_LANE_KEY = "lane"

Lane for which the IL is calculated.

If this is left unspecified, ILs for all lanes will be calculated.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.160 str KMIL_CONF_GLOBAL_MIN_AMPLITUDE_KEY = "min_amplitude___V"

Minimum signal amplitude for IL calculation in volts.

The sum of channels specified in KMIL_CONF_GLOBAL_CHANNELS_KEY must be at least this value for the IL calculation to proceed.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.161 str KMIL_CONF_GLOBAL_NUMBER_OF_SAMPLES_FOR_SMOOTHING_KEY = "number_of_samples_for_smoothing"

Number of samples for signal smoothing.

If non-zero, the signal will be smoothed by a centred moving average before IL calculation.

Type: unsigned
System default: 0
Template default: UNKNOWN

3.1.1.162 str KMIL_CONF_GLOBAL_SUBTRACT_LINEAR_TREND_MAX_DELTA_KEY = "subtract_linear_trend_max_delta___%"

Maximum change allowed for linear trend subtraction in percent of the maximum signal.

The default value does not place any limits on the linear trend subtraction.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.163 str KMIL_CONF_GLOBAL_SUBTRACT_LOCAL_LINEAR_TREND_ENABLED_KEY = "subtract_local_linear_trend_enabled"

Enable subtraction of local linear trend.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.164 str KMIL_CONF_GLOBAL_SUBTRACT_LOCAL_LINEAR_TREND_SECONDS_KEY = "subtract_local_linear_trend___s"

Local linear trend subtraction interval.

If subtraction is enabled, a length of signal just before and just after the vehicle-on-bridge interval is summed and a linear function between these two points subtracted from the signal before using it in the calculation.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.165 str KMIL_CONF_GLOBAL_SUBTRACT_GLOBAL_LINEAR_TREND↩_ENABLED_KEY = "subtract_global_linear_trend_enabled"

Enable subtraction of global linear trend.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.166 str KMIL_CONF_GLOBAL_SUBTRACT_GLOBAL_LINEAR_TREND↩_SECONDS_KEY = "subtract_global_linear_trend___s"

Global linear trend subtraction interval.

If subtraction is enabled, a length of signal at the start and at the end of the whole event is summed and a linear function between these two points subtracted from the signal before using it in the calculation.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.167 str KMIL_CONF_GLOBAL_WRITE_TEXT_FILE_KEY = "write_text_file"

Write the calculated ILs as a text file.

If enabled an ASCII tab- and comma-delimited file, suitable for import into MATLAB, is written. The fields are:

- 1..7: Interval timestamp in the format YYYY,MM,DD,HH,MM,SS,mmm
- 8: Axle groups
- 9..15: The 6 measured temperatures (T1 through T5 and Ti)
- 16..: The ordinates of the IL, one per centimetre

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.168 str KMIL_CONF_NUMERICAL_SECTION = "numerical"

Section name.

This section controls the numerical options for the kmil core

3.1.1.169 str KMIL_CONF_NUMERICAL_AUTO_INTERVALS = "auto_intervals"

Enable minimisaton of calculation intervals.

The idea is that:

- the part to the left of the peak of the IL is completely defined by the part of the signal from the start of the signal to the position of the left-most peak and

- the prt to the right of the peak of the IL is completely defined by the part of the signal from the rightmost peak to the end of the signal,

so in principle the calculation could be performed only on those two intervals. This has been partly tested during development of this core, but has never been used in practice.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.170 str KMIL_CONF_NUMERICAL_CALC_METHOD_KEY = "calc_method"

Calculation method.

This parameter has two legal values:

- 1: Classical MkII equidistant calculation
- 2: Cubic spline interpolation

Type: unsigned
System default: 2
Template default: UNKNOWN

3.1.1.171 str KMIL_CONF_NUMERICAL_CHISQR_FTOL_KEY = "chisqr_ftol"

χ^2 tolerance for iteration termination

IL calculation is a non-linear optimisation process. When the relative change of χ^2 changes by less than this ammount between iterations, the approximation is deemed acceptable and iteration stopped.

Type: float
System default: 1e-2
Template default: UNKNOWN

3.1.1.172 str KMIL_CONF_NUMERICAL_INTERP_METHOD_KEY = "interp_method"

Interpolation method.

This parameter has two legal values:

- 0: Linear interpolation between points
- 1: Cubic spline interpolation between points

If the KMIL_CONF_NUMERICAL_CALC_METHOD_KEY is set to 2 (cubic splines), this parameter is automatically set to 1.

Type: unsigned
System default: 1
Template default: UNKNOWN

3.1.1.173 str KMIL_CONF_NUMERICAL_ITERATION_DIAGS_KEY = "iteration_diags"

Controls whether iteration diagnostics are written to the event.

Note that this can produce very large events and should only be used for development abd debugging purposes.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.174 str KMIL_CONF_NUMERICAL_LINMIN_FTOL_KEY = "linmin_ftol"

Numerical tolerance for the interal linear minimisation step of optimisation.

Type: float
System default: 1e-2
Template default: UNKNOWN

3.1.1.175 str KMIL_CONF_NUMERICAL_MODEL_PEAK = "model_peak"

Controls whether the IL peak is modelled.

If true, the peak is modelled as an arc, using 5 points, where the horizontal distance between the outermost points is equal to the KMIL_CONF_BRIDGE_THICKNESS value. If KMIL_CONF_NUMERICAL_MODEL_PEAK_EDGE is false, the arc subtends 90°.

If false, the peak is represented as a single point.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.176 str KMIL_CONF_NUMERICAL_MODEL_PEAK_EDGE = "model_peak_edge"

Controls whether the IL peak edge is modelled.

If true, KMIL_CONF_NUMERICAL_MODEL_PEAK is automatically turned on and the peak is modelled as a part of a circle using 5 points. The angle subtended by the arc depends on the derivatives of the adjoining lines.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.177 str KMIL_CONF_NUMERICAL_MODEL_SUPPORT_WIDTH = "model_support_width"

Controls whether the support width is modelled.

If true, there are additional two points placed to the left and right of each support. The distance of the points from the support point is taken from KMIL_CONF_BRIDGE_THICKNESS.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.178 str KMIL_CONF_NUMERICAL_MULTIPLE_SPLINES = "multiple_splines"

Controls whether multiple splines are stitched together.

This is a remnant of development and is currently not used.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.179 str KMIL_CONF_NUMERICAL_NUMBER_OF_OPT_POINTS_KEY = "number_of_opt_points"

Number of optimisation points for the equidistant calculation method,.

This is a required parameter only if KMIL_CONF_NUMERICAL_CALC_METHOD_KEY is set to equidistant mode. For the spline method it is ignored.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.180 str KMIL_CONF_NUMERICAL_OPTIMISE_AD = "optimise_axle_distance"

Controls whether the axle distance is optimised.

The speed of the vehicle is allowed to vary, compensating for possible errors in speed calculation. This is incompatible with KMIL_CONF_GLOBAL_ALLOW_MP_KEY and will turn the other option off.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.181 str KMIL_CONF_NUMERICAL_OPTIMISE_LEFT_BOUNDARY_POS↩ ITION = "optimise_left_boundary_position"

Controls whether the left boundary position is optimised.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.182 str KMIL_CONF_NUMERICAL_OPTIMISE_LEFT_SUPPORT_POSIT↩ IONS = "optimise_left_support_positions"

Controls whether the left support positions are optimised.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.183 str KMIL_CONF_NUMERICAL_OPTIMISE_PEAK_HEIGHT = "optimise_peak_height"

Controls whether the peak height is optimised.

This has the effect of representing the peak with an elliptical arc instead of a circular arc

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.184 str KMIL_CONF_NUMERICAL_OPTIMISE_PEAK_POSITION = "optimise_peak_position"

Controls whether the peak position is optimised.

Type: bool
System default: false

Template default: UNKNOWN

3.1.1.185 str KMIL_CONF_NUMERICAL_OPTIMISE_PEAK_WIDTH = "optimise_peak_width"

Controls whether the peak width is optimised.

This multiplies the KMIL_CONF_BRIDGE_THICKNESS by a factor between 2 and 0.5 and thus widens or narrows the arc representing the peak.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.186 str KMIL_CONF_NUMERICAL_OPTIMISE_RIGHT_BOUNDARY_PO↩SITION = "optimise_right_boundary_position"

Controls whether the right boundary position is optimised.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.187 str KMIL_CONF_NUMERICAL_OPTIMISE_RIGHT_SUPPORT_POSI↩TIONS = "optimise_right_support_positions"

Controls whether the right support positions are optimised.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.188 str KMIL_CONF_DEFAULT_IL_SECTION_REGEX = "^default_il_lane\\d+$"

Section name.

A regular expression. An example of a match is `default_il_lane1`.

3.1.1.189 str KMIL_CONF_DEFAULT_IL_SECTION_DELIMITERS = "_adefilntu"

Section name separators.

Used for for parsing the section name

3.1.1.190 str KMIL_CONF_DEFAULT_IL_LEFT_BOUNDARY_KEY = "left_boundary___m"

The position of the leftmost point of the IL.

Defined in the Coordinate System.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.191 str KMIL_CONF_DEFAULT_IL_LEFT_DERIVATIVE_KEY = "left_derivative"

The derivative at the leftmost point of the IL.

If this value is not present or `1.e99`, the derivative is arbitrary (not fixed), Otherwise it is forced to this value.

Type: float
System default: 1.e99
Template default: UNKNOWN

3.1.1.192 str KMIL_CONF_DEFAULT_IL_LEFT_SUPPORT_POSITIONS_KEY = "left_support_positions___m"

A list of support positions between the left boundary and the peak of the IL.

Defined in the Coordinate System. Can be empty for a single-span bridge.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.193 str KMIL_CONF_DEFAULT_IL_PEAK_POSITION_KEY = "peak_position___m"

The position of the peak of the IL.

Defined in the Coordinate System. This should generally be left at its default value. If the peak of the IL and the peak in signals do not coincide, use VEHICLE_AD_CONF_LANE_AD_CORRE↩ CTION_KEY or VEHICLE_FAD_CONF_LANE_ADMP_CORRECTION_KEY to shift the entire IL.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.194 str KMIL_CONF_DEFAULT_IL_RIGHT_BOUNDARY_KEY = "right_boundary___m"

The position of the rightmost point of the IL.

Defined in the Coordinate System.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.195 str KMIL_CONF_DEFAULT_IL_RIGHT_DERIVATIVE_KEY = "right_derivative"

The derivative at the rightmost point of the IL.

If this value is not present or `1.e99`, the derivative is arbitrary (not fixed), Otherwise it is forced to this value.

Type: float
System default: 1.e99
Template default: UNKNOWN

3.1.1.196 str KMIL_CONF_DEFAULT_IL_RIGHT_SUPPORT_POSITIONS_KEY = "right_support_positions___m"

A list of support positions between the peak of the IL and the right boundary.

Defined in the Coordinate System. Can be empty for a single-span bridge.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.197 str KMIL_CONF_DEFAULT_IL_OPT_POINTS_KEY = "opt_points"

A list of optimisation point positions.

The floating point values in this list have a special meaning.

The integer part of the value, including the sign, defines the interval of the IL:

- -0 is the interval between the peak and the first support to the left of the peak (or the left boundary for a single-span bridge),
- +0 is the interval between the peak and the first support (or boundary) to the right of the peak,
- -1 is the interval between the first and second supports to the left of the peak,
- +1 is the interval between the first and second supports to the right of the peak,
- -2 is the interval between second and third supports to the left of the peak,
- ...

The fracional part of the value defines the position of the optimisation point in the interval in percent of the interval length. The position is always measured from the peak position.

For example, on a three-span bridge with the left span length of 10m, the centre span length of 20m and the right span length of 10m, a list `-1.5,-0.2,0.2,1.5` would define four points:

1. in the middle of the left span, 15m to the left of the peak.
2. 2m to the left of the peak
3. 2m to the right of the pak
4. in the middle of the right span, 15m to the right of the peak.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.198 str M1WEIGH_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the m1weigh core

3.1.1.199 str M1WEIGH_CONF_GLOBAL_AUTOGROUP_KEY = "autogroup___m"

Axle autogroup distance in metres.

Automatically groups axles if they are less than this distance apart. All the axles in a group will have the same weight.

See also M1WEIGH_CONF_GLOBAL_RECONSTRUCTION_DISABLES_AUTOGROUP_KEY.

Type: float
System default: 0 (disabled)
Template default: UNKNOWN

3.1.1.200 str M1WEIGH_CONF_GLOBAL_AUTO_TIMESTEP_ENABLED_KEY = "auto_timestep_enabled"

Enables automatic timestep calculation.

See Automatic Timestep

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.201 str M1WEIGH_CONF_GLOBAL_AUTO_TIMESTEP_LENGTH_KEY = "auto_timestep_length___m"

Automatic timestep length in metres.

See Automatic Timestep

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.202 str M1WEIGH_CONF_GLOBAL_CHANNELS_KEY = "channels"

List of channels for weighing.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.203 str M1WEIGH_CONF_GLOBAL_DIAGS_ONLY_KEY = "diags_only"

Use previously calcualated weights to generate diagnostic data.

This is incompatible with M1WEIGH_CONF_GLOBAL_NO_DIAGS_KEY.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.204 str M1WEIGH_CONF_GLOBAL_ISO_TIMESTAMP_KEY = "iso_timestamp"

Enables ISO 8601 timestamp format suitable for use with numpy.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.205 str M1WEIGH_CONF_GLOBAL_NO_DIAGS_KEY = "no_diags"

Does not generate diagnostic data.

Enabling this option can speed up calculations appreciably for longer bridges and especially for RailWIM systems.

This is incompatible with M1WEIGH_CONF_GLOBAL_DIAGS_ONLY_KEY.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.206 str M1WEIGH_CONF_GLOBAL_RECONSTRUCTION_ENABLED_KEY = "reconstruction_enabled"

Enables reconstruction.

See reconstruct.conf

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.207 str M1WEIGH_CONF_GLOBAL_RECONSTRUCTION_DIAGS_KEY = "reconstruction_diags"

If reconstruction is enabled, generate reconstruction diagnostics.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.208 str M1WEIGH_CONF_GLOBAL_RECONSTRUCTION_DISABLES_AU↩TOGROUP_KEY = "reconstruction_disables_autogroup"

If reconstruction is enabled, disable autogroup.

When adding an axle during reconstruction,, e.g., a missing middle axle in a tridem, the added axle can cause the axles to become part of a group, whereas previously they were considered as separate axles. This reduces the number of degrees of freedom in the system and can cause an "artificial" rejection of otherwise better results.

This option can prevent autogrouping in such cases.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.209 str M1WEIGH_CONF_GLOBAL_RECONSTRUCTION_MIN_CS_IMP↩ROVEMENT_KEY = "reconstruction_min_cs_improvement___%"

Minimum impovement in χ^2 for the reconstruction to be considered as potentially useful.

N.B.: The default value is 0, meaning that no reconstruction will be accepted. You need to set this value depending on the bridge.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.210 str M1WEIGH_CONF_GLOBAL_REWEIGH_VEHICLES_KEY = "reweigh_vehicles"

Reweigh vehicles.

This option causes the core to use weighed vehicles (instead of detected vehicles) as the input for weighing. Used in SiWIM-D when manually adjusting axle positions.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.211 str M1WEIGH__CONF__GLOBAL__SUBTRACT__LINEAR__TREND__MAX↩ __DELTA__KEY = "subtract__linear__trend__max__delta___%"

Maximum change allowed for linear trend subtraction in percent of the maximum signal.

The default value does not place any limits on the linear trend subtraction.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.212 str M1WEIGH__CONF__GLOBAL__SUBTRACT__LOCAL__LINEAR__TRE↩ ND__ENABLED__KEY = "subtract__local__linear__trend__enabled"

Enable subtraction of local linear trend.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.213 str M1WEIGH__CONF__GLOBAL__SUBTRACT__LOCAL__LINEAR__TRE↩ ND__SECONDS__KEY = "subtract__local__linear__trend___s"

Local linear trend subtraction interval.

If subtraction is enabled, a length of signal just before and just after the vehicle-on-bridge interval is summed and a linear function between these two points subtracted from the signal before using it in the calculation.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.214 str M1WEIGH__CONF__GLOBAL__SUBTRACT__GLOBAL__LINEAR__TR↩ END__ENABLED__KEY = "subtract__global__linear__trend__enabled"

Enable subtraction of global linear trend.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.215 str M1WEIGH__CONF__GLOBAL__SUBTRACT__GLOBAL__LINEAR__TR↩ END__SECONDS__KEY = "subtract__global__linear__trend___s"

Global linear trend subtraction interval.

If subtraction is enabled, a length of signal at the start and at the end of the whole event is summed and a linear function between these two points subtracted from the signal before using it in the calculation.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.216 str M1WEIGH_CONF_GLOBAL_WRITE_AVERAGE_DISTS_KEY = "write_average_dists"

Enable on-the-fly averaging of transverse distributions.

Type: bool
System default: true
Template default: UNKNOWN

3.1.1.217 str M1WEIGH_CONF_GLOBAL_WRITE_DISTS_KEY = "write_dists"

Enable writing individual transverse distributions.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.218 str M1WEIGH_CONF_GLOBAL_WRITE_DISTS_TXT_KEY = "write_dists_txt"

Enable writing transverse distributions to a text file.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.219 str M1WEIGH_CONF_DEFAULT_IL_SECTION = "default_il"

Section name.

This section controls the default influence line options for the m1weigh core

3.1.1.220 str M1WEIGH_CONF_DEFAULT_IL_XL_KEY = "xL___m"

Position of the left boundary of the default IL.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.221 str M1WEIGH_CONF_DEFAULT_IL_XR_KEY = "xR___m"

Position of the right boundary of the default IL.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.222 str M1WEIGH_CONF_STRIPS_SECTION = "strips"

Section name.

This section controls the strips options for the m1weigh core

3.1.1.223 str M1WEIGH_CONF_STRIPS_ENABLED_KEY = "enabled"

Enable use of strips.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.224 str M1WEIGH_CONF_STRIPS_USE_ONLY_FOR_MP_KEY = "use_only_for_MP"

Use strips only for MP events.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.225 str MATH_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the math core

3.1.1.226 str MATH_CONF_GLOBAL_SAVE_TSV_KEY = "save_tsv"

Save an ASCII tab-separated file of data after processing.

The file has the fields:

- 1: Time of the sample in seconds from the start of the event
- 2, ...: Samples for each channel of data

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.227 str MATH_CONF_GLOBAL_TSV_DOWNSAMPLE_KEY = "tsv_downsample"

Downsample before saving TSV.

Reduce sampling frequency by this much. Needs to be a power of 2.

Type: int
System default: 1
Template default: UNKNOWN

3.1.1.228 str MATH_CONF_INVERT_SECTION_REGEX = "^\\d+_invert$"

Section name.

This section controls the invert options for the math core

3.1.1.229 str MATH_CONF_INVERT_SECTION_CHANNELS_KEY = "channels"

List of channels to be inverted.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.230 str MATH_CONF_MULTIPLY_SECTION_REGEX = "^\\d+_multiply$"

Section name.

This section controls the multiply options for the math core.

3.1.1.231 str MATH_CONF_MULTIPLY_CHANNEL_FACTOR_REGEX = "^ch(\\d+)_factor$"

Factor for each channel.

The key name is a regular expression matching, e.g., `ch2_factor`.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.232 str MATH_CONF_MULTIPLY_REMOVE_OFFSET_KEY = "remove_offset"

Remove offset before multiplying.

If this is true, values have the offsets subtracted before multiplication and the offsets set to 0. If false, values and offsets are multiplied.

Type: float
System default: true
Template default: UNKNOWN

3.1.1.233 str MATH_CONF_MULTIPLY_ALL_SECTION_REGEX = "^\\d+_multiply_all$"

Section name.

This section controls the multiply_all options for the math core.

3.1.1.234 str MATH_CONF_MULTIPLY_ALL_FACTOR_KEY = "factor"

Multipication factor.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.235 str MATH_CONF_MULTIPLY_ALL_REMOVE_OFFSET_KEY = "remove_offset"

Remove offset before multiplying.

If this is true, values have the offsets subtracted before multiplication and the offsets set to 0. If false, values and offsets are multiplied.

Type: float
System default: true
Template default: UNKNOWN

3.1.1.236 str MATH_CONF_RELABEL_SECTION_REGEX = "^\\d+_relabel$"

Section name.

This section controls the relabel options for the math core.

3.1.1.237 str MATH_CONF_RELABEL_CLEAR_ALL_KEY = "clear_all"

Clear all labels before relabelling.

Type: float
System default: false
Template default: UNKNOWN

3.1.1.238 str MATH_CONF_RELABEL_CHANNEL_LABEL_REGEX = "^ch(\\d+)_label$"

New label for each channel.

The key name is a regular expression matching, e.g., `ch2_label`.

Type: string
System default: NONE
Template default: UNKNOWN

3.1.1.239 str MATH_CONF_RMS_SECTION_REGEX = "^\\d+_rms$"

Section name.

This section controls the RMS options for the math core.

3.1.1.240 str MATH_CONF_RMS_ENABLED_KEY = "enabled"

RMS enabled.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.241 str MATH_CONF_STATS_SECTION_REGEX = "^\\d+_write_stats$"

Section name.

This section controls the write_stats options for the math core. See documentation for math core for the formats of files produced by this option.

3.1.1.242 str MATH_CONF_STATS_ISO_TIMESTAMP_KEY = "iso_timestamp"

Enables ISO 8601 timestamp format suitable for use with numpy.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.243 str MATH_CONF_STATS_INVALID_CHANNELS_KEY = "invalid_channels"

Write invalid_channels file.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.244 str MATH_CONF_STATS_MAXMINAVG_KEY = "maxminavg"

Write maxminavg file.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.245 str MATH_CONF_STATS_OFFSET_KEY = "offset"

Write offset file.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.246 str MATH_CONF_STATS_DELTA_KEY = "delta"

Write delta file.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.247 str MODULES_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the SiWIM-E module construction

3.1.1.248 str MODULES_CONF_GLOBAL_MODULES_KEY = "modules"

A list of modules to be launched.

Each module listed here must have its own section, defining at least MODULES_CONF_MODULE_↩ CORE_KEY. All the modules must be linked together into an arborescence via the MODULES_CO↩ NF_MODULE_DOWNSTREAM_MODULES_KEY values.

Type: list of string
System default: NONE
Template default: UNKNOWN

3.1.1.249 str MODULES_CONF_MODULE_ARGS_KEY = "args"

Command-line arguments.

This option is only used if the module is launched externally.

Type: string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.250 str MODULES_CONF_MODULE_DOWNSTREAM_MODULES_KEY = "downstream_modules"

List of downstream modules.

Type: list of string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.251 str MODULES_CONF_MODULE_CAN_DROP_SWU_DATA_KEY = "can_drop_swu_data"

Module can drop data.

Normally a module that has not yet finished processing data will block upstream modules from sending it further data until the processing is finished.

By turning this option on, the upstream modules will discard data instead of waiting. This is useful for modules containing the kmil core, in which the calculations can take along time, but it is not critical that all the data is processed.

N.B.: This value is overridden in replay mode, where modules are not allowed to drop data.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.252 str MODULES_CONF_MODULE_CORE_KEY = "core"

Core name.

Type: string
System default: NONE
Template default: UNKNOWN

3.1.1.253 str MODULES_CONF_MODULE_ENABLED_KEY = "enabled"

Module is enabled.

If a module is disabled, it cannot accept data. If all modules that are downstream from a particular module are disabled, that module will automatically dump all the data to disk.

Type: bool
System default: true
Template default: UNKNOWN

3.1.1.254 str MODULES_CONF_MODULE_SAVE_SWU_DATA_KEY = "save_swu_data"

Filter for saving data.

This is a regular expression that determines the types of data that will get saved by this module, e.g., `event|il` would save events and influence lines and just pass the other data along the chain.

Note that some modules automatically write data. In those cases this filter has no effect.

Type: string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.255 str MP_FACTORS_CONF_GLOBAL_SECTION = "global"

Section name.

This section contains the global options for this file. See mp_factors.conf for details.

3.1.1.256 str MP_FACTORS_CONF_GLOBAL_NUMBER_OF_CHANNELS_KEY = "number_of_channels"

Number of channels.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.257 str MP_FACTORS_CONF_FACTORS_SECTION = "factors"

Section name.

This section contains the MP factors for this file. See mp_factors.conf for details.

3.1.1.258 str MP_FACTORS_CONF_FACTORS_FACTOR_REGEX = "ch(\\d+)_factor"

Factor for each channel.

The key name is a regular expression matching, e.g., `ch2_factor`.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.259 str MP_TIME_FACTORS_CONF_GLOBAL_SECTION = "global"

Section name.

This section contains the global options for this file. See mp_time_factors.conf for details.

3.1.1.260 str MP_TIME_FACTORS_CONF_GLOBAL_DATESTAMPS = "datestamps"

"Datestamps" of the correction points

The value is a list of dates in the format `YYYY-MM-DD`.

Type: list of string
System default: NONE
Template default: UNKNOWN

3.1.1.261 str MP_TIME_FACTORS_CONF_FACTORS_SECTON = "factors"

Section name.

This section contains the MP correction factors. See mp_time_factors.conf for details.

3.1.1.262 str MP_TIME_FACTORS_CONF_FACTORS_FACTOR_REGEX = "ch(\\d+)_f"

Correction values for channels.

The key name is a regular expression matching, e.g., `ch2_f`.

The value is a list factors. The lenght of the list must match the lenght of the list in MP_TIME_FACTORS_CONF_GLOBAL_DATESTAMPS.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.263 str OFFSET_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the offset core.

3.1.1.264 str OFFSET_CONF_GLOBAL_HOLD_KEY = "hold"

Hold offsets.

If this is true, the old values of offsets are used if the current block cannot have offsets calculated. This is useful for situations where the offset present, but very stable.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.265 str OFFSET_CONF_GLOBAL_MAX_DELTA_KEY = "max_delta___V"

Maximum delta in volts.

Specifies maximum signal change where offset calculation is allowed.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.266 str OFFSET_CONF_GLOBAL_NACQ_MODE_KEY = "nacq_mode"

NACQ mode.

`NACQ` files from the old MkII SiWIM system had their offsets calculated on-the-fly, This mode emulates MkII calculation by using the first 0.1s of each channel to calculate offset, regardless of the Δ of that channel.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.267 str OFFSET_CONF_GLOBAL_SUBINTERVAL_LENGTH_KEY = "subinterval_length___s"

Subinterval length in seconds.

Specifies the subinterval length on which offset will be calculated

Type: float
System default: 1
Template default: UNKNOWN

3.1.1.268 str OFFSET_CONF_GLOBAL_MAX_SUBINTERVALS_KEY = "max_subintervals"

Maximum number of subintervals.

Specifies the number of subintervals for which offset will be calculated.

Type: unsigned
System default: 1
Template default: UNKNOWN

3.1.1.269 str RECONSTRUCT_CONF_ITEM_SECTION_REGEX = "^item_\\d+$"

Section name.

This is a regular expression matching, e.g., `item_1`. See reconstruct.conf for details.

3.1.1.270 str RECONSTRUCT_CONF_ITEM_SECTION_ROOT = "item_"

Section name root.

Used for writing the configuration file.

3.1.1.271 str RECONSTRUCT_CONF_ITEM_SECTION_SEPARATORS = "eimt_"

Section name separators.

Used for for parsing the section name

3.1.1.272 str RECONSTRUCT_CONF_ITEM_COMMENT = "comment"

Item comment.

This value is not used by SiWIM-E and can be omitted. It is intended as a help for maintainers of the reconstruction definitions.

Type: string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.273 str RECONSTRUCT_CONF_ITEM_MAX_AXLE_DISTANCE_KEY = "max_axle_distance___m"

Maximum axle distances.

Upper limits for axle distances for this item

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.274 str RECONSTRUCT_CONF_ITEM_MIN_AXLE_DISTANCE_KEY = "min_axle_distance___m"

Minimum axle distances.

Lower limits for axle distances for this item

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.275 str RECONSTRUCT_CONF_ITEM_FORCE_KEY = "force"

Force this reconstruction.

If this is true, the reconstruction is performed regardless of the increase or decrease of χ^2 value.

Only one rule can be present in this case

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.276 str RECONSTRUCT_CONF_ITEM_RATIO_REFERENCE_AXLE_KEY = "ratio_reference_axle"

Reference axle for ratio rule.

If not defined, the ratio plays no role in selection

Type: integer
System default: NONE
Template default: UNKNOWN

3.1.1.277 str RECONSTRUCT_CONF_ITEM_RATIO_RATIOS_KEY = "ratio_ratios"

Ratio limits.

The rule is only applied if the ratios of specified axles' bare weights to the reference axle weight are within the limits

The value contains a list of semicolon (`;`) separated rules. Each rule consists of three comma-separated values. The first value is the axle index. The second and third values are the minimum and maximum limits for this rule to be applied.

Type: list of tuple<integer, float, float>
System default: NONE
Template default: UNKNOWN

3.1.1.278 str RECONSTRUCT_CONF_ITEM_RULE_REGEX = "^rule_\\d+$"

Reconstruction rule.

This is a regular expression matching, e.g., `rule_1`.

There can be an arbitrary number of rules for each item and all rules are applied in turn for a vehicle matching the axle distances for this item.

The rules contain a list of operations, separated by semicolons (`;`) and applied from the leftmost to the rightmost.

Each operation consists of a comma-separated pair, <axle, distance>. The sign of the axle value determines the type of operation. A negative axle value means "move the axle". In this case the distance is added or subtracted to the axle position. Examples:

- `-3,-0.5`: Move the third axle by 0.5m towards the front of the vehicle
- `-2,1.2`: Move the second axle by 1.2m towards the back of the vehicle

A positive axle value means "add an axle". In this case the sign of the distance determines whether the axle is added before of after the axle in question. Examples:

- `3,-1.2`: Add an axle 1.2m *before* the third axle
- `3,1.2`: Add an axle 1,2m *after* the third axle.

N.B.: The operations are applied from left to right, thus the axle numbers (except in the first operation) apply to the intermediate results. E.g., `rule_1=3,0.12;3,-1.2;4,+1.2` applied to a vehicle with three single axles (111) would define three operations:

1. Move the third axle by 0.12m towards the front of the vehicle (the vehicle still has 3 axles, 111)
2. Add an axle 1.2m before the third axle (the vehicle now has 4 axles, 112)
3. Add an axle 1.2m after the fourth axle (the vehicle now has 5 axles, 113)

The operations are not uniquely determined. E.g., the same rule could also be written as `rule_2=-3,0.12;3,+1.2;3,-1.2`, which would define three different operations, but which would result in a same vehicle:

1. Move the third axle by 0.12m towards the front of the vehicle (the vehicle still has 3 axles, 111)
2. Add an axle 1.2m after the third axle (the vehicle now has 4 axles, 112)
3. Add an axle 1.2m before the third axle (the vehicle now has 5 axles, 113)

Type: list of pair<integer, float>
System default: NONE
Template default: UNKNOWN

3.1.1.279 str RECONSTRUCT__CONF__ITEM__RULE__ROOT = "rule_"

Reconstruction rule name root.

Used for writing the configuration file.

3.1.1.280 str RECONSTRUCT__CONF__ITEM__RULE__SEPARATORS = "elru_"

Reconstruction rule name separators.

Used for for parsing the reconstruction rule name

3.1.1.281 str REPLAY__CONF__GLOBAL__SECTION = "global"

Section name.

This section controls the global options for replay setup.

3.1.1.282 str REPLAY__CONF__GLOBAL__INPUT__STAGE__KEY = "input_stage"

Input stage for reprocessing.

Type: unsigned
System default: 0
Template default: UNKNOWN

3.1.1.283 str REPLAY__CONF__GLOBAL__OUTPUT__STAGE__KEY = "output_stage"

Output stage for reprocessing.

Type: unsigned
System default: 1
Template default: UNKNOWN

3.1.1.284 str SIWIM_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for SiWIM-E.

3.1.1.285 str SIWIM_CONF_GLOBAL_SITE_KEY = "site"

Site name.

This is a string containing the name of the site. It is also the name of the directory containing all the files pertaining to that site. Thus the name shouldn't consist of any special characters other than `A-Z`, `a-z`, `0-9`, _ and `-`.

See also SIWIM_CONF_FS_DATA_ROOT_KEY.

Type: string
System default: NONE
Template default: UNKNOWN

3.1.1.286 str SIWIM_CONF_GLOBAL_RESTART_COMMAND_KEY = "restart_command"

Restart command.

This is used only on linux systems. The current method of restarting executables leaves behind zombie processes (quite unsanitary). This value will contain the restart command that will make a clean restart.

Type: string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.287 str SIWIM_CONF_ENGINE_SECTION = "engine"

Section name.

This section controls the siwim_engine.exe options for SiWIM-E.

3.1.1.288 str SIWIM_CONF_ENGINE_DUMP_MEMORY_STATE_KEY = "dump_memory_state"

Dump memory state.

If true, the current ammount of free memory will be written to a log file every minute. This was used in the development stages, where memory leaks occasionally occurred and should now be probably left off.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.289 str SIWIM_CONF_ENGINE_INTERNAL_CHAIN_KEY = "internal_chain"

Dump memory state.

If true, the chain of Modules is constructed within the main program, otherwise modules are spawned as separate executables.

Type: bool
System default: true
Template default: UNKNOWN

3.1.1.290 str SIWIM_CONF_ENGINE_DEFAULT_TIMEOUT_KEY = "default_timeout___s"

Default timeout.

If a module is inactive for longer than this number of seconds, it is considered to be "dead" and is restarted.

Type: unsigned
System default: 120
Template default: UNKNOWN

3.1.1.291 str SIWIM_CONF_ENGINE_DATATAX_SERVER = "datatax_server"

Enable DataTax server.

If true, SiWIM-E opens another TCP server on port 9090 for connection from DataTax software. Each time a vehicle is weighed, a plain-text line is sent over this connection. Lines are separated by a single newline character `'\n'`. The line consists of tab-delimited fields:

- 1: Header "VEH"
- 2: Timestamp of the vehicle in format `YYYYMMDDHHMMSSmmm`
- 3: Difference between vehicle and event timestamp in format `<seconds>.<milliseconds>`
- 4: Lane
- 5: Gross weight in kN to 1 decimal place
- 6: Classification
- 7: Speed in m/s to 1 decimal place
- 8: Flags in hexadecimal representation from `0` to `FFFFFFFF`
- 9: Number of axles N
- 10..(10+N-1): Individual axle weights in kN to one decimal place
- (10+N)..(10+2N-2): Axle spacings in metres to 2 decimal places

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.292 str SIWIM_CONF_ENGINE_GPS_SERVER = "gps_server"

Enable GPS server.

If true, SiWIM-E opens another TCP server on port 9091 for connection from GPS software running in a calibration vehicle. Currently this data is discarded, but in future it will be used to mark calibration vehicle passages.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.293 str SIWIM_CONF_FS_SECTION = "fs"

Section name.

This section controls the filesystem options for SiWIM-E.

3.1.1.294 str SIWIM_CONF_FS_DATA_ROOT_KEY = "data_root"

Data root.

This is the root directory for the site data. It can contain drive letter, e.g., `d:/sites`. Data from each separate site is contained within a subdirectory of the data root.

Type: string
System default: NONE
Template default: UNKNOWN

3.1.1.295 str SIWIM_CONF_FS_BACKUP_ROOTS_KEY = "backup_roots"

Backup roots.

Similar to SIWIM_CONF_FS_DATA_ROOT_KEY, but used for backups. E.g., if a USB disk or key is mounted in a SiWIM system as drive `e:`, and this entry contains `e:/backup`, then everything that is written to, e.g., `d:/sites`, will also get written to the backup directory.

The benefit is that once the measurement is completed, it is not necessary to copy all the data off the main disk, it is sufficient to remove the USB key and copy the data later in the office.

More than one backup root can be present. In this case backups will be written to all locations.

N.B.: If a system cannot write to a backup location, this is *not* considered as an error, it is merely logged in the site log files.

Type: list of string
System default: EMPTY_STRING
Template default: UNKNOWN

3.1.1.296 str SIWIM_CONF_LOGGER_SECTION = "logger"

Section name.

This section controls the siwim_logger.exe options for SiWIM-E.

3.1.1.297 str SIWIM_CONF_LOGGER_DISABLE_INFO_ECHO_KEY = "disable_info_echo"

Disable INFO echo.

If true, only log messages above log priority INFO, i.e., NOTICE, ERROR,... will get echoed to SiW↩ IM-F. This is mostly useful for reprocessing, where a large number of INFO messages will get sent to SiWIM-F and could slow down reprocessing speed.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.298 str SIWIM_CONF_PARALLEL_SECTION = "parallel"

Section name.

This section controls the parallel modules options for SiWIM-E.

Except for the two special values described below, all the names of values must correspond to names of modules (not cores). E.g., `speed=2` will create two parallel instances of the module speed.

N.B.: The name must match the module name exactly, i.e., it is not a regular expression.

3.1.1.299 str SIWIM_CONF_PARALLEL_REPLAY_KEY = "REPLAY"

Number of replay threads.

This value controls the number of threads reading from disk.

Type: unsigned
System default: 1
Template default: UNKNOWN

3.1.1.300 str SIWIM_CONF_PARALLEL_ENABLED_KEY = "ENABLED"

Enable parallel processing.

If this is false, then only a single instance of each module and the reader thread is launched. Used mostly for debugging purposes.

Type: bool
System default: true
Template default: UNKNOWN

3.1.1.301 str SIWIM_D_CONF_FORMS_SECTION = "forms"

Section name.

This section controls the form positions for SiWIM-D.

3.1.1.302 str SIWIM_D_CONF_FORMS_FRM_MAIN_HEIGHT = "frm_main_height"

Main form height.

Type: unsigned
System default: NONE

3.1.1.303 str SIWIM_D_CONF_FORMS_FRM_MAIN_LIST_COLUMN_WIDHTS = "frm_main_list_column_widths"

Main form vehicle list column widths.

Type: list of unsigned
System default: NONE

3.1.1.304 str SIWIM_D_CONF_FORMS_FRM_MAIN_LIST_HEIGHT = "frm_main_list_height"

Main form vehicle list height.

Type: unsigned
System default: NONE

3.1.1.305 str SIWIM_D_CONF_FORMS_FRM_MAIN_WIDTH = "frm_main_width"

Main form width.

Type: unsigned
System default: NONE

3.1.1.306 str SIWIM_D_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for SiWIM-D.

3.1.1.307 str SIWIM_D_CONF_GLOBAL_SITES_DIR_KEY = "sites_dir"

Sites directory.

Specifies SiWIM-D sites directory.

Type: string
System default: EMPTY_STRING

3.1.1.308 str SIWIM_F_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for SiWIM-F.

3.1.1.309 str SIWIM_F_CONF_GLOBAL_AUTOCACHE_KEY = "autocache"

Autocache.

If true, all files received from Engine are saved to sites/site_name directory.

Type: bool
System default: false

3.1.1.310 str SIWIM_F_CONF_GLOBAL_DISPLAY_MODULE_QUEUE_SIZE_↩ KEY = "display_module_queue_size"

Display module queue size.

Specifies whether the module configurator will display module queue sizes. Experts only.

Type: bool
System default: false

3.1.1.311 str SIWIM_F_CONF_GLOBAL_FORCE_UPDATE_KEY = "force_update"

Force update.

Forces SiWIM-F to update to the latest version on the update server even if the running version is the same.

Type: bool
System default: false

3.1.1.312 str SIWIM_F_CONF_GLOBAL_LANGUAGE_KEY = "language"

Language.

Language DLL to be used. Legal values are currently `croatian.dll`, `english.dll` and `slovenian.dll`.

Type: string
System default: english.dll

3.1.1.313 str SIWIM_F_CONF_GLOBAL_LOCAL_E_PARAMETERS_KEY = "local_e_parameters"

Local SiWIM-E parameters.

Command line parameters used when starting local SiWIM-E.

Type: string
System default: EMPTY_STRING

3.1.1.314 str SIWIM_F_CONF_GLOBAL_MDI_KEY = "mdi"

MDI mode.

Defines whether SiWIM-F will use multiple window or single window layout.

Type: bool
System default: false

3.1.1.315 str SIWIM_F_CONF_GLOBAL_RXTXLOG_VISIBLE_KEY = "rxtxlog_visible"

Display TCP sent and received logs.

If true, F log will contain TCP sent and received logs. Used for debugging.

Type: bool
System default: false

3.1.1.316 str SIWIM_F_CONF_GLOBAL_RECONNECT_BUTTON_KEY = "reconnect_button"

Reconnect button.

If true, the Reconnect button in the System menu is visible. Experts only.

Type: bool
System default: false

3.1.1.317 str SIWIM_F_CONF_GLOBAL_SITES_DIR_KEY = "sites_dir"

Sites directory.

Specifies SiWIM-F sites directory.

N.B.: If this value is the same as the value of SIWIM_CONF_FS_DATA_ROOT_KEY, SIWIM_F_CONF_GLOBAL_AUTOCACHE_KEY autocache is true and SiWIM-F connects to a site that exists in this directory, then SiWIM-F will overwrite any local modifications made to the `./conf` directory.

Type: string
System default: EMPTY_STRING

3.1.1.318 str SIWIM_F_CONF_GLOBAL_VEHICLE_CLASSES_KEY = "vehicle_classes"

Vehicle classes file.

Specifies vehicle_classes configuration file used when creating a new site.

Type: string
System default: NONE

3.1.1.319 str SIWIM_F_CONF_GLOBAL_UPDATE_CHANNEL_KEY = "update_channel"

Update channel.

When updating, update to stable (0) or testing (1) SiWIM-F version.

Type: unsigned
System default: 0

3.1.1.320 str SIWIM_F_CONF_SYSTEMS_SECTION = "systems"

Section name.

Entries here appear under in the menu "System|Connect" in SiWIM-F.

3.1.1.321 str SIWIM_F_CONF_SYSTEMS_SYSTEM_REGEX = "^system(\\d)$"

System entry.

A regular expression. An example of a match is `system1`. The system entries must be numbered sequentially, i.e., 1, 2,...

The entry consists of two to four comma-separated values

- 1: The system's name (displayed in the menu)
- 2: The system's LAN IP address
- 3: The system's optional VPN IP address
- 4: The system's optional other IP address

Type: list of string
System default: NONE

3.1.1.322 str SPEED_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the speed core

3.1.1.323 str SPEED_CONF_GLOBAL_MAX_SPEED_KEY = "max_speed___m/s"

Maximum absolute speed in metres per second.

Any speed above this value will be discarded as invalid.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.324 str SPEED_CONF_GLOBAL_MIN_SPEED_KEY = "min_speed___m/s"

Minimum absolute speed in metres per second.

Any speed below this value will be discarded as invalid.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.325 str SPEED_CONF_GLOBAL_NO_DIAGS_KEY = "no_diags"

Does not generate diagnostic data.

Enabling this option can speed up calculations appreciably for longer bridges, especially for RailWIM systems and when SPEED_CONF_GLOBAL_VEHICLE_MODE_KEY is enabled.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.326 str SPEED_CONF_GLOBAL_ACCELERATION_MODE_KEY = "acceleration_mode_enabled"

Enable acceleration mode.

In this mode individual vehicles' speeds are calculated with acceleration estimated and taken into account. The option SPEED_CONF_GLOBAL_VEHICLE_MODE_KEY must be enabled for this to work.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.327 str SPEED_CONF_GLOBAL_VEHICLE_MODE_KEY = "vehicle_mode_enabled"

Enable vehicle mode.

In this mode individual vehicles' speeds are calculated as opposed to general event speed for each lane.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.328 str SPEED_CONF_GLOBAL_PREVENT_NEGATIVE_SPEEDS_KEY = "prevent_negative_speeds"

Prevents negative speeds.

If true, the speed core only searches for positive speeds in the cross-correlation. This makes sense only on uni-directional bridges.

On bi-directional bridges, especially ones that are instrumented using axle detectors, negative speeds may mean that a vehicle, e.g., a car, was on lane 2, overtaking another vehicle on lane 1. In that case the speed on lane 2 would be negative with respect to normal traffic speeds, but the system of equations would still be consistent and would correctly account for the multiple presence. If the negative speeds were prevented, the car's speed (and the car itself) would be discarded and the multiple presence not detected.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.329 str SPEED_CONF_SPEED_SECTION_REGEX = "^lane\\d_speed\\d+$"

Section name.

This section controls the speed calculatin options The section name is a regular expression, matching, e.g., `lane1_speed1`.

3.1.1.330 str SPEED_CONF_SPEED_SECTION_DELIMITERS = "_adelnps"

Section name separators.

Used for parsing the section name

3.1.1.331 str SPEED_CONF_SPEED_DIGITAL_KEY = "digital"

Digital speed signals.

If true, the speed signals are assumed to be digital values produced by ADs.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.332 str SPEED_CONF_SPEED_ENABLED_KEY = "enabled"

Enable this speed calculation.

A simple way of enabling/disabling a particular speed calculation

Type: bool
System default: true
Template default: UNKNOWN

3.1.1.333 str SPEED_CONF_SPEED_FACTOR_KEY = "factor"

Correction factor.

Many times, especially on FAD installations, the effective distance between the SMPs is not the same as the measured distance.

on AD installations a similar effect is produced by pneumo tubes of unequal lenghts. In this case the effect depends on the speed, so it is important to keep the tube lenghts for both SMPs on each lane as close as possible to each other.

These errors have the effect of multiplying the calculated speed and thus the axle distances by a constant factor. By using a vehicle with known axle distances this factor can be deduced and a correction factor applied to speed calculations.

Type: float
System default: 1
Template default: UNKNOWN

3.1.1.334 str SPEED_CONF_SPEED_CH1_KEY = "smp1_ch"

SMP1 channel.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.335 str SPEED_CONF_SPEED_CH2_KEY = "smp2_ch"

SMP2 channel.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.336 str SPEED_CONF_SPEED_X1_KEY = "smp1_x__m"

Longitudinal position of SMP1.

Defined in the Coordinate System.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.337 str SPEED_CONF_SPEED_X2_KEY = "smp2_x__m"

Longitudinal position of SMP2.

Defined in the Coordinate System.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.338 str SPEED_CONF_SPEED_MA1_KEY = "smp1_moving_average__s"

Length of moving average for SMP1 in seconds.

Not used for digital SMP channels.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.339 str SPEED_CONF_SPEED_MA2_KEY = "smp2_moving_average__s"

Length of moving average for SMP2 in seconds.

Not used for digital SMP channels.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.340 str SPEED_CONF_SPEED_TRIGGER1_ABOVE_KEY = "smp1_trigger_above__V"

Trigger level for SMP1.

Only if SMP1 signal rises above this value (or below SPEED_CONF_SPEED_TRIGGER1_BELOW_KEY, if defined), is the speed calculation performed.

Not used for digital SMP channels.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.341 str SPEED_CONF_SPEED_TRIGGER2_ABOVE_KEY = "smp2_trigger_above__V"

Trigger level for SMP2.

Only if SMP2 signal rises above this value (or below SPEED_CONF_SPEED_TRIGGER2_BELOW_KEY, if defined), is the speed calculation performed.

Not used for digital SMP channels.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.342 str SPEED_CONF_SPEED_TRIGGER1_BELOW_KEY = "smp1_trigger_below___V"

Trigger level for SMP1.

Only if SMP1 signal drops below this value (or above SPEED_CONF_SPEED_TRIGGER1_ABOV←E_KEY, if defined), is the speed calculation performed.

Not used for digital SMP channels.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.343 str SPEED_CONF_SPEED_TRIGGER2_BELOW_KEY = "smp2_trigger_below___V"

Trigger level for SMP2.

Only if SMP2 signal drops below this value (or above SPEED_CONF_SPEED_TRIGGER2_ABOV←E_KEY, if defined), is the speed calculation performed.

Not used for digital SMP channels.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.344 str SPEED_CONF_SPEED_SUBTRACT_ENVELOPE1_KEY = "smp1_subtract_envelope"

Enables envelope subtraction for SMP1.

Not used for digital SMP channels.

N.B.: The SPEED_CONF_SPEED_TRIGGER1_BELOW_KEY needs to be set for the envelope subtraction to proceed.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.345 str SPEED_CONF_SPEED_SUBTRACT_ENVELOPE2_KEY = "smp2_subtract_envelope"

Enables envelope subtraction for SMP2.

Not used for digital SMP channels.

N.B.: The SPEED_CONF_SPEED_TRIGGER2_BELOW_KEY needs to be set for the envelope subtraction to proceed.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.346 str SPEED_CONF_SPEED_USE_IN_VEHICLE_MODE = "use_in_vehicle_mode"

Use this SMP pair in vehicle mode.

Only one pair of SMPs can be used in vehicle mode.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.347 str SPEED_CONF_VEHICLE_MODE_SECTION = "vehicle_mode"

Section name.

This section controls the Vehicle Mode options for the speed core

3.1.1.348 str SPEED_CONF_VEHICLE_MODE_MOVING_AVERAGE_POINTS↩ _KEY = "moving_average_points"

Number of points for moving average.

Since the speed, especially in RailWIM cannot vary arbitrarily between carraiges, this is used to smooth out the speed values by taking a centered moving average of speeds.

The value must be either 0 (no averaging, for road usage) or an an odd value. It is not advisable to use this together with the SPEED_CONF_VEHICLE_MODE_SPEED_ERROR_CORRECTION.

Type: unsigned
System default: 0
Template default: UNKNOWN

3.1.1.349 str SPEED_CONF_VEHICLE_MODE_SPEED_ERROR_CORRECTION = "speed_error_correction_enabled"

Enable speed error correction.

Since the speed, especially in RailWIM cannot vary arbitrarily between carriages, it is possible to check for any errors in speed calculation. If a speed of any single carriage deviates too much from the average speed of carriages surrounding it, it is replaced by the average speed of those carriages.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.350 str SPEED_CONF_VEHICLE_MODE_SPEED_ERROR_AVERAGE_P↩ OINTS_KEY = "speed_error_average_points"

Number of points for speed error correction average.

The value must be either 0 (no averaging) or an an odd value. It is not advisable to use this together with the SPEED_CONF_VEHICLE_MODE_MOVING_AVERAGE_POINTS_KEY.

Type: unsigned
System default: 0
Template default: UNKNOWN

3.1.1.351 str SPEED_CONF_VEHICLE_MODE_SPEED_ERROR_MAX_DEVIA↩ TION_KEY = "speed_error_max_deviation___%"

Max speed deviation in percent of the average.

If the speed of a single carriage deviates from the average by more than this, it is replaced by the average.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.352 str SPEED_CONF_VEHICLE_MODE_WRITE_DIAGS_KEY = "write_diags"

Enable writing of a diagnostic file.

If this is true, an ASCII tab-delimited file is written for each event. The file contains fields:

- 1: Serial number of the vehicle in the event
- 2: Timestamp of the vehicle in the format HHMMSS.mmm
- 3: Original speed
- 4: Corrected speed

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.353 str SPEED_CONF_VEHICLE_MODE_XL_KEY = "xL___m"

Defines the left edge for the cross-correlation.

When calculating the event speed, the entire signal from the SMPs is taken into account, while for vehicle speeds, only a portion is used. This parameter defines the position of the leftmost point, relative to the SMP position, defined in the Coordinate System.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.354 str SPEED_CONF_VEHICLE_MODE_XR_KEY = "xR___m"

Defines the right edge for the cross-correlation.

See SPEED_CONF_VEHICLE_MODE_XL_KEY for explanation.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.355 str STRIPS_CONF_GLOBAL_SECTION = "global"

Section name.

This section contains the global options for this file. See strips.conf for details.

3.1.1.356 str STRIPS_CONF_GLOBAL_NUMBER_OF_STRIPS_KEY = "number_of_strips"

Number of strips.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.357 str STRIPS_CONF_GLOBAL_STRIP_CHANNELS_ROOT = "channels_strip(\\d+)$"

Channels in N-th strip.

The name is a regular expression, matching, e.g., `channels_strip1`.

The value is a list of channels contained in the N-th strip. It is also possible to distribute a channel among two strips by preceeding the channel number with a ratio and a percent sign. E.g., the definitions:

- `channels_strip1=1,25%2`
- `channels_strip2=75%2,3`

would assign channel 1 to strip 1, channel 3 to strip 2, and would distribute 25% of channel 2 to strip 1 and 75% of channel 2 to strip 2.

N.B.: Channels for each strip must be defined, i.e., there must be the same number of `channels_↩ strip<N>` definitions as there are strips defined by STRIPS_CONF_GLOBAL_NUMBER_OF_S↩ TRIPS_KEY, and the union of strip channels must correspond to the channels in data and in the .dists file.

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.358 str TSPLIT_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the tsplit core

3.1.1.359 str TSPLIT_CONF_GLOBAL_COMPLEMENT_TRIGGER_KEY = "complement_trigger"

Complement trigger.

The normal mode of operation is to pass data containing time intervals in which there is something on the bridge.

For special purposes this operation can be complemented, so that the data passed contains those time intervals in which there is nothing on the bridge.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.360 str TSPLIT_CONF_GLOBAL_BLOCK_MODE_KEY = "block_mode"

Enables the block trigger mode.

In this mode the incoming blocks are individually examined for triggers and the resulting blocks are at least TSPLIT_CONF_BLOCK_BLOCK_LENGTH_KEY long, including TSPLIT_CONF_BLOC↩ K_PREPEND_KEY of prepend.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.361 str TSPLIT_CONF_GLOBAL_CHOP_MODE_KEY = "chop_mode"

Enables the chop mode.

In this mode the incoming blocks are chopped into blocks of a exactly TSPLIT_CONF_CHOP_BLO↩
CK_LENGTH_KEY length. Only block of full length are passed on, so a 3s block could be chopped into 3x1s blocks, 1x2s block (and the remaining 1s block thrown away), 1x3s block and requests for blocks larger than 3s would result in nothing being passed.

N.B.: This mode is only valid for events, as chopping a 1s block of ACQ data makes no sense.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.362 str TSPLIT_CONF_GLOBAL_MAX_EVENT_LENGTH_KEY = "max_event_length___s"

Maximum event length.

If one of the trigger channels drifts by more than the trigger level, that channel would be triggered forever and the resulting events could be arbitrarily long.

To prevent this, in case the event length exceeds a predetermined length, it is discarded.

N.B.: This value is ignored in the block trigger mode

Type: unsigned
System default: 120
Template default: UNKNOWN

3.1.1.363 str TSPLIT_CONF_GLOBAL_MIN_IDLE_TIME_KEY = "min_idle_time___s"

Minimum idle time between events.

If the interval between triggered parts of signals is less than this value, the triggered parts are considered as one event.

This value also detemines the length of untriggered data in the event before and after the triggered part.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.364 str TSPLIT_CONF_GLOBAL_NO_DIAGS_KEY = "no_diags"

Does not generate diagnostic data.

Enabling this option can speed up calculations appreciably for longer bridges and especially for RailWIM systems.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.365 str TSPLIT_CONF_GLOBAL_PSDP_MODE_KEY = "psdp_mode"

Enables the power spectrum density preprocessor mode.

This is similar to the TSPLIT_CONF_GLOBAL_COMPLEMENT_TRIGGER_KEY mode, but the output is not a block of continuous data, but is split up into blocks of predetermined length.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.366 str TSPLIT_CONF_GLOBAL_SAVE_TXT_FILES_KEY = "save_txt_files"

Save text representation of data.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.367 str TSPLIT_CONF_BLOCK_SECTION = "block"

Section name.

This section controls the block options for the tsplit core

3.1.1.368 str TSPLIT_CONF_BLOCK_BLOCK_LENGTH_KEY = "block_length___s"

Length of the event in seconds.

Events produced by block mode are at least this number of seconds long.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.369 str TSPLIT_CONF_BLOCK_PREPEND_KEY = "prepend___s"

Length of prepended data.

Prepend this many seconds before the first triggered block.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.370 str TSPLIT_CONF_BLOCK_UNCONDITIONAL_KEY = "unconditional"

Unconditional mode.

This is to be used for unconditionally producing blocks of a certain length. In this case everything except TSPLIT_CONF_BLOCK_BLOCK_LENGTH_KEY is ignored.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.371 str TSPLIT_CONF_CHOP_SECTION = "chop"

Section name.

This section controls the block options for the tsplit core

3.1.1.372 str TSPLIT_CONF_CHOP_BLOCK_LENGTH_KEY = "block_length___s"

Length of the event in seconds.

Events produced by chop mode are at exactly this number of seconds long.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.373 str TSPLIT_CONF_PSDP_SECTION = "psdp"

Section name.

This section controls the PSDP options for the tsplit core

3.1.1.374 str TSPLIT_CONF_PSDP_BLOCK_LENGTH_KEY = "block_length___s"

Length of the event in seconds.

Events produced by PSDP mode are exactly this number of seconds long.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.375 str TSPLIT_CONF_PSDP_MIN_LENGTH_KEY = "min_length___%"

Minimum lenght of untriggered data.

On bridges with heavy traffic loads it is sometimes hard to find long intervals of nothing on the bridge. By setting this value, a certain portion of triggered data can be included in the PSD blocks. E.g., `min_length___%=90` would mean that at most 10% of the data in the block can be triggered data for this block to be sent for the PSD calculation.

Type: unsigned
System default: 100
Template default: UNKNOWN

3.1.1.376 str TSPLIT_CONF_PSDP_DUMP_UNUSED_DATA_KEY = "dump_unused_data"

Dump unused data.

If, for whatever reason, the data unused for PSD is needed, this can enable dumping of this data.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.377 str TSPLIT_CONF_TRIGGER_ABOVE_SECTION = "trigger_above___V"

Section name.

This section contains the channels number and levels above which the data from those channels must rise in order to trigger the event.

3.1.1.378 str TSPLIT_CONF_TRIGGER_BELOW_SECTION = "trigger_below___V"

Section name.

This section contains the channels number and levels below which the data from those channels must fall in order to trigger the event.

3.1.1.379 str TSPLIT_CONF_TRIGGER_CHANNEL_KEY_BASE = "ch"

The base name for channel/level value in volts.

These values are located in TSPLIT_CONF_TRIGGER_ABOVE_SECTION and TSPLIT_CONF↩_TRIGGER_BELOW_SECTION and determine the values above/below which that particular channel is triggered.

E.g., `ch1=-2.1` in the TSPLIT_CONF_TRIGGER_BELOW_SECTION section would indicate that the value in the first channel needs to fall below -2.1V for it to be triggered.

Note that the same channel can be present in both sections. In this case it is checked against both levels.

3.1.1.380 str TSPLIT_CONF_CONDITIONS_SECTION = "conditions"

Section name.

This section cotains the conditions that must be true for the tsplit to trigger. Each of the conditions in the section is examined in turn and the union of all conditions is the final trigger (thus a single triggered condition will trigger the overall trigger).

3.1.1.381 str TSPLIT_CONF_CONDITIONS_CONDITION_KEY_BASE = "condition"

The base name for trigger conditions.

A list of channels that must trigger for this condition to trigger. All channels must trigger for this condition to trigger, i.e., the resulting trigger is the intersection of the individual channel triggers.

E.g., `condition1=1` would mean that channel 1 must trigger (the trigger level or levels are defined in the TSPLIT_CONF_TRIGGER_ABOVE_SECTION and TSPLIT_CONF_TRIGGER_BELOW_↩SECTION sections).

But `condition1=1,2,3` would mean that the condition is triggered in those regions where channels 1, 2 and 3 have all triggered at the same time.

3.1.1.382 str VEHICLE_AD_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the vehicle_ad core

3.1.1.383 str VEHICLE_AD_CONF_GLOBAL_MAX_AXLES_KEY = "max_axles"

Maximum number of axles in a vehicle.

If a vehicle with more than this number of axles is constructed, it is treated as an error and discarded.

Type: unsigned
System default: 12
Template default: UNKNOWN

3.1.1.384 str VEHICLE_AD_CONF_LANE_AD_SECTION_REGEX = "^lane\\d_ad\\d+$"

Section name.

This section controls the ADMP options The section name is a regular expression, matching, e.g., `lane1↩_ad1`.

Note that only one AD per lane is currenlty supported.

3.1.1.385 str VEHICLE_AD_CONF_LANE_AD_SECTION_DELIMITERS = "_adeln"

Section name separators.

Used for for parsing the section name

3.1.1.386 str VEHICLE_AD_CONF_LANE_AD_CORRECTION_KEY = "correction___m"

Correction of the AD position in metres.

Use this correction to place the peak of MP response at $x = 0$.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.387 str VEHICLE_AD_CONF_LANE_TUBE_LENGTH_KEY = "tube_length___m"

Length of pneumo tube.

This parameter is necessary to include the effect speed of sound, at which the pulse travels within a pneumo tube, on axle positions.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.388 str VEHICLE_AD_CONF_LANE_AD_X_KEY = "x___m"

ADMP location defined in the Coordinate System.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.389 str VEHICLE_FAD_CONF_GLOBAL_SECTION = "global"

Section name.

This section controls the global options for the vehicle_fad core

3.1.1.390 str VEHICLE_FAD_CONF_GLOBAL_MAX_AXLES_KEY = "max_axles"

Maximum number of axles in a vehicle.

If a vehicle with more than this number of axles is constructed, it is treated as an error and discarded.

Type: unsigned
System default: 12
Template default: UNKNOWN

3.1.1.391 str VEHICLE_FAD_CONF_GLOBAL_NO_DIAGS_KEY = "no_diags"

Does not generate diagnostic data.

Enabling this option can speed up calculations appreciably for longer bridges and especially for RailWIM systems.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.392 str VEHICLE_FAD_CONF_GLOBAL_CARRIAGE_MODE_KEY = "carriage_mode"

Enables carriage mode.

See Carriage Mode

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.393 str VEHICLE_FAD_CONF_GLOBAL_FORCE_SPLIT_AFTER_AXLE↩ _KEY = "force_split_after_axle"

Force the start of another vehicle after a certain number of axles.

This is purely for experimental reasons and should never be used in a live system.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.394 str VEHICLE_FAD_CONF_GLOBAL_SPLIT_SINGLE_AXLE_KEY = "split_single_axle"

Enables forced split of a single axle into a double axle.

See Axle Substitution

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.395 str VEHICLE_FAD_CONF_GLOBAL_SPLIT_SINGLE_AXLE_DISTA↩ NCE_KEY = "split_single_axle_distance___m"

Split single axle distance in metres.

Type: float
System default: none
Template default: UNKNOWN

3.1.1.396 str VEHICLE_FAD_CONF_LANE_ADMP_SECTION_REGEX = "^lane\\d_admp\\d+$"

Section name.

This section controls the ADMP options for one ADMP on one lane. The section name is a regular expression, matching, e.g., `lane1_admp2`.

3.1.1.397 str VEHICLE_FAD_CONF_LANE_ADMP_SECTION_DELIMITERS = "_adelmnp"

Section name separators.

Used for for parsing the section name

3.1.1.398 str VEHICLE_FAD_CONF_LANE_ADMP_AUTOSHIFT_KEY = "autoshift"

Enables autoshift for multiple ADMP channels.

See Multiple Channels per ADMP

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.399 str VEHICLE_FAD_CONF_LANE_ADMP_CHANNEL_FACTORS_KEY = "channel_factors"

Channel factors for combining several channels into one ADMP.

See Multiple Channels per ADMP

Type: list of float
System default: 1
Template default: UNKNOWN

3.1.1.400 str VEHICLE_FAD_CONF_LANE_ADMP_CH_KEY = "ch"

One or more channels to use for this ADMP.

See Multiple Channels per ADMP

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.401 str VEHICLE_FAD_CONF_LANE_ADMP_COMPARISON_FACTOR_↩ KEY = "comparison_factor"

Comparison factor for choosing between ADMPs.

See Multiple ADMPs

Type: float
System default: 1
Template default: UNKNOWN

3.1.1.402 str VEHICLE_FAD_CONF_LANE_ADMP_CORRECTION_KEY = "correction___m"

Correction of the ADMP position in metres.

Use this correction to place the peak of MP response at $x = 0$.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.403 str VEHICLE_FAD_CONF_LANE_ADMP_X_KEY = "x___m"

ADMP location defined in the Coordinate System.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.404 str VEHICLE_FAD_CONF_LANE_ADMP_ATS_CUTOFF_KEY = "ats_cutoff"

ATS cutoff for axle detection.

See ATS Algorithm

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.405 str VEHICLE_FAD_CONF_LANE_ADMP_ATS_START_KEY = "ats_start___%"

ATS starting level in percent of the maximum of the conditioned signal.

See ATS Algorithm

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.406 str VEHICLE_FAD_CONF_LANE_ADMP_ATS_STEP_KEY = "ats_step___%"

ATS step in percent of the maximum of the conditioned signal.

See ATS Algorithm

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.407 str VEHICLE_FAD_CONF_LANE_ADMP_ATS_STOP_KEY = "ats_stop___%"

ATS stopping level in percent of the maximum of the conditioned signal.

See ATS Algorithm

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.408 str VEHICLE_FAD_CONF_LANE_ADMP_AVERAGE1_KEY = "average1___m"

First averaging length in metres.

See ATS Algorithm

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.409 str VEHICLE_FAD_CONF_LANE_ADMP_AVERAGE1_TYPE_KEY = "average1_type"

First averaging type.

See Averaging

Legal values are:

- 0: Square window
- 1: Triangular window
- 2: Savitzky-Golay smoothing

Type: unsigned
System default: 0
Template default: UNKNOWN

3.1.1.410 str VEHICLE_FAD_CONF_LANE_ADMP_AVERAGE2_KEY = "average2___m"

Second averaging length in metres.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.411 str VEHICLE_FAD_CONF_LANE_ADMP_AVERAGE2_TYPE_KEY = "average2_type"

Second averaging type.

See Averaging

Legal values are:

- 0: Square window
- 1: Triangular window
- 2: Savitzky-Golay smoothing

Type: unsigned
System default: 0
Template default: UNKNOWN

3.1.1.412 str VEHICLE_FAD_CONF_LANE_ADMP_DIFFERENCE_TYPE_KEY = "difference_type"

Difference type.

See Alternate Difference

Legal values are:

- 0: Default difference
- 1: Alternate difference

Type: unsigned
System default: 0
Template default: UNKNOWN

3.1.1.413 str VEHICLE_FAD_CONF_LANE_ADMP_QMM_ENABLED_KEY = "qmm_enabled"

Enable QMM algorithm.

See Qualified Maximum/Minimum

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.414 str VEHICLE_FAD_CONF_LANE_ADMP_QMM_MIN_VALUE_KEY = "qmm_min_value___%"

QMM minimum value for a peak in percent of the maximum of the conditioned signal.

See Qualified Maximum/Minimum

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.415 str VEHICLE_FAD_CONF_LANE_ADMP_QMM_DELTA_KEY = "qmm_delta___%"

QMM Δ value in percent of the maximum of the conditioned signal.

See Qualified Maximum/Minimum

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.416 str VEHICLE_FAD_CONF_LANE_ADMP_SG1_ORDER_KEY = "sg1_order"

Order of the Savitzky-Golay smoothing polynomial for the first average.

See Averaging

Type: unsigned
System default: 2
Template default: UNKNOWN

3.1.1.417 str VEHICLE_FAD_CONF_LANE_ADMP_SG2_ORDER_KEY = "sg2_order"

Order of the Savitzky-Golay smoothing polynomial for the second average.

See Averaging

Type: unsigned
System default: 2
Template default: UNKNOWN

3.1.1.418 str VEHICLE_FAD_CONF_LANE_ADMP_SUBTRACT_ENVELOPE_↩ ENABLED_KEY = "subtract_envelope_enabled"

Enable envelope subtraction.

See Envelope Subtraction

Type: unsigned
System default: 2
Template default: UNKNOWN

3.1.1.419 str VEHICLE_FAD_CONF_LANE_ADMP_SUBTRACT_ENVELOPE_↩ BELOW_KEY = "subtract_envelope_below___%"

The level in percent of the minimum of the signal.

See Envelope Subtraction

The points at which the signal drop beneath this level define the triangle verteces L and R.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.420 str VEHICLE_FAD_CONF_LANE_ADMP_SUBTRACT_ENVELOPE_↩ SPLIT_KEY = "subtract_envelope_split___m"

Treat intervals longer than this as separate.

See Envelope Subtraction

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.421 str VEHICLE_FAD_CONF_LANE_ADMP_SUBTRA↩ CT_GLOBAL_LINEAR_TREND_ENABLED_KEY = "subtract_global_linear_trend_enabled"

Enable subtraction of global linear trend.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.422 str VEHICLE_FAD_CONF_LANE_ADMP_SUBTRACT_GLOBAL_LI↩ NEAR_TREND_SECONDS_KEY = "subtract_global_linear_trend___s"

Global linear trend subtraction interval.

If subtraction is enabled, a length of signal at the start and at the end of the whole event is summed and a linear function between these two points subtracted from the signal before using it in the calculation.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.423 str VEHICLE_FAD_CONF_LANE_ADMP_TPT_ENABLED_KEY = "tpt_enabled"

Enable TPT algorithm.

See Two-Phase Threshold

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.424 str VEHICLE_FAD_CONF_LANE_ADMP_TPT_THRESHOLD_KEY = "tpt_threshold___%"

TPT threshold in percent of the maximum of the conditioned signal.

See Two-Phase Threshold

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.425 str VEHICLE_FAD_CONF_LANE_ADMP_TRIGGER_ABOVE_KEY = "trigger_above___V"

Process signal if it rises above this value in volts.

At least one of VEHICLE_FAD_CONF_LANE_ADMP_TRIGGER_ABOVE_KEY and VEHICL↩ E_FAD_CONF_LANE_ADMP_TRIGGER_BELOW_KEY must be defined.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.426 str VEHICLE_FAD_CONF_LANE_ADMP_TRIGGER_BELOW_KEY = "trigger_below___V"

Process signal if it drops below this value in volts.

At least one of VEHICLE_FAD_CONF_LANE_ADMP_TRIGGER_ABOVE_KEY and VEHICL↩ E_FAD_CONF_LANE_ADMP_TRIGGER_BELOW_KEY must be defined.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.427 str VEHICLE_FAD_CONF_LANE_ADMP_USE_MAX_DIFF_KEY = "use_max_diff"

Use the location of the maximum of the conditioned signal as axle location.

By default the midpoint of the interval of the signal above the threshold is used as the axle location.

Type: bool
System default: false
Template default: UNKNOWN

3.1.1.428 str VEHICLE_CLASSES_GLOBAL_SECTION = "classes"

Section name.

This section contains the global options for this file. See vehicle_classes.conf for details.

Some of the values are there for backward compatibility only and are not used by the SiWIM-E system.

3.1.1.429 str VEHICLE_CLASSES_GLOBAL_COUNTRY_CODE_KEY = "country_code"

ISO Alpha-2 country code code.

This is used by SiWIM-D when reading bitmaps for displaying vehicle information.

Type: string
System default: NONE
Template default: UNKNOWN

3.1.1.430 str VEHICLE_CLASSES_GLOBAL_GROUP_MAX_DISTANCE_KEY = "group_max_distance___m"

Group maximum distance in metres.

When determining axle groups from axle spacings all axles further than this distance apart are considered single axles, otherwise they are a part of a group of axles.

Type: float
System default: 1.75
Template default: UNKNOWN

3.1.1.431 str VEHICLE_CLASSES_GLOBAL_MAX_AXLE_DISTANCE_KEY = "max_axle_distance___m"

Maximum axle distance in metres.

Any axles spaced more than this distance apart are considered as belonging to separate vehicles.

By default the `vehicle_classes.conf` file is parsed and this value determined from the data for subclasses, but for special purposes this can be overriden.

Type: float
System default: ∞
Template default: UNKNOWN

3.1.1.432 str VEHICLE_CLASSES_GLOBAL_MIN_AXLE_DISTANCE_KEY = "min_axle_distance___m"

Minimum axle distance in metres.

Any axle spaced less than this distance from the previous one is considered a bogus axle and discarded.

By default the `vehicle_classes.conf` file is parsed and this value determined from the data for subclasses, but for special purposes this can be overriden.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.433 str VEHICLE_CLASSES_GLOBAL_NO_CLASS_KEY = "unclassified_vehicle_subclass"

Unclassified vehicle subclass.

If axle distances do not place a vehicle in any of the defined subclasses, it is assigned this subclass number.

Type: unsigned
System default: 140
Template default: UNKNOWN

3.1.1.434 str VEHICLE_CLASSES_SUBCLASS_SECTION_REGEX = "^subclass_\\d+$"

Section name.

This is a regular expression matching, e.g., `subclass_113`.

The number defines the subclass ID assigned to vehicle whose axle distances match the ones specified in this item.

3.1.1.435 str VEHICLE_CLASSES_SUBCLASS_SECTION_ROOT = "subclass_"

Section name root.

Used for writing the configuration file.

3.1.1.436 str VEHICLE_CLASSES_SUBCLASS_SECTION_SEPARATORS = "abclsu_"

Section name separators.

Used for for parsing the section name

3.1.1.437 str VEHICLE_CLASSES_SUBCLASS_NUMBER_OF_AXLES_KEY = "number_of_axles"

Number of axles in this subclass.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.438 str VEHICLE_CLASSES_SUBCLASS_CATEGORY_KEY = "category"

Category for this subclass.

Type: unsigned
System default: NONE
Template default: UNKNOWN

3.1.1.439 str VEHICLE_CLASSES_SUBCLASS_LOWER_GVW_LIMIT_KEY = "lower_GVW_limit___kN"

Lower GVW limit in kN.

If the vehicle is identified by its axle distances as belonging to this subclass, but its GVW is below this limit, it is reclassified to subclass defined by VEHICLE_CLASSES_SUBCLASS_RECLASSIFY_DO↩WN_TO_SUBCLASS_KEY.

Type: float
System default: 0
Template default: UNKNOWN

3.1.1.440 str VEHICLE_CLASSES_SUBCLASS_MAX_AXLE_WEIGHT_KEY = "max_axle_weight___kN"

Maximum group or individual weights for this subclass.

This entry is a semicolon-separated list that defines the maximum individual axle weghts. The first part of a value is a +-separated list of axles whose weights are to be summed up and the second part is the maximum weight in kN. The list of axles can contain a single axle.

E.g., `1,7.5;2,20;3,20;2+3,35;4+5+6,70` for a six-axle truck would describe the following limits:

- The first axle can weigh at most 7.5 kN,

- The second and third axles can individually weigh at most 20 kN each, provided that the sum of their weights does not exceed 35 kN.
- The sum of last three axles can be at most 70 kN, regadless of their individual weights.

Type: list of pair<unsigned[+unsigned...], float>
System default: NONE
Template default: UNKNOWN

3.1.1.441 str VEHICLE_CLASSES_SUBCLASS_MAX_GVW_KEY = "max_GVW__kN"

Maximum GVW for this subclass.

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.442 str VEHICLE_CLASSES_SUBCLASS_MAX_AXLE_DISTANCE_KEY = "max_axle_distance__m"

Maximum axle distances in metres.

Vehicle axle distances are compared against this list and, if they not greater than the values in the list and not smaller than the values in the VEHICLE_CLASSES_SUBCLASS_MIN_AXLE_DISTANCE_KEY list, the vehicle is assigned this subclass.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.443 str VEHICLE_CLASSES_SUBCLASS_MIN_AXLE_DISTANCE_KEY = "min_axle_distance__m"

Minimum axle distances in metres.

Vehicle axle distances are compared against this list and, if they not smaller than the values in the list and not greater than the values in the VEHICLE_CLASSES_SUBCLASS_MAX_AXLE_DISTANCE_KEY list, the vehicle is assigned this subclass.

Type: list of float
System default: NONE
Template default: UNKNOWN

3.1.1.444 str VEHICLE_CLASSES_SUBCLASS_GROUP_WEIGHT_RATIO_KEY_ROOT = "group"

Read, but not used by SiWIM-E.

This will need to be investigated!

Type: float
System default: NONE
Template default: UNKNOWN

3.1.1.445 str VEHICLE_CLASSES_SUBCLASS_RECLASSIFY_DOWN_TO_SUBCLASS_KEY = "reclassify_down_to_subclass"

Reclassify light vehicles to this subclass.

If the vehicle is identified by its axle distances as belonging to this subclass, but its GVW is below VEHICLE_CLASSES_SUBCLASS_LOWER_GVW_LIMIT_KEY, and this value is not 0, it is reclassified to subclass defined by this value.

Type: unsigned
System default: 0
Template default: UNKNOWN

3.1.1.446 str VEHICLE_CLASSES_SUBCLASS_RECLASSIFY_UP_TO_SUBCLASS_KEY = "reclassify_up_to_subclass"

Reclassify heavy vehicles to this subclass.

If the vehicle is identified by its axle distances as belonging to this subclass, but its GVW is above VEHICLE_CLASSES_SUBCLASS_UPPER_GVW_LIMIT_KEY, and this value is not 0, it is reclassified to subclass defined by this value.

Type: unsigned
System default: 0
Template default: UNKNOWN

3.1.1.447 str VEHICLE_CLASSES_SUBCLASS_UPPER_GVW_LIMIT_KEY = "upper_GVW_limit___kN"

Lower GVW limit in kN.

If the vehicle is identified by its axle distances as belonging to this subclass, but its GVW is above this limit, it is reclassified to subclass defined by VEHICLE_CLASSES_SUBCLASS_RECLASSIFY_UP_TO_SUBCLASS_KEY.

Type: float
System default: ∞
Template default: UNKNOWN

3.1.1.448 str VEHICLE_CLASSES_SUBCLASS_TYRE_TYPE_KEY = "tyre_type"

Tyre type.

Used for calculating ESALs. The legal values are:

1. single
2. double
3. wide single

Type: list of unsigned
System default: NONE
Template default: UNKNOWN

3.2 local_definitions.h File Reference

3.2.1 Detailed Description

This file is used for develper-dependent compilation settings.

One of the settings `RELEASE_SETTINGS`, `DEVEL_MATEJ_SETTINGS` or `DEVEL_JAN_SETTINGS` must be defined, or the compilation will be aborted with an error.

The following definitions are available for controlling options:

- F_SW_PROT
  Turns on SiWIM-F Senselock protection
- E_SW_PROT
  Turns on SiWIM-E Senselock protection
- USE_ACQ_BMC_SENSELOCK
  "acq_bmc" core uses Senselock to generate zeroing addresses and commands.
  This will be removed once the protection is completely implemented
- EMULATE_ACQ_BMC_SENSELOCK
  Senselock is emulated for development purposes
- DEVELOPMENT_BUTTONS_AND_TABS
  Sets F's buttons and tabs to development settings
- MAX_MESSAGE_SIZE_B
  Used for RailWIM where you can have minutes-long events.
  An appropriate value is 512*1024*1024
- FLOAT2STR_SPEED_TO_ONE_DECIMAL_PLACE
  Return speed formatted to 1 decimal place in Convert::float2str()
- FLOAT2STR_DONT_TRIM_TRAILING_ZEROS
  Don't trim trailing zeroes and the decimal point in Convert::float2str() with units
- DONT_DROP_NON_LOCAL_SERVER_CONNECTION
  Used to prevent SWM_TCP_Server from dropping connection on full queue to non-local connections
- F_TEST_CONNECTIONS
  F sends SWM_MSG_Test_Connection message across idle log connection to detect dropped connections
- DIE_ON_NI_ERROR
  If any error occurs while getting data from a NI device, acq_mni core will cause the Engine to die with either a runtime error or a critical error instead of waiting forever
- ALLOW_LS_R_IN_LIVE_MODE
  If this is on, then ls-R will also work in live mode. Use this for vibration measurements where the disk workload is low and the PC's power is high.