The Open Master Hearing Aid (openMHA)

4.16.1

Plugin Developers' Manual

© 2005-2021 by HörTech gGmbH, Marie-Curie-Str. 2, D–26129 Oldenburg, Germany

The Open Master Hearing Aid (openMHA) – Plugin Developers' Manual
HörTech gGmbH
Marie-Curie-Str. 2
D–26129 Oldenburg

LICENSE AGREEMENT

This file is part of the HörTech Open Master Hearing Aid (openMHA)
Copyright © 2005 2006 2007 2008 2009 2010 2012 2013 2014 2015 2016 HörTech gGmbH.
Copyright © 2017 2018 2019 2020 2021 HörTech gGmbH.

openMHA is free software: you can redistribute it and/or modify it under the terms of the GNU Affero General Public License as published by the Free Software Foundation, version 3 of the License.

openMHA is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Affero General Public License, version 3 for more details.

You should have received a copy of the GNU Affero General Public License, version 3 along with openMHA. If not, see <http://www.gnu.org/licenses/>.

Contents

1 Overview **1**
1.1 Structure . . . 1
1.2 Platform Services and Conventions . . . 2

2 Deprecated List **4**

3 Module Documentation **4**
3.1 Concept of Variables and Data Exchange in the openMHA . . . 4
3.2 Writing openMHA Plugins. A step-by-step tutorial . . . 6
3.3 The MHA Framework interface . . . 22
3.4 Communication between algorithms . . . 23
3.5 Error handling in the openMHA . . . 28
3.6 The openMHA configuration language . . . 30
3.7 The openMHA Toolbox library . . . 31
3.8 Vector and matrix processing toolbox . . . 33
3.9 Complex arithmetics in the openMHA . . . 58
3.10 Fast Fourier Transform functions . . . 70

4 Namespace Documentation **78**
4.1 ac2lsl Namespace Reference . . . 78
4.2 ac_proc Namespace Reference . . . 78
4.3 acmon Namespace Reference . . . 79
4.4 acsave Namespace Reference . . . 79
4.5 addsndfile Namespace Reference . . . 79
4.6 ADM Namespace Reference . . . 81
4.7 audiometerbackend Namespace Reference . . . 82
4.8 AuditoryProfile Namespace Reference . . . 84
4.9 coherence Namespace Reference . . . 84
4.10 cpuload Namespace Reference . . . 85
4.11 dbasync_native Namespace Reference . . . 85
4.12 dc Namespace Reference . . . 86
4.13 dc_simple Namespace Reference . . . 86
4.14 delay Namespace Reference . . . 89
4.15 delaysum Namespace Reference . . . 89
4.16 delaysum_spec Namespace Reference . . . 89
4.17 double2acvar Namespace Reference . . . 90
4.18 DynComp Namespace Reference . . . 90
4.19 equalize Namespace Reference . . . 91
4.20 fader_wave Namespace Reference . . . 92
4.21 fftfbpow Namespace Reference . . . 92
4.22 fftfilter Namespace Reference . . . 92
4.23 fftfilterbank Namespace Reference . . . 94
4.24 fshift Namespace Reference . . . 94
4.25 fshift_hilbert Namespace Reference . . . 95
4.26 gain Namespace Reference . . . 95
4.27 gsc_adaptive_stage Namespace Reference . . . 95
4.28 gtfb_analyzer Namespace Reference . . . 96
4.29 level_matching Namespace Reference . . . 96
4.30 lsl2ac Namespace Reference . . . 96
4.31 matlab_wrapper Namespace Reference . . . 97

4.32 matrixmixer Namespace Reference . . . 98
4.33 mconv Namespace Reference . . . 98
4.34 MHA_AC Namespace Reference . . . 98
4.35 mha_error_helpers Namespace Reference . . . 99
4.36 MHA_TCP Namespace Reference . . . 100
4.37 mha_tcp Namespace Reference . . . 103
4.38 mhachain Namespace Reference . . . 104
4.39 MHAEvents Namespace Reference . . . 104
4.40 MHAFilter Namespace Reference . . . 104
4.41 MHAIOJack Namespace Reference . . . 109
4.42 MHAIOJackdb Namespace Reference . . . 110
4.43 MHAIOPortAudio Namespace Reference . . . 110
4.44 mhaioutils Namespace Reference . . . 111
4.45 MHAJack Namespace Reference . . . 111
4.46 MHAKernel Namespace Reference . . . 114
4.47 MHAMultiSrc Namespace Reference . . . 115
4.48 MHAOvlFilter Namespace Reference . . . 115
4.49 MHAOvlFilter::barkscale Namespace Reference . . . 116
4.50 MHAOvlFilter::FreqScaleFun Namespace Reference . . . 117
4.51 MHAOvlFilter::ShapeFun Namespace Reference . . . 120
4.52 MHAParser Namespace Reference . . . 122
4.53 MHAParser::StrCnv Namespace Reference . . . 128
4.54 MHAPlugin Namespace Reference . . . 134
4.55 MHAPlugin_Resampling Namespace Reference . . . 134
4.56 MHAPlugin_Split Namespace Reference . . . 135
4.57 MHASignal Namespace Reference . . . 136
4.58 MHASndFile Namespace Reference . . . 150
4.59 MHATableLookup Namespace Reference . . . 150
4.60 MHAUtils Namespace Reference . . . 150
4.61 MHAWindow Namespace Reference . . . 153
4.62 multibandcompressor Namespace Reference . . . 155
4.63 noise_psd_estimator Namespace Reference . . . 155
4.64 overlapadd Namespace Reference . . . 156
4.65 plingploing Namespace Reference . . . 156
4.66 PluginLoader Namespace Reference . . . 157
4.67 plugins Namespace Reference . . . 158
4.68 plugins::hoertech Namespace Reference . . . 158
4.69 plugins::hoertech::acrec Namespace Reference . . . 158
4.70 rmslevel Namespace Reference . . . 158
4.71 rohBeam Namespace Reference . . . 159
4.72 route Namespace Reference . . . 160
4.73 shadowfilter_begin Namespace Reference . . . 160
4.74 shadowfilter_end Namespace Reference . . . 161
4.75 smooth_cepstrum Namespace Reference . . . 161
4.76 smoothgains_bridge Namespace Reference . . . 161
4.77 testplugin Namespace Reference . . . 161
4.78 windnoise Namespace Reference . . . 161

5 Class Documentation **162**
5.1 ac2lsl::ac2lsl_t Class Reference . . . 162
5.2 ac2lsl::cfg_t Class Reference . . . 166
5.3 ac2lsl::save_var_base_t Class Reference . . . 169

© 2005-2021 HörTech gGmbH, Oldenburg

5.4 ac2lsl::save_var_t< T > Class Template Reference . . . 171
5.5 ac2lsl::save_var_t< mha_complex_t > Class Reference . . . 175
5.6 ac2lsl::type_info Struct Reference . . . 179
5.7 ac2osc_t Class Reference . . . 180
5.8 ac2wave_if_t Class Reference . . . 185
5.9 ac2wave_t Class Reference . . . 188
5.10 ac_mul_t Class Reference . . . 191
5.11 ac_proc::interface_t Class Reference . . . 196
5.12 acConcat_wave Class Reference . . . 199
5.13 acConcat_wave_config Class Reference . . . 202
5.14 acmon::ac_monitor_t Class Reference . . . 203
5.15 acmon::acmon_t Class Reference . . . 207
5.16 acPooling_wave Class Reference . . . 211
5.17 acPooling_wave_config Class Reference . . . 215
5.18 acsave::acsave_t Class Reference . . . 218
5.19 acsave::cfg_t Class Reference . . . 222
5.20 acsave::mat4head_t Struct Reference . . . 224
5.21 acsave::save_var_t Class Reference . . . 225
5.22 acSteer Class Reference . . . 228
5.23 acSteer_config Class Reference . . . 231
5.24 acTransform_wave Class Reference . . . 233
5.25 acTransform_wave_config Class Reference . . . 237
5.26 adaptive_feedback_canceller Class Reference . . . 240
5.27 adaptive_feedback_canceller_config Class Reference . . . 244
5.28 addsndfile::addsndfile_if_t Class Reference . . . 250
5.29 addsndfile::level_adapt_t Class Reference . . . 255
5.30 addsndfile::resampled_soundfile_t Class Reference . . . 257
5.31 addsndfile::sndfile_t Class Reference . . . 259
5.32 addsndfile::waveform_proxy_t Class Reference . . . 260
5.33 ADM::ADM< F > Class Template Reference . . . 261
5.34 ADM::Delay< F > Class Template Reference . . . 265
5.35 ADM::Linearphase_FIR< F > Class Template Reference . . . 268
5.36 adm_if_t Class Reference . . . 271
5.37 adm_rtconfig_t Class Reference . . . 275
5.38 algo_comm_t Struct Reference . . . 279
5.39 alsa_base_t Class Reference . . . 286
5.40 alsa_dev_par_parser_t Class Reference . . . 289
5.41 alsa_t< T > Class Template Reference . . . 291
5.42 altconfig_t Class Reference . . . 295
5.43 altplugs_t Class Reference . . . 300
5.44 analysepath_t Class Reference . . . 306
5.45 analysispath_if_t Class Reference . . . 310
5.46 attenuate20_t Class Reference . . . 313
5.47 audiometerbackend::audiometer_if_t Class Reference . . . 315
5.48 audiometerbackend::level_adapt_t Class Reference . . . 318
5.49 audiometerbackend::lnn3rdoct_t Class Reference . . . 320
5.50 audiometerbackend::signal_gen_t Class Reference . . . 322
5.51 audiometerbackend::sine_t Class Reference . . . 323
5.52 AuditoryProfile::fmap_t Class Reference . . . 324
5.53 AuditoryProfile::parser_t Class Reference . . . 325
5.54 AuditoryProfile::parser_t::ear_t Class Reference . . . 327

© 2005-2021 HörTech gGmbH, Oldenburg

5.55 AuditoryProfile::parser_t::fmap_t Class Reference . . . 329
5.56 AuditoryProfile::profile_t Class Reference . . . 331
5.57 AuditoryProfile::profile_t::ear_t Class Reference . . . 332
5.58 bbcalib_interface_t Class Reference . . . 333
5.59 calibrator_runtime_layer_t Class Reference . . . 335
5.60 calibrator_t Class Reference . . . 338
5.61 calibrator_variables_t Class Reference . . . 341
5.62 cfg_t Class Reference . . . 343
5.63 coherence::cohflt_if_t Class Reference . . . 347
5.64 coherence::cohflt_t Class Reference . . . 349
5.65 coherence::vars_t Class Reference . . . 353
5.66 combc_if_t Class Reference . . . 355
5.67 combc_t Class Reference . . . 357
5.68 comm_var_t Struct Reference . . . 359
5.69 complex_scale_channel_t Class Reference . . . 361
5.70 cpuload::cpuload_cfg_t Class Reference . . . 363
5.71 cpuload::cpuload_if_t Class Reference . . . 366
5.72 db_if_t Class Reference . . . 369
5.73 db_t Class Reference . . . 371
5.74 dbasync_native::db_if_t Class Reference . . . 373
5.75 dbasync_native::dbasync_t Class Reference . . . 376
5.76 dbasync_native::delay_check_t Class Reference . . . 379
5.77 dc::dc_if_t Class Reference . . . 380
5.78 dc::dc_t Class Reference . . . 382
5.79 dc::dc_vars_t Class Reference . . . 387
5.80 dc::dc_vars_validator_t Class Reference . . . 391
5.81 dc_simple::dc_if_t Class Reference . . . 392
5.82 dc_simple::dc_t Class Reference . . . 397
5.83 dc_simple::dc_t::line_t Class Reference . . . 401
5.84 dc_simple::dc_vars_t Class Reference . . . 403
5.85 dc_simple::dc_vars_validator_t Class Reference . . . 406
5.86 dc_simple::level_smoother_t Class Reference . . . 407
5.87 delay::interface_t Class Reference . . . 410
5.88 delaysum::delaysum_wave_if_t Class Reference . . . 412
5.89 delaysum::delaysum_wave_t Class Reference . . . 415
5.90 delaysum_spec::delaysum_spec_if_t Class Reference . . . 418
5.91 delaysum_spec::delaysum_t Class Reference . . . 420
5.92 doasvm_classification Class Reference . . . 421
5.93 doasvm_classification_config Class Reference . . . 425
5.94 doasvm_feature_extraction Class Reference . . . 427
5.95 doasvm_feature_extraction_config Class Reference . . . 430
5.96 double2acvar::double2acvar_t Class Reference . . . 433
5.97 dropgen_t Class Reference . . . 437
5.98 droptect_t Class Reference . . . 440
5.99 ds_t Class Reference . . . 444
5.100 dynamiclib_t Class Reference . . . 446
5.101 DynComp::dc_afterburn_rt_t Class Reference . . . 450
5.102 DynComp::dc_afterburn_t Class Reference . . . 453
5.103 DynComp::dc_afterburn_vars_t Class Reference . . . 456
5.104 DynComp::gaintable_t Class Reference . . . 458
5.105 equalize::cfg_t Class Reference . . . 463

© 2005-2021 HörTech gGmbH, Oldenburg

5.106 equalize::freqgains_t Class Reference . . . 464
5.107 example1_t Class Reference . . . 467
5.108 example2_t Class Reference . . . 469
5.109 example3_t Class Reference . . . 473
5.110 example4_t Class Reference . . . 477
5.111 example5_t Class Reference . . . 481
5.112 example6_t Class Reference . . . 482
5.113 example7_t Class Reference . . . 485
5.114 expression_t Class Reference . . . 486
5.115 fader_if_t Class Reference . . . 487
5.116 fader_wave::fader_wave_if_t Class Reference . . . 489
5.117 fader_wave::level_adapt_t Class Reference . . . 492
5.118 fftfbpow::fftfbpow_interface_t Class Reference . . . 494
5.119 fftfbpow::fftfbpow_t Class Reference . . . 497
5.120 fftfilter::fftfilter_t Class Reference . . . 499
5.121 fftfilter::interface_t Class Reference . . . 501
5.122 fftfilterbank::fftfb_interface_t Class Reference . . . 504
5.123 fftfilterbank::fftfb_plug_t Class Reference . . . 507
5.124 fshift::fshift_config_t Class Reference . . . 509
5.125 fshift::fshift_t Class Reference . . . 512
5.126 fshift_hilbert::frequency_translator_t Class Reference . . . 515
5.127 fshift_hilbert::hilbert_shifter_t Class Reference . . . 518
5.128 fw_t Class Reference . . . 523
5.129 fw_vars_t Class Reference . . . 532
5.130 gain::gain_if_t Class Reference . . . 533
5.131 gain::scaler_t Class Reference . . . 536
5.132 gsc_adaptive_stage::gsc_adaptive_stage Class Reference . . . 537
5.133 gsc_adaptive_stage::gsc_adaptive_stage_if Class Reference . . . 545
5.134 gtfb_analyzer::gtfb_analyzer_cfg_t Struct Reference . . . 549
5.135 gtfb_analyzer::gtfb_analyzer_t Class Reference . . . 553
5.136 gtfb_simd_cfg_t Class Reference . . . 556
5.137 gtfb_simd_t Class Reference . . . 562
5.138 gtfb_simple_rt_t Class Reference . . . 564
5.139 gtfb_simple_t Class Reference . . . 570
5.140 hanning_ramps_t Class Reference . . . 574
5.141 identity_t Class Reference . . . 576
5.142 iirfilter_t Class Reference . . . 578
5.143 io_alsa_t Class Reference . . . 579
5.144 io_asterisk_fwcb_t Class Reference . . . 584
5.145 io_asterisk_parser_t Class Reference . . . 589
5.146 io_asterisk_sound_t Class Reference . . . 596
5.147 io_asterisk_t Class Reference . . . 600
5.148 io_file_t Class Reference . . . 605
5.149 io_lib_t Class Reference . . . 611
5.150 io_parser_t Class Reference . . . 616
5.151 io_tcp_fwcb_t Class Reference . . . 621
5.152 io_tcp_parser_t Class Reference . . . 626
5.153 io_tcp_sound_t Class Reference . . . 633
5.154 io_tcp_sound_t::float_union Union Reference . . . 638
5.155 io_tcp_t Class Reference . . . 638
5.156 io_wrapper Class Reference . . . 643

© 2005-2021 HörTech gGmbH, Oldenburg

5.157 latex_doc_t Class Reference . . . 645
5.158 level_matching::channel_pair Class Reference . . . 648
5.159 level_matching::level_matching_config_t Class Reference . . . 651
5.160 level_matching::level_matching_t Class Reference . . . 654
5.161 levelmeter_t Class Reference . . . 658
5.162 lpc Class Reference . . . 661
5.163 lpc_bl_predictor Class Reference . . . 664
5.164 lpc_bl_predictor_config Class Reference . . . 668
5.165 lpc_burglattice Class Reference . . . 670
5.166 lpc_burglattice_config Class Reference . . . 673
5.167 lpc_config Class Reference . . . 676
5.168 lsl2ac::cfg_t Class Reference . . . 679
5.169 lsl2ac::lsl2ac_t Class Reference . . . 681
5.170 lsl2ac::save_var_t Class Reference . . . 686
5.171 matlab_wrapper::callback Class Reference . . . 693
5.172 matlab_wrapper::matlab_wrapper_rt_cfg_t Class Reference . . . 695
5.173 matlab_wrapper::matlab_wrapper_t Class Reference . . . 697
5.174 matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t Class Reference . . . 703
5.175 matlab_wrapper::types< T > Struct Template Reference . . . 711
5.176 matlab_wrapper::types< MHA_SPECTRUM > Struct Reference . . . 711
5.177 matlab_wrapper::types< MHA_WAVEFORM > Struct Reference . . . 712
5.178 matrixmixer::cfg_t Class Reference . . . 713
5.179 matrixmixer::matmix_t Class Reference . . . 714
5.180 mconv::MConv Class Reference . . . 717
5.181 MHA_AC::ac2matrix_helper_t Class Reference . . . 721
5.182 MHA_AC::ac2matrix_t Class Reference . . . 723
5.183 MHA_AC::acspace2matrix_t Class Reference . . . 725
5.184 MHA_AC::double_t Class Reference . . . 729
5.185 MHA_AC::float_t Class Reference . . . 731
5.186 MHA_AC::int_t Class Reference . . . 733
5.187 MHA_AC::spectrum_t Class Reference . . . 735
5.188 MHA_AC::stat_t Class Reference . . . 737
5.189 MHA_AC::waveform_t Class Reference . . . 739
5.190 mha_audio_descriptor_t Struct Reference . . . 741
5.191 mha_audio_t Struct Reference . . . 743
5.192 mha_channel_info_t Struct Reference . . . 744
5.193 mha_complex_t Struct Reference . . . 745
5.194 mha_complex_test_array_t Struct Reference . . . 746
5.195 mha_dblbuf_t< FIFO > Class Template Reference . . . 747
5.196 mha_direction_t Struct Reference . . . 755
5.197 mha_drifter_fifo_t< T > Class Template Reference . . . 756
5.198 MHA_Error Class Reference . . . 764
5.199 mha_fifo_lf_t< T > Class Template Reference . . . 767
5.200 mha_fifo_lw_t< T > Class Template Reference . . . 770
5.201 mha_fifo_posix_threads_t Class Reference . . . 774
5.202 mha_fifo_t< T > Class Template Reference . . . 776
5.203 mha_fifo_thread_guard_t Class Reference . . . 783
5.204 mha_fifo_thread_platform_t Class Reference . . . 784
5.205 mha_real_test_array_t Struct Reference . . . 788
5.206 mha_rt_fifo_element_t< T > Class Template Reference . . . 788
5.207 mha_rt_fifo_t< T > Class Template Reference . . . 790

© 2005-2021 HörTech gGmbH, Oldenburg

5.208 mha_spec_t Struct Reference . . . 794
5.209 mha_stash_environment_variable_t Class Reference . . . 796
5.210 MHA_TCP::Async_Notify Class Reference . . . 798
5.211 mha_tcp::buffered_socket_t Class Reference . . . 799
5.212 MHA_TCP::Client Class Reference . . . 801
5.213 MHA_TCP::Connection Class Reference . . . 803
5.214 MHA_TCP::Event_Watcher Class Reference . . . 811
5.215 MHA_TCP::OS_EVENT_TYPE Struct Reference . . . 814
5.216 MHA_TCP::Server Class Reference . . . 815
5.217 mha_tcp::server_t Class Reference . . . 819
5.218 MHA_TCP::sock_init_t Class Reference . . . 825
5.219 MHA_TCP::Sockaccept_Event Class Reference . . . 825
5.220 MHA_TCP::Sockread_Event Class Reference . . . 826
5.221 MHA_TCP::Sockwrite_Event Class Reference . . . 827
5.222 MHA_TCP::Thread Class Reference . . . 828
5.223 MHA_TCP::Timeout_Event Class Reference . . . 833
5.224 MHA_TCP::Timeout_Watcher Class Reference . . . 834
5.225 MHA_TCP::Wakeup_Event Class Reference . . . 836
5.226 mha_tictoc_t Struct Reference . . . 839
5.227 mha_wave_t Struct Reference . . . 840
5.228 mhachain::chain_base_t Class Reference . . . 842
5.229 mhachain::mhachain_t Class Reference . . . 846
5.230 mhachain::plugs_t Class Reference . . . 847
5.231 mhaconfig_t Struct Reference . . . 851
5.232 MHAEvents::connector_base_t Class Reference . . . 853
5.233 MHAEvents::connector_t< receiver_t > Class Template Reference . . . 856
5.234 MHAEvents::emitter_t Class Reference . . . 859
5.235 MHAEvents::patchbay_t< receiver_t > Class Template Reference . . . 861
5.236 MHAFilter::adapt_filter_param_t Class Reference . . . 863
5.237 MHAFilter::adapt_filter_state_t Class Reference . . . 864
5.238 MHAFilter::adapt_filter_t Class Reference . . . 866
5.239 MHAFilter::blockprocessing_polyphase_resampling_t Class Reference . . . 868
5.240 MHAFilter::complex_bandpass_t Class Reference . . . 872
5.241 MHAFilter::diff_t Class Reference . . . 876
5.242 MHAFilter::fftfilter_t Class Reference . . . 877
5.243 MHAFilter::fftfilterbank_t Class Reference . . . 882
5.244 MHAFilter::filter_t Class Reference . . . 887
5.245 MHAFilter::gamma_flt_t Class Reference . . . 892
5.246 MHAFilter::iir_filter_state_t Class Reference . . . 897
5.247 MHAFilter::iir_filter_t Class Reference . . . 898
5.248 MHAFilter::iir_ord1_real_t Class Reference . . . 902
5.249 MHAFilter::o1_ar_filter_t Class Reference . . . 906
5.250 MHAFilter::o1flt_lowpass_t Class Reference . . . 910
5.251 MHAFilter::o1flt_maxtrack_t Class Reference . . . 913
5.252 MHAFilter::o1flt_mintrack_t Class Reference . . . 915
5.253 MHAFilter::partitioned_convolution_t Class Reference . . . 917
5.254 MHAFilter::partitioned_convolution_t::index_t Struct Reference . . . 921
5.255 MHAFilter::polyphase_resampling_t Class Reference . . . 923
5.256 MHAFilter::resampling_filter_t Class Reference . . . 928
5.257 MHAFilter::smoothspec_t Class Reference . . . 930
5.258 MHAFilter::thirdoctave_analyzer_t Class Reference . . . 934

© 2005-2021 HörTech gGmbH, Oldenburg

5.259 MHAFilter::transfer_function_t Struct Reference 937
5.260 MHAFilter::transfer_matrix_t Struct Reference 940
5.261 MHAIOJack::io_jack_t Class Reference 941
5.262 MHAIOJackdb::io_jack_t Class Reference 948
5.263 MHAIOPortAudio::device_info_t Class Reference 956
5.264 MHAIOPortAudio::io_portaudio_t Class Reference 959
5.265 MHAIOPortAudio::stream_info_t Class Reference 965
5.266 MHAJack::client_avg_t Class Reference 967
5.267 MHAJack::client_noncont_t Class Reference 971
5.268 MHAJack::client_t Class Reference 974
5.269 MHAJack::port_t Class Reference 984
5.270 MHAKernel::algo_comm_class_t Class Reference 988
5.271 MHAKernel::comm_var_map_t Class Reference 994
5.272 MHAMultiSrc::base_t Class Reference 995
5.273 MHAMultiSrc::channel_t Class Reference 997
5.274 MHAMultiSrc::channels_t Class Reference 997
5.275 MHAMultiSrc::spectrum_t Class Reference 998
5.276 MHAMultiSrc::waveform_t Class Reference 1000
5.277 MHAOvlFilter::band_descriptor_t Class Reference 1001
5.278 MHAOvlFilter::barkscale::bark2hz_t Class Reference 1003
5.279 MHAOvlFilter::barkscale::hz2bark_t Class Reference 1004
5.280 MHAOvlFilter::fftfb_ac_info_t Class Reference 1005
5.281 MHAOvlFilter::fftfb_t Class Reference 1006
5.282 MHAOvlFilter::fftfb_vars_t Class Reference 1010
5.283 MHAOvlFilter::fscale_bw_t Class Reference 1014
5.284 MHAOvlFilter::fscale_t Class Reference 1016
5.285 MHAOvlFilter::fspacing_t Class Reference 1018
5.286 MHAOvlFilter::overlap_save_filterbank_analytic_t Class Reference 1022
5.287 MHAOvlFilter::overlap_save_filterbank_t Class Reference 1023
5.288 MHAOvlFilter::overlap_save_filterbank_t::vars_t Class Reference 1026
5.289 MHAOvlFilter::scale_var_t Class Reference 1028
5.290 MHAParser::base_t Class Reference 1030
5.291 MHAParser::base_t::replace_t Class Reference 1044
5.292 MHAParser::bool_mon_t Class Reference 1045
5.293 MHAParser::bool_t Class Reference 1047
5.294 MHAParser::c_ifc_parser_t Class Reference 1050
5.295 MHAParser::commit_t< receiver_t > Class Template Reference 1053
5.296 MHAParser::complex_mon_t Class Reference 1055
5.297 MHAParser::complex_t Class Reference 1057
5.298 MHAParser::entry_t Class Reference 1059
5.299 MHAParser::expression_t Class Reference 1060
5.300 MHAParser::float_mon_t Class Reference 1061
5.301 MHAParser::float_t Class Reference 1063
5.302 MHAParser::int_mon_t Class Reference 1066
5.303 MHAParser::int_t Class Reference 1069
5.304 MHAParser::keyword_list_t Class Reference 1071
5.305 MHAParser::kw_t Class Reference 1075
5.306 MHAParser::mcomplex_mon_t Class Reference 1079
5.307 MHAParser::mcomplex_t Class Reference 1081
5.308 MHAParser::mfloat_mon_t Class Reference 1083
5.309 MHAParser::mfloat_t Class Reference 1085

© 2005-2021 HörTech gGmbH, Oldenburg

5.310 MHAParser::mhaconfig_mon_t Class Reference 1088
5.311 MHAParser::mhapluginloader_t Class Reference 1090
5.312 MHAParser::mint_mon_t Class Reference 1095
5.313 MHAParser::mint_t Class Reference 1097
5.314 MHAParser::monitor_t Class Reference 1100
5.315 MHAParser::parser_t Class Reference 1102
5.316 MHAParser::range_var_t Class Reference 1108
5.317 MHAParser::string_mon_t Class Reference 1113
5.318 MHAParser::string_t Class Reference 1115
5.319 MHAParser::variable_t Class Reference 1118
5.320 MHAParser::vcomplex_mon_t Class Reference 1120
5.321 MHAParser::vcomplex_t Class Reference 1122
5.322 MHAParser::vfloat_mon_t Class Reference 1125
5.323 MHAParser::vfloat_t Class Reference 1127
5.324 MHAParser::vint_mon_t Class Reference 1130
5.325 MHAParser::vint_t Class Reference 1132
5.326 MHAParser::vstring_mon_t Class Reference 1134
5.327 MHAParser::vstring_t Class Reference 1136
5.328 MHAParser::window_t Class Reference 1138
5.329 mhaplug_cfg_t Class Reference 1142
5.330 MHAPlugin::cfg_node_t< runtime_cfg_t > Class Template Reference 1143
5.331 MHAPlugin::config_t< runtime_cfg_t > Class Template Reference 1146
5.332 MHAPlugin::plugin_t< runtime_cfg_t > Class Template Reference 1151
5.333 MHAPlugin_Resampling::resampling_if_t Class Reference 1157
5.334 MHAPlugin_Resampling::resampling_t Class Reference 1160
5.335 MHAPlugin_Split::domain_handler_t Class Reference 1163
5.336 MHAPlugin_Split::dummy_threads_t Class Reference 1169
5.337 MHAPlugin_Split::posix_threads_t Class Reference 1171
5.338 MHAPlugin_Split::split_t Class Reference 1176
5.339 MHAPlugin_Split::splitted_part_t Class Reference 1183
5.340 MHAPlugin_Split::thread_platform_t Class Reference 1188
5.341 MHAPlugin_Split::uni_processor_t Class Reference 1191
5.342 mhaserver_t Class Reference 1193
5.343 mhaserver_t::tcp_server_t Class Reference 1197
5.344 MHASignal::async_rmslevel_t Class Reference 1199
5.345 MHASignal::delay_spec_t Class Reference 1202
5.346 MHASignal::delay_t Class Reference 1203
5.347 MHASignal::delay_wave_t Class Reference 1206
5.348 MHASignal::doublebuffer_t Class Reference 1207
5.349 MHASignal::fft_t Class Reference 1211
5.350 MHASignal::hilbert_fftw_t Class Reference 1216
5.351 MHASignal::hilbert_t Class Reference 1218
5.352 MHASignal::loop_wavefragment_t Class Reference 1220
5.353 MHASignal::matrix_t Class Reference 1225
5.354 MHASignal::minphase_t Class Reference 1237
5.355 MHASignal::quantizer_t Class Reference 1238
5.356 MHASignal::ringbuffer_t Class Reference 1240
5.357 MHASignal::schroeder_t Class Reference 1244
5.358 MHASignal::spectrum_t Class Reference 1248
5.359 MHASignal::stat_t Class Reference 1254
5.360 MHASignal::subsample_delay_t Class Reference 1256

© 2005-2021 HörTech gGmbH, Oldenburg

5.361 MHASignal::uint_vector_t Class Reference 1259
5.362 MHASignal::waveform_t Class Reference 1263
5.363 MHASndFile::sf_t Class Reference 1276
5.364 MHASndFile::sf_wave_t Class Reference 1277
5.365 MHATableLookup::linear_table_t Class Reference 1278
5.366 MHATableLookup::table_t Class Reference 1283
5.367 MHATableLookup::xy_table_t Class Reference 1285
5.368 MHAWindow::bartlett_t Class Reference 1290
5.369 MHAWindow::base_t Class Reference 1291
5.370 MHAWindow::blackman_t Class Reference 1293
5.371 MHAWindow::fun_t Class Reference 1295
5.372 MHAWindow::hamming_t Class Reference 1296
5.373 MHAWindow::hanning_t Class Reference 1297
5.374 MHAWindow::rect_t Class Reference 1299
5.375 MHAWindow::user_t Class Reference 1300
5.376 multibandcompressor::fftfb_plug_t Class Reference 1302
5.377 multibandcompressor::interface_t Class Reference 1304
5.378 multibandcompressor::plugin_signals_t Class Reference 1307
5.379 nlms_t Class Reference 1309
5.380 noise_psd_estimator::noise_psd_estimator_if_t Class Reference 1312
5.381 noise_psd_estimator::noise_psd_estimator_t Class Reference 1315
5.382 noise_t Class Reference 1318
5.383 osc2ac_t Class Reference 1321
5.384 osc_server_t Class Reference 1324
5.385 osc_variable_t Class Reference 1326
5.386 overlapadd::overlapadd_if_t Class Reference 1330
5.387 overlapadd::overlapadd_t Class Reference 1335
5.388 parser_int_dyn Class Reference 1338
5.389 plingploing::if_t Class Reference 1340
5.390 plingploing::plingploing_t Class Reference 1344
5.391 plug_t Class Reference 1349
5.392 plug_wrapper Class Reference 1350
5.393 plug_wrapperl Class Reference 1352
5.394 plugin_interface_t Class Reference 1354
5.395 pluginbrowser_t Class Reference 1356
5.396 plugindescription_t Class Reference 1359
5.397 pluginlib_t Class Reference 1360
5.398 PluginLoader::config_file_splitter_t Class Reference 1362
5.399 PluginLoader::fourway_processor_t Class Reference 1365
5.400 PluginLoader::mhapluginloader_t Class Reference 1368
5.401 pluginloader_t Class Reference 1376
5.402 plugins::hoertech::acrec::acrec_t Class Reference 1377
5.403 plugins::hoertech::acrec::acwriter_t Class Reference 1382
5.404 proc_counter_t Class Reference 1388
5.405 rmslevel::mon_t Class Reference 1390
5.406 rmslevel::rmslevel_if_t Class Reference 1392
5.407 rmslevel::rmslevel_t Class Reference 1394
5.408 rohBeam::configOptions Struct Reference 1397
5.409 rohBeam::rohBeam Class Reference 1399
5.410 rohBeam::rohConfig Class Reference 1406
5.411 route::interface_t Class Reference 1412

© 2005-2021 HörTech gGmbH, Oldenburg

5.412 route::process_t Class Reference 1415
5.413 rt_nlms_t Class Reference 1417
5.414 save_spec_t Class Reference 1422
5.415 save_wave_t Class Reference 1424
5.416 shadowfilter_begin::cfg_t Class Reference 1425
5.417 shadowfilter_begin::shadowfilter_begin_t Class Reference 1427
5.418 shadowfilter_end::cfg_t Class Reference 1429
5.419 shadowfilter_end::shadowfilter_end_t Class Reference 1431
5.420 sine_cfg_t Struct Reference 1433
5.421 sine_t Class Reference 1435
5.422 smooth_cepstrum::smooth_cepstrum_if_t Class Reference 1438
5.423 smooth_cepstrum::smooth_cepstrum_t Class Reference 1443
5.424 smooth_cepstrum::smooth_params Class Reference 1450
5.425 smoothgains_bridge::overlapadd_if_t Class Reference 1453
5.426 smoothgains_bridge::smoothspec_wrap_t Class Reference 1456
5.427 softclip_t Class Reference 1458
5.428 softclipper_t Class Reference 1460
5.429 softclipper_variables_t Class Reference 1463
5.430 spec2wave_if_t Class Reference 1466
5.431 spec2wave_t Class Reference 1468
5.432 spec_fader_t Class Reference 1471
5.433 speechnoise_t Class Reference 1472
5.434 steerbf Class Reference 1475
5.435 steerbf_config Class Reference 1478
5.436 testplugin::ac_parser_t Class Reference 1480
5.437 testplugin::config_parser_t Class Reference 1483
5.438 testplugin::if_t Class Reference 1486
5.439 testplugin::signal_parser_t Class Reference 1489
5.440 us_t Class Reference 1490
5.441 wave2spec_if_t Class Reference 1492
5.442 wave2spec_t Class Reference 1497
5.443 wavrec_t Class Reference 1503
5.444 wavwriter_t Class Reference 1506
5.445 windnoise::cfg_t Class Reference 1509
5.446 windnoise::if_t Class Reference 1514
5.447 windowselector_t Class Reference 1518

6 File Documentation **1522**
6.1 ac2lsl.cpp File Reference 1522
6.2 ac2osc.cpp File Reference 1523
6.3 ac2wave.cpp File Reference 1523
6.4 ac_monitor_type.cpp File Reference 1523
6.5 ac_monitor_type.hh File Reference 1523
6.6 ac_mul.cpp File Reference 1524
6.7 ac_mul.hh File Reference 1524
6.8 ac_proc.cpp File Reference 1525
6.9 acConcat_wave.cpp File Reference 1525
6.10 acConcat_wave.h File Reference 1526
6.11 acmon.cpp File Reference 1526
6.12 acPooling_wave.cpp File Reference 1526
6.13 acPooling_wave.h File Reference 1527
6.14 acrec.cpp File Reference 1527

© 2005-2021 HörTech gGmbH, Oldenburg

6.15 acrec.hh File Reference . . . 1527
6.16 acsave.cpp File Reference . . . 1527
6.17 acSteer.cpp File Reference . . . 1529
6.18 acSteer.h File Reference . . . 1529
6.19 acTransform_wave.cpp File Reference . . . 1529
6.20 acTransform_wave.h File Reference . . . 1530
6.21 adaptive_feedback_canceller.cpp File Reference . . . 1530
6.22 adaptive_feedback_canceller.h File Reference . . . 1531
6.23 addsndfile.cpp File Reference . . . 1531
6.24 adm.cpp File Reference . . . 1532
6.25 adm.hh File Reference . . . 1533
6.26 altconfig.cpp File Reference . . . 1534
6.27 altconfig.hh File Reference . . . 1534
6.28 altplugs.cpp File Reference . . . 1534
6.29 analysemhaplugin.cpp File Reference . . . 1535
6.30 analysispath.cpp File Reference . . . 1536
6.31 attenuate20.cpp File Reference . . . 1536
6.32 audiometerbackend.cpp File Reference . . . 1537
6.33 auditory_profile.cpp File Reference . . . 1538
6.34 auditory_profile.h File Reference . . . 1538
6.35 browsemhaplugins.cpp File Reference . . . 1538
6.36 coherence.cpp File Reference . . . 1539
6.37 combinechannels.cpp File Reference . . . 1539
6.38 compiler_id.cpp File Reference . . . 1540
6.39 compiler_id.hh File Reference . . . 1540
6.40 complex_filter.cpp File Reference . . . 1541
6.41 complex_filter.h File Reference . . . 1541
6.42 complex_scale_channel.cpp File Reference . . . 1542
6.43 cpuload.cpp File Reference . . . 1542
6.44 db.cpp File Reference . . . 1542
6.45 dbasync.cpp File Reference . . . 1542
6.46 dc.cpp File Reference . . . 1543
6.47 dc.hh File Reference . . . 1544
6.48 dc_afterburn.cpp File Reference . . . 1544
6.49 dc_afterburn.h File Reference . . . 1544
6.50 dc_simple.cpp File Reference . . . 1545
6.51 dc_simple.hh File Reference . . . 1545
6.52 delay.cpp File Reference . . . 1546
6.53 delay.hh File Reference . . . 1546
6.54 delaysum_spec.cpp File Reference . . . 1547
6.55 delaysum_wave.cpp File Reference . . . 1547
6.56 doasvm_classification.cpp File Reference . . . 1547
6.57 doasvm_classification.h File Reference . . . 1548
6.58 doasvm_feature_extraction.cpp File Reference . . . 1548
6.59 doasvm_feature_extraction.h File Reference . . . 1549
6.60 doc_appendix.h File Reference . . . 1549
6.61 doc_examples.h File Reference . . . 1549
6.62 doc_frameworks.h File Reference . . . 1549
6.63 doc_general.h File Reference . . . 1549
6.64 doc_kernel.h File Reference . . . 1549
6.65 doc_matlab.h File Reference . . . 1549

© 2005-2021 HörTech gGmbH, Oldenburg

6.66 doc_mhamain.h File Reference 1549
6.67 doc_parser.h File Reference 1549
6.68 doc_plugins.h File Reference 1549
6.69 doc_system.h File Reference 1549
6.70 doc_toolbox.h File Reference 1549
6.71 double2acvar.cpp File Reference 1549
6.72 downsample.cpp File Reference 1550
6.73 dropgen.cpp File Reference 1550
6.74 droptect.cpp File Reference 1550
6.75 equalize.cpp File Reference 1550
6.76 example1.cpp File Reference 1550
6.77 example2.cpp File Reference 1551
6.78 example3.cpp File Reference 1551
6.79 example4.cpp File Reference 1551
6.80 example5.cpp File Reference 1551
6.81 example6.cpp File Reference 1551
6.82 example7.cpp File Reference 1552
6.83 example7.hh File Reference 1552
6.84 fader_spec.cpp File Reference 1552
6.85 fader_wave.cpp File Reference 1552
6.86 fftfbpow.cpp File Reference 1553
6.87 fftfilter.cpp File Reference 1553
6.88 fftfilterbank.cpp File Reference 1553
6.89 fshift.cpp File Reference 1554
6.90 fshift.hh File Reference 1554
6.91 fshift_hilbert.cpp File Reference 1554
6.92 gain.cpp File Reference 1555
6.93 gaintable.cpp File Reference 1555
6.94 gaintable.h File Reference 1555
6.95 generatemhaplugindoc.cpp File Reference 1556
6.96 gsc_adaptive_stage.cpp File Reference 1558
6.97 gsc_adaptive_stage.hh File Reference 1558
6.98 gsc_adaptive_stage_if.cpp File Reference 1559
6.99 gsc_adaptive_stage_if.hh File Reference 1559
6.100 gtfb_analyzer.cpp File Reference 1559
6.101 gtfb_simd.cpp File Reference 1561
6.102 gtfb_simple_bridge.cpp File Reference 1567
6.103 hann.cpp File Reference 1567
6.104 hann.h File Reference 1568
6.105 identity.cpp File Reference 1568
6.106 ifftshift.cpp File Reference 1569
6.107 ifftshift.h File Reference 1569
6.108 iirfilter.cpp File Reference 1569
6.109 level_matching.cpp File Reference 1569
6.110 level_matching.hh File Reference 1570
6.111 levelmeter.cpp File Reference 1570
6.112 lpc.cpp File Reference 1571
6.113 lpc.h File Reference 1572
6.114 lpc_bl_predictor.cpp File Reference 1572
6.115 lpc_bl_predictor.h File Reference 1572
6.116 lpc_burg-lattice.cpp File Reference 1573

© 2005-2021 HörTech gGmbH, Oldenburg

6.117 lpc_burg-lattice.h File Reference 1573
6.118 lsl2ac.cpp File Reference 1574
6.119 lsl2ac.hh File Reference 1574
6.120 matlab_wrapper.cpp File Reference 1574
6.121 matlab_wrapper.hh File Reference 1574
6.122 matrixmixer.cpp File Reference 1575
6.123 mconv.cpp File Reference 1575
6.124 mha.cpp File Reference 1576
6.125 mha.hh File Reference 1576
6.126 mha_algo_comm.cpp File Reference 1584
6.127 mha_algo_comm.h File Reference 1585
6.128 mha_algo_comm.hh File Reference 1587
6.129 mha_defs.h File Reference 1588
6.130 mha_errno.c File Reference 1590
6.131 mha_errno.h File Reference 1591
6.132 mha_error.cpp File Reference 1592
6.133 mha_error.hh File Reference 1593
6.134 mha_event_emitter.h File Reference 1594
6.135 mha_events.cpp File Reference 1595
6.136 mha_events.h File Reference 1595
6.137 mha_fttfb.cpp File Reference 1595
6.138 mha_fttfb.hh File Reference 1597
6.139 mha_fifo.cpp File Reference 1598
6.140 mha_fifo.h File Reference 1598
6.141 mha_filter.cpp File Reference 1599
6.142 mha_filter.hh File Reference 1599
6.143 mha_generic_chain.cpp File Reference 1601
6.144 mha_generic_chain.h File Reference 1601
6.145 mha_git_commit_hash.cpp File Reference 1602
6.146 mha_git_commit_hash.hh File Reference 1603
6.147 mha_io_ifc.h File Reference 1603
6.148 mha_io_utils.cpp File Reference 1606
6.149 mha_io_utils.hh File Reference 1606
6.150 mha_multisrc.cpp File Reference 1606
6.151 mha_multisrc.h File Reference 1606
6.152 mha_os.cpp File Reference 1607
6.153 mha_os.h File Reference 1609
6.154 mha_parser.cpp File Reference 1614
6.155 mha_parser.hh File Reference 1618
6.156 mha_plugin.cpp File Reference 1622
6.157 mha_plugin.hh File Reference 1622
6.158 mha_profiling.c File Reference 1626
6.159 mha_profiling.h File Reference 1627
6.160 mha_ruby.cpp File Reference 1628
6.161 mha_signal.cpp File Reference 1629
6.162 mha_signal.hh File Reference 1633
6.163 mha_signal_fft.h File Reference 1643
6.164 mha_tablelookup.cpp File Reference 1643
6.165 mha_tablelookup.hh File Reference 1643
6.166 mha_tcp.cpp File Reference 1644
6.167 mha_tcp.hh File Reference 1647

© 2005-2021 HörTech gGmbH, Oldenburg

6.168 mha_tcp_server.cpp File Reference 1648
6.169 mha_tcp_server.hh File Reference 1648
6.170 mha_toolbox.h File Reference 1649
6.171 mha_utils.cpp File Reference 1649
6.172 mha_utils.hh File Reference 1649
6.173 mha_windowparser.cpp File Reference 1649
6.174 mha_windowparser.h File Reference 1650
6.175 mhachain.cpp File Reference 1651
6.176 mhafw_lib.cpp File Reference 1651
6.177 mhafw_lib.h File Reference 1651
6.178 MHAIOalsa.cpp File Reference 1651
6.179 MHAIOAsterisk.cpp File Reference 1656
6.180 MHAIOFile.cpp File Reference 1661
6.181 MHAIOJack.cpp File Reference 1666
6.182 MHAIOJackdb.cpp File Reference 1670
6.183 MHAIOParser.cpp File Reference 1674
6.184 MHAIOPortAudio.cpp File Reference 1678
6.185 MHAIOTCP.cpp File Reference 1683
6.186 mhajack.cpp File Reference 1688
6.187 mhajack.h File Reference 1689
6.188 mhamain.cpp File Reference 1691
6.189 mhapluginloader.cpp File Reference 1693
6.190 mhapluginloader.h File Reference 1693
6.191 mhasndfile.cpp File Reference 1693
6.192 mhasndfile.h File Reference 1694
6.193 multibandcompressor.cpp File Reference 1695
6.194 nlms_wave.cpp File Reference 1695
6.195 noise.cpp File Reference 1697
6.196 noise_psd_estimator.cpp File Reference 1697
6.197 osc2ac.cpp File Reference 1697
6.198 overlapadd.cpp File Reference 1698
6.199 overlapadd.hh File Reference 1698
6.200 plingploing.cpp File Reference 1698
6.201 pluginbrowser.cpp File Reference 1699
6.202 pluginbrowser.h File Reference 1699
6.203 proc_counter.cpp File Reference 1699
6.204 resampling.cpp File Reference 1699
6.205 rmslevel.cpp File Reference 1699
6.206 rohBeam.cpp File Reference 1700
6.207 rohBeam.hh File Reference 1700
6.208 route.cpp File Reference 1701
6.209 save_spec.cpp File Reference 1701
6.210 save_wave.cpp File Reference 1701
6.211 shadowfilter_begin.cpp File Reference 1702
6.212 shadowfilter_end.cpp File Reference 1702
6.213 sine.cpp File Reference 1702
6.214 smooth_cepstrum.cpp File Reference 1702
6.215 smooth_cepstrum.hh File Reference 1703
6.216 smoothgains_bridge.cpp File Reference 1703
6.217 softclip.cpp File Reference 1704
6.218 spec2wave.cpp File Reference 1704

© 2005-2021 HörTech gGmbH, Oldenburg

6.219 speechnoise.cpp File Reference 1704
6.220 speechnoise.h File Reference 1708
6.221 split.cpp File Reference 1709
6.222 steerbf.cpp File Reference 1710
6.223 steerbf.h File Reference 1711
6.224 testalsadevice.c File Reference 1711
6.225 testplugin.cpp File Reference 1711
6.226 transducers.cpp File Reference 1712
6.227 upsample.cpp File Reference 1713
6.228 wave2spec.cpp File Reference 1713
6.229 wave2spec.hh File Reference 1713
6.230 wavrec.cpp File Reference 1714
6.231 windnoise.cpp File Reference 1714
6.232 windnoise.hh File Reference 1714
6.233 windowselector.cpp File Reference 1714
6.234 windowselector.h File Reference 1714

Index **1715**

© 2005-2021 HörTech gGmbH, Oldenburg

1 Overview

The HörTech Open Master Hearing Aid (openMHA), is a development and evaluation software platform that is able to execute hearing aid signal processing in real-time on standard computing hardware with a low delay between sound input and output.

1.1 Structure

The openMHA can be split into four major components :

- **The openMHA command line application (MHA)** (p. 30)
- Signal processing plugins
- Audio input-output (IO) plugins (see **io_file_t** (p. 605), **MHAIOJack** (p. 109), **io_parser_t** (p. 616), **io_tcp_parser_t** (p. 626))
- **The openMHA toolbox library** (p. 31)

Figure 1 openMHA structure

The openMHA command line application (MHA) (p. 30) acts as a plugin host. It can load signal processing plugins as well as audio input-output (IO) plugins. Additionally, it provides the command line configuration interface and a TCP/IP based configuration interface. Several IO plugins exist: For real-time signal processing, commonly the openMHA **MHAIOJack** (p. 109) plugin (see plugins' manual) is used, which provides an interface to the Jack Audio Connection Kit (JACK). Other IO plugins provide audio file access or TCP/IP-based processing.

openMHA plugins provide the audio signal processing capabilities and audio signal handling. Typically, one openMHA plugin implements one specific algorithm. The complete virtual hearing aid signal processing can be achieved by a combination of several openMHA plugins.

© 2005-2021 HörTech gGmbH, Oldenburg

1.2 Platform Services and Conventions

The openMHA platform offers some services and conventions to algorithms implemented in plugins, that make it especially well suited to develop hearing aid algorithms, while still supporting general-purpose signal processing.

1.2.1 Audio Signal Domains

As in most other plugin hosts, the audio signal in the openMHA is processed in audio chunks. However, plugins are not restricted to propagate audio signal as blocks of audio samples in the time domain another option is to propagate the audio signal in the short time Fourier transform (STFT) domain, i.e. as spectra of blocks of audio signal, so that not every plugin has to perform its own STFT analysis and synthesis. Since STFT analysis and re-synthesis of acceptable audio quality always introduces an algorithmic delay, sharing STFT data is a necessity for a hearing aid signal processing platform, because the overall delay of the complete processing has to be as short as possible.

Similar to some other platforms, the openMHA allows also arbitrary data to be exchanged between plugins through a mechanism called **algorithm communication variables** (p. 23) or short "AC vars". This mechanism is commonly used to share data such as filter coefficients or filter states.

1.2.2 Real-Time Safe Complex Configuration Changes

Hearing aid algorithms in the openMHA can export configuration settings that may be changed by the user at run time.

To ensure real-time safe signal processing, the audio processing will normally be done in a signal processing thread with real-time priority, while user interaction with configuration parameters would be performed in a configuration thread with normal priority, so that the audio processing does not get interrupted by configuration tasks. Two types of problems may occur when the user is changing parameters in such a setup:

- The change of a simple parameter exposed to the user may cause an involved recalculation of internal runtime parameters that the algorithm actually uses in processing. The duration required to perform this recalculation may be a significant portion of (or take even longer than) the time available to process one block of audio signal. In hearing aid usage, it is not acceptable to halt audio processing for the duration that the recalculation may require.
- If the user needs to change multiple parameters to reach a desired configuration state of an algorithm from the original configuration state, then it may not be acceptable that processing is performed while some of the parameters have already been changed while others still retain their original values. It is also not acceptable to interrupt signal processing until all pending configuration changes have been performed.

© 2005-2021 HörTech gGmbH, Oldenburg

The openMHA provides a mechanism in its toolbox library to enable real-time safe configuration changes in openMHA plugins:

Basically, existing runtime configurations are used in the processing thread until the work of creating an updated runtime configuration has been completed in the configuration thread.

In hearing aids, it is more acceptable to continue to use an outdated configuration for a few more milliseconds than blocking all processing.

The openMHA toolbox library provides an easy-to-use mechanism to integrate real-time safe runtime configuration updates into every plugin.

1.2.3 Plugins can Themselves Host Other Plugins

An openMHA plugin can itself act as a plugin host. This allows to combine analysis and re-synthesis methods in a single plugin. We call plugins that can themselves load other plugins `‘bridge plugins'' in the openMHA.

When such a bridge plugin is then called by the openMHA to process one block of signal, it will first perform its analysis, then invoke (as a function call) the signal processing in the loaded plugin to process the block of signal in the analysis domain, wait to receive a processed block of signal in the analysis domain back from the loaded plugin when the signal processing function call to that plugin returns, then perform the re-synthesis transform, and finally return the block of processed signal in the original domain back to the caller of the bridge plugin.

1.2.4 Central Calibration

The purpose of hearing aid signal processing is to enhance the sound for hearing impaired listeners. Hearing impairment generally means that people suffering from it have increased hearing thresholds, i.e. soft sounds that are audible for normal hearing listeners may be imperceptible for hearing impaired listeners. To provide accurate signal enhancement for hearing impaired people, hearing aid signal processing algorithms have to be able to determine the absolute physical sound pressure level corresponding to a digital signal given to any openM↩HA plugin for processing. Inside the openMHA, we achieve this with the following convention: The single-precision floating point time-domain sound signal samples, that are processed inside the openMHA plugins in blocks of short durations, have the physical pressure unit Pascal ($1\mathrm{Pa} = 1\mathrm{N/m^2}$). With this convention in place, all plugins can determine the absolute physical sound pressure level from the sound samples that they process: E.g. plugins can compute the rms (root mean squared) sound pressure in Pascal of the current block by computing $\mathrm{rms} = \sqrt{\sum_i x_i^2}$, where x_i refer to the samples of the audio signal in a single audio channel, and the corresponding free-field sound pressure level L in dB SPL FF as $\mathrm{L} = 20\log_{10}(rms/20\mu\mathrm{Pa})$. ($20\mu\mathrm{Pa}$ is the sound pressure at 0dB SPL FF).

A derived convention is employed in the spectral domain for STFT signals: The sum of the squared magnitudes of all spectral bins computes rms of the signal in the current STFT block: $\mathrm{rms} = \sqrt{\sum_i |X_i|^2}$, the X_i refer to the complex values of the STFT spectral bins. The STFT bins

© 2005-2021 HörTech gGmbH, Oldenburg

of the negative frequencies are not stored, since they contain the complex conjugate values of the corresponding positive frequencies. When summing over all bins to compute the rms as above, care must be taken to also account for the negative frequencies. Note that the bins corresponding to 0Hz and to the Nyquist frequency have no corresponding negative frequency bin. The sound pressure level in dB can be computed from the rms in the same way as in the time domain.

Due to the dependency of the calibration on the hardware used, it is the responsibility of the user of the openMHA to perform calibration measurements and adapt the openMHA settings to make sure that this calibration convention is met. We provide the plugin `transducers` which can be configured to perform the necessary signal adjustments.

2 Deprecated List

Member MHAParser::base_t::base_t (p. 1034) **(const base_t** (p. 1030) **&)**

Copying parser nodes makes little sense, avoid wherever possible

3 Module Documentation

3.1 Concept of Variables and Data Exchange in the openMHA

Accessibility of configuration variables and data exchange between plugins (processing blocks) are an important issue in the openMHA. In general, variable types in the openMHA are distinguished by their different access methods. The variable types in the openMHA are:

Accessibility of configuration variables and data exchange between plugins (processing blocks) are an important issue in the openMHA. In general, variable types in the openMHA are distinguished by their different access methods. The variable types in the openMHA are:

- **Configuration variables** : Read and write accesses are possible through the openM↩HA configuration language interface. Configuration variables are implemented as C++ classes with a public data member of the underlying C type. Configuration variables can be read and modified from `‘outside'' using the configuration language. The plugin which provides the configuration variable can use the exposed data member directly. All accesses through the openMHA configuration language are checked for data type, valid range, and access restrictions.
- **Monitor variables** : Read access is possible through the openMHA configuration language. Write access is only possible from the C++ code. Internally, monitor variables have a similar C++ class interface as configuration variables.
- **AC variables** (**algorithm communication variables** (p. 23)): Any C or C++ data structure can be shared within an openMHA chain. Access management and name space is realised in openMHA chain plugin ('mhachain'). AC variables are not available to the openMHA configuration language interface, although a read-only converter plugin `acmon` is available.

© 2005-2021 HörTech gGmbH, Oldenburg

- **Runtime configuration** : Algorithms usually derive more parameters (runtime configuration) from the openMHA configuration language variables. When a configuration variable changes through configuration language write access, then the runtime configuration has to be recomputed. Plugin developers are encouraged to encapsulate the runtime configuration in a C++ class, which recomputes the runtime configuration from configuration variables in the constructor. The openMHA supports lock-free and thread-safe replacement of the runtime configuration instance (see **example5.cpp** (p. 15) and references therein).

Figure 2 Variable types in the openMHA

Variables that describe physical facts to the MHA user should be given in SI units, e.g. meters for distances (not centimeters or inches), seconds for times (not milliseconds or minutes) etc for reasons of uniformity and simplicity of handling derived units.

The C++ data types are shown in the figure below. These variables can be accessed via the openMHA host application using the openMHA configuration language. For more details see the openMHA application manual.

Figure 3 MHAParser elements

© 2005-2021 HörTech gGmbH, Oldenburg

3.2 Writing openMHA Plugins. A step-by-step tutorial

Plugins are C++ code that is compiled and linked against the openMHA library. The compiler needs be instructed on how to find the openMHA headers and library and to link against the openMHA library. There are two possible options: One can compile openMHA and then create a copy of an example plugin directory and customize from there. See COMPILATION.md for more information on how to compile openMHA.

Plugins are C++ code that is compiled and linked against the openMHA library. The compiler needs be instructed on how to find the openMHA headers and library and to link against the openMHA library. There are two possible options: One can compile openMHA and then create a copy of an example plugin directory and customize from there. See COMPILATION.md for more information on how to compile openMHA.

On Ubuntu is is also possible to install the openmha-dev package and include config.mk into the user's Makefile. Example 21 provides an example plugin and Makefile for this scenario.

openMHA contains a small number of example plugins as C++ source code. They are meant to help developers in understanding the concepts of openMHA plugin programming starting from the simplest example and increasing in complexity. This tutorial explains the basic parts of the example files.

3.2.1 example1.cpp

The example plugin file **`example1.cpp`** (p. 1550) demonstrates the easiest way to implement an openMHA Plugin. It attenuates the sound signal in the first channel by multiplying the sound samples with a factor. The plugin class **MHAPlugin::plugin_t** (p. 1151) exports several methods, but only two of them need a non-empty implementation:

- `prepare()`
  - The `prepare()` method in the parent class `mha_plugin_t<>` is a pure virtual method and needs an implementation so that the plugin class can be instantiated.
- `process()`
  - The `process()` method is called whenever a new block of audio arrives and needs signal processing by this plugin.

```
#include "mha_plugin.hh"
class example1_t : public MHAPlugin::plugin_t<int> {
public:
  example1_t(algo_comm_t iac, const std::string & configured_name)
      : MHAPlugin::plugin_t<int>("",iac)
  {(void)configured_name;/* ignore 2nd parameter */}
  void release(void)
  {/* Do nothing in release */}
```

Every plugin implementation should include the '**mha_plugin.hh** (p. 1622)' header file. C++ helper classes for plugin development are declared in this header file, and most header files needed for plugin development are included by **mha_plugin.hh** (p. 1622).

© 2005-2021 HörTech gGmbH, Oldenburg

The class **example1_t** (p. 467) inherits from the class **MHAPlugin::plugin_t** (p. 1151), which in turn inherits from **MHAParser::parser_t** (p. 1102) – the configuration language interface in the method "parse". Our plugin class therefore inherits the "parse" method from **MHAParser::parser_t** (p. 1102), which integrates the plugin into the global openMHA configuration tree.

The constructor has to accept two parameters of types **algo_comm_t** (p. 279) and `std::string`, respectively. In this simple example, we do not make use of them.

The `release()` method is used to free resources after signal processing. In this simple example, we do not allocate resources, so there is no need to free them.

3.2.1.1 The prepare method

```
void prepare(mhaconfig_t & signal_info)
{
  if (signal_info.domain != MHA_WAVEFORM)
    throw MHA_Error(__FILE__, __LINE__,
                    "This plugin can only process waveform signals.");
  if (signal_info.channels < 1)
    throw MHA_Error(__FILE__,__LINE__,
                    "This plugin requires at least one input channel.");
}
```

Parameters

signal_info	Contains information about the input signal's dimensions, see **mhaconfig_t** (p. 851).

The `prepare()` method of the plugin is called before the signal processing starts, when the input signal dimensions like domain, number of channels, frames per block, and sampling rate are known. The `prepare()` method can check these values and raise an exception if the plugin cannot cope with them, as is done here. The plugin can also change these values if the signal processing performed in the plugin results in an output signal with different parameters. This plugin does not change the signal's parameters, therefore they are not modified here.

3.2.1.2 The signal processing method

```
mha_wave_t * process(mha_wave_t * signal)
{
  unsigned int channel = 0; // channels and frames counting starts with 0
  float factor = 0.1f;
  unsigned int frame;
  // Scale channel number "channel" by "factor":
  for(frame = 0; frame < signal->num_frames; frame++) {
    // Waveform channels are stored interleaved.
    signal->buf[signal->num_channels * frame + channel] *= factor;
  }
  // Algorithms may process data in-place and return the input signal
  // structure as their output signal:
  return signal;
}
};
```

Parameters

signal	Pointer to the input signal structure **mha_wave_t** (p. 840).

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

Pointer to the output signal structure. The input signal structure may be reused if the signal has the same domain and dimensions.

The plugin works with time domain input signal (indicated by the data type **mha_wave_t** (p. 840) of the process method's parameter). It scales the first channel by a factor of 0.1. The output signal reuses the structure that previously contained the input signal (in-place processing).

3.2.1.3 Connecting the C++ class with the C plugin interface Plugins have to export C functions as their interface (to avoid C++ name-mangling issues and other incompatibilities when mixing plugins compiled with different C++ compilers).

```
MHAPLUGIN_CALLBACKS(example1,example1_t,wave,wave)
```

This macro takes care of accessing the C++ class from the C functions required as the plugin's interface. It implements the C funtions and calls the corresponding C++ instance methods. Plugin classes should be derived from the template class **MHAPlugin::plugin_t** (p. 1151) to be compatible with the C interface wrapper.

This macro also catches C++ exceptions of type **MHA_Error** (p. 764), when raised in the methods of the plugin class, and reports the error using an error flag as the return value of the underlying C function. It is therefore important to note that only C++ exceptions of type **MH↩A_Error** (p. 764) may be raised by your plugin. If your code uses different Exception classes, you will have to catch them yourself before control leaves your plugin class, and maybe report the error by throwing an instance of **MHA_Error** (p. 764). This is important, because: (1) C++ exceptions cannot cross the plugin interface, which is in C, and (2) there is no error handling code for your exception classes in the openMHA framework anyways.

3.2.2 example2.cpp

This is another simple example of openMHA plugin written in C++. This plugin also scales one channel of the input signal, working in the time domain. The scale factor and which channel to scale (index number) are made accessible to the configuration language.

The algorithm is again implemented as a C++ class.

```
class example2_t : public MHAPlugin::plugin_t<int> {
    MHAParser::int_t scale_ch;
    MHAParser::float_t factor;
public:
    example2_t(algo_comm_t iac, const std::string & configured_name);
    void prepare(mhaconfig_t & signal_info);
    void release(void);
    mha_wave_t * process(mha_wave_t * signal);
};
```

Parameters

scale_ch	– the channel number to be scaled
factor	– the scale factor of the scaling.

© 2005-2021 HörTech gGmbH, Oldenburg

This class again inherits from the template class **MHAPlugin::plugin_t** (p. 1151) for intergration with the openMHA configuration language. The two data members serve as externally visible configuration variables. All methods of this class have a non-empty implementation.

3.2.2.1 Constructor

```
      example2_t::example2_t(algo_comm_t iac, const std::string & configured_name)
    : MHAPlugin::plugin_t<int>("This plugin multiplies the sound signal"
                               " in one audio channel by a factor",iac),
      scale_ch("Index of audio channel to scale. Indices start from 0.",
               "0",
               "[0,[inf[[),
      factor("The scaling factor that is applied to the selected channel.",
             "0.1",
             "[0,[inf[[)
{
    insert_item("channel", &scale_ch);
    insert_item("factor", &factor);
    (void)configured_name; // Ignore 2nd parameter
}
```

The constructor invokes the superclass constructor with a string parameter. This string parameter serves as the help text that describes the functionality of the plugin. The constructor registers configuration variables with the openMHA configuration tree and sets their default values and permitted ranges. The minimum permitted value for both variables is zero, and there is no maximum limit (apart from the limitations of the underlying C data type). The configuration variables have to be registered with the parser node instance using the **MHAParser::parser↩_t::insert_item** (p. 1104) method.

3.2.2.2 The prepare method

```
void example2_t::prepare(mhaconfig_t & signal_info)
{
    if (signal_info.domain != MHA_WAVEFORM)
        throw MHA_Error(__FILE__, __LINE__,
                        "This plugin can only process waveform signals.");
    // The user may have configured scale_ch before prepare is called.
    // Check that the configured channel is present in the input signal.
    if (signal_info.channels <= unsigned(scale_ch.data))
        throw MHA_Error(__FILE__,__LINE__,
                        "This plugin requires at least %d input channels.",
                        scale_ch.data + 1);
    // Adjust the range of the channel configuration variable so that it
    // cannot be set to an out-of-range value during processing.
    using MHAParser::StrCnv::val2str;
    scale_ch.set_range("[0," + val2str(int(signal_info.channels)) + "[");
}
```

Parameters

signal_info	– contains information about the input signal's parameters, see **mhaconfig_t** (p. 851).

The user may have changed the configuration variables before preparing the openMHA plugin. A consequence of this is that it is not sufficient any more to check if the input signal has at least 1 audio channel.

Instead, this prepare method checks that the input signal has enough channels so that the current value of `scale_ch.data` is a valid channel index, i.e. $0 \leq$ `scale_ch.data` $<$ `signal↩_info.channels`. The prepare method does not have to check that $0 \leq$ `scale_ch.data`, since this is guaranteed by the valid range setting of the configuration variable.

© 2005-2021 HörTech gGmbH, Oldenburg

The prepare method then modifies the valid range of the `scale_ch` variable, it modifies the upper bound so that the user cannot set the variable to a channel index higher than the available channels. Setting the range is done using a string parameter. The prepare method contatenates a string of the form "[0,n[". n is the number of channels in the input signal, and is used here as an exclusive upper boundary. To convert the number of channels into a string, a helper function for string conversion from the openMHA Toolbox is used. This function is overloaded and works for several data types.

It is safe to assume that the value of configuration variables does not change while the prepare method executes, since openMHA preparation is triggered from a configuration language command, and the openMHA configuration language parser is busy and cannot accept other commands until all openMHA plugins are prepared (or one of them stops the process by raising an exception). As we will see later in this tutorial, the same assumption cannot be made for the process method.

3.2.2.3 The release method

```
void example2_t::release(void)
{
    scale_ch.set_range("[0,[");
}
```

The release method should undo the state changes that were performed by the prepare method. In this example, the prepare method has reduced the valid range of the `scale_ch`, so that only valid channels could be selected during signal processing.

The release method reverts this change by setting the valid range back to its original value, "[0,[".

3.2.2.4 The signal processing method

```
mha_wave_t * example2_t::process(mha_wave_t * signal)
{
    unsigned int frame;
    for(frame = 0; frame < signal->num_frames; frame++)
        value(signal,frame,scale_ch.data) *= factor.data;
    return signal;
}
```

The processing function uses the current values of the configuration variables to scale every frame in the selected audio channel.

Note that the value of each configuration variable can change while the processing method executes, since the process method usually executes in a different thread than the configuration interface.

For this simple plugin, this is not a problem, but for more advanced plugins, it has to be taken into consideration. The next section takes a closer look at the problem.

© 2005-2021 HörTech gGmbH, Oldenburg

Consistency Assume that one thread reads the value stored in a variable while another thread writes a new value to that variable concurrently. In this case, you may have a consistency problem. You would perhaps expect that the value retrieved from the variable either (a) the old value, or (b) the new value, but not (c) something else. Yet generally case (c) is a possibility.

Fortunately, for some data types on PC systems, case (c) cannot happen. These are 32bit wide data types with a 4-byte alignment. Therefore, the values in **MHAParser::int_t** (p. 1069) and **MHAParser::float_t** (p. 1063) are always consistent, but this is not the case for vectors, strings, or complex values. With these, you can get a mixture of the bit patterns of old and new values, or you can even cause a memory access violation in case a vector or string grows and has to be reallocated to a different memory address.

There is also a consistency problem if you take the combination of two "safe" datatypes. The openMHA provides a mechanism that can cope with these types of problems. This thread-safe runtime configuration update mechanism is introduced in example 5.

3.2.3 example3.cpp

This example introduces the openMHA Event mechanism. Plugins that provide configuration variable can receive a callback from the parser base class when a configuration variable is accessed through the configuration language interface.

The third example performes the same processing as before, but now only even channel indices are permitted when selecting the audio channel to scale. This restriction cannot be ensured by setting the range of the channel index configuration variable. Instead, the event mechanism of openMHA configuration variables is used. Configuration variables emit 4 different events, and your plugin can connect callback methods that are called when the events are triggered. These events are:

writeaccess

- triggered on write access to a configuration variable.

valuechanged

- triggered when write access to a configuration variable actually changes the value of this variable.

readaccess

- triggered after the value of the configuration variable has been read.

prereadaccess

- triggered before the value of a configuration variable is read, i.e. the value of the requested variable can be changed by the callback to implement computation on demand.

All of these callbacks are executed in the configuration thread. Therefore, the callback implementation does not have to be realtime-safe. No other updates of configuration language variables through the configuration language can happen in parallel, but your processing method can execute in parallel and may change values.

© 2005-2021 HörTech gGmbH, Oldenburg

3.2.3.1 Data member declarations

```
class example3_t : public MHAPlugin::plugin_t<int> {
    MHAParser::int_t scale_ch;
    MHAParser::float_t factor;
    MHAParser::int_mon_t prepared;
    MHAEvents::patchbay_t<example3_t> patchbay;
```

This plugin exposes another configuration variable, `"prepared"`, that keeps track of the prepared state of the plugin. This is a read-only (monitor) integer variable, i.e. its value can only be changed by your plugin's C++ code. When using the configuration language interface, the value of this variable can only be read, but not changed.

The patchbay member is an instance of a connector class that connects event sources with callbacks.

3.2.3.2 Method declarations

```
    /* Callbacks triggered by Events */
    void on_scale_ch_writeaccess();
    void on_scale_ch_valuechanged();
    void on_scale_ch_readaccess();
    void on_prereadaccess();
public:
    example3_t(algo_comm_t iac, const std::string & configured_name);
    void prepare(mhaconfig_t & signal_info);
    void release(void);
    mha_wave_t * process(mha_wave_t * signal);
};
```

This plugin exposes 4 callback methods that are triggered by events. Multiple events (from the same or different configuration variables) can be connected to the same callback method, if desired.

This example plugin uses the `valuechanged` event to check that the `scale_ch` configuration variable is only set to valid values.

The other callbacks only cause log messages to stdout, but the comments in the logging callbacks give a hint when listening on the events would be useful.

3.2.3.3 Example 3 constructor

```
      example3_t::example3_t(algo_comm_t iac, const std::string &)
    : MHAPlugin::plugin_t<int>("This plugin multiplies the sound signal"
                               " in one audio channel by a factor",iac),
      scale_ch("Index of audio channel to scale. Indices start from 0."
               " Only channels with even indices may be scaled.",
               "0",
               "[0,[“),
      factor("The scaling factor that is applied to the selected channel.",
             "0.1",
             "[0,[“),
      prepared("State of this plugin: 0 = unprepared, 1 = prepared")
{
    insert_item("channel", &scale_ch);
    insert_item("factor", &factor);
    prepared.data = 0;
    insert_item("prepared", &prepared);

    patchbay.connect(&scale_ch.writeaccess, this,
                     &example3_t::on_scale_ch_writeaccess);
    patchbay.connect(&scale_ch.valuechanged, this,
                     &example3_t::on_scale_ch_valuechanged);
    patchbay.connect(&scale_ch.readaccess, this,
                     &example3_t::on_scale_ch_readaccess);
    patchbay.connect(&scale_ch.prereadaccess, this,
                     &example3_t::on_prereadaccess);
```

© 2005-2021 HörTech gGmbH, Oldenburg

```
    patchbay.connect(&factor.prereadaccess, this,
                     &example3_t::on_prereadaccess);
    patchbay.connect(&prepared.prereadaccess, this,
                     &example3_t::on_prereadaccess);
}
```

The constructor of a monitor variable does not require a parameter for setting the initial value. The only parameter here is the help text describing the contents of the read-only variable. If the initial value should differ from 0, then the `.data` member of the configuration variable has to be set to the initial value in the plugin constructor's body explicitly, as is done here for demonstration although the initial value of this monitor variable is 0.

Events and callback methods are then connected using the patchbay member variable.

3.2.3.4 The prepare method

```
void example3_t::prepare(mhaconfig_t & signal_info)
{
    if (signal_info.domain != MHA_WAVEFORM)
        throw MHA_Error(__FILE__, __LINE__,
                        "This plugin can only process waveform signals.");
    // The user may have configured scale_ch before prepare is called.
    // Check that the configured channel is present in the input signal.
    if (signal_info.channels <= unsigned(scale_ch.data))
        throw MHA_Error(__FILE__,__LINE__,
                        "This plugin requires at least %d input channels.",
                        scale_ch.data + 1);
    // bookkeeping
    prepared.data = 1;
}
```

The prepare method checks wether the current setting of the scale_ch variable is possible with the input signal dimension. It does not adjust the range of the variable, since the range alone is not sufficient to ensure all future settings are also valid: The scale channel index has to be even.

3.2.3.5 The release method

```
void example3_t::release(void)
{
  prepared.data = 0;
}
```

The release method is needed for tracking the prepared state only in this example.

3.2.3.6 The signal processing method

```
      mha_wave_t * example3_t::process(mha_wave_t * signal)
{
    unsigned int frame;
    for(frame = 0; frame < signal->num_frames; frame++)
        value(signal,frame,scale_ch.data) *= factor.data;
    return signal;
}
```

The signal processing member function is the same as in example 2.

© 2005-2021 HörTech gGmbH, Oldenburg

3.2.3.7 The callback methods

```
void example3_t::on_scale_ch_writeaccess()
{
    printf("Write access: Attempt to set scale_ch=%d.\n", scale_ch.data);
    // Can be used to track any writeaccess to the configuration, even
    // if it does not change the value.  E.g. setting the name of the
    // sound file in a string configuration variable can cause a sound
    // file player plugin to start playing the sound file from the
    // beginning.
}
void example3_t::on_scale_ch_valuechanged()
{
    if (scale_ch.data & 1)
        throw MHA_Error(__FILE__,__LINE__,
                        "Attempt to set scale_ch to non-even value %d",
                        scale_ch.data);
    // Can be used to recompute a runtime configuration only if some
    // configuration variable actually changed.
}
void example3_t::on_scale_ch_readaccess()
{
    printf("scale_ch has been read.\n");
    // A configuration variable used as an accumulator can be reset
    // after it has been read.
}
void example3_t::on_prereadaccess()
{
    printf("A configuration language variable is about to be read.\n");
    // Can be used to compute the value on demand.
}
MHAPLUGIN_CALLBACKS(example3,example3_t,wave,wave)
```

When the `writeaccess` or `valuechanged` callbacks throw an MHAError exception, then the change made to the value of the configuration variable is reverted.

If multiple event sources are connected to a single callback method, then it is not possible to determine which event has caused the callback to execute. Often, this information is not crucial, i.e. when the answer to a change of any variable in a set of variables is the same, e.g. the recomputation of a new runtime configuration that takes all variables of this set as input.

3.2.4 example4.cpp

This plugin is the same as example 3 except that it works on the spectral domain (STFT).

3.2.4.1 The Prepare method

```
void example4_t::prepare(mhaconfig_t & signal_info)
{
    if (signal_info.domain != MHA_SPECTRUM)
        throw MHA_Error(__FILE__, __LINE__,
                        "This plugin can only process spectrum signals.");
    // The user may have configured scale_ch before prepare is called.
    // Check that the configured channel is present in the input signal.
    if (signal_info.channels <= unsigned(scale_ch.data))
        throw MHA_Error(__FILE__,__LINE__,
                        "This plugin requires at least %d input channels.",
                        scale_ch.data + 1);
    // bookkeeping
    prepared.data = 1;
}
```

The prepare method now checks that the signal domain is MHA_SPECTRUM.

© 2005-2021 HörTech gGmbH, Oldenburg

3.2.4.2 The signal processing method

```
mha_spec_t * example4_t::process(mha_spec_t * signal)
{
    unsigned int bin;
    // spectral signal is stored non-interleaved.
    mha_complex_t * channeldata =
        signal->buf + signal->num_frames * scale_ch.data;
    for(bin = 0; bin < signal->num_frames; bin++)
        channeldata[bin] *= factor.data;
    return signal;
}
```

The signal processing member function works on the spectral signal instead of the wave signal as before.

The **mha_spec_t** (p. 794) instance stores the complex (**mha_complex_t** (p. 745)) spectral signal for positive frequences only (since the waveform signal is always real). The num_frames member of **mha_spec_t** (p. 794) actually denotes the number of STFT bins.

Please note that different from **mha_wave_t** (p. 840), a multichannel signal in **mha_spec_t** (p. 794) is stored non-interleaved in the signal buffer.

Some arithmetic operations are defined on struct **mha_complex_t** (p. 745) to facilitate efficient complex computations. The $*=$ operator used here (defined for real and for complex arguments) is one of them.

3.2.4.3 Connecting the C++ class with the C plugin interface

```
MHAPLUGIN_CALLBACKS(example4,example4_t,spec,spec)
```

When connecting a class that performs spectral processing with the C interface, use `spec` instead of `wave` as the domain indicator.

3.2.5 example5.cpp

Many algorithms use complex operations to transform the user space variables into run time configurations. If this takes a noticeable time (e.g. more than 100-500 μ sec), the update of the runtime configuration can not take place in the real time processing thread. Furthermore, the parallel access to complex structures may cause unpredictable results if variables are read while only parts of them are written to memory (cf. section **Consistency** (p. 11)). To handle these situations, a special C++ template class **MHAPlugin::plugin_t** (p. 1151) was designed. This class helps keeping all access to the configuration language variables in the **configuration** thread rather than in the **processing** thread.

The runtime configuration class **`example5_t`** (p. 481) is the parameter of the template class **MHAPlugin::plugin_t** (p. 1151). Its constructor converts the user variables into a runtime configuration. Because the constructor executes in the configuration thread, there is no harm if the constructor takes a long time. All other member functions and data members of the runtime configurations are accessed only from the signal processing thread (real-time thread).

```
class example5_t {
public:
    example5_t(unsigned int,unsigned int,mha_real_t);
    mha_spec_t* process(mha_spec_t*);
private:
    unsigned int channel;
```

© 2005-2021 HörTech gGmbH, Oldenburg

```
    mha_real_t scale;
};
```

The plugin interface class inherits from the plugin template class **MHAPlugin::plugin_t** (p. 1151), parameterised by the runtime configuration. Configuration changes (write access to the variables) will emit a write access event of the changed variables. These events can be connected to member functions of the interface class by the help of a **MHAEvents::patchbay_t** (p. 861) instance.

```
class plugin_interface_t : public MHAPlugin::plugin_t<example5_t> {
public:
    plugin_interface_t(algo_comm_t iac, const std::string & configured_name);
    mha_spec_t* process(mha_spec_t*);
    void prepare(mhaconfig_t&);
private:
    void update_cfg();
    /* integer variable of MHA-parser: */
    MHAParser::int_t scale_ch;
    /* float variable of MHA-parser: */
    MHAParser::float_t factor;
    /* patch bay for connecting configuration parser
       events with local member functions: */
    MHAEvents::patchbay_t<plugin_interface_t> patchbay;
};
```

The constructor of the runtime configuration analyses and validates the user variables. If the configuration is invalid, an exception of type **MHA_Error** (p. 764) is thrown. This will cause the openMHA configuration language command which caused the change to fail: The modified configuration language variable is then reset to its original value, and the error message will contain the message string of the **MHA_Error** (p. 764) exception.

```
example5_t::example5_t(unsigned int ichannel,
                       unsigned int numchannels,
                       mha_real_t iscale)
    : channel(ichannel),scale(iscale)
{
    if( channel >= numchannels )
        throw MHA_Error(__FILE__,__LINE__,
                        "Invalid channel number %u (only %u channels configured).",
                        channel,numchannels);
}
```

In this example, the run time configuration class **`example5_t`** (p. 481) has a signal processing member function. In this function, the selected channel is scaled by the given scaling factor.

```
mha_spec_t* example5_t::process(mha_spec_t* spec)
{
    /* Scale channel number "scale_ch" by "factor": */
    for(unsigned int fr = 0; fr < spec->num_frames; fr++){
        spec->buf[fr + channel * spec->num_frames].re *= scale;
        spec->buf[fr + channel * spec->num_frames].im *= scale;
    }
    return spec;
}
```

The constructor of the example plugin class is similar to the previous examples. A callback triggered on write access to the variables is registered using the **MHAEvents::patchbay_t** (p. 861) instance.

```
plugin_interface_t::plugin_interface_t(algo_comm_t iac, const std::string &)
    : MHAPlugin::plugin_t<example5_t>("example plugin scaling a spectral signal",iac),
      /* initialzing variable 'scale_ch' with MHAParser::int_t(char* name, .... ) */
      scale_ch("channel number to be scaled","0","[0,[["),
      /* initialzing variable 'factor' with MHAParser::float_t(char* name, .... ) */
      factor("scale factor","1.0","[0,2]")
{
    /* Register variables to the configuration parser: */
    insert_item("channel",&scale_ch);
```

© 2005-2021 HörTech gGmbH, Oldenburg

```
    insert_item("factor",&factor);
    /*
     * On write access to the parser variables a notify callback of
     * this class will be called. That funtion will update the runtime
     * configuration.
     */
    patchbay.connect(&scale_ch.writeaccess,this,&plugin_interface_t::update_cfg);
    patchbay.connect(&factor.writeaccess,this,&plugin_interface_t::update_cfg);
}
```

The processing function can gather the latest valid runtime configuration by a call of `poll_config`. On success, the class member `cfg` points to this configuration. On error, if there is no usable runtime configuration instance, an exception is thrown. In this example, the prepare method ensures that there is a valid runtime configuration, so that in this example, no error can be raised at this point. The prepare method is always executed before the process method is called. The runtime configuration class in this example provides a signal processing method. The process method of the plugin interface calls the process method of this instance to perform the actual signal processing.

```
mha_spec_t* plugin_interface_t::process(mha_spec_t* spec)
{
    poll_config();
    return cfg->process(spec);
}
```

The prepare method ensures that a valid runtime configuration exists by creating a new runtime configuration from the current configuration language variables. If the configuraion is invalid, then an exception of type **MHA_Error** (p. 764) is raised and the preparation of the openMHA fails with an error message.

```
void plugin_interface_t::prepare(mhaconfig_t& tfcfg)
{
    if( tfcfg.domain != MHA_SPECTRUM )
        throw MHA_Error(__FILE__,__LINE__,
                        "Example5: Only spectral processing is supported.");
    /* remember the transform configuration (i.e. channel numbers): */
    tftype = tfcfg;
    /* make sure that a valid runtime configuration exists: */
    update_cfg();
}
```

The update_cfg member function is called when the value of a configuration language variable changes, or from the prepare method. It allocates a new runtime configuration and registers it for later access from the real time processing thread. The function **push_config** (p. 1150) stores the configuration in a FiFo queue of runtime configurations. Once they are inserted in the FiFo, the **MHAPlugin::plugin_t** (p. 1151) template is responsible for deleting runtime configuration instances stored in the FiFo. You don't need to keep track of the created instances, and you must not delete them yourself.

```
void plugin_interface_t::update_cfg()
{
    if( tftype.channels )
        push_config(new example5_t(scale_ch.data,tftype.channels,factor.data));
}
```

In the end of the example code file, the macro **MHAPLUGIN_CALLBACKS** (p. 1625) defines all ANSI-C interface functions and passes them to the corresponding C++ class member functions (partly defined by the **MHAPlugin::plugin_t** (p. 1151) template class). All exceptions of type **MHA_Error** (p. 764) are caught and transformed into an appropriate error code and error message.

```
MHAPLUGIN_CALLBACKS(example5,plugin_interface_t,spec,spec)
```

© 2005-2021 HörTech gGmbH, Oldenburg

3.2.6 example6.cpp

This example is the same as the previous one, except that it additionally creates an 'Algorithm Communication Variable' (AC variable). It calculates the RMS level of a given channel and stores it into this variable. The variable can be accessed by any other algorithm in the same chain. To store the data onto disk, the 'acsave' plugin can be used. 'acmon' is a plugin which converts AC variables into parsable monitor variables.

In the constructor of the plugin class the variable `rmsdb` is registered under the name `example6_rmslev` as a one-dimensional AC variable of type float. For registration of other types, read access and other detailed informations please see **Communication between algorithms** (p. 23).

```
example6_t::example6_t(algo_comm_t iac, const std::string &)
    : MHAPlugin::plugin_t<cfg_t>("Example rms level meter plugin",iac),
      /* initialzing variable 'channel_no' with MHAParser::int_t(char* name, .... ) */
      channel_no("channel in which the RMS level is measured","0","[0,[")
{
    /* Register variables to the configuration parser: */
    insert_item("channel",&channel_no);
    /*
     * On write access to the parser variables a notify callback of
     * this class will be called. That funtion will update the runtime
     * configuration.
     */
    patchbay.connect(&channel_no.writeaccess,this,&example6_t::update_cfg);
    /*
     * Propagate the level variable to all algorithms in the
     * processing chain. If multiple instances of this algorithm are
     * required, than it is necessary to use different names for this
     * variable (i.e. prefixing the name with the algorithm name
     * passed to MHAInit).
     */
    ac.insert_var_float( ac.handle, "example6_rmslev", &rmsdb );
}
```

© 2005-2021 HörTech gGmbH, Oldenburg

3.2.7 Debugging openMHA plugins

Suppose you would want to step through the code of your openMHA plugin with a debugger. This example details how to use the GDB debugger to inspect the **`example6_t::prepare()`** (p. 483) and **`example6_t::process()`** (p. 483) routines of **example6.cpp** (p. 18) example 6.

First, make sure that your plugin is compiled with the compiler option to include debugging symbols: Apply the -ggdb switch to all gcc, g++ invocations.

Once the plugin is compiled with debugging symbols, create a test configuration. For example 6, assuming there is an audio file named input.wav in your working directory, you could create a configuration file named 'debugexample6.cfg', with the following content:

```
# debugexample6.cfg
fragsize = 64
srate = 44100
nchannels_in = 2
iolib = MHAIOFile

io.in = input.wav
io.out = output.wav
mhalib = example6
mha.channel = 1
cmd=start
```

Assuming all your binaries and shared-object libraries are in your 'bin' directory (see READ↩ ME.md), you could start gdb using

```
$ export MHA_LIBRARY_PATH=$PWD/bin
$ gdb $MHA_LIBRARY_PATH/mha
```

Set breakpoints in prepare and process methods, and start execution. Note that specifying the breakpoint by symbol (**`example6_t::prepare`** (p. 483)) does not yet work, as the symbol lives in the openMHA plugin that has not yet been loaded. Specifying by line number works, however. Specifying the breakpoint by symbol also works once the plugin is loaded (i.e. when the debugger stops in the first break point). You can set the breakpoints like this (example shown here is run in gdb version 7.11.1):

```
(gdb) run ?read:debugexample6.cfg
Starting program: {openMHA_directory}/bin/mha ?read:debugexample6.cfg
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
The Open Master Hearing Aid (openMHA) server
Copyright (c) 2005-2021 HoerTech gGmbH, D-26129 Oldenburg, Germany

This program comes with ABSOLUTELY NO WARRANTY; for details see file COPYING.
This is free software, and you are welcome to redistribute it
under the terms of the GNU AFFERO GENERAL PUBLIC LICENSE, Version 3;
for details see file COPYING.
```

© 2005-2021 HörTech gGmbH, Oldenburg

```
Breakpoint 1, example6_t::prepare (this=0x6478b0, tfcfg=...)
    at example6.cpp:192
192         if( tfcfg.domain != MHA_WAVEFORM )
(gdb) b example6.cpp:162
Breakpoint 2 at 0x7ffff589744a: file example6.cpp, line 162.
(gdb) c
Continuing.
```

Where '{openMHA_directory}' is the directory where openMHA is located (which should also be your working directory in this case). Next stop is the `process()` method. You can now examine and change the variables, step through the program as needed (using, for example 'n' to step in the next line):

```
Breakpoint 2, example6_t::process (this=0x7ffff6a06c0d, wave=0x10a8b550)
    at example6.cpp:162
162     {
(gdb) n
163         poll_config();
(gdb)
```

3.2.8 Writing unit tests for openMHA plugins

This section introduces how to test a plugin with C++ unit tests using the Googletest framework. In order to execute the tests, navigate to the openMHA root directory and run `make unit-tests` in your terminal. Afterwards you may execute `make unit-tests` in the plugin directory in order to only execute the very test you are working on.

3.2.8.1 example7 As an example, unit tests for plugin **`example7.cpp`** (p. 1552) are written, which is functionally the same as plugin **`example1.cpp`** (p. 1550) (see section **example1.cpp** (p. 6)). In order to write unit tests for your plugin it must have its class/function declarations in a header file (.hh) so you can include it in the unit test file. The class/function definitions are contained in the respective source file (.cpp).
The unit tests are written using a test fixture class (here: `example7_testing`) which will be inherited by the individual tests (`TEST_F`). This enables us to use the members in `example7_testing` in multiple tests without the need for redundant declarations.

```
#include "mha_algo_comm.hh"
#include "mha_signal.hh"
#include "example7.hh"
#include <gtest/gtest.h>
class example7_testing : public ::testing::Test {
public:
  // AC variable space
  MHAKernel::algo_comm_class_t acspace{};
  // C handle to AC variable space
  algo_comm_t ac {acspace.get_c_handle()};
  // example input to prepare method
  mhaconfig_t signal_properties {
      .channels = 2U,
      .domain = MHA_WAVEFORM,
      .fragsize = 10U,
      .wndlen = 0U,
      .fftlen = 0U,
      .srate = 44100.0f
      };
```

© 2005-2021 HörTech gGmbH, Oldenburg

```
  //Plugin instance
  example7_t ex7{ac,"algo"};
  MHASignal::waveform_t wave_input{signal_properties.fragsize,signal_properties.channels};
};
```

The test fixture class is derived from the **::testing::Test** class declared in `gtest.h`. The constructor of **`example7_t`** (p. 485) needs three parameters, namely a handle to the algorithm communication variable space and two strings. A container for audio signals for repeatedly passing blocks of the input signal to the plugin under test is also allocated by the test fixture class. It is defined as an instance of **`MHASignal::waveform_t`** (p. 1263) with the name `wave_input` and its values are zero upon initialization.

```
TEST_F(example7_testing,test_state_methods){
  EXPECT_FALSE(ex7.is_prepared());
  ex7.prepare_(signal_properties);
  EXPECT_TRUE(ex7.is_prepared());
  ex7.release_();
  EXPECT_FALSE(ex7.is_prepared());
}
```

The first test checks whether the state methods work as expected. Next to the actual processing there are often certain variables in each individual openMHA plugin that need to be allocated beforehand or wiped from memory afterwards. The methods that are used to do this are `prepare()` and `release()`. In order to assert that they were called and that we switched states accordingly we use the methods `prepare_()` and `release_()` (Note: the underscore!) that are defined in the plugin base class `mha_plugin_t<>`. These methods keep track of the state, call `prepare()` and `release()` and do additional bookkeeping. To ensure that the state methods work as expected the Googletest methods `EXPECT_FALSE` and `EXPECT_TRUE` are used.

```
TEST_F(example7_testing,test_functionality){
  ex7.prepare_(signal_properties);
  wave_input.assign(1.0f);
  EXPECT_FLOAT_EQ(1.0f,value(wave_input,4,0));
  EXPECT_FLOAT_EQ(1.0f,value(wave_input,5,0));
  EXPECT_FLOAT_EQ(1.0f,value(wave_input,4,1));
  EXPECT_FLOAT_EQ(1.0f,value(wave_input,5,1));
  ex7.process(&wave_input);
  EXPECT_FLOAT_EQ(0.1f,value(wave_input,4,0));
  EXPECT_FLOAT_EQ(0.1f,value(wave_input,5,0));
  EXPECT_FLOAT_EQ(1.0f,value(wave_input,4,1));
  EXPECT_FLOAT_EQ(1.0f,value(wave_input,5,1));
  ex7.release_();
}
```

In this test the goal is to assess the main feature of the plugin (**`example7_t`** (p. 485)), which is the same as in **`example1_t`** (p. 467), namely altering the signal's first channel by a constant factor of 0.1. The variable `wave_input` is the signal that will be processed by the plugin. In order to assert the success, the elements in `wave_input` are set to a constant value of 1, because they are 0 upon initialization. During the `process()` function all elements of the first channel of `wave_input` are multiplied by the factor 0.1. Before `process()` is called the value assigned to `wave_input` is checked via the method `EXPECT_FLOAT_EQ` provided by Googletest. The values of `wave_input` are retrieved by the method **`value()`** (p. 46) by passing the desired sample position and channel number as second and third input parameter, respectively. Here, we checked the values of two frames in each channel to show the difference before and after processing; the frame indices were chosen randomly. After calling `process()`, the values contained in `wave_input` are checked again to make sure that the plugin worked as intended.

© 2005-2021 HörTech gGmbH, Oldenburg

3.3 The MHA Framework interface

© 2005-2021 HörTech gGmbH, Oldenburg

3.4 Communication between algorithms

Algorithms within one chain can share variables for communication with other algorithms. This mechanism allows interaction between algorithms (i.e. separation of noise estimation and noise reduction algorithms, combination of dynamic compression and noise estimation). Through a set of simple C functions, algorithms can propagate variables of any type, even C++ classes, to other algorithms.

Collaboration diagram for Communication between algorithms:

Files

- file **mha_algo_comm.h**
  Header file for Algorithm Communication.

Namespaces

- **MHA_AC**
  Functions and classes for Algorithm Communication (AC) support.

Classes

- class **MHA_AC::spectrum_t**
  *Insert a **MHASignal::spectrum_t** (p. 1248) class into the AC space.*
- class **MHA_AC::waveform_t**
  *Insert a **MHASignal::waveform_t** (p. 1263) class into the AC space.*
- class **MHA_AC::int_t**
  Insert a integer variable into the AC space.
- class **MHA_AC::float_t**
  Insert a float point variable into the AC space.
- class **MHA_AC::double_t**
  Insert a double precision floating point variable into the AC space.
- class **MHA_AC::ac2matrix_t**
  Copy AC variable to a matrix.
- class **MHA_AC::acspace2matrix_t**
  Copy all or a subset of all numeric AC variables into an array of matrixes.
- struct **algo_comm_t**
  A reference handle for algorithm communication variables.
- struct **comm_var_t**
  Algorithm communication variable structure.

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- **mha_spec_t MHA_AC::get_var_spectrum** (**algo_comm_t** ac, const std::string &name)

  Convert an AC variable into a spectrum.
- **mha_wave_t MHA_AC::get_var_waveform** (**algo_comm_t** ac, const std::string &name)

  Convert an AC variable into a waveform.
- int **MHA_AC::get_var_int** (**algo_comm_t** ac, const std::string &name)

  Return value of an integer scalar AC variable.
- float **MHA_AC::get_var_float** (**algo_comm_t** ac, const std::string &name)

  Return value of an floating point scalar AC variable.
- std::vector< float > **MHA_AC::get_var_vfloat** (**algo_comm_t** ac, const std::string &name)

  Return value of an floating point vector AC variable as standard vector of floats.

3.4.1 Detailed Description

Algorithms within one chain can share variables for communication with other algorithms. This mechanism allows interaction between algorithms (i.e. separation of noise estimation and noise reduction algorithms, combination of dynamic compression and noise estimation). Through a set of simple C functions, algorithms can propagate variables of any type, even C++ classes, to other algorithms.

An algorithm communication handle (**algo_comm_t** (p. 279)) is passed at initialisation time to the constructor of each plugin class **constructor** (p. 1151). This handle contains a reference handle, **algo_comm_t::handle** (p. 280), and a number of function pointers, **algo_comm_t↩ ::insert_var** (p. 281) etc.. An algorithm communication variable is accessed through objects of type **comm_var_t** (p. 359).

For openMHA users, openMHA provides generic plugins to inspect and store AC variables of numeric types:

- plugin acmon mirrors AC variables of numeric types in readonly configuration variables (called monitors),
- plugin acsave stores AC variables into Matlab or text files. Plugin developers may also want to use these plugins to inspect any AC variables published by their own plugins during testing.

As a developer of openMHA plugin(s), please observe the following best practices in plugins using AC variables:

1. Plugins publishing AC variables:
   - insert all variables during prepare()

© 2005-2021 HörTech gGmbH, Oldenburg

- re-insert all variables during each process()
- memory used for storing AC variable values is allocated and owned by the publishing plugin and needs to remain valid until the next call to process() or release() of the same plugin.

2. Plugins consuming AC variable published by other plugins:
   - poll required variables (and check validity) again during each process() before accessing their values.

3.4.2 Function Documentation

3.4.2.1 get_var_spectrum()

```
mha_spec_t MHA_AC::get_var_spectrum (
    algo_comm_t ac,
    const std::string & name )
```

Convert an AC variable into a spectrum.

This function reads an AC variable and tries to convert it into a valid spectrum. The Spectrum variable is granted to be valid only for one call of the processing function.

Parameters

ac	AC handle
name	Name of the variable

Returns

Spectrum structure

3.4.2.2 get_var_waveform()

```
mha_wave_t MHA_AC::get_var_waveform (
    algo_comm_t ac,
    const std::string & name )
```

Convert an AC variable into a waveform.

This function reads an AC variable and tries to convert it into a valid waveform. The waveform variable is granted to be valid only for one call of the processing function.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

ac	AC handle
name	Name of the variable

Returns

waveform structure

3.4.2.3 get_var_int()

```
int MHA_AC::get_var_int (
    algo_comm_t ac,
    const std::string & name )
```

Return value of an integer scalar AC variable.

Parameters

ac	AC handle
name	Name of the variable

Returns

Variable value

3.4.2.4 get_var_float()

```
float MHA_AC::get_var_float (
    algo_comm_t ac,
    const std::string & name )
```

Return value of an floating point scalar AC variable.

Parameters

ac	AC handle
name	Name of the variable

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

Variable value

3.4.2.5 get_var_vfloat()

```
std::vector< float > MHA_AC::get_var_vfloat (
    algo_comm_t ac,
    const std::string & name )
```

Return value of an floating point vector AC variable as standard vector of floats.

Parameters

ac	AC handle
name	Name of the variable

Returns

Variable value

© 2005-2021 HörTech gGmbH, Oldenburg

3.5 Error handling in the openMHA

Errors are reported to the user via the **MHA_Error** (p. 764) exception.

Collaboration diagram for Error handling in the openMHA:

Classes

- class **MHA_Error**
  Error reporting exception class.

Macros

- #define **MHA_ErrorMsg**(x) **MHA_Error**(__FILE__,__LINE__,"%s",x)
  Throw an openMHA error with a text message.
- #define **MHA_assert**(x) if(!(x)) throw **MHA_Error**(__FILE__,__LINE__,"\"%s\" is false.",#x)
  *Assertion macro, which throws an **MHA_Error** (p. 764).*
- #define **MHA_assert_equal**(a, b) if(a != b) throw **MHA_Error**(__FILE__,__LINE__↩,"\"%s == %s\" is false (%s = %g, %s = %g).",#a,#b,#a,(double)(a),#b,(double)(b))
  *Equality assertion macro, which throws an **MHA_Error** (p. 764) with the values.*

Functions

- void **mha_debug** (const char ∗fmt,...) __attribute__((__format__(printf
  Print an info message (stderr on Linux, OutputDebugString in Windows).

3.5.1 Detailed Description

Errors are reported to the user via the **MHA_Error** (p. 764) exception.

3.5.2 Macro Definition Documentation

3.5.2.1 MHA_ErrorMsg

```
#define MHA_ErrorMsg(
    x ) MHA_Error(__FILE__,__LINE__,"%s",x)
```

Throw an openMHA error with a text message.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

x	Text message.

3.5.2.2 MHA_assert

```
#define MHA_assert(
    x ) if(!(x)) throw MHA_Error(__FILE__,__LINE__,"\"%s\" is false.",#x)
```

Assertion macro, which throws an **MHA_Error** (p. 764).

Parameters

x	Boolean expression which should be true.

3.5.2.3 MHA_assert_equal

```
#define MHA_assert_equal(
    a,
    b ) if( a != b ) throw MHA_Error(__FILE__,__LINE__,"\"%s == %s\" is
false (%s = %g, %s = %g).",#a,#b,#a,(double)(a),#b,(double)(b))
```

Equality assertion macro, which throws an **MHA_Error** (p. 764) with the values.

Parameters

a	Numeric expression which can be converted to double (for printing).
b	Numeric expression which should be equal to a

3.5.3 Function Documentation

3.5.3.1 mha_debug()

```
void mha_debug (
    const char * fmt,
    ... )
```

Print an info message (stderr on Linux, OutputDebugString in Windows).

© 2005-2021 HörTech gGmbH, Oldenburg

3.6 The openMHA configuration language

openMHA Plugins that should use the openMHA configuration language for their configuration have to be implemented in C++ and need to include **mha_parser.hh** (p. 1618). All required classes and functions for parser access are declared in the namespace **MHAParser** (p. 122). The plugin class should be derived from the class **MHAParser::parser_t** (p. 1102) (or **M↩ HAPlugin::plugin_t** (p. 1151)), which symbolises a sub-parser node in the openMHA script hierarchy. Variables of many types can be registered to the sub-parser node by calling the member function **insert_item** (p. 1104).

openMHA Plugins that should use the openMHA configuration language for their configuration have to be implemented in C++ and need to include **mha_parser.hh** (p. 1618). All required classes and functions for parser access are declared in the namespace **MHAParser** (p. 122). The plugin class should be derived from the class **MHAParser::parser_t** (p. 1102) (or **M↩ HAPlugin::plugin_t** (p. 1151)), which symbolises a sub-parser node in the openMHA script hierarchy. Variables of many types can be registered to the sub-parser node by calling the member function **insert_item** (p. 1104).

The openMHA Plugin template class **MHAPlugin::plugin_t** (p. 1151) together with the Plugin macro **MHAPLUGIN_CALLBACKS** (p. 1625) provide the callback mappings and correct inheritance. If your plugin is based on that template class, you simply have to use the insert_item command to give access to your variables, everything else is managed internally.

A complete list of all openMHA script items is given in the description of the **MHAParser** (p. 122) namespace.

© 2005-2021 HörTech gGmbH, Oldenburg

3.7 The openMHA Toolbox library

The openMHA toolbox is a static C++ library which makes it more comfortable to develop openMHA plugins. It contains the openMHA script language classes.

Collaboration diagram for The openMHA Toolbox library:

Modules

- **Error handling in the openMHA**

  *Errors are reported to the user via the **MHA_Error** (p. 764) exception.*
- **Vector and matrix processing toolbox**

  *The vector and matrix processing toolbox consists of a number of classes defined in the namespace **MHASignal** (p. 136), and many functions and operators for use with the structures **mha_wave_t** (p. 840) and **mha_spec_t** (p. 794).*
- **Complex arithmetics in the openMHA**
- **Fast Fourier Transform functions**

Files

- file **mha_algo_comm.h**

  Header file for Algorithm Communication.
- file **mha_filter.hh**

  Header file for IIR filter classes.
- file **mha_signal.hh**

  Header file for audio signal handling and processing classes.
- file **mha_tablelookup.hh**

  Header file for table lookup classes.

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **MHAOvlFilter**
  Namespace for overlapping FFT based filter bank classes and functions.
- **MHAFilter**
  Namespace for IIR and FIR filter classes.
- **MHAParser**
  Name space for the openMHA-Parser configuration language.
- **MHASignal**
  Namespace for audio signal handling and processing classes.
- **MHATableLookup**
  Namespace for table lookup classes.

3.7.1 Detailed Description

The openMHA toolbox is a static C++ library which makes it more comfortable to develop openMHA plugins. It contains the openMHA script language classes.

© 2005-2021 HörTech gGmbH, Oldenburg

3.8 Vector and matrix processing toolbox

The vector and matrix processing toolbox consists of a number of classes defined in the namespace **MHASignal** (p. 136), and many functions and operators for use with the structures **mha_wave_t** (p. 840) and **mha_spec_t** (p. 794).

Collaboration diagram for Vector and matrix processing toolbox:

Namespaces

- **MHASignal**

  Namespace for audio signal handling and processing classes.
- **MHAWindow**

  Collection of Window types.

Classes

- struct **mha_wave_t**

  Waveform signal structure.
- struct **mha_audio_descriptor_t**

  *Description of an audio fragment (planned as a replacement of **mhaconfig_t** (p. 851)).*
- struct **mha_audio_t**

  *An audio fragment in the openMHA (planned as a replacement of **mha_wave_t** (p. 840) and **mha_spec_t** (p. 794)).*
- class **MHASignal::spectrum_t**

  *a signal processing class for spectral data (based on **mha_spec_t** (p. 794))*
- class **MHASignal::waveform_t**

  *signal processing class for waveform data (based on **mha_wave_t** (p. 840))*
- class **MHASignal::doublebuffer_t**

  Double-buffering class.
- class **MHASignal::hilbert_t**

  Hilbert transformation of a waveform segment.
- class **MHASignal::minphase_t**

  Minimal phase function.
- class **MHASignal::uint_vector_t**

  Vector of unsigned values, used for size and index description of n-dimensional matrixes.

© 2005-2021 HörTech gGmbH, Oldenburg

- class **MHASignal::matrix_t**

  n-dimensional matrix with real or complex floating point values.
- class **MHAParser::window_t**

  MHA configuration interface for a window function generator.
- class **MHASignal::delay_wave_t**

  Delayline containing wave fragments.
- class **MHASignal::async_rmslevel_t**

  Class for asynchronous level metering.

Typedefs

- typedef float **mha_real_t**

  openMHA type for real numbers

Functions

- **mha_wave_t range** (**mha_wave_t** s, unsigned int k0, unsigned int len)

  Return a time interval from a waveform chunk.
- **mha_spec_t channels** (**mha_spec_t** s, unsigned int ch_start, unsigned int nch)

  Return a channel interval from a spectrum.
- **mha_real_t MHASignal::bin2freq** (**mha_real_t** bin, unsigned fftlen, **mha_real_t** srate)

  conversion from fft bin index to frequency
- **mha_real_t MHASignal::freq2bin** (**mha_real_t** freq, unsigned fftlen, **mha_real_t** srate)

  conversion from frequency to fft bin index
- **mha_real_t MHASignal::smp2rad** (**mha_real_t** samples, unsigned bin, unsigned fftlen)

  conversion from delay in samples to phase shift
- **mha_real_t MHASignal::rad2smp** (**mha_real_t** phase_shift, unsigned bin, unsigned fftlen)

  conversion from phase shift to delay in samples
- template<class elem_type >
  std::vector< elem_type > **MHASignal::dupvec** (std::vector< elem_type > vec, unsigned n)

  Duplicate last vector element to match desired size.
- template<class elem_type >
  std::vector< elem_type > **MHASignal::dupvec_chk** (std::vector< elem_type > vec, unsigned n)

  Duplicate last vector element to match desired size, check for dimension.
- bool **equal_dim** (const **mha_wave_t** &a, const **mha_wave_t** &b)

  Test for equal dimension of waveform structures.
- bool **equal_dim** (const **mha_wave_t** &a, const **mhaconfig_t** &b)

  Test for match of waveform dimension with mhaconfig structure.

© 2005-2021 HörTech gGmbH, Oldenburg

- bool **equal_dim** (const **mha_spec_t** &a, const **mha_spec_t** &b)
  Test for equal dimension of spectrum structures.
- bool **equal_dim** (const **mha_spec_t** &a, const **mhaconfig_t** &b)
  Test for match of spectrum dimension with mhaconfig structure.
- bool **equal_dim** (const **mha_wave_t** &a, const **mha_spec_t** &b)
  Test for equal dimension of waveform/spectrum structures.
- bool **equal_dim** (const **mha_spec_t** &a, const **mha_wave_t** &b)
  Test for equal dimension of waveform/spectrum structures.
- void **integrate** (**mha_wave_t** &s)
  Numeric integration of a signal vector (real values)
- void **integrate** (**mha_spec_t** &s)
  Numeric integration of a signal vector (complex values)
- unsigned int **size** (const **mha_wave_t** &s)
  Return size of a waveform structure.
- unsigned int **size** (const **mha_spec_t** &s)
  Return size of a spectrum structure.
- unsigned int **size** (const **mha_wave_t** ∗s)
  Return size of a waveform structure.
- unsigned int **size** (const **mha_spec_t** ∗s)
  Return size of a spectrum structure.
- void **clear** (**mha_wave_t** &s)
  Set all values of waveform to zero.
- void **clear** (**mha_wave_t** ∗s)
  Set all values of waveform to zero.
- void **clear** (**mha_spec_t** &s)
  Set all values of spectrum to zero.
- void **clear** (**mha_spec_t** ∗s)
  Set all values of spectrum to zero.
- void **assign** (**mha_wave_t** self, **mha_real_t** val)
  Set all values of waveform 'self' to 'val'.
- void **assign** (**mha_wave_t** self, const **mha_wave_t** &val)
  Set all values of waveform 'self' to 'val'.
- void **assign** (**mha_spec_t** self, const **mha_spec_t** &val)
  Set all values of spectrum 'self' to 'val'.
- void **timeshift** (**mha_wave_t** &self, int shift)
  Time shift of waveform chunk.
- **mha_real_t** & **value** (**mha_wave_t** ∗s, unsigned int fr, unsigned int ch)
  Access an element of a waveform structure.
- const **mha_real_t** & **value** (const **mha_wave_t** ∗s, unsigned int fr, unsigned int ch)
  Constant access to an element of a waveform structure.
- **mha_complex_t** & **value** (**mha_spec_t** ∗s, unsigned int fr, unsigned int ch)
  Access to an element of a spectrum.
- const **mha_complex_t** & **value** (const **mha_spec_t** ∗s, unsigned int fr, unsigned int ch)
  Constant access to an element of a spectrum.
- **mha_real_t** & **value** (**mha_wave_t** &s, unsigned int fr, unsigned int ch)

© 2005-2021 HörTech gGmbH, Oldenburg

Access to an element of a waveform structure.

- const **mha_real_t** & **value** (const **mha_wave_t** &s, unsigned int fr, unsigned int ch)

  Constant access to an element of a waveform structure.

- **mha_complex_t** & **value** (**mha_spec_t** &s, unsigned int fr, unsigned int ch)

  Access to an element of a spectrum.

- const **mha_complex_t** & **value** (const **mha_spec_t** &s, unsigned int fr, unsigned int ch)

  Constant access to an element of a spectrum.

- std::vector< float > **std_vector_float** (const **mha_wave_t** &)

  *Converts a **mha_wave_t** (p. 840) structure into a std::vector<float> (interleaved order).*

- std::vector< std::vector< float > > **std_vector_vector_float** (const **mha_wave_t** &)

  *Converts a **mha_wave_t** (p. 840) structure into a std::vector< std::vector<float> > (outer vector represents channels).*

- std::vector< std::vector< **mha_complex_t** > > **std_vector_vector_complex** (const **mha_spec_t** &)

  *Converts a **mha_spec_t** (p. 794) structure into a std::vector< std::vector<mha_complex_t> > (outer vector represents channels).*

- **mha_wave_t** & **operator+=** (**mha_wave_t** &, const **mha_real_t** &)

  Addition operator.

- **mha_wave_t** & **operator+=** (**mha_wave_t** &, const **mha_wave_t** &)

  Addition operator.

- **mha_wave_t** & **operator-=** (**mha_wave_t** &, const **mha_wave_t** &)

  Subtraction operator.

- **mha_spec_t** & **operator-=** (**mha_spec_t** &, const **mha_spec_t** &)

  Subtraction operator.

- **mha_wave_t** & **operator∗=** (**mha_wave_t** &, const **mha_real_t** &)

  Element-wise multiplication operator.

- **mha_wave_t** & **operator∗=** (**mha_wave_t** &, const **mha_wave_t** &)

  Element-wise multiplication operator.

- **mha_spec_t** & **operator∗=** (**mha_spec_t** &, const **mha_real_t** &)

  Element-wise multiplication operator.

- **mha_spec_t** & **operator∗=** (**mha_spec_t** &, const **mha_wave_t** &)

  Element-wise multiplication operator.

- **mha_spec_t** & **operator∗=** (**mha_spec_t** &, const **mha_spec_t** &)

  Element-wise multiplication operator.

- **mha_spec_t** & **operator/=** (**mha_spec_t** &, const **mha_spec_t** &)

  Element-wise division operator.

- **mha_wave_t** & **operator/=** (**mha_wave_t** &, const **mha_wave_t** &)

  Element-wise division operator.

- **mha_spec_t** & **operator+=** (**mha_spec_t** &, const **mha_spec_t** &)

  Addition operator.

- **mha_spec_t** & **operator+=** (**mha_spec_t** &, const **mha_real_t** &)

  Addition operator.

- **mha_wave_t** & **operator^=** (**mha_wave_t** &self, const **mha_real_t** &arg)

  Exponent operator.

© 2005-2021 HörTech gGmbH, Oldenburg

- void **MHASignal::copy_channel** (**mha_spec_t** &self, const **mha_spec_t** &src, unsigned sch, unsigned dch)

  Copy one channel of a source signal.
- void **MHASignal::copy_channel** (**mha_wave_t** &self, const **mha_wave_t** &src, unsigned src_channel, unsigned dest_channel)

  Copy one channel of a source signal.
- **mha_real_t MHASignal::rmslevel** (const **mha_spec_t** &s, unsigned int channel, unsigned int fftlen)

  Return RMS level of a spectrum channel.
- **mha_real_t MHASignal::colored_intensity** (const **mha_spec_t** &s, unsigned int channel, unsigned int fftlen, **mha_real_t** *sqfreq_response=nullptr)

  Colored spectrum intensity.
- **mha_real_t MHASignal::maxabs** (const **mha_spec_t** &s, unsigned int channel)

  Find maximal absolute value.
- **mha_real_t MHASignal::rmslevel** (const **mha_wave_t** &s, unsigned int channel)

  Return RMS level of a waveform channel.
- **mha_real_t MHASignal::maxabs** (const **mha_wave_t** &s, unsigned int channel)

  Find maximal absolute value.
- **mha_real_t MHASignal::maxabs** (const **mha_wave_t** &s)

  Find maximal absolute value.
- **mha_real_t MHASignal::max** (const **mha_wave_t** &s)

  Find maximal value.
- **mha_real_t MHASignal::min** (const **mha_wave_t** &s)

  Find minimal value.
- **mha_real_t MHASignal::sumsqr_channel** (const **mha_wave_t** &s, unsigned int channel)

  Calculate sum of squared values in one channel.
- **mha_real_t MHASignal::sumsqr_frame** (const **mha_wave_t** &s, unsigned int frame)

  Calculate sum over all channels of squared values.
- void **conjugate** (**mha_spec_t** &self)

  *Replace (!) the value of this **mha_spec_t** (p. 794) with its conjugate.*

3.8.1 Detailed Description

The vector and matrix processing toolbox consists of a number of classes defined in the namespace **MHASignal** (p. 136), and many functions and operators for use with the structures **mha_wave_t** (p. 840) and **mha_spec_t** (p. 794).

3.8.2 Typedef Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

3.8.2.1 mha_real_t `typedef float` **`mha_real_t`**

openMHA type for real numbers

This type is expected to be allways the C-type 'float' (IEEE 754 single).

3.8.3 Function Documentation

3.8.3.1 range()

```
mha_wave_t range (
    mha_wave_t s,
    unsigned int k0,
    unsigned int len )
```

Return a time interval from a waveform chunk.

A waveform chunk containing a time intervall of a larger waveform chunk is returned. The number of channels remains constant. The data of the output waveform structure points to the data of the input structure, i.e., write access to the output waveform chunk modifies the corresponding entries in the input chunk.

Parameters

s	Waveform structure
k0	Index of first value in output
len	Number of frames in output

Returns

Waveform structure representing the sub-interval.

3.8.3.2 channels()

```
mha_spec_t channels (
    mha_spec_t s,
    unsigned int ch_start,
    unsigned int nch )
```

Return a channel interval from a spectrum.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s	Input spectrum
ch_start	Index of first channel in output
nch	Number of channels in output

Returns

Spectrum structure representing the sub-interval.

3.8.3.3 bin2freq()

```
mha_real_t MHASignal::bin2freq (
    mha_real_t bin,
    unsigned fftlen,
    mha_real_t srate ) [inline]
```

conversion from fft bin index to frequency

Parameters

bin	index of fft bin, index 0 has dc
fftlen	FFT length
srate	sampling frequency / Hz

Returns

frequency of fft bin / Hz

3.8.3.4 freq2bin()

```
mha_real_t MHASignal::freq2bin (
    mha_real_t freq,
    unsigned fftlen,
    mha_real_t srate ) [inline]
```

conversion from frequency to fft bin index

Parameters

freq	frequency / Hz
fftlen	FFT length
srate	sampling frequency / Hz

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

0-based index of fft bin, generally has non-zero fractional part

3.8.3.5 smp2rad()

```
mha_real_t MHASignal::smp2rad (
    mha_real_t samples,
    unsigned bin,
    unsigned fftlen ) [inline]
```

conversion from delay in samples to phase shift

Compute phase shift that needs to be applied to fft spectrum to achieve the desired delay.

Parameters

samples	delay in samples. Positive delay: shift current signal to future.
bin	index of fft bin, index 0 has dc (index 0 and nyqvist bin cannot be delayed)
fftlen	FFT length

Returns

The phase shift in radiant that needs to be applied to fft bin to achieve the desired delay. A positive delay requires a negative phase shift. If required phase shift is >pi or <-pi, then the desired delay cannot be applied in the fft domain with given parameters. Required phase shifts close to pi should not be used. If bin is 0 or nyqvist, returns 0 phase shift.

3.8.3.6 rad2smp()

```
mha_real_t MHASignal::rad2smp (
    mha_real_t phase_shift,
    unsigned bin,
    unsigned fftlen ) [inline]
```

conversion from phase shift to delay in samples

Compute delay in samples that is achieved by a phase shift.

Parameters

phase_shift	phase shift in radiant
bin	index of fft bin, index 0 has dc (index 0 and nyqvist bin cannot be delayed)
fftlen	FFT length

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

The delay in samples achieved by applying the phase shift. A negative phase shift causes a positive delay: shifts current signal to future.

3.8.3.7 dupvec()

```
template<class elem_type >
std::vector<elem_type> MHASignal::dupvec (
            std::vector< elem_type > vec,
            unsigned n )
```

Duplicate last vector element to match desired size.

Parameters

vec	Input vector.
n	Target number of elements.

Return values

Resized	vector.

3.8.3.8 dupvec_chk()

```
template<class elem_type >
std::vector<elem_type> MHASignal::dupvec_chk (
            std::vector< elem_type > vec,
            unsigned n )
```

Duplicate last vector element to match desired size, check for dimension.

The input dimension can be either 1 or the target length.

Parameters

vec	Input vector.
n	Target number of elements.

Return values

Resized	vector.

© 2005-2021 HörTech gGmbH, Oldenburg

3.8.3.9 equal_dim() [1/6]

```
bool equal_dim (
    const mha_wave_t & a,
    const mha_wave_t & b ) [inline]
```

Test for equal dimension of waveform structures.

3.8.3.10 equal_dim() [2/6]

```
bool equal_dim (
    const mha_wave_t & a,
    const mhaconfig_t & b ) [inline]
```

Test for match of waveform dimension with mhaconfig structure.

3.8.3.11 equal_dim() [3/6]

```
bool equal_dim (
    const mha_spec_t & a,
    const mha_spec_t & b ) [inline]
```

Test for equal dimension of spectrum structures.

3.8.3.12 equal_dim() [4/6]

```
bool equal_dim (
    const mha_spec_t & a,
    const mhaconfig_t & b ) [inline]
```

Test for match of spectrum dimension with mhaconfig structure.

3.8.3.13 equal_dim() [5/6]

```
bool equal_dim (
    const mha_wave_t & a,
    const mha_spec_t & b ) [inline]
```

Test for equal dimension of waveform/spectrum structures.

Warning

Waveform structures **mha_wave_t** (p. 840) use interleaved data order, while spectrum structures **mha_spec_t** (p. 794) use non-interleaved.

© 2005-2021 HörTech gGmbH, Oldenburg

3.8.3.14 equal_dim() [6/6]

```
bool equal_dim (
    const mha_spec_t & a,
    const mha_wave_t & b ) [inline]
```

Test for equal dimension of waveform/spectrum structures.

Warning

Waveform structures **mha_wave_t** (p. 840) use interleaved data order, while spectrum structures **mha_spec_t** (p. 794) use non-interleaved.

3.8.3.15 integrate() [1/2]

```
void integrate (
    mha_wave_t & s )
```

Numeric integration of a signal vector (real values)

Parameters

s	Input signal vector

3.8.3.16 integrate() [2/2]

```
void integrate (
    mha_spec_t & s )
```

Numeric integration of a signal vector (complex values)

Parameters

s	Input signal vector

3.8.3.17 size() [1/4]

```
unsigned int size (
    const mha_wave_t & s ) [inline]
```

Return size of a waveform structure.

© 2005-2021 HörTech gGmbH, Oldenburg

3.8.3.18 size() [2/4] unsigned int size (const mha_spec_t & s) [inline]

Return size of a spectrum structure.

3.8.3.19 size() [3/4] unsigned int size (const mha_wave_t * s) [inline]

Return size of a waveform structure.

3.8.3.20 size() [4/4] unsigned int size (const mha_spec_t * s) [inline]

Return size of a spectrum structure.

3.8.3.21 clear() [1/4] void clear (mha_wave_t & s) [inline]

Set all values of waveform to zero.

3.8.3.22 clear() [2/4] void clear (mha_wave_t * s) [inline]

Set all values of waveform to zero.

3.8.3.23 clear() [3/4] void clear (mha_spec_t & s) [inline]

Set all values of spectrum to zero.

3.8.3.24 clear() [4/4] void clear (mha_spec_t * s) [inline]

Set all values of spectrum to zero.

3.8.3.25 assign() [1/3] void assign (mha_wave_t self, mha_real_t val) [inline]

Set all values of waveform 'self' to 'val'.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

self	Waveform to be modified.
val	Value to be assigned to all entries of waveform.

3.8.3.26 assign() [2/3]

```
void assign (
    mha_wave_t self,
    const mha_wave_t & val )
```

Set all values of waveform 'self' to 'val'.

Parameters

self	Waveform to be modified.
val	Source waveform structure.

3.8.3.27 assign() [3/3]

```
void assign (
    mha_spec_t self,
    const mha_spec_t & val )
```

Set all values of spectrum 'self' to 'val'.

Parameters

self	Spectrum to be modified.
val	Source spectrum.

3.8.3.28 timeshift()

```
void timeshift (
    mha_wave_t & self,
    int shift )
```

Time shift of waveform chunk.

Shifted areas are filled with zeros.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

self	Waveform chunk to be shifted
shift	Shift amount, positive values shift to later times

3.8.3.29 value() [1/8]

```
mha_real_t& value (
    mha_wave_t * s,
    unsigned int fr,
    unsigned int ch ) [inline]
```

Access an element of a waveform structure.

Parameters

s	Waveform structure
fr	Frame number
ch	Channel number

Returns

Reference to element

3.8.3.30 value() [2/8]

```
const mha_real_t& value (
    const mha_wave_t * s,
    unsigned int fr,
    unsigned int ch ) [inline]
```

Constant access to an element of a waveform structure.

Parameters

s	Waveform structure
fr	Frame number
ch	Channel number

Returns

Reference to element

© 2005-2021 HörTech gGmbH, Oldenburg

3.8.3.31 value() [3/8]

```
mha_complex_t& value (
    mha_spec_t * s,
    unsigned int fr,
    unsigned int ch ) [inline]
```

Access to an element of a spectrum.

Parameters

s	Spectrum structure
fr	Bin number
ch	Channel number

Returns

Reference to element

3.8.3.32 value() [4/8]

```
const mha_complex_t& value (
    const mha_spec_t * s,
    unsigned int fr,
    unsigned int ch ) [inline]
```

Constant access to an element of a spectrum.

Parameters

s	Spectrum structure
fr	Bin number
ch	Channel number

Returns

Reference to element

3.8.3.33 value() [5/8]

```
mha_real_t& value (
    mha_wave_t & s,
    unsigned int fr,
    unsigned int ch ) [inline]
```

Access to an element of a waveform structure.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s	Waveform structure
fr	Frame number
ch	Channel number

Returns

Reference to element

3.8.3.34 value() [6/8]

```
const mha_real_t& value (
    const mha_wave_t & s,
    unsigned int fr,
    unsigned int ch ) [inline]
```

Constant access to an element of a waveform structure.

Parameters

s	Waveform structure
fr	Frame number
ch	Channel number

Returns

Reference to element

3.8.3.35 value() [7/8]

```
mha_complex_t& value (
    mha_spec_t & s,
    unsigned int fr,
    unsigned int ch ) [inline]
```

Access to an element of a spectrum.

Parameters

s	Spectrum structure
fr	Bin number
ch	Channel number

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

Reference to element

3.8.3.36 value() [8/8]

```
const mha_complex_t& value (
    const mha_spec_t & s,
    unsigned int fr,
    unsigned int ch ) [inline]
```

Constant access to an element of a spectrum.

Parameters

s	Spectrum structure
fr	Bin number
ch	Channel number

Returns

Reference to element

3.8.3.37 std_vector_float()

```
std::vector<float> std_vector_float (
    const mha_wave_t & )
```

Converts a **mha_wave_t** (p. 840) structure into a std::vector<float> (interleaved order).

Warning

This function is not real-time safe. Do not use in signal processing thread.

3.8.3.38 std_vector_vector_float()

```
std::vector<std::vector<float> > std_vector_↩
vector_float (
    const mha_wave_t & )
```

Converts a **mha_wave_t** (p. 840) structure into a std::vector< std::vector<float> > (outer vector represents channels).

Warning

This function is not real-time safe. Do not use in signal processing thread.

© 2005-2021 HörTech gGmbH, Oldenburg

3.8.3.39 std_vector_vector_complex()

```
std::vector<std::vector< mha_complex_t> >
std_vector_vector_complex (
            const  mha_spec_t &  )
```

Converts a **mha_spec_t** (p. 794) structure into a std::vector< std::vector<mha_complex_t> > (outer vector represents channels).

Warning

This function is not real-time safe. Do not use in signal processing thread.

3.8.3.40 operator+=() [1/4]

```
mha_wave_t& operator+= (
            mha_wave_t & ,
            const  mha_real_t &  )
```

Addition operator.

3.8.3.41 operator+=() [2/4]

```
mha_wave_t& operator+= (
            mha_wave_t & ,
            const  mha_wave_t &  )
```

Addition operator.

3.8.3.42 operator-=() [1/2]

```
mha_wave_t& operator-= (
            mha_wave_t & ,
            const  mha_wave_t &  )
```

Subtraction operator.

3.8.3.43 operator-=() [2/2]

```
mha_spec_t& operator-= (
            mha_spec_t & ,
            const  mha_spec_t &  )
```

Subtraction operator.

© 2005-2021 HörTech gGmbH, Oldenburg

3.8.3.44 operator*=() [1/5]

```
mha_wave_t& operator*= (
    mha_wave_t & ,
    const mha_real_t & )
```

Element-wise multiplication operator.

3.8.3.45 operator*=() [2/5]

```
mha_wave_t& operator*= (
    mha_wave_t & ,
    const mha_wave_t & )
```

Element-wise multiplication operator.

3.8.3.46 operator*=() [3/5]

```
mha_spec_t& operator*= (
    mha_spec_t & ,
    const mha_real_t & )
```

Element-wise multiplication operator.

3.8.3.47 operator*=() [4/5]

```
mha_spec_t& operator*= (
    mha_spec_t & ,
    const mha_wave_t & )
```

Element-wise multiplication operator.

3.8.3.48 operator*=() [5/5]

```
mha_spec_t& operator*= (
    mha_spec_t & ,
    const mha_spec_t & )
```

Element-wise multiplication operator.

© 2005-2021 HörTech gGmbH, Oldenburg

3.8.3.49 operator/=() [1/2]

```
mha_spec_t& operator/= (
    mha_spec_t & ,
    const mha_spec_t & )
```

Element-wise division operator.

3.8.3.50 operator/=() [2/2]

```
mha_wave_t& operator/= (
    mha_wave_t & ,
    const mha_wave_t & )
```

Element-wise division operator.

3.8.3.51 operator+=() [3/4]

```
mha_spec_t& operator+= (
    mha_spec_t & ,
    const mha_spec_t & )
```

Addition operator.

3.8.3.52 operator+=() [4/4]

```
mha_spec_t& operator+= (
    mha_spec_t & ,
    const mha_real_t & )
```

Addition operator.

3.8.3.53 operator^=()

```
mha_wave_t& operator^= (
    mha_wave_t & self,
    const mha_real_t & arg )
```

Exponent operator.

Warning

This overwrites the xor operator!

3.8.3.54 copy_channel() [1/2]

```
void MHASignal::copy_channel (
    mha_spec_t & self,
    const mha_spec_t & src,
    unsigned sch,
    unsigned dch )
```

Copy one channel of a source signal.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

self	Destination.
src	Source
sch	Source channel number
dch	Destination channel number

3.8.3.55 copy_channel() [2/2]

```
void MHASignal::copy_channel (
    mha_wave_t & self,
    const mha_wave_t & src,
    unsigned src_channel,
    unsigned dest_channel )
```

Copy one channel of a source signal.

Parameters

self	Destination.
src	Source
src_channel	Source channel number
dest_channel	Destination channel number

3.8.3.56 rmslevel() [1/2]

```
mha_real_t MHASignal::rmslevel (
    const mha_spec_t & s,
    unsigned int channel,
    unsigned int fftlen )
```

Return RMS level of a spectrum channel.

Computes the RMS level of the signal in Pascal in the given channel.

Takes into account the the negative frequency bins that are not stored (**Central Calibration** (p. 3)).

Parameters

s	Input spectrum
channel	Channel number to be tested
fftlen	FFT length (to correctly count the level of the Nyquist bin)

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

RMS level in Pa

3.8.3.57 colored_intensity()

```
mha_real_t MHASignal::colored_intensity (
    const mha_spec_t & s,
    unsigned int channel,
    unsigned int fftlen,
    mha_real_t * sqfreq_response = nullptr )
```

Colored spectrum intensity.

computes the squared sum of the spectrum after filtering with the frequency response. Takes into account the negative frequency bins that are not stored (**Central Calibration** (p. 3)).

Parameters

s	Input spectrum
channel	Channel number to be tested
fftlen	FFT length (to correctly count the level of the Nyquist bin)
sqfreq_response	An array with one squared weighting factor for every fft bin. Array length must be equal to s->num_frames. nullptr can be given for equal weighting of all frequencies.

Returns

sum of squares. Root of this is the colored level in Pa

3.8.3.58 maxabs() [1/3]

```
mha_real_t MHASignal::maxabs (
    const mha_spec_t & s,
    unsigned int channel )
```

Find maximal absolute value.

Parameters

s	Input signal
channel	Channel to be tested

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

maximum absolute value

3.8.3.59 rmslevel() [2/2]

```
mha_real_t MHASignal::rmslevel (
    const mha_wave_t & s,
    unsigned int channel )
```

Return RMS level of a waveform channel.

Parameters

s	Input waveform signal
channel	Channel number to be tested

Returns

RMS level in Pa

3.8.3.60 maxabs() [2/3]

```
mha_real_t MHASignal::maxabs (
    const mha_wave_t & s,
    unsigned int channel )
```

Find maximal absolute value.

Parameters

s	Input signal
channel	Channel to be tested

Returns

maximum absolute value

3.8.3.61 maxabs() [3/3]

```
mha_real_t MHASignal::maxabs (
    const mha_wave_t & s )
```

Find maximal absolute value.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s	Input signal

Returns

maximum absolute value

3.8.3.62 max()

```
mha_real_t MHASignal::max (
    const mha_wave_t & s )
```

Find maximal value.

Parameters

s	Input signal

Returns

maximum absolute value

3.8.3.63 min()

```
mha_real_t MHASignal::min (
    const mha_wave_t & s )
```

Find minimal value.

Parameters

s	Input signal

Returns

maximum absolute value

© 2005-2021 HörTech gGmbH, Oldenburg

3.8.3.64 sumsqr_channel()

```
mha_real_t MHASignal::sumsqr_channel (
    const mha_wave_t & s,
    unsigned int channel )
```

Calculate sum of squared values in one channel.

Parameters

s	Input signal
channel	Channel

Returns

$\sum x^2$

3.8.3.65 sumsqr_frame()

```
mha_real_t MHASignal::sumsqr_frame (
    const mha_wave_t & s,
    unsigned int frame )
```

Calculate sum over all channels of squared values.

Parameters

s	Input signal
frame	Frame number

Returns

$\sum x^2$

3.8.3.66 conjugate()

```
void conjugate (
    mha_spec_t & self ) [inline]
```

Replace (!) the value of this **mha_spec_t** (p. 794) with its conjugate.

© 2005-2021 HörTech gGmbH, Oldenburg

3.9 Complex arithmetics in the openMHA

Collaboration diagram for Complex arithmetics in the openMHA:

Classes

- struct **mha_complex_t**

  Type for complex floating point values.

Functions

- **mha_complex_t** & **set** (**mha_complex_t** &self, **mha_real_t** real, **mha_real_t** imag=0)

  *Assign real and imaginary parts to a **mha_complex_t** (p. 745) variable.*
- **mha_complex_t** **mha_complex** (**mha_real_t** real, **mha_real_t** imag=0)

  *Create a new **mha_complex_t** (p. 745) with specified real and imaginary parts.*
- **mha_complex_t** & **set** (**mha_complex_t** &self, const std::complex< **mha_real_t** > & **stdcomplex**)

  *Assign a **mha_complex_t** (p. 745) variable from a std::complex.*
- std::complex< **mha_real_t** > **stdcomplex** (const **mha_complex_t** &self)

  *Create a std::complex from **mha_complex_t** (p. 745).*
- **mha_complex_t** & **expi** (**mha_complex_t** &self, **mha_real_t** **angle**)

  *replaces the value of the given **mha_complex_t** (p. 745) with exp(i∗b).*
- double **angle** (const **mha_complex_t** &self)

  Computes the angle of a complex number in the complex plane.
- **mha_complex_t** & **operator+=** (**mha_complex_t** &self, const **mha_complex_t** &other)

  Addition of two complex numbers, overwriting the first.
- **mha_complex_t** **operator+** (const **mha_complex_t** &self, const **mha_complex_t** &other)

  Addition of two complex numbers, result is a temporary object.
- **mha_complex_t** & **operator+=** (**mha_complex_t** &self, **mha_real_t** other_real)

  Addition of a complex and a real number, overwriting the complex.
- **mha_complex_t** **operator+** (const **mha_complex_t** &self, **mha_real_t** other_real)

  Addition of a complex and a real number, result is a temporary object.

© 2005-2021 HörTech gGmbH, Oldenburg

- **mha_complex_t** & **operator-=** (**mha_complex_t** &self, const **mha_complex_t** &other)

  Subtraction of two complex numbers, overwriting the first.
- **mha_complex_t** **operator-** (const **mha_complex_t** &self, const **mha_complex_t** &other)

  Subtraction of two complex numbers, result is a temporary object.
- **mha_complex_t** & **operator-=** (**mha_complex_t** &self, **mha_real_t** other_real)

  Subtraction of a complex and a real number, overwriting the complex.
- **mha_complex_t** **operator-** (const **mha_complex_t** &self, **mha_real_t** other_real)

  Subtraction of a complex and a real number, result is a temporary object.
- **mha_complex_t** & **operator∗=** (**mha_complex_t** &self, const **mha_complex_t** &other)

  Multiplication of two complex numbers, overwriting the first.
- **mha_complex_t** **operator∗** (const **mha_complex_t** &self, const **mha_complex_t** &other)

  Multiplication of two complex numbers, result is a temporary object.
- **mha_complex_t** & **operator∗=** (**mha_complex_t** &self, **mha_real_t** other_real)

  Multiplication of a complex and a real number, overwriting the complex.
- **mha_complex_t** & **expi** (**mha_complex_t** &self, **mha_real_t** **angle**, **mha_real_t** factor)

  *replaces (!) the value of the given **mha_complex_t** (p. 745) with a ∗ exp(i∗b)*
- **mha_complex_t** **operator∗** (const **mha_complex_t** &self, **mha_real_t** other_real)

  Multiplication of a complex and a real number, result is a temporary object.
- **mha_real_t** **abs2** (const **mha_complex_t** &self)

  Compute the square of the absolute value of a complex value.
- **mha_real_t** **abs** (const **mha_complex_t** &self)

  Compute the absolute value of a complex value.
- **mha_complex_t** & **operator/=** (**mha_complex_t** &self, **mha_real_t** other_real)

  Division of a complex and a real number, overwriting the complex.
- **mha_complex_t** **operator/** (const **mha_complex_t** &self, **mha_real_t** other_real)

  Division of a complex and a real number, result is a temporary object.
- **mha_complex_t** & **safe_div** (**mha_complex_t** &self, const **mha_complex_t** &other, **mha_real_t** eps, **mha_real_t** eps2)
- **mha_complex_t** & **operator/=** (**mha_complex_t** &self, const **mha_complex_t** &other)

  Division of two complex numbers, overwriting the first.
- **mha_complex_t** **operator/** (const **mha_complex_t** &self, const **mha_complex_t** &other)

  Division of two complex numbers, result is a temporary object.
- **mha_complex_t** **operator-** (const **mha_complex_t** &self)

  Unary minus on a complex results in a negative temporary object.
- bool **operator==** (const **mha_complex_t** &x, const **mha_complex_t** &y)

  Compare two complex numbers for equality.
- bool **operator!=** (const **mha_complex_t** &x, const **mha_complex_t** &y)

  Compare two complex numbers for inequality.
- void **conjugate** (**mha_complex_t** &self)

  *Replace (!) the value of this **mha_complex_t** (p. 745) with its conjugate.*
- **mha_complex_t** **_conjugate** (const **mha_complex_t** &self)

© 2005-2021 HörTech gGmbH, Oldenburg

Compute the cojugate of this complex value.

- void **reciprocal** (**mha_complex_t** &self)

  Replace the value of this complex with its reciprocal.

- **mha_complex_t** **_reciprocal** (const **mha_complex_t** &self)

  compute the reciprocal of this complex value.

- void **normalize** (**mha_complex_t** &self)

  Divide a complex by its absolute value, thereby normalizing it (projecting onto the unit circle).

- void **normalize** (**mha_complex_t** &self, **mha_real_t** margin)

  Divide a complex by its absolute value, thereby normalizing it (projecting onto the unit circle), with a safety margin.

- bool **almost** (const **mha_complex_t** &self, const **mha_complex_t** &other, **mha_real_t** times_epsilon=1e2)

  Compare two complex numbers for equality except for a small relative error.

- bool **operator**< (const **mha_complex_t** &x, const **mha_complex_t** &y)

  Compares the absolute values of two complex numbers.

3.9.1 Detailed Description

3.9.2 Function Documentation

3.9.2.1 set() [1/2]

```
mha_complex_t& set (
    mha_complex_t & self,
    mha_real_t real,
    mha_real_t imag = 0 ) [inline]
```

Assign real and imaginary parts to a **mha_complex_t** (p. 745) variable.

Parameters

self	The **mha_complex_t** (p. 745) variable whose value is about to change.
real	The new real part.
imag	The new imaginary part.

Returns

A reference to the changed variable.

© 2005-2021 HörTech gGmbH, Oldenburg

3.9.2.2 mha_complex()

```
mha_complex_t mha_complex (
    mha_real_t real,
    mha_real_t imag = 0 ) [inline]
```

Create a new **mha_complex_t** (p. 745) with specified real and imaginary parts.

Parameters

real	The real part.
imag	The imaginary part.

Returns

The new value.

3.9.2.3 set() [2/2]

```
mha_complex_t& set (
    mha_complex_t & self,
    const std::complex< mha_real_t > & stdcomplex ) [inline]
```

Assign a **mha_complex_t** (p. 745) variable from a std::complex.

Parameters

self	The **mha_complex_t** (p. 745) variable whose value is about to change.
stdcomplex	The new complex value.

Returns

A reference to the changed variable.

3.9.2.4 stdcomplex()

```
std::complex< mha_real_t> stdcomplex (
    const mha_complex_t & self ) [inline]
```

Create a std::complex from **mha_complex_t** (p. 745).

3.9.2.5 expi() [1/2]

```
mha_complex_t& expi (
    mha_complex_t & self,
    mha_real_t angle ) [inline]
```

replaces the value of the given **mha_complex_t** (p. 745) with exp(i∗b).

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

self	The **mha_complex_t** (p. 745) variable whose value is about to change.
angle	The angle in the complex plane [rad].

Returns

A reference to the changed variable.

3.9.2.6 angle()

```
double angle (
    const mha_complex_t & self ) [inline]
```

Computes the angle of a complex number in the complex plane.

Parameters

self	The complex number whose angle is needed.

Returns

The angle of a complex number in the complex plane.

3.9.2.7 operator+=() [1/2]

```
mha_complex_t& operator+= (
    mha_complex_t & self,
    const mha_complex_t & other ) [inline]
```

Addition of two complex numbers, overwriting the first.

3.9.2.8 operator+() [1/2]

```
mha_complex_t operator+ (
    const mha_complex_t & self,
    const mha_complex_t & other ) [inline]
```

Addition of two complex numbers, result is a temporary object.

© 2005-2021 HörTech gGmbH, Oldenburg

3.9.2.9 operator+=() [2/2]

```
mha_complex_t& operator+= (
    mha_complex_t & self,
    mha_real_t other_real ) [inline]
```

Addition of a complex and a real number, overwriting the complex.

3.9.2.10 operator+() [2/2]

```
mha_complex_t operator+ (
    const mha_complex_t & self,
    mha_real_t other_real ) [inline]
```

Addition of a complex and a real number, result is a temporary object.

3.9.2.11 operator-=() [1/2]

```
mha_complex_t& operator-= (
    mha_complex_t & self,
    const mha_complex_t & other ) [inline]
```

Subtraction of two complex numbers, overwriting the first.

3.9.2.12 operator-() [1/3]

```
mha_complex_t operator- (
    const mha_complex_t & self,
    const mha_complex_t & other ) [inline]
```

Subtraction of two complex numbers, result is a temporary object.

3.9.2.13 operator-=() [2/2]

```
mha_complex_t& operator-= (
    mha_complex_t & self,
    mha_real_t other_real ) [inline]
```

Subtraction of a complex and a real number, overwriting the complex.

© 2005-2021 HörTech gGmbH, Oldenburg

3.9.2.14 operator-() [2/3]

```
mha_complex_t operator- (
    const mha_complex_t & self,
    mha_real_t other_real ) [inline]
```

Subtraction of a complex and a real number, result is a temporary object.

3.9.2.15 operator∗=() [1/2]

```
mha_complex_t& operator*= (
    mha_complex_t & self,
    const mha_complex_t & other ) [inline]
```

Multiplication of two complex numbers, overwriting the first.

3.9.2.16 operator∗() [1/2]

```
mha_complex_t operator* (
    const mha_complex_t & self,
    const mha_complex_t & other ) [inline]
```

Multiplication of two complex numbers, result is a temporary object.

3.9.2.17 operator∗=() [2/2]

```
mha_complex_t& operator*= (
    mha_complex_t & self,
    mha_real_t other_real ) [inline]
```

Multiplication of a complex and a real number, overwriting the complex.

3.9.2.18 expi() [2/2]

```
mha_complex_t& expi (
    mha_complex_t & self,
    mha_real_t angle,
    mha_real_t factor ) [inline]
```

replaces (!) the value of the given **mha_complex_t** (p. 745) with a ∗ exp(i∗b)

Parameters

self	The **mha_complex_t** (p. 745) variable whose value is about to change.
angle	The imaginary exponent.
factor	The absolute value of the result.

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

A reference to the changed variable.

3.9.2.19 operator*() [2/2]

```
mha_complex_t operator* (
    const mha_complex_t & self,
    mha_real_t other_real ) [inline]
```

Multiplication of a complex and a real number, result is a temporary object.

3.9.2.20 abs2()

```
mha_real_t abs2 (
    const mha_complex_t & self ) [inline]
```

Compute the square of the absolute value of a complex value.

Returns

The square of the absolute value of self.

3.9.2.21 abs()

```
mha_real_t abs (
    const mha_complex_t & self ) [inline]
```

Compute the absolute value of a complex value.

Returns

The absolute value of self.

3.9.2.22 operator/=() [1/2]

```
mha_complex_t& operator/= (
    mha_complex_t & self,
    mha_real_t other_real ) [inline]
```

Division of a complex and a real number, overwriting the complex.

© 2005-2021 HörTech gGmbH, Oldenburg

3.9.2.23 operator/() [1/2]

```
mha_complex_t operator/ (
    const mha_complex_t & self,
    mha_real_t other_real ) [inline]
```

Division of a complex and a real number, result is a temporary object.

3.9.2.24 safe_div()

```
mha_complex_t& safe_div (
    mha_complex_t & self,
    const mha_complex_t & other,
    mha_real_t eps,
    mha_real_t eps2 ) [inline]
```

Safe division of two complex numbers, overwriting the first. If abs(divisor) $<$ eps, then divisor is replaced by eps. eps2 = eps$*$eps.

3.9.2.25 operator/=() [2/2]

```
mha_complex_t& operator/= (
    mha_complex_t & self,
    const mha_complex_t & other ) [inline]
```

Division of two complex numbers, overwriting the first.

3.9.2.26 operator/() [2/2]

```
mha_complex_t operator/ (
    const mha_complex_t & self,
    const mha_complex_t & other ) [inline]
```

Division of two complex numbers, result is a temporary object.

3.9.2.27 operator-() [3/3]

```
mha_complex_t operator- (
    const mha_complex_t & self ) [inline]
```

Unary minus on a complex results in a negative temporary object.

© 2005-2021 HörTech gGmbH, Oldenburg

3.9.2.28 operator==()

```
bool operator== (
    const mha_complex_t & x,
    const mha_complex_t & y ) [inline]
```

Compare two complex numbers for equality.

3.9.2.29 operator"!=()

```
bool operator!= (
    const mha_complex_t & x,
    const mha_complex_t & y ) [inline]
```

Compare two complex numbers for inequality.

3.9.2.30 conjugate()

```
void conjugate (
    mha_complex_t & self ) [inline]
```

Replace (!) the value of this **mha_complex_t** (p. 745) with its conjugate.

3.9.2.31 _conjugate()

```
mha_complex_t _conjugate (
    const mha_complex_t & self ) [inline]
```

Compute the cojugate of this complex value.

Returns

A temporary object holding the conjugate value.

3.9.2.32 reciprocal()

```
void reciprocal (
    mha_complex_t & self ) [inline]
```

Replace the value of this complex with its reciprocal.

© 2005-2021 HörTech gGmbH, Oldenburg

3.9.2.33 _reciprocal()

```
mha_complex_t _reciprocal (
    const mha_complex_t & self ) [inline]
```

compute the reciprocal of this complex value.

Returns

A temporary object holding the reciprocal value.

3.9.2.34 normalize() [1/2]

```
void normalize (
    mha_complex_t & self ) [inline]
```

Divide a complex by its absolute value, thereby normalizing it (projecting onto the unit circle).

3.9.2.35 normalize() [2/2]

```
void normalize (
    mha_complex_t & self,
    mha_real_t margin ) [inline]
```

Divide a complex by its absolute value, thereby normalizing it (projecting onto the unit circle), with a safety margin.

3.9.2.36 almost()

```
bool almost (
    const mha_complex_t & self,
    const mha_complex_t & other,
    mha_real_t times_epsilon = 1e2 ) [inline]
```

Compare two complex numbers for equality except for a small relative error.

Parameters

self	The first complex number.
other	The second complex number.
times_epsilon	Permitted relative error is this number multiplied with the machine accuracy for this Floating point format (std::numeric_limits<mha_real_t>::epsilon)

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

true if the relative difference is below times_epsilon ∗ std::numeric_limits<mha_real_t>↩ ::epsilon

3.9.2.37 operator<()

```
bool operator< (
    const mha_complex_t & x,
    const mha_complex_t & y ) [inline]
```

Compares the absolute values of two complex numbers.

© 2005-2021 HörTech gGmbH, Oldenburg

3.10 Fast Fourier Transform functions

Collaboration diagram for Fast Fourier Transform functions:

Typedefs

- typedef void ∗ **mha_fft_t**
  Handle for an FFT object.

Functions

- **mha_fft_t mha_fft_new** (unsigned int n)
  Create a new FFT handle.
- void **mha_fft_free** (**mha_fft_t** h)
  Destroy an FFT handle.
- void **mha_fft_wave2spec** (**mha_fft_t** h, const **mha_wave_t** ∗in, **mha_spec_t** ∗out)
  Tranform waveform segment into spectrum.
- void **mha_fft_wave2spec** (**mha_fft_t** h, const **mha_wave_t** ∗in, **mha_spec_t** ∗out, bool swaps)
  Tranform waveform segment into spectrum.
- void **mha_fft_spec2wave** (**mha_fft_t** h, const **mha_spec_t** ∗in, **mha_wave_t** ∗out)
  Tranform spectrum into waveform segment.
- void **mha_fft_spec2wave** (**mha_fft_t** h, const **mha_spec_t** ∗in, **mha_wave_t** ∗out, unsigned int offset)
  Tranform spectrum into waveform segment. out may have fewer number of frames than needed for a complete iFFT. Only as many frames are written into out as fit, starting with offset offset of the complete iFFT.
- void **mha_fft_forward** (**mha_fft_t** h, **mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)
  Complex to complex FFT (forward).
- void **mha_fft_backward** (**mha_fft_t** h, **mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)
  Complex to complex FFT (backward).
- void **mha_fft_forward_scale** (**mha_fft_t** h, **mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)
  Complex to complex FFT (forward).
- void **mha_fft_backward_scale** (**mha_fft_t** h, **mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)
  Complex to complex FFT (backward).

© 2005-2021 HörTech gGmbH, Oldenburg

- void **mha_fft_wave2spec_scale** (**mha_fft_t** h, const **mha_wave_t** ∗in, **mha_spec_t** ∗out)

  Tranform waveform segment into spectrum.
- void **mha_fft_spec2wave_scale** (**mha_fft_t** h, const **mha_spec_t** ∗in, **mha_wave_t** ∗out)

  Tranform spectrum into waveform segment.

3.10.1 Detailed Description

3.10.2 Typedef Documentation

3.10.2.1 mha_fft_t `typedef void*` **`mha_fft_t`**

Handle for an FFT object.

This FFT object is used by the functions mha_fft_wave2spec and mha_fft_spec2wave. The F↩ FT back-end is the FFTW library. The back-end is completely hidden, including external header files or linking external libraries is not required.

3.10.3 Function Documentation

3.10.3.1 mha_fft_new() **`mha_fft_t`** `mha_fft_new (`
`unsigned int` *`n`* `)`

Create a new FFT handle.

Parameters

n	FFT length.

Create a new FFT handle.

Parameters

n	FFT length

© 2005-2021 HörTech gGmbH, Oldenburg

Return values

FFT	object

3.10.3.2 mha_fft_free()

```
void mha_fft_free (
    mha_fft_t h )
```

Destroy an FFT handle.

Parameters

h	Handle to be destroyed.

Destroy an FFT handle.

Parameters

h	FFT object to be removed

3.10.3.3 mha_fft_wave2spec() [1/2]

```
void mha_fft_wave2spec (
    mha_fft_t h,
    const mha_wave_t * in,
    mha_spec_t * out )
```

Tranform waveform segment into spectrum.

Parameters

h	FFT handle.
in	Input waveform segment.
out	Output spectrum.

Tranform waveform segment into spectrum.

Parameters

h	FFT object handle
in	pointer to input waveform signal
out	pointer to output spectrum signal (has to be allocated)

© 2005-2021 HörTech gGmbH, Oldenburg

3.10.3.4 mha_fft_wave2spec() [2/2]

```
void mha_fft_wave2spec (
    mha_fft_t h,
    const mha_wave_t * in,
    mha_spec_t * out,
    bool swaps )
```

Tranform waveform segment into spectrum.

Like normal wave2spec, but swaps wave buffer halves before transforming if the swaps parameter is true.

Warning: These openMHA FFTs adopt a nonstandard scaling scheme in which the forward transform scales by 1/N and the backward does not scale. We would recommend using the '_scale' methods instead.

Parameters

h	FFT handle.
in	Input waveform segment.
out	Output spectrum.
swaps	Function swaps the first and second half of the waveform buffer before the FFT transform when this parameter is set to true.

3.10.3.5 mha_fft_spec2wave() [1/2]

```
void mha_fft_spec2wave (
    mha_fft_t h,
    const mha_spec_t * in,
    mha_wave_t * out )
```

Tranform spectrum into waveform segment.

Warning: These openMHA FFTs adopt a nonstandard scaling scheme in which the forward transform scales by 1/N and the backward does not scale. We would recommend using the '_scale' methods instead.

Parameters

h	FFT handle.
in	Input spectrum.
out	Output waveform segment.

© 2005-2021 HörTech gGmbH, Oldenburg

Tranform spectrum into waveform segment.

Parameters

h	FFT object handle
in	pointer to input spectrum
out	pointer to output waveform signal (has to be allocated)

3.10.3.6 mha_fft_spec2wave() [2/2]

```
void mha_fft_spec2wave (
    mha_fft_t h,
    const mha_spec_t * in,
    mha_wave_t * out,
    unsigned int offset )
```

Tranform spectrum into waveform segment. out may have fewer number of frames than needed for a complete iFFT. Only as many frames are written into out as fit, starting with offset offset of the complete iFFT.

Warning: These openMHA FFTs adopt a nonstandard scaling scheme in which the forward transform scales by 1/N and the backward does not scale. We would recommend using the '_scale' methods instead.

Parameters

h	FFT handle.
in	Input spectrum.
out	Output waveform segment.
offset	Offset into iFFT wave buffer

Tranform spectrum into waveform segment. out may have fewer number of frames than needed for a complete iFFT. Only as many frames are written into out as fit, starting with offset offset of the complete iFFT.

Only part of the iFFT is tranferred into the out buffer.

Out may have fewer number of freames than needed for a complete iFFT. Only as many frames are written into out as fit, starting with offset offset of the complete iFFT.

Parameters

h	FFT object handle
in	pointer to input spectrum
out	pointer to output waveform signal (has to be allocated)
offset	Offset into complete iFFT buffer.

© 2005-2021 HörTech gGmbH, Oldenburg

3.10.3.7 mha_fft_forward()

```
void mha_fft_forward (
    mha_fft_t h,
    mha_spec_t * sIn,
    mha_spec_t * sOut )
```

Complex to complex FFT (forward).

sIn and sOut need to have nfft bins (please note that **mha_spec_t** (p. 794) typically has nfft/2+1 bins for half-complex representation).

Warning: These openMHA FFTs adopt a nonstandard scaling scheme in which the forward transform scales by 1/N and the backward does not scale. We would recommend using the '_scale' methods instead.

Parameters

h	FFT handle.
sIn	Input spectrum.
sOut	Output spectrum.

3.10.3.8 mha_fft_backward()

```
void mha_fft_backward (
    mha_fft_t h,
    mha_spec_t * sIn,
    mha_spec_t * sOut )
```

Complex to complex FFT (backward).

sIn and sOut need to have nfft bins (please note that **mha_spec_t** (p. 794) typically has nfft/2+1 bins for half-complex representation).

Warning: These openMHA FFTs adopt a nonstandard scaling scheme in which the forward transform scales by 1/N and the backward does not scale. We would recommend using the '_scale' methods instead.

Parameters

h	FFT handle.
sIn	Input spectrum.
sOut	Output spectrum.

© 2005-2021 HörTech gGmbH, Oldenburg

3.10.3.9 mha_fft_forward_scale()

```
void mha_fft_forward_scale (
    mha_fft_t h,
    mha_spec_t * sIn,
    mha_spec_t * sOut )
```

Complex to complex FFT (forward).

sIn and sOut need to have nfft bins (please note that **mha_spec_t** (p. 794) typically has nfft/2+1 bins for half-complex representation).

The _scale methods use standard DFT scaling: There is no scaling in the forward transformation, and 1/N scaling for the backward.

Parameters

h	FFT handle.
sIn	Input spectrum.
sOut	Output spectrum.

3.10.3.10 mha_fft_backward_scale()

```
void mha_fft_backward_scale (
    mha_fft_t h,
    mha_spec_t * sIn,
    mha_spec_t * sOut )
```

Complex to complex FFT (backward).

sIn and sOut need to have nfft bins (please note that **mha_spec_t** (p. 794) typically has nfft/2+1 bins for half-complex representation).

The _scale methods use standard DFT scaling: There is no scaling in the forward transformation, and 1/N scaling for the backward.

Parameters

h	FFT handle.
sIn	Input spectrum.
sOut	Output spectrum.

© 2005-2021 HörTech gGmbH, Oldenburg

3.10.3.11 mha_fft_wave2spec_scale()

```
void mha_fft_wave2spec_scale (
    mha_fft_t h,
    const mha_wave_t * in,
    mha_spec_t * out )
```

Tranform waveform segment into spectrum.

The _scale methods use standard DFT scaling: There is no scaling in the forward transformation, and 1/N scaling for the backward.

Parameters

h	FFT handle.
in	Input waveform segment.
out	Output spectrum.

3.10.3.12 mha_fft_spec2wave_scale()

```
void mha_fft_spec2wave_scale (
    mha_fft_t h,
    const mha_spec_t * in,
    mha_wave_t * out )
```

Tranform spectrum into waveform segment.

The _scale methods use standard DFT scaling: There is no scaling in the forward transformation, and 1/N scaling for the backward.

Parameters

h	FFT handle.
in	Input spectrum.
out	Output waveform segment.

© 2005-2021 HörTech gGmbH, Oldenburg

4 Namespace Documentation

4.1 ac2lsl Namespace Reference

All types for the **ac2lsl** (p. 78) plugins live in this namespace.

Classes

- class **ac2lsl_t**
  *Plugin class of **ac2lsl** (p. 78).*
- class **cfg_t**
  *Runtime configuration class of the **ac2lsl** (p. 78) plugin.*
- class **save_var_base_t**
  Interface for ac to lsl bridge variable.
- class **save_var_t**
  Implementation for all ac to lsl bridges except complex types.
- class **save_var_t**< **mha_complex_t** >
  *Template specialization of the **ac2lsl** (p. 78) bridge to take care of complex numbers.*
- struct **type_info**

Variables

- const std::map< int, **type_info** > **types**

4.1.1 Detailed Description

All types for the **ac2lsl** (p. 78) plugins live in this namespace.

4.1.2 Variable Documentation

4.1.2.1 types `const std::map<int, type_info> ac2lsl::types`

4.2 ac_proc Namespace Reference

Classes

- class **interface_t**

© 2005-2021 HörTech gGmbH, Oldenburg

4.3 acmon Namespace Reference

Namespace for displaying ac variables as parser monitors.

Classes

- class **ac_monitor_t**
  A class for converting AC variables to Parser monitors of correct type.
- class **acmon_t**

4.3.1 Detailed Description

Namespace for displaying ac variables as parser monitors.

4.4 acsave Namespace Reference

Classes

- class **acsave_t**
- class **cfg_t**
- struct **mat4head_t**
- class **save_var_t**

4.5 addsndfile Namespace Reference

Classes

- class **addsndfile_if_t**
- class **level_adapt_t**
- class **resampled_soundfile_t**
  Reads sound from file and resamples it if necessary and wanted.
- class **sndfile_t**
- class **waveform_proxy_t**
  Class helps to specify which instance of MHASignal_waveform_t parent instance is meant in ***resampled_soundfile_t*** *(p. 257).*

Typedefs

- typedef **MHAPlugin::config_t**< **level_adapt_t** > **level_adaptor**
- typedef **MHAPlugin::plugin_t**< **sndfile_t** > **wave_reader**

© 2005-2021 HörTech gGmbH, Oldenburg

Enumerations

- enum **addsndfile_resampling_mode_t** { **DONT_RESAMPLE_PERMISSIVE**, **DON↩T_RESAMPLE_STRICT**, **DO_RESAMPLE** }

  Specifies the resampling mode in resampled_soundfile_t.

Functions

- static unsigned **resampled_num_frames** (unsigned num_source_frames, float source↩_rate, float target_rate, **addsndfile_resampling_mode_t** resampling_mode)

4.5.1 Typedef Documentation

4.5.1.1 level_adaptor `typedef` **`MHAPlugin::config_t< level_adapt_t> addsndfile↩::level_adaptor`**

4.5.1.2 wave_reader `typedef` **`MHAPlugin::plugin_t< sndfile_t> addsndfile::wave_↩reader`**

4.5.2 Enumeration Type Documentation

4.5.2.1 addsndfile_resampling_mode_t `enum` **`addsndfile::addsndfile_resampling_↩mode_t`**

Specifies the resampling mode in **resampled_soundfile_t** (p. 257).

Enumerator

DONT_RESAMPLE_PERMISSIVE	
DONT_RESAMPLE_STRICT	Do not resample, if the sample rate of the MHA differs from the sample rate of the sound file, raise an error.
DO_RESAMPLE	Resample.

© 2005-2021 HörTech gGmbH, Oldenburg

4.5.3 Function Documentation

4.5.3.1 resampled_num_frames()

```
static unsigned addsndfile::resampled_num_frames
(
        unsigned num_source_frames,
        float source_rate,
        float target_rate,
         addsndfile_resampling_mode_t resampling_mode ) [static]
```

4.6 ADM Namespace Reference

Classes

- class **ADM**
  Adaptive differential microphone, working for speech frequency range.
- class **Delay**
  A delay-line class.
- class **Linearphase_FIR**
  An efficient linear-phase fir filter implementation.

Functions

- static double **subsampledelay_coeff** (double samples, double f_design, double fs=1.0)
  compute IIR coefficient for subsample delay

Variables

- const double **PI** = 3.14159265358979312
- const double **C** = 340
- const double **DELAY_FREQ** = 2000
- const double **START_BETA** = 0.5

4.6.1 Function Documentation

4.6.1.1 subsampledelay_coeff()

```
static double ADM::subsampledelay_coeff (
        double samples,
        double f_design,
        double fs = 1.0 ) [static]
```

compute IIR coefficient for subsample delay

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

samples	Constraint: 0.0 <= samples < 1.0; Amount of sub-sample delay
f_design	design frequency (subsample delay is accurate for this frequency)
fs	sampling rate

Returns

IIR coefficient for subsample delay

4.6.2 Variable Documentation

4.6.2.1 PI `const double ADM::PI = 3.14159265358979312`

4.6.2.2 C `const double ADM::C = 340`

4.6.2.3 DELAY_FREQ `const double ADM::DELAY_FREQ = 2000`

4.6.2.4 START_BETA `const double ADM::START_BETA = 0.5`

4.7 audiometerbackend Namespace Reference

Classes

- class **audiometer_if_t**
- class **level_adapt_t**
- class **lnn3rdoct_t**
- class **signal_gen_t**
- class **sine_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Typedefs

- typedef **MHAPlugin::config_t**< **level_adapt_t** > **level_adaptor**
- typedef **MHAPlugin::plugin_t**< **signal_gen_t** > **generator**

Functions

- static unsigned int **gcd** (unsigned int a, unsigned int b)
- **MHASignal::waveform_t return_sig** (unsigned int sigtype, unsigned int fs, unsigned int f)

4.7.1 Typedef Documentation

4.7.1.1 level_adaptor `typedef MHAPlugin::config_t< level_adapt_t> audiometerbackend↩ ::level_adaptor`

4.7.1.2 generator `typedef MHAPlugin::plugin_t< signal_gen_t> audiometerbackend↩ ::generator`

4.7.2 Function Documentation

4.7.2.1 gcd()

```
static unsigned int audiometerbackend::gcd (
    unsigned int a,
    unsigned int b ) [inline], [static]
```

4.7.2.2 return_sig()

```
MHASignal::waveform_t audiometerbackend::return_sig (
    unsigned int sigtype,
    unsigned int fs,
    unsigned int f )
```

© 2005-2021 HörTech gGmbH, Oldenburg

4.8 AuditoryProfile Namespace Reference

Namespace for classes and functions around the auditory profile (e.g., audiogram handling)

Classes

- class **fmap_t**
  A class to store frequency dependent data (e.g., HTL and UCL).
- class **parser_t**
  Class to make the auditory profile accessible through the parser interface.
- class **profile_t**
  The Auditory Profile class.

4.8.1 Detailed Description

Namespace for classes and functions around the auditory profile (e.g., audiogram handling)

The auditory profile as defined by HearCom or BMBF Modellbasierte Hoergeraete is stored in the class **AuditoryProfile::profile_t** (p. 331). Until a complete definition is available, only the currently needed elements are implemented.

4.9 coherence Namespace Reference

Classes

- class **cohflt_if_t**
- class **cohflt_t**
- class **vars_t**

Functions

- void **getcipd** (**mha_complex_t** &c, **mha_real_t** &a, const **mha_complex_t** &xl, const **mha_complex_t** &xr)

4.9.1 Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

4.9.1.1 getcipd()

```
void coherence::getcipd (
    mha_complex_t & c,
    mha_real_t & a,
    const mha_complex_t & xl,
    const mha_complex_t & xr ) [inline]
```

4.10 cpuload Namespace Reference

Classes

- class **cpuload_cfg_t**
- class **cpuload_if_t**

4.11 dbasync_native Namespace Reference

Classes

- class **db_if_t**
- class **dbasync_t**
- class **delay_check_t**

Enumerations

- enum { **INVALID_THREAD_PRIORITY** = 999999999 }

Functions

- static void * **thread_start** (void *instance)
- static unsigned **gcd** (unsigned a, unsigned b)

4.11.1 Enumeration Type Documentation

4.11.1.1 anonymous enum

```
anonymous enum
```

© 2005-2021 HörTech gGmbH, Oldenburg

Enumerator

INVALID_THREAD_PRIORITY	

4.11.2 Function Documentation

4.11.2.1 thread_start()

```
static void* dbasync_native::thread_start (
    void * instance ) [static]
```

4.11.2.2 gcd()

```
static unsigned dbasync_native::gcd (
    unsigned a,
    unsigned b ) [inline], [static]
```

4.12 dc Namespace Reference

Classes

- class **dc_if_t**
- class **dc_t**
- class **dc_vars_t**
- class **dc_vars_validator_t**

4.13 dc_simple Namespace Reference

Classes

- class **dc_if_t**

  *interface class for **dc_simple*** (p. 86)
- class **dc_t**

  *Runtime config class for **dc_simple*** (p. 86) *plugin.*
- class **dc_vars_t**

  *class for **dc_simple*** (p. 86) *plugin which registers variables to **MHAParser*** (p. 122).
- class **dc_vars_validator_t**

  Helper class to check sizes of configuration variable vectors.
- class **level_smoother_t**

  Class which computes smoothed input levels on individual bands, using an attack and release filter, which are a first order low pass filter and a maximum tracker filter, respectively.

© 2005-2021 HörTech gGmbH, Oldenburg

Typedefs

- typedef **MHAPlugin::plugin_t**< **dc_t** > **DC**
  Define alternate name for runtime_cfg_t.
- typedef **MHAPlugin::config_t**< **level_smoother_t** > **LEVEL**
  Define alternate name for config_t.

Functions

- void **test_fail** (const std::vector< float > &v, unsigned int s, const std::string &name)
  Checks size of vector.
- std::vector< float > **force_resize** (const std::vector< float > &v, unsigned int s, const std::string &name)
  Creates a copy of vector `v` with `s` elements, provided that \v has either s elements or 1 elements.
- **mha_real_t not_zero** (**mha_real_t** x, const std::string &comment)
  Helper function to throw an error if `x` is 0.

4.13.1 Typedef Documentation

4.13.1.1 DC `typedef` **`MHAPlugin::plugin_t< dc_t> dc_simple::DC`**

Define alternate name for runtime_cfg_t.

4.13.1.2 LEVEL `typedef` **`MHAPlugin::config_t< level_smoother_t> dc_simple::LEVEL`**

Define alternate name for config_t.

4.13.2 Function Documentation

4.13.2.1 test_fail()

```
void dc_simple::test_fail (
    const std::vector< float > & v,
    unsigned int s,
    const std::string & name )
```

Checks size of vector.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

in	*v*	The vector to check the size of.
in	*s*	Expected size of vector v.
in	*name*	Name of vector to include in error message when size does not match.

Exceptions

MHA_Error *(p. 764)*	if the size of v is neither s nor 1.

4.13.2.2 force_resize()

```
std::vector< float > dc_simple::force_resize (
    const std::vector< float > & v,
    unsigned int s,
    const std::string & name )
```

Creates a copy of vector v with s elements, provided that \v has either s elements or 1 elements.

Parameters

in	*v*	The vector to copy elements from.
in	*s*	The desired number of elements in the output vector.
in	*name*	Name of vector to include in error message when input size does not match expectation.

Returns

A copy of v with s elements.

Exceptions

MHA_Error *(p. 764)*	if size of v is neither s nor 1.

4.13.2.3 not_zero()

```
mha_real_t dc_simple::not_zero (
    mha_real_t x,
    const std::string & comment )
```

Helper function to throw an error if x is 0.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

in	*x*	The value to check.
in	*comment*	Optional explanation for error message.

Exceptions

MHA_Error *(p. 764)*	if x == 0.

4.14 delay Namespace Reference

Classes

- class **interface_t**

4.15 delaysum Namespace Reference

This namespace contains the delaysum plugin.

Classes

- class **delaysum_wave_if_t**
  Interface class for the delaysum plugin.
- class **delaysum_wave_t**
  Runtime configuration of the delaysum_wave plugin.

4.15.1 Detailed Description

This namespace contains the delaysum plugin.

4.16 delaysum_spec Namespace Reference

Classes

- class **delaysum_spec_if_t**
- class **delaysum_t**

© 2005-2021 HörTech gGmbH, Oldenburg

4.17 double2acvar Namespace Reference

Classes

- class **double2acvar_t**
  *Plugin interface class for **double2acvar** (p. 90).*

4.18 DynComp Namespace Reference

dynamic compression related classes and functions

Classes

- class **dc_afterburn_rt_t**
  Real-time class for after burn effect.
- class **dc_afterburn_t**
  Afterburn class, to be defined as a member of compressors.
- class **dc_afterburn_vars_t**
  *Variables for **dc_afterburn_t** (p. 453) class.*
- class **gaintable_t**
  Gain table class.

Functions

- **mha_real_t interp1** (const std::vector< **mha_real_t** > &vX, const std::vector< **mha_real_t** > &vY, **mha_real_t** X)
  One-dimensional linear interpolation.
- **mha_real_t interp2** (const std::vector< **mha_real_t** > &vX, const std::vector< **mha_real_t** > &vY, const std::vector< std::vector< **mha_real_t** > > &mZ, **mha_real_t** X, **mha_real_t** Y)
  Linear interpolation in a two-dimensional field.

4.18.1 Detailed Description

dynamic compression related classes and functions

4.18.2 Function Documentation

4.18.2.1 interp1()

```
mha_real_t DynComp::interp1 (
    const std::vector< mha_real_t > & vX,
    const std::vector< mha_real_t > & vY,
    mha_real_t X )
```

One-dimensional linear interpolation.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

vX	Vector with input samples.
vY	Vector with values at input samples.
X	Input value to be interpolated.

Return values

Interpolated	value Y(X) at position X.

4.18.2.2 interp2()

```
mha_real_t DynComp::interp2 (
    const std::vector< mha_real_t > & vX,
    const std::vector< mha_real_t > & vY,
    const std::vector< std::vector< mha_real_t > > & mZ,
    mha_real_t X,
    mha_real_t Y )
```

Linear interpolation in a two-dimensional field.

Parameters

vX	Vector with input samples, first dimension.
vY	Vector with input samples, second dimension.
mZ	Field with values at input samples.
X	First dimension of input value to be interpolated.
Y	Second dimension of input value to be interpolated.

Return values

Interpolated	value Z(X,Y) at position X,Y.

4.19 equalize Namespace Reference

Classes

- class **cfg_t**
- class **freqgains_t**

© 2005-2021 HörTech gGmbH, Oldenburg

4.20 fader_wave Namespace Reference

Classes

- class **fader_wave_if_t**
- class **level_adapt_t**

Typedefs

- typedef **MHAPlugin::plugin_t**< **level_adapt_t** > **level_adaptor**

4.20.1 Typedef Documentation

4.20.1.1 level_adaptor `typedef` **`MHAPlugin::plugin_t< level_adapt_t> fader_wave↩ ::level_adaptor`**

4.21 ffttfbpow Namespace Reference

Namespace for the ffttfbpow plugin.

Classes

- class **ffttfbpow_interface_t**
  Interface class for ffttfbpow plugin.
- class **ffttfbpow_t**
  Run time configuration for the ffttfbpow plugin.

4.21.1 Detailed Description

Namespace for the ffttfbpow plugin.

4.22 ffttfilter Namespace Reference

Classes

- class **ffttfilter_t**
- class **interface_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- unsigned int **irs_length** (const **MHAParser::mfloat_t** &irs)
- unsigned int **irs_validator** (const **MHAParser::mfloat_t** &irs, const unsigned int & **channels**, const unsigned int &fragsize, const unsigned int &fftlen)

4.22.1 Function Documentation

4.22.1.1 irs_length()

```
unsigned int fftfilter::irs_length (
    const MHAParser::mfloat_t & irs )
```

Return the length of the longest vector in irs.

Parameters

irs	"Matrix" of floats parser variable

Returns

length of the longest vector in irs, or 1 if all vectors are empty

4.22.1.2 irs_validator()

```
unsigned int fftfilter::irs_validator (
    const MHAParser::mfloat_t & irs,
    const unsigned int & channels,
    const unsigned int & fragsize,
    const unsigned int & fftlen )
```

Validity checks. Throws Error if parameters are invalid: Number of channels must be $>$ 0, fftlen must be $>=$ fragsize. The number of rows in irs has to match the number of channels, or has to be exactly 1. None of the row vectors may be empty. The longest supported impulse response is (fftlen - fragsize + 1). Impulse responses longer than (fftlen - fragsize + 1) would cause temporal aliasing.

Parameters

irs	The matrix containing the impulse responses (one response per channel, or the same response for every channels) as set by the parser.
channels	The number of prepared audio channels for this MHA plugin.
fragsize	The block size (samples per channel) for waveform audio data
fftlen	FFT length used for filtering.

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

the length of the longest impulse response vector in irs.

4.23 fftfilterbank Namespace Reference

Classes

- class **fftfb_interface_t**
- class **fftfb_plug_t**

4.24 fshift Namespace Reference

All types for the fshift plugin live in this namespace.

Classes

- class **fshift_config_t**
  fshift runtime config class
- class **fshift_t**
  fshift plugin interface class

Functions

- int **fft_find_bin** (**mha_real_t** frequency, unsigned fftlen, **mha_real_t** srate)
  Finds bin number of FFT bin nearest to the given frequency.

4.24.1 Detailed Description

All types for the fshift plugin live in this namespace.

4.24.2 Function Documentation

4.24.2.1 fft_find_bin()

```
int fshift::fft_find_bin (
    mha_real_t frequency,
    unsigned fftlen,
    mha_real_t srate )
```

Finds bin number of FFT bin nearest to the given frequency.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

frequency	The frequency for which to look. Has to be in range [-srate/2,+srate/2]
fftlen	Length of the FFT.
srate	Sampling rate of the waveform from which the FFT originates.

Returns

Bin number of the FFT bin corresponding to frequency

4.25 fshift_hilbert Namespace Reference

All types for the hilbert frequency shifter live in this namespace.

Classes

- class **frequency_translator_t**
- class **hilbert_shifter_t**

4.25.1 Detailed Description

All types for the hilbert frequency shifter live in this namespace.

4.26 gain Namespace Reference

Classes

- class **gain_if_t**
- class **scaler_t**

4.27 gsc_adaptive_stage Namespace Reference

Classes

- class **gsc_adaptive_stage**
- class **gsc_adaptive_stage_if**

  Plugin interface class.

© 2005-2021 HörTech gGmbH, Oldenburg

Variables

- constexpr **mha_real_t DELT** =1e-12
  Small constant to ensure no division by zero occurs.

4.27.1 Variable Documentation

4.27.1.1 DELT `constexpr mha_real_t gsc_adaptive_stage::DELT =1e-12 [constexpr]`

Small constant to ensure no division by zero occurs.

4.28 gtfb_analyzer Namespace Reference

Classes

- struct **gtfb_analyzer_cfg_t**
  Configuration for Gammatone Filterbank Analyzer.
- class **gtfb_analyzer_t**
  Gammatone Filterbank Analyzer Plugin.

4.29 level_matching Namespace Reference

Classes

- class **channel_pair**
- class **level_matching_config_t**
- class **level_matching_t**

4.30 lsl2ac Namespace Reference

Classes

- class **cfg_t**
  *Runtime configuration class of the **lsl2ac** (p. 96) plugin.*
- class **lsl2ac_t**
  *Plugin class of **lsl2ac** (p. 96).*
- class **save_var_t**
  LSL to AC bridge variable.

© 2005-2021 HörTech gGmbH, Oldenburg

Enumerations

- enum **overrun_behavior** { **overrun_behavior::Discard** =0, **overrun_behavior::Ignore** }

4.30.1 Enumeration Type Documentation

4.30.1.1 overrun_behavior `enum` **`lsl2ac::overrun_behavior`** `[strong]`

Enumerator

Discard	
Ignore	

4.31 matlab_wrapper Namespace Reference

Namespace where all classes of the matlab wrapper plugin live.

Classes

- class **callback**

  Utility class connecting a user_config_t instance to its corresponding configuration parser.
- class **matlab_wrapper_rt_cfg_t**

  Thin wrapper around the emxArray containing the user defined configuration variables.
- class **matlab_wrapper_t**

  Matlab wraper plugin interface class.
- struct **types**
- struct **types**< **MHA_SPECTRUM** >
- struct **types**< **MHA_WAVEFORM** >

4.31.1 Detailed Description

Namespace where all classes of the matlab wrapper plugin live.

© 2005-2021 HörTech gGmbH, Oldenburg

4.32 matrixmixer Namespace Reference

Classes

- class **cfg_t**
- class **matmix_t**

4.33 mconv Namespace Reference

Classes

- class **MConv**

4.34 MHA_AC Namespace Reference

Functions and classes for Algorithm Communication (AC) support.

Classes

- class **ac2matrix_helper_t**
- class **ac2matrix_t**

  Copy AC variable to a matrix.
- class **acspace2matrix_t**

  Copy all or a subset of all numeric AC variables into an array of matrixes.
- class **double_t**

  Insert a double precision floating point variable into the AC space.
- class **float_t**

  Insert a float point variable into the AC space.
- class **int_t**

  Insert a integer variable into the AC space.
- class **spectrum_t**

  *Insert a **MHASignal::spectrum_t** (p. 1248) class into the AC space.*
- class **stat_t**
- class **waveform_t**

  *Insert a **MHASignal::waveform_t** (p. 1263) class into the AC space.*

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- **mha_spec_t get_var_spectrum** (**algo_comm_t** ac, const std::string &name)
  Convert an AC variable into a spectrum.
- **mha_wave_t get_var_waveform** (**algo_comm_t** ac, const std::string &name)
  Convert an AC variable into a waveform.
- int **get_var_int** (**algo_comm_t** ac, const std::string &name)
  Return value of an integer scalar AC variable.
- float **get_var_float** (**algo_comm_t** ac, const std::string &name)
  Return value of an floating point scalar AC variable.
- std::vector< float > **get_var_vfloat** (**algo_comm_t** ac, const std::string &name)
  Return value of an floating point vector AC variable as standard vector of floats.

4.34.1 Detailed Description

Functions and classes for Algorithm Communication (AC) support.

4.35 mha_error_helpers Namespace Reference

Functions

- unsigned **digits** (unsigned n)
  Compute number of decimal digits required to represent an unsigned integer.
- unsigned **snprintf_required_length** (const char *formatstring,...)
  snprintf_required_length Compute the number of bytes (excluding the terminating nul) required to store the result of an snprintf.

4.35.1 Function Documentation

4.35.1.1 digits() `unsigned mha_error_helpers::digits ( unsigned n )`

Compute number of decimal digits required to represent an unsigned integer.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

n	The unsigned integer that we want to know the number of required decimal digits for. return The number of decimal digits in `n`.

4.35.1.2 snprintf_required_length()

```
unsigned mha_error_helpers::snprintf_required_length (
            const char * formatstring,
             ... )
```

snprintf_required_length Compute the number of bytes (excluding the terminating nul) required to store the result of an snprintf.

Parameters

formatstring	The format string with standard printf formatstring

Returns

the number of bytes required by printf without the terminating nul

4.36 MHA_TCP Namespace Reference

A Namespace for TCP helper classes.

Classes

- class **Async_Notify**
  Portable Multiplexable cross-thread notification.
- class **Client**
  A portable class for a tcp client connections.
- class **Connection**
  Connection *(p. 803) handles Communication between client and server, is used on both sides.*
- class **Event_Watcher**
  OS-independent event watcher, uses select on Unix and WaitForMultipleObjects on Windows.
- struct **OS_EVENT_TYPE**
- class **Server**
- class **sock_init_t**
- class **Sockaccept_Event**

© 2005-2021 HörTech gGmbH, Oldenburg

- class **Sockread_Event**

  Watch socket for incoming data.
- class **Sockwrite_Event**
- class **Thread**

  A very simple class for portable threads.
- class **Timeout_Event**
- class **Timeout_Watcher**

  OS-independent event watcher with internal fixed-end-time timeout.
- class **Wakeup_Event**

  A base class for asynchronous wakeup events.

Typedefs

- typedef int **SOCKET**

Functions

- std::string **STRERROR** (int err)

  Portable conversion from error number to error string.
- std::string **HSTRERROR** (int err)

  Portable conversion from hostname error number to error string.
- int **N_ERRNO** ()

  Portable access to last network error number.
- int **H_ERRNO** ()

  Portable access to last hostname error number.
- int **G_ERRNO** ()

  Portable access to last non-network error number.
- double **dtime** ()

  Time access function for system's high resolution time, retrieve current time as double.
- double **dtime** (const struct timeval &tv)

  Time access function for unix' high resolution time, converts struct timeval to double.
- struct timeval **stime** (double d)

  Time access function for unix' high resolution time, converts time from double to struct timeval.

Variables

- class **MHA_TCP::sock_init_t sock_initializer**

4.36.1 Detailed Description

A Namespace for TCP helper classes.

© 2005-2021 HörTech gGmbH, Oldenburg

4.36.2 Typedef Documentation

4.36.2.1 SOCKET `typedef int` **`MHA_TCP::SOCKET`**

4.36.3 Function Documentation

4.36.3.1 STRERROR() `std::string MHA_TCP::STRERROR ( int err )`

Portable conversion from error number to error string.

4.36.3.2 HSTRERROR() `std::string MHA_TCP::HSTRERROR ( int err )`

Portable conversion from hostname error number to error string.

4.36.3.3 N_ERRNO() `int MHA_TCP::N_ERRNO ( )`

Portable access to last network error number.

4.36.3.4 H_ERRNO() `int MHA_TCP::H_ERRNO ( )`

Portable access to last hostname error number.

© 2005-2021 HörTech gGmbH, Oldenburg

4.36.3.5 G_ERRNO() `int MHA_TCP::G_ERRNO ( )`

Portable access to last non-network error number.

4.36.3.6 dtime() [1/2] `double MHA_TCP::dtime ( )`

Time access function for system's high resolution time, retrieve current time as double.

4.36.3.7 dtime() [2/2] `double MHA_TCP::dtime ( const struct timeval & tv )`

Time access function for unix' high resolution time, converts struct timeval to double.

4.36.3.8 stime() `struct timeval MHA_TCP::stime ( double d )`

Time access function for unix' high resolution time, converts time from double to struct timeval.

4.36.4 Variable Documentation

4.36.4.1 sock_initializer `class MHA_TCP::sock_init_t MHA_TCP::sock_initializer`

4.37 mha_tcp Namespace Reference

namespace for network communication classes of MHA

Classes

- class **buffered_socket_t**

  An asio TCP socket with an associated streambuf buffer for receiving lines of text, as well as string buffers for sending responses.
- class **server_t**

  Class for accepting TCP connections from clients.

© 2005-2021 HörTech gGmbH, Oldenburg

4.37.1 Detailed Description

namespace for network communication classes of MHA

4.38 mhachain Namespace Reference

Classes

- class **chain_base_t**
- class **mhachain_t**
- class **plugs_t**

4.39 MHAEvents Namespace Reference

Collection of event handling classes.

Classes

- class **connector_base_t**
- class **connector_t**
- class **emitter_t**

  Class for emitting openMHA events.
- class **patchbay_t**

  Patchbay which connects any event emitter with any member function of the parameter class.

4.39.1 Detailed Description

Collection of event handling classes.

4.40 MHAFilter Namespace Reference

Namespace for IIR and FIR filter classes.

© 2005-2021 HörTech gGmbH, Oldenburg

Classes

- class **adapt_filter_param_t**
- class **adapt_filter_state_t**
- class **adapt_filter_t**

  Adaptive filter.
- class **blockprocessing_polyphase_resampling_t**

  A class that does polyphase resampling and takes into account block processing.
- class **complex_bandpass_t**

  Complex bandpass filter.
- class **diff_t**

  Differentiator class (non-normalized)
- class **fftfilter_t**

  FFT based FIR filter implementation.
- class **fftfilterbank_t**

  FFT based FIR filterbank implementation.
- class **filter_t**

  Generic IIR filter class.
- class **gamma_flt_t**

  Class for gammatone filter.
- class **iir_filter_state_t**
- class **iir_filter_t**

  IIR filter class wrapper for integration into parser structure.
- class **iir_ord1_real_t**

  First order recursive filter.
- class **o1_ar_filter_t**

  First order attack-release lowpass filter.
- class **o1flt_lowpass_t**

  First order low pass filter.
- class **o1flt_maxtrack_t**

  First order maximum tracker.
- class **o1flt_mintrack_t**

  First order minimum tracker.
- class **partitioned_convolution_t**

  A filter class for partitioned convolution.
- class **polyphase_resampling_t**

  A class that performs polyphase resampling.
- class **resampling_filter_t**

  Hann shaped low pass filter for resampling.
- class **smoothspec_t**

  Smooth spectral gains, create a windowed impulse response.
- class **thirdoctave_analyzer_t**
- struct **transfer_function_t**

  a structure containing a source channel number, a target channel number, and an impulse response.
- struct **transfer_matrix_t**

  A sparse matrix of transfer function partitionss.

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- template<typename T , typename std::enable_if< std::is_floating_point< T >::value, T >::type * = nullptr> void **make_friendly_number** (T &x)
- void **o1_lp_coeffs** (const **mha_real_t** tau, const **mha_real_t** fs, **mha_real_t** &c1, **mha_real_t** &c2)

  Set first order filter coefficients from time constant and sampling rate.
- void **butter_stop_ord1** (double *A, double *B, double f1, double f2, double fs)

  Setup a first order butterworth band stop filter.
- std::vector< float > **fir_lp** (float f_pass_, float f_stop_, float fs_, unsigned order_)

  Setup a nth order fir low pass filter.
- **MHASignal::waveform_t** * **spec2fir** (const **mha_spec_t** *spec, const unsigned int fftlen, const **MHAWindow::base_t** &window, const bool minphase)

  Create a windowed impulse response/FIR filter coefficients from a spectrum.
- unsigned **gcd** (unsigned a, unsigned b)

  greatest common divisor
- double **sinc** (double x)

  sin(x)/x function, coping with x=0.
- std::pair< unsigned, unsigned > **resampling_factors** (float source_sampling_rate, float target_sampling_rate, float factor=1.0f)

  Computes rational resampling factor from two sampling rates.

4.40.1 Detailed Description

Namespace for IIR and FIR filter classes.

4.40.2 Function Documentation

4.40.2.1 make_friendly_number()

```
template<typename T , typename std::enable_if<
std::is_floating_point< T >::value, T >::type * = nullptr>
void MHAFilter::make_friendly_number (
            T & x )  [inline]
```

4.40.2.2 o1_lp_coeffs()

```
void MHAFilter::o1_lp_coeffs (
            const  mha_real_t tau,
            const  mha_real_t fs,
             mha_real_t & c1,
             mha_real_t & c2 )
```

Set first order filter coefficients from time constant and sampling rate.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

tau	Time constant
fs	Sampling rate

Return values

c1	Recursive filter coefficient
c2	Non-recursive filter coefficient

4.40.2.3 butter_stop_ord1()

```
void MHAFilter::butter_stop_ord1 (
    double * A,
    double * B,
    double f1,
    double f2,
    double fs )
```

Setup a first order butterworth band stop filter.

This function calculates the filter coefficients of a first order butterworth band stop filter.

Return values

A	recursive filter coefficients
B	non recursive filter coefficients

Parameters

f1	lower frequency
f2	upper frequency
fs	sample frequency

4.40.2.4 fir_lp()

```
std::vector< float > MHAFilter::fir_lp (
    float f_pass_,
    float f_stop_,
    float fs_,
    unsigned order_ )
```

Setup a nth order fir low pass filter.

© 2005-2021 HörTech gGmbH, Oldenburg

This function calculates the filter coefficients of a nth order fir low pass filter filter. Frequency arguments above the nyquist frequency are accepted but the spectral response is truncated at the nyquist frequency

Returns

vector containing filter coefficients

Precondition

f_pass_ must be smaller or equal to f_stop_.

Parameters

*f_pass_*	Upper passband frequency
f_stop-	Lower stopband frequency
*fs_*	sample frequency

4.40.2.5 spec2fir()

```
MHASignal::waveform_t * MHAFilter::spec2fir (
    const mha_spec_t * spec,
    const unsigned int fftlen,
    const MHAWindow::base_t & window,
    const bool minphase )
```

Create a windowed impulse response/FIR filter coefficients from a spectrum.

Parameters

spec	Input spectrum
fftlen	FFT length of spectrum
window	Window shape (with length, e.g. initialized with MHAWindow::hanning(54)).
minphase	Flag, true if original phase should be discarded and replaced by a minimal phase function.

4.40.2.6 gcd()

```
unsigned MHAFilter::gcd (
    unsigned a,
    unsigned b ) [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

greatest common divisor

4.40.2.7 sinc()

```
double MHAFilter::sinc (
    double x )
```

sin(x)/x function, coping with x=0.

This is the historical sinc function, not the normalized sinc function.

4.40.2.8 resampling_factors()

```
std::pair< unsigned, unsigned > MHAFilter::resampling↩
_factors (
    float source_sampling_rate,
    float target_sampling_rate,
    float factor = 1.0f )
```

Computes rational resampling factor from two sampling rates.

The function will fail if either sampling_rate $*$ factor is not an integer

Parameters

source_sampling_rate	The original sampling rate
target_sampling_rate	The desired sampling rate
factor	A helper factor to use for non-integer sampling rates

Returns

a pair that contains first the upsampling factor and second the downsampling factor required for the specified resampling.

Exceptions

MHA_Error *(p. 764)*	if no rational resampling factor can be found.

4.41 MHAIOJack Namespace Reference

JACK IO.

© 2005-2021 HörTech gGmbH, Oldenburg

Classes

- class **io_jack_t**
  Main class for JACK IO.

4.41.1 Detailed Description

JACK IO.

4.42 MHAIOJackdb Namespace Reference

Classes

- class **io_jack_t**
  Main class for JACK IO.

4.43 MHAIOPortAudio Namespace Reference

Classes

- class **device_info_t**
- class **io_portaudio_t**
  Main class for Portaudio sound IO.
- class **stream_info_t**

Functions

- static std::string **parserFriendlyName** (const std::string &in)

4.43.1 Function Documentation

4.43.1.1 parserFriendlyName()

```
static std::string MHAIOPortAudio::parserFriendly↩
Name (
            const std::string & in )  [static]
```

© 2005-2021 HörTech gGmbH, Oldenburg

4.44 mhaioutils Namespace Reference

Functions

- template<typename T >
  T **to_int_clamped** (float val)

4.44.1 Function Documentation

4.44.1.1 to_int_clamped() template<typename T >

```
T mhaioutils::to_int_clamped (
            float val )
```

4.45 MHAJack Namespace Reference

Classes and functions for openMHA and JACK interaction.

Classes

- class **client_avg_t**
  Generic JACK client for averaging a system response across time.
- class **client_noncont_t**
  Generic client for synchronous playback and recording of waveform fragments.
- class **client_t**
  Generic asynchronous JACK client.
- class **port_t**
  Class for one channel/port.

Functions

- void **io** (**mha_wave_t** ∗s_out, **mha_wave_t** ∗s_in, const std::string &name, const std↩ ::vector< std::string > &p_out, const std::vector< std::string > &p_in, float ∗srate=NULL, unsigned int ∗fragsize=NULL, bool use_jack_transport=false)
  Functional form of generic client for synchronous playback and recording of waveform fragments.
- std::vector< unsigned int > **get_port_capture_latency** (const std::vector< std::string > &ports)
  Return the JACK port latency of ports.
- std::vector< int > **get_port_capture_latency_int** (const std::vector< std::string > &ports)
  Return the JACK port latency of ports.
- std::vector< unsigned int > **get_port_playback_latency** (const std::vector< std::string > &ports)
  Return the JACK port latency of ports.
- std::vector< int > **get_port_playback_latency_int** (const std::vector< std::string > &ports)

© 2005-2021 HörTech gGmbH, Oldenburg

4.45.1 Detailed Description

Classes and functions for openMHA and JACK interaction.

4.45.2 Function Documentation

4.45.2.1 io()

```
void MHAJack::io (
    mha_wave_t * s_out,
    mha_wave_t * s_in,
    const std::string & name,
    const std::vector< std::string > & p_out,
    const std::vector< std::string > & p_in,
    float * srate = NULL,
    unsigned int * fragsize = NULL,
    bool use_jack_transport = false )
```

Functional form of generic client for synchronous playback and recording of waveform fragments.

4.45.2.2 get_port_capture_latency()

```
std::vector< unsigned int > MHAJack::get_port_capture_latency (
    const std::vector< std::string > & ports )
```

Return the JACK port latency of ports.

Parameters

ports	Ports to be tested

Returns

Latency vector (one entry for each port)

4.45.2.3 get_port_capture_latency_int()

```
std::vector< int > MHAJack::get_port_capture_latency_int (
    const std::vector< std::string > & ports )
```

© 2005-2021 HörTech gGmbH, Oldenburg

Return the JACK port latency of ports.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

ports	Ports to be tested

Returns

Latency vector (one entry for each port)

4.45.2.4 get_port_playback_latency()

```
std::vector< unsigned int > MHAJack::get_↩
port_playback_latency (
            const std::vector< std::string > & ports )
```

Return the JACK port latency of ports.

Parameters

ports	Ports to be tested

Returns

Latency vector (one entry for each port)

4.45.2.5 get_port_playback_latency_int()

```
std::vector< int > MHAJack::get_port_↩
playback_latency_int (
            const std::vector< std::string > & ports )
```

4.46 MHAKernel Namespace Reference

Classes

- class **algo_comm_class_t**
- class **comm_var_map_t**

Functions

- **algo_comm_class_t** ∗ **algo_comm_safe_cast** (void ∗)

© 2005-2021 HörTech gGmbH, Oldenburg

4.46.1 Function Documentation

4.46.1.1 algo_comm_safe_cast()

```
MHAKernel::algo_comm_class_t * MHAKernel::algo_↩
comm_safe_cast (
            void * h )
```

4.47 MHAMultiSrc Namespace Reference

Collection of classes for selecting audio chunks from multiple sources.

Classes

- class **base_t**
  Base class for source selection.
- class **channel_t**
- class **channels_t**
- class **spectrum_t**
- class **waveform_t**

4.47.1 Detailed Description

Collection of classes for selecting audio chunks from multiple sources.

4.48 MHAOvlFilter Namespace Reference

Namespace for overlapping FFT based filter bank classes and functions.

Namespaces

- **barkscale**
- **FreqScaleFun**
  Transform functions from linear scale in Hz to new frequency scales.
- **ShapeFun**
  Shape functions for overlapping filters.

© 2005-2021 HörTech gGmbH, Oldenburg

Classes

- class **band_descriptor_t**
- class **fftfb_ac_info_t**
- class **fftfb_t**

  FFT based overlapping filter bank.
- class **fftfb_vars_t**

  Set of configuration variables for FFT-based overlapping filters.
- class **fscale_bw_t**
- class **fscale_t**
- class **fspacing_t**

  Class for frequency spacing, used by filterbank shape generator class.
- class **overlap_save_filterbank_analytic_t**
- class **overlap_save_filterbank_t**

  *A time-domain minimal phase filter bank with frequency shapes from **MHAOvlFilter::fftfb_t** (p. 1006).*
- class **scale_var_t**

Typedefs

- typedef **mha_real_t**() **scale_fun_t**(**mha_real_t**)

4.48.1 Detailed Description

Namespace for overlapping FFT based filter bank classes and functions.

4.48.2 Typedef Documentation

4.48.2.1 scale_fun_t `typedef mha_real_t() MHAOvlFilter::scale_fun_t( mha_real_t)`

4.49 MHAOvlFilter::barkscale Namespace Reference

Classes

- class **bark2hz_t**
- class **hz2bark_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Variables

- **mha_real_t vfreq** [**BARKSCALE_ENTRIES**]
- **mha_real_t vbark** [**BARKSCALE_ENTRIES**]

4.49.1 Variable Documentation

4.49.1.1 vfreq `mha_real_t MHAOvlFilter::barkscale::vfreq`

4.49.1.2 vbark `mha_real_t MHAOvlFilter::barkscale::vbark`

4.50 MHAOvlFilter::FreqScaleFun Namespace Reference

Transform functions from linear scale in Hz to new frequency scales.

Functions

- **mha_real_t hz2hz** (**mha_real_t** x)
  Dummy scale transformation Hz to Hz.
- **mha_real_t hz2khz** (**mha_real_t** x)
- **mha_real_t hz2octave** (**mha_real_t** x)
- **mha_real_t hz2third_octave** (**mha_real_t** x)
- **mha_real_t hz2bark** (**mha_real_t** x)
  Transformation to bark scale.
- **mha_real_t hz2bark_analytic** (**mha_real_t**)
- **mha_real_t hz2erb** (**mha_real_t**)
- **mha_real_t hz2erb_glasberg1990** (**mha_real_t**)
- **mha_real_t hz2log** (**mha_real_t** x)
  Third octave frequency scale.
- **mha_real_t inv_scale** (**mha_real_t**, **mha_real_t**(∗)(**mha_real_t**))

4.50.1 Detailed Description

Transform functions from linear scale in Hz to new frequency scales.

© 2005-2021 HörTech gGmbH, Oldenburg

4.50.2 Function Documentation

4.50.2.1 hz2hz()

```
mha_real_t MHAOvlFilter::FreqScaleFun::hz2hz (
    mha_real_t x )
```

Dummy scale transformation Hz to Hz.

This function implements a dummy scale transformation (linear frequency scale).

Parameters

x	Input frequency in Hz

Returns

Frequency in Hz

4.50.2.2 hz2khz()

```
mha_real_t MHAOvlFilter::FreqScaleFun::hz2khz (
    mha_real_t x )
```

4.50.2.3 hz2octave()

```
mha_real_t MHAOvlFilter::FreqScaleFun::hz2octave (
    mha_real_t x )
```

4.50.2.4 hz2third_octave()

```
mha_real_t MHAOvlFilter::FreqScaleFun::hz2third_octave
(
    mha_real_t x )
```

4.50.2.5 hz2bark()

```
mha_real_t MHAOvlFilter::FreqScaleFun::hz2bark (
    mha_real_t x )
```

Transformation to bark scale.

This function implements a critical band rate (bark) scale.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

x	Input frequency in Hz

Returns

Critical band rate in Bark

4.50.2.6 hz2bark_analytic()

```
mha_real_t MHAOvlFilter::FreqScaleFun::hz2bark_analytic (
    mha_real_t x )
```

4.50.2.7 hz2erb()

```
mha_real_t MHAOvlFilter::FreqScaleFun::hz2erb (
    mha_real_t x )
```

4.50.2.8 hz2erb_glasberg1990()

```
mha_real_t MHAOvlFilter::FreqScaleFun::hz2erb_↩
glasberg1990 (
    mha_real_t x )
```

4.50.2.9 hz2log()

```
mha_real_t MHAOvlFilter::FreqScaleFun::hz2log (
    mha_real_t x )
```

Third octave frequency scale.

This function implements a third octave scale. Frequencies below 16 Hz are mapped to 16 Hz.

Parameters

x	Frequency in Hz

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

Third octaves relative to 1000 Hz

4.50.2.10 inv_scale()

```
mha_real_t MHAOvlFilter::FreqScaleFun::inv_scale (
    mha_real_t y,
    mha_real_t(*)( mha_real_t) fun )
```

4.51 MHAOvlFilter::ShapeFun Namespace Reference

Shape functions for overlapping filters.

Functions

- **mha_real_t rect** (**mha_real_t** x)
  Filter shape function for rectangular filters.
- **mha_real_t linear** (**mha_real_t** x)
  Filter shape function for sawtooth filters.
- **mha_real_t hann** (**mha_real_t** x)
  Filter shape function for hanning shaped filters.
- **mha_real_t expflt** (**mha_real_t**)
- **mha_real_t gauss** (**mha_real_t**)

4.51.1 Detailed Description

Shape functions for overlapping filters.

4.51.2 Function Documentation

4.51.2.1 rect()

```
mha_real_t MHAOvlFilter::ShapeFun::rect (
    mha_real_t x )
```

Filter shape function for rectangular filters.

This function creates rectangular filter shapes. The edge is exactly half way between two center frequencies (on a given scale).

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

x	Input value in the range [-1,1].

Returns

Weigth function in the range [0,1]

4.51.2.2 linear()

```
mha_real_t MHAOvlFilter::ShapeFun::linear (
    mha_real_t x )
```

Filter shape function for sawtooth filters.

This function creates sawtooth filter shapes. They rise linearly form 0 to 1 in the interval from the lower neighbor center frequency to the band center frequency and from 1 to 0 in the interval from the band center frequency to the upper neighbour band center frequency. Linear means linear on a given frequency scale.

Parameters

x	Input value in the range [-1,1].

Returns

Weigth function in the range [0,1]

4.51.2.3 hann()

```
mha_real_t MHAOvlFilter::ShapeFun::hann (
    mha_real_t x )
```

Filter shape function for hanning shaped filters.

This function creates hanning window shaped filters.

Parameters

x	Input value in the range [-1,1].

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

Weigth function in the range [0,1]

4.51.2.4 expflt()

`mha_real_t MHAOvlFilter::ShapeFun::expflt ( mha_real_t x )`

4.51.2.5 gauss()

`mha_real_t MHAOvlFilter::ShapeFun::gauss ( mha_real_t x )`

4.52 MHAParser Namespace Reference

Name space for the openMHA-Parser configuration language.

Namespaces

- **StrCnv**
  String converter namespace.

Classes

- class **base_t**
  Base class for all parser items.
- class **bool_mon_t**
  Monitor with string value.
- class **bool_t**
  Variable with a boolean value ("yes"/"no")
- class **c_ifc_parser_t**
- class **commit_t**
  Parser variable with event-emission functionality.
- class **complex_mon_t**
  Monitor with complex value.
- class **complex_t**
  Variable with complex value.
- class **entry_t**
- class **expression_t**

© 2005-2021 HörTech gGmbH, Oldenburg

- class **float_mon_t**
  Monitor with float value.
- class **float_t**
  Variable with float value.
- class **int_mon_t**
  Monitor variable with int value.
- class **int_t**
  Variable with integer value.
- class **keyword_list_t**
  Keyword list class.
- class **kw_t**
  Variable with keyword list value.
- class **mcomplex_mon_t**
  Matrix of complex numbers monitor.
- class **mcomplex_t**
  Matrix variable with complex value.
- class **mfloat_mon_t**
  Matrix of floats monitor.
- class **mfloat_t**
  Matrix variable with float value.
- class **mhaconfig_mon_t**
- class **mhapluginloader_t**
  Class to create a plugin loader in a parser, including the load logic.
- class **mint_mon_t**
  Matrix of ints monitor.
- class **mint_t**
  Matrix variable with int value.
- class **monitor_t**
  Base class for monitors and variable nodes.
- class **parser_t**
  Parser node class.
- class **range_var_t**
  Base class for all variables with a numeric value range.
- class **string_mon_t**
  Monitor with string value.
- class **string_t**
  Variable with a string value.
- class **variable_t**
  Base class for variable nodes.
- class **vcomplex_mon_t**
  Monitor with vector of complex values.
- class **vcomplex_t**
  Vector variable with complex value.
- class **vfloat_mon_t**
  Vector of floats monitor.

© 2005-2021 HörTech gGmbH, Oldenburg

- class **vfloat_t**

  Vector variable with float value.
- class **vint_mon_t**

  Vector of ints monitor.
- class **vint_t**

  Variable with vector<int> value.
- class **vstring_mon_t**

  Vector of monitors with string value.
- class **vstring_t**

  Vector variable with string values.
- class **window_t**

  MHA configuration interface for a window function generator.

Typedefs

- typedef std::string(base_t::∗ **opact_t**) (**expression_t** &)
- typedef std::string(base_t::∗ **query_t**) (const std::string &)
- typedef std::map< std::string, **opact_t** > **opact_map_t**
- typedef std::map< std::string, **query_t** > **query_map_t**
- typedef std::list< **entry_t** > **entry_map_t**
- typedef int(∗ **c_parse_cmd_t**) (void ∗, const char ∗, char ∗, unsigned int)

Functions

- int **get_precision** ()
- std::string **commentate** (const std::string &s)
- void **trim** (std::string &s)
- std::string **cfg_dump** (**base_t** ∗, const std::string &)
- std::string **cfg_dump_short** (**base_t** ∗, const std::string &)
- std::string **all_dump** (**base_t** ∗, const std::string &)
- std::string **mon_dump** (**base_t** ∗, const std::string &)
- std::string **all_ids** (**base_t** ∗, const std::string &, const std::string &="")
- void **strreplace** (std::string &, const std::string &, const std::string &)

  string replace function
- void **envreplace** (std::string &s)

Variables

- const typedef char ∗(∗ **c_parse_err_t**)(void ∗, int)

© 2005-2021 HörTech gGmbH, Oldenburg

4.52.1 Detailed Description

Name space for the openMHA-Parser configuration language.

This namespace contains all classes which are needed for the implementation of the open↩ MHA configuration language. For details on the script language itself please see section **The openMHA configuration language** (p. 30).

4.52.2 List of valid MHAParser items

- **Sub-parser**: **parser_t** (p. 1102)
- **Variables**:
  Numeric variables: **int_t** (p. 1069), **vint_t** (p. 1132), **float_t** (p. 1063), **vfloat_t** (p. 1127), **mfloat_t** (p. 1085)
  Other variables: **string_t** (p. 1115), **vstring_t** (p. 1136), **kw_t** (p. 1075), **bool_t** (p. 1047)
- **Monitors**:
  Numeric monitors: **int_mon_t** (p. 1066), **vint_mon_t** (p. 1130), **float_mon_t** (p. 1061), **vfloat_mon_t** (p. 1125)
  mfloat_mon_t (p. 1083)
  mcomplex_mon_t (p. 1079)
  Other monitors: **bool_mon_t** (p. 1045), **string_mon_t** (p. 1113), **vstring_mon_↩ t** (p. 1134)

Members can be inserted into the configuration namespace by using MHAParser::insert_item() or the **insert_member()** (p. 1622) macro.

4.52.3 Typedef Documentation

4.52.3.1 opact_t `typedef std::string(base_t::* MHAParser::opact_t) ( expression_t &)`

4.52.3.2 query_t `typedef std::string(base_t::* MHAParser::query_t) (const std↩ ::string &)`

© 2005-2021 HörTech gGmbH, Oldenburg

4.52.3.3 opact_map_t `typedef std::map<std::string, opact_t> MHAParser::opact_map_t`

4.52.3.4 query_map_t `typedef std::map<std::string, query_t> MHAParser::query_map_t`

4.52.3.5 entry_map_t `typedef std::list< entry_t> MHAParser::entry_map_t`

4.52.3.6 c_parse_cmd_t `typedef int(* MHAParser::c_parse_cmd_t) (void *, const char *, char *, unsigned int)`

4.52.4 Function Documentation

4.52.4.1 get_precision() `int MHAParser::get_precision ( )`

4.52.4.2 commentate() `std::string MHAParser::commentate ( const std::string & s )`

4.52.4.3 trim() `void MHAParser::trim ( std::string & s )`

4.52.4.4 cfg_dump() `std::string MHAParser::cfg_dump ( base_t * p, const std::string & pref )`

© 2005-2021 HörTech gGmbH, Oldenburg

4.52.4.5 cfg_dump_short()

```
std::string MHAParser::cfg_dump_short (
    base_t * p,
    const std::string & pref )
```

4.52.4.6 all_dump()

```
std::string MHAParser::all_dump (
    base_t * p,
    const std::string & pref )
```

4.52.4.7 mon_dump()

```
std::string MHAParser::mon_dump (
    base_t * p,
    const std::string & pref )
```

4.52.4.8 all_ids()

```
std::string MHAParser::all_ids (
    base_t * p,
    const std::string & pref,
    const std::string & id = "" )
```

4.52.4.9 strreplace()

```
void MHAParser::strreplace (
    std::string & s,
    const std::string & arg,
    const std::string & rep )
```

string replace function

Parameters

s	target string
arg	search pattern
rep	replace pattern

4.52.4.10 envreplace()

```
void MHAParser::envreplace (
    std::string & s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

4.52.5 Variable Documentation

4.52.5.1 c_parse_err_t `const typedef char*(* MHAParser::c_parse_err_t) (void *, int)`

4.53 MHAParser::StrCnv Namespace Reference

String converter namespace.

Functions

- int **num_brackets** (const std::string &s)
  count number of brackets
- int **bracket_balance** (const std::string &s)
- void **str2val** (const std::string &, bool &)
  Convert from string.
- void **str2val** (const std::string &, float &)
  Convert from string.
- void **str2val** (const std::string &, **mha_complex_t** &)
  Convert from string.
- void **str2val** (const std::string &, int &)
  Convert from string.
- void **str2val** (const std::string &, **keyword_list_t** &)
  Convert from string.
- void **str2val** (const std::string &, std::string &)
  Convert from string.
- template<class arg_t >
  void **str2val** (const std::string &s, std::vector< arg_t > &val)
  Converter for vector types.
- template<> void **str2val**< **mha_real_t** > (const std::string &s, std::vector< **mha_real_t** > &v)
  Converter for vector<mha_real_t> with Matlab-style expansion.
- template<class arg_t >
  void **str2val** (const std::string &s, std::vector< std::vector< arg_t > > &val)
  Converter for matrix types.
- std::string **val2str** (const bool &)
  Convert to string.
- std::string **val2str** (const float &)
  Convert to string.
- std::string **val2str** (const **mha_complex_t** &)

© 2005-2021 HörTech gGmbH, Oldenburg

Convert to string.

- std::string **val2str** (const int &)

  Convert to string.
- std::string **val2str** (const **keyword_list_t** &)

  Convert to string.
- std::string **val2str** (const std::string &)

  Convert to string.
- std::string **val2str** (const std::vector< float > &)

  Convert to string.
- std::string **val2str** (const std::vector< **mha_complex_t** > &)

  Convert to string.
- std::string **val2str** (const std::vector< int > &)

  Convert to string.
- std::string **val2str** (const std::vector< std::vector< int > > &)

  Convert to string.
- std::string **val2str** (const std::vector< std::string > &)

  Convert to string.
- std::string **val2str** (const std::vector< std::vector< float > > &)

  Convert to string.
- std::string **val2str** (const std::vector< std::vector< **mha_complex_t** > > &)

  Convert to string.

4.53.1 Detailed Description

String converter namespace.

The functions defined in this namespace manage the conversions from C++ variables to strings and back. It was tried to keep a matlab compatible string format for vectors and vectors of vectors.

4.53.2 Function Documentation

4.53.2.1 num_brackets()

```
int MHAParser::StrCnv::num_brackets (
    const std::string & s )
```

count number of brackets

Return number of brackets according to layer depth (vector:2, matrix:4, etc)

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s	String

Returns

Number of brackets, or -1 for empty string, or -2 for invalid brackets

4.53.2.2 bracket_balance()

```
int MHAParser::StrCnv::bracket_balance (
    const std::string & s )
```

4.53.2.3 str2val() [1/8]

```
void MHAParser::StrCnv::str2val (
    const std::string & s,
    bool & v )
```

Convert from string.

4.53.2.4 str2val() [2/8]

```
void MHAParser::StrCnv::str2val (
    const std::string & s,
    float & v )
```

Convert from string.

4.53.2.5 str2val() [3/8]

```
void MHAParser::StrCnv::str2val (
    const std::string & s,
    mha_complex_t & v )
```

Convert from string.

© 2005-2021 HörTech gGmbH, Oldenburg

4.53.2.6 str2val() [4/8]

```
void MHAParser::StrCnv::str2val (
    const std::string & s,
    int & v )
```

Convert from string.

4.53.2.7 str2val() [5/8]

```
void MHAParser::StrCnv::str2val (
    const std::string & s,
    MHAParser::keyword_list_t & v )
```

Convert from string.

4.53.2.8 str2val() [6/8]

```
void MHAParser::StrCnv::str2val (
    const std::string & s,
    std::string & v )
```

Convert from string.

4.53.2.9 str2val() [7/8]

```
template<class arg_t >
void MHAParser::StrCnv::str2val (
    const std::string & s,
    std::vector< arg_t > & val )
```

Converter for vector types.

4.53.2.10 str2val< mha_real_t >()

```
template<>
void MHAParser::StrCnv::str2val< mha_real_t > (
    const std::string & s,
    std::vector< mha_real_t > & v )
```

Converter for vector<mha_real_t> with Matlab-style expansion.

© 2005-2021 HörTech gGmbH, Oldenburg

4.53.2.11 str2val() [8/8] template<class arg_t >

```
void MHAParser::StrCnv::str2val (
            const std::string & s,
            std::vector< std::vector< arg_t > > & val )
```

Converter for matrix types.

4.53.2.12 val2str() [1/13] std::string MHAParser::StrCnv::val2str (

```
            const bool & v )
```

Convert to string.

4.53.2.13 val2str() [2/13] std::string MHAParser::StrCnv::val2str (

```
            const float & v )
```

Convert to string.

4.53.2.14 val2str() [3/13] std::string MHAParser::StrCnv::val2str (

```
            const mha_complex_t & v )
```

Convert to string.

4.53.2.15 val2str() [4/13] std::string MHAParser::StrCnv::val2str (

```
            const int & v )
```

Convert to string.

4.53.2.16 val2str() [5/13] std::string MHAParser::StrCnv::val2str (

```
            const keyword_list_t & v )
```

Convert to string.

© 2005-2021 HörTech gGmbH, Oldenburg

4.53.2.17 val2str() [6/13] std::string MHAParser::StrCnv::val2str (const std::string & *v*)

Convert to string.

4.53.2.18 val2str() [7/13] std::string MHAParser::StrCnv::val2str (const std::vector< float > & *v*)

Convert to string.

4.53.2.19 val2str() [8/13] std::string MHAParser::StrCnv::val2str (const std::vector< **mha_complex_t** > & *v*)

Convert to string.

4.53.2.20 val2str() [9/13] std::string MHAParser::StrCnv::val2str (const std::vector< int > & *v*)

Convert to string.

4.53.2.21 val2str() [10/13] std::string MHAParser::StrCnv::val2str (const std::vector< std::vector< int > > & *v*)

Convert to string.

4.53.2.22 val2str() [11/13] std::string MHAParser::StrCnv::val2str (const std::vector< std::string > & *v*)

Convert to string.

© 2005-2021 HörTech gGmbH, Oldenburg

4.53.2.23 val2str() [12/13] std::string MHAParser::StrCnv::val2str (const std::vector< std::vector< float > > & v)

Convert to string.

4.53.2.24 val2str() [13/13] std::string MHAParser::StrCnv::val2str (const std::vector< std::vector< mha_complex_t > > & v)

Convert to string.

4.54 MHAPlugin Namespace Reference

Namespace for openMHA plugin class templates and thread-safe runtime configurations.

Classes

- class **cfg_node_t**

  A node class for storing MHA plugin runtime configurations as a singly linked list, where the nodes store besides the usual "next" and "data" pointers also a flag that indicates whether this node can be deleted.
- class **config_t**

  Template class for thread safe configuration.
- class **plugin_t**

  The template class for C++ openMHA plugins.

4.54.1 Detailed Description

Namespace for openMHA plugin class templates and thread-safe runtime configurations.

4.55 MHAPlugin_Resampling Namespace Reference

Classes

- class **resampling_if_t**
- class **resampling_t**

© 2005-2021 HörTech gGmbH, Oldenburg

4.56 MHAPlugin_Split Namespace Reference

Classes

- class **domain_handler_t**

  Handles domain-specific partial input and output signal.
- class **dummy_threads_t**

  Dummy specification of a thread platform: This class implements everything in a single thread.
- class **posix_threads_t**

  Posix threads specification of thread platform.
- class **split_t**

  Implements split plugin.
- class **splitted_part_t**

  *The **splitted_part_t** (p. 1183) instance manages the plugin that performs processing on the reduced set of channels.*
- class **thread_platform_t**

  Basic interface for encapsulating thread creation, thread priority setting, and synchronization on any threading platform (i.e., pthreads or win32threads).
- class **uni_processor_t**

  An interface to a class that sports a process method with no parameters and no return value.

Enumerations

- enum { **INVALID_THREAD_PRIORITY** = 999999999 }

  Invalid thread priority.

4.56.1 Detailed Description

A namespace for the split plugin. Helps testability and documentation.

4.56.2 Enumeration Type Documentation

4.56.2.1 anonymous enum `anonymous enum`

Invalid thread priority.

© 2005-2021 HörTech gGmbH, Oldenburg

Enumerator

INVALID_THREAD_PRIORITY	

4.57 MHASignal Namespace Reference

Namespace for audio signal handling and processing classes.

Classes

- class **async_rmslevel_t**
  Class for asynchronous level metering.
- class **delay_spec_t**
- class **delay_t**
  Class to realize a simple delay of waveform streams.
- class **delay_wave_t**
  Delayline containing wave fragments.
- class **doublebuffer_t**
  Double-buffering class.
- class **fft_t**
- class **hilbert_fftw_t**
- class **hilbert_t**
  Hilbert transformation of a waveform segment.
- class **loop_wavefragment_t**
  Copy a fixed waveform fragment to a series of waveform fragments of other size.
- class **matrix_t**
  n-dimensional matrix with real or complex floating point values.
- class **minphase_t**
  Minimal phase function.
- class **quantizer_t**
  Simple simulation of fixpoint quantization.
- class **ringbuffer_t**
  A ringbuffer class for time domain audio signal, which makes no assumptions with respect to fragment size.
- class **schroeder_t**
  Schroeder tone complex class.
- class **spectrum_t**
  *a signal processing class for spectral data (based on **mha_spec_t** (p. 794))*
- class **stat_t**
- class **subsample_delay_t**
  implements subsample delay in spectral domain.
- class **uint_vector_t**
  Vector of unsigned values, used for size and index description of n-dimensional matrixes.
- class **waveform_t**
  *signal processing class for waveform data (based on **mha_wave_t** (p. 840))*

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- void **for_each** (**mha_wave_t** ∗s, **mha_real_t**(∗fun)(**mha_real_t**))
  *Apply a function to each element of a **mha_wave_t*** (p. 840).
- **mha_real_t lin2db** (**mha_real_t** x, **mha_real_t** eps)
  Conversion from linear scale to dB (no SPL reference)
- **mha_real_t lin2db** (**mha_real_t** x)
  Conversion from linear scale to dB (no SPL reference)
- **mha_real_t db2lin** (**mha_real_t** x)
  Conversion from dB scale to linear (no SPL reference)
- **mha_real_t sq2db** (**mha_real_t** x, **mha_real_t** eps=0.0f)
  conversion from squared values to dB (no SPL reference)
- **mha_real_t db2sq** (**mha_real_t** x)
  conversion from dB to squared values (no SPL reference)
- **mha_real_t pa2dbspl** (**mha_real_t** x, **mha_real_t** eps)
  Conversion from linear Pascal scale to dB SPL.
- **mha_real_t pa2dbspl** (**mha_real_t** x)
  Conversion from linear Pascal scale to dB SPL.
- **mha_real_t dbspl2pa** (**mha_real_t** x)
  Conversion from dB SPL to linear Pascal scale.
- **mha_real_t pa22dbspl** (**mha_real_t** x, **mha_real_t** eps=0.0f)
  Conversion from squared Pascal scale to dB SPL.
- **mha_real_t dbspl2pa2** (**mha_real_t** x)
  conversion from dB SPL to squared Pascal scale
- **mha_real_t smp2sec** (**mha_real_t** n, **mha_real_t** srate)
  conversion from samples to seconds
- **mha_real_t sec2smp** (**mha_real_t** sec, **mha_real_t** srate)
  conversion from seconds to samples
- **mha_real_t bin2freq** (**mha_real_t** bin, unsigned fftlen, **mha_real_t** srate)
  conversion from fft bin index to frequency
- **mha_real_t freq2bin** (**mha_real_t** freq, unsigned fftlen, **mha_real_t** srate)
  conversion from frequency to fft bin index
- **mha_real_t smp2rad** (**mha_real_t** samples, unsigned bin, unsigned fftlen)
  conversion from delay in samples to phase shift
- **mha_real_t rad2smp** (**mha_real_t** phase_shift, unsigned bin, unsigned fftlen)
  conversion from phase shift to delay in samples
- template<class elem_type >
  std::vector< elem_type > **dupvec** (std::vector< elem_type > vec, unsigned n)
  Duplicate last vector element to match desired size.
- template<class elem_type >
  std::vector< elem_type > **dupvec_chk** (std::vector< elem_type > vec, unsigned n)
  Duplicate last vector element to match desired size, check for dimension.
- void **copy_channel** (**mha_spec_t** &self, const **mha_spec_t** &src, unsigned sch, unsigned dch)
  Copy one channel of a source signal.

© 2005-2021 HörTech gGmbH, Oldenburg

- void **copy_channel** (**mha_wave_t** &self, const **mha_wave_t** &src, unsigned src_↩channel, unsigned dest_channel)

  Copy one channel of a source signal.
- **mha_real_t rmslevel** (const **mha_spec_t** &s, unsigned int channel, unsigned int fftlen)

  Return RMS level of a spectrum channel.
- **mha_real_t colored_intensity** (const **mha_spec_t** &s, unsigned int channel, unsigned int fftlen, **mha_real_t** ∗sqfreq_response=nullptr)

  Colored spectrum intensity.
- **mha_real_t maxabs** (const **mha_spec_t** &s, unsigned int channel)

  Find maximal absolute value.
- **mha_real_t rmslevel** (const **mha_wave_t** &s, unsigned int channel)

  Return RMS level of a waveform channel.
- **mha_real_t maxabs** (const **mha_wave_t** &s, unsigned int channel)

  Find maximal absolute value.
- **mha_real_t maxabs** (const **mha_wave_t** &s)

  Find maximal absolute value.
- **mha_real_t max** (const **mha_wave_t** &s)

  Find maximal value.
- **mha_real_t min** (const **mha_wave_t** &s)

  Find minimal value.
- **mha_real_t sumsqr_channel** (const **mha_wave_t** &s, unsigned int channel)

  Calculate sum of squared values in one channel.
- **mha_real_t sumsqr_frame** (const **mha_wave_t** &s, unsigned int frame)

  Calculate sum over all channels of squared values.
- void **scale** (**mha_spec_t** ∗dest, const **mha_wave_t** ∗src)
- void **limit** (**mha_wave_t** &s, const **mha_real_t** & **min**, const **mha_real_t** & **max**)

  Limit the singal in the waveform buffer to the range [min, max].
- template<class elem_type >
  elem_type **kth_smallest** (elem_type array[ ], unsigned n, unsigned k)

  Fast search for the kth smallest element of an array.
- template<class elem_type >
  elem_type **median** (elem_type array[ ], unsigned n)

  Fast median search.
- template<class elem_type >
  elem_type **mean** (const std::vector< elem_type > &data, elem_type start_val)

  Calculate average of elements in a vector.
- template<class elem_type >
  std::vector< elem_type > **quantile** (std::vector< elem_type > data, const std::vector< elem_type > &p)

  Calculate quantile of elements in a vector.
- void **saveas_mat4** (const **mha_spec_t** &data, const std::string &varname, FILE ∗fh)

  Save a openMHA spectrum as a variable in a Matlab4 file.
- void **saveas_mat4** (const **mha_wave_t** &data, const std::string &varname, FILE ∗fh)

  Save a openMHA waveform as a variable in a Matlab4 file.
- void **saveas_mat4** (const std::vector< **mha_real_t** > &data, const std::string &varname, FILE ∗fh)

© 2005-2021 HörTech gGmbH, Oldenburg

Save a float vector as a variable in a Matlab4 file.

- void **copy_permuted** (**mha_wave_t** ∗dest, const **mha_wave_t** ∗src)

  Copy contents of a waveform to a permuted waveform.

Variables

- unsigned long int **signal_counter** = 0

  Signal counter to produce signal ID strings.

4.57.1 Detailed Description

Namespace for audio signal handling and processing classes.

4.57.2 Function Documentation

4.57.2.1 for_each()

```
void MHASignal::for_each (
    mha_wave_t * s,
    mha_real_t(*)( mha_real_t) fun ) [inline]
```

Apply a function to each element of a **mha_wave_t** (p. 840).

Parameters

s	Pointer to a **mha_wave_t** (p. 840) structure
fun	Function to be applied (one argument)

4.57.2.2 lin2db() [1/2]

```
mha_real_t MHASignal::lin2db (
    mha_real_t x,
    mha_real_t eps ) [inline]
```

Conversion from linear scale to dB (no SPL reference)

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

x	Linear input
eps	minimum linear value (if x < eps --> convert eps instead), eps < 0 not allowed

Returns

NaN if x < 0 (log not defined for negative)

Exceptions

MHA_Error *(p. 764)*	if eps < 0

4.57.2.3 lin2db() [2/2]

```
mha_real_t MHASignal::lin2db ( mha_real_t x ) [inline]
```

Conversion from linear scale to dB (no SPL reference)

Parameters

x	Linear input.

Returns

NaN if x < 0 (log not defined for negative)

4.57.2.4 db2lin()

```
mha_real_t MHASignal::db2lin ( mha_real_t x ) [inline]
```

Conversion from dB scale to linear (no SPL reference)

Parameters

x	dB input.

© 2005-2021 HörTech gGmbH, Oldenburg

4.57.2.5 sq2db()

```
mha_real_t MHASignal::sq2db (
    mha_real_t x,
    mha_real_t eps = 0.0f ) [inline]
```

conversion from squared values to dB (no SPL reference)

Parameters

x	squared value input
eps	minimum squared value (if x < eps --> convert eps instead), eps < 0 not allowed

Returns

NaN if x < 0 (log not defined for negative)

Exceptions

MHA_Error *(p. 764)*	if eps < 0

4.57.2.6 db2sq()

```
mha_real_t MHASignal::db2sq (
    mha_real_t x ) [inline]
```

conversion from dB to squared values (no SPL reference)

Parameters

x	dB input

4.57.2.7 pa2dbspl() [1/2]

```
mha_real_t MHASignal::pa2dbspl (
    mha_real_t x,
    mha_real_t eps ) [inline]
```

Conversion from linear Pascal scale to dB SPL.

Parameters

x	Linear input
eps	minimum pascal value (if x < eps --> convert eps instead),

© 2005-2021 HörTech gGmbH, Oldenburg

Precondition

eps >= 0

Returns

NaN if x < 0 (logarithm not defined for negative numbers)

Exceptions

MHA_Error *(p. 764)*	if eps < 0

4.57.2.8 pa2dbspl() [2/2]

```
mha_real_t MHASignal::pa2dbspl (
    mha_real_t x ) [inline]
```

Conversion from linear Pascal scale to dB SPL.

Parameters

x	Linear input

Returns

NaN if x < 0 (log not defined for negative)

4.57.2.9 dbspl2pa()

```
mha_real_t MHASignal::dbspl2pa (
    mha_real_t x ) [inline]
```

Conversion from dB SPL to linear Pascal scale.

Parameters

x	Linear input.

4.57.2.10 pa22dbspl()

```
mha_real_t MHASignal::pa22dbspl (
```

© 2005-2021 HörTech gGmbH, Oldenburg

```
mha_real_t x,
mha_real_t eps = 0.0f )  [inline]
```

Conversion from squared Pascal scale to dB SPL.

Parameters

x	squared pascal input
eps	minimum squared-pascal value (if x < eps --> convert eps instead), eps < 0 not allowed

Returns

NaN if x < 0 (log not defined for negative)

Exceptions

MHA_Error *(p. 764)*	if eps < 0

4.57.2.11 dbspl2pa2()

```
mha_real_t MHASignal::dbspl2pa2 (
mha_real_t x )  [inline]
```

conversion from dB SPL to squared Pascal scale

Parameters

x	dB SPL input

4.57.2.12 smp2sec()

```
mha_real_t MHASignal::smp2sec (
mha_real_t n,
mha_real_t srate )  [inline]
```

conversion from samples to seconds

Parameters

n	number of samples
srate	sampling rate / Hz

© 2005-2021 HörTech gGmbH, Oldenburg

4.57.2.13 sec2smp()

```
mha_real_t MHASignal::sec2smp (
    mha_real_t sec,
    mha_real_t srate ) [inline]
```

conversion from seconds to samples

Parameters

sec	time in seconds
srate	sampling rate / Hz

Returns

number of samples, generally has non-zero fractional part

4.57.2.14 scale()

```
void MHASignal::scale (
    mha_spec_t * dest,
    const mha_wave_t * src )
```

4.57.2.15 limit()

```
void MHASignal::limit (
    mha_wave_t & s,
    const mha_real_t & min,
    const mha_real_t & max )
```

Limit the singal in the waveform buffer to the range [min, max].

Parameters

s	The signal to limit. The signal in this wave buffer is modified.
min	lower limit
max	upper limit

4.57.2.16 kth_smallest()

```
template<class elem_type >
elem_type MHASignal::kth_smallest (
```

© 2005-2021 HörTech gGmbH, Oldenburg

```
elem_type array[],
unsigned n,
unsigned k )
```

Fast search for the kth smallest element of an array.

The order of elements is altered, but not completely sorted. Using the algorithm from N. Wirth, published in "Algorithms + data structures = programs", Prentice-Hall, 1976

Parameters

array	Element array

Postcondition

The order of elements in the array is altered. array[k] then holds the result.

Parameters

n	number of elements in array

Precondition

n >= 1

Parameters

k	The k'th smalles element is returned: k = 0 returns the minimum, k = (n-1)/2 returns the median, k=(n-1) returns the maximum

Precondition

k < n

Returns

The kth smallest array element

© 2005-2021 HörTech gGmbH, Oldenburg

4.57.2.17 median()

```
template<class elem_type >
elem_type MHASignal::median (
            elem_type array[],
            unsigned n )  [inline]
```

Fast median search.

The order of elements is altered, but not completely sorted.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

array	Element array

Postcondition

The order of elements in the array is altered. array[(n-1)/2] then holds the median.

Parameters

n	number of elements in array

Precondition

n >= 1

Returns

The median of the array elements

4.57.2.18 mean()

```
template<class elem_type >
elem_type MHASignal::mean (
            const std::vector< elem_type > & data,
            elem_type start_val ) [inline]
```

Calculate average of elements in a vector.

Parameters

data	Input vector
start_val	Value for initialization of the return value before sum.

Returns

The average of the vector elements

© 2005-2021 HörTech gGmbH, Oldenburg

4.57.2.19 quantile()

```
template<class elem_type >
std::vector<elem_type> MHASignal::quantile (
            std::vector< elem_type > data,
            const std::vector< elem_type > & p ) [inline]
```

Calculate quantile of elements in a vector.

Parameters

data	Input vector
p	Vector of probability values.

Returns

Vector of quantiles of input data, one entry for each probability value.

4.57.2.20 saveas_mat4() [1/3]

```
void MHASignal::saveas_mat4 (
            const mha_spec_t & data,
            const std::string & varname,
            FILE * fh )
```

Save a openMHA spectrum as a variable in a Matlab4 file.

Parameters

data	openMHA spectrum to be saved.
varname	Matlab variable name (Matlab4 limitations on maximal length are not checked).
fh	File handle to Matlab4 file.

4.57.2.21 saveas_mat4() [2/3]

```
void MHASignal::saveas_mat4 (
            const mha_wave_t & data,
            const std::string & varname,
            FILE * fh )
```

Save a openMHA waveform as a variable in a Matlab4 file.

Parameters

data	openMHA waveform to be saved.
varname	Matlab variable name (Matlab4 limitations on maximal length are not checked).
fh	File handle to Matlab4 file.

© 2005-2021 HörTech gGmbH, Oldenburg

4.57.2.22 saveas_mat4() [3/3]

```
void MHASignal::saveas_mat4 (
    const std::vector< mha_real_t > & data,
    const std::string & varname,
    FILE * fh )
```

Save a float vector as a variable in a Matlab4 file.

Parameters

data	Float vector to be saved.
varname	Matlab variable name (Matlab4 limitations on maximal length are not checked).
fh	File handle to Matlab4 file.

4.57.2.23 copy_permuted()

```
void MHASignal::copy_permuted (
    mha_wave_t * dest,
    const mha_wave_t * src )
```

Copy contents of a waveform to a permuted waveform.

Parameters

dest	Destination waveform
src	Source waveform

The total size of src and dest must be the same, num_frames and num_channels must be exchanged in dest.

4.57.3 Variable Documentation

4.57.3.1 signal_counter

```
unsigned long int MHASignal::signal_counter = 0
```

Signal counter to produce signal ID strings.

© 2005-2021 HörTech gGmbH, Oldenburg

4.58 MHASndFile Namespace Reference

Classes

- class **sf_t**
- class **sf_wave_t**

4.59 MHATableLookup Namespace Reference

Namespace for table lookup classes.

Classes

- class **linear_table_t**

  Class for interpolation with equidistant x values.
- class **table_t**
- class **xy_table_t**

  Class for interpolation with non-equidistant x values.

4.59.1 Detailed Description

Namespace for table lookup classes.

4.60 MHAUtils Namespace Reference

Functions

- bool **is_multiple_of** (const unsigned big, const unsigned small)
- bool **is_power_of_two** (const unsigned n)
- bool **is_multiple_of_by_power_of_two** (const unsigned big, const unsigned small)
- std::string **strip** (const std::string &line)
- std::string **remove** (const std::string &str_, char c)
- bool **is_denormal** (**mha_real_t** x)

  Get the normal-ness of a mha_real_t.
- bool **is_denormal** (const **mha_complex_t** &x)

  Get the normal-ness of a complex number.
- bool **is_denormal** (const std::complex< **mha_real_t** > &x)

  Get the normal-ness of a complex number.
- **mha_real_t** **spl2hl** (**mha_real_t** f)

  Get the offset of between dB(SPL) and dB(HL) for a given frequency according to ISO 389-7↩ :2005 (freefield); e.g.

© 2005-2021 HörTech gGmbH, Oldenburg

4.60.1 Function Documentation

4.60.1.1 is_multiple_of()

```
bool MHAUtils::is_multiple_of (
    const unsigned big,
    const unsigned small ) [inline]
```

4.60.1.2 is_power_of_two()

```
bool MHAUtils::is_power_of_two (
    const unsigned n ) [inline]
```

4.60.1.3 is_multiple_of_by_power_of_two()

```
bool MHAUtils::is_multiple_of_by_power↩
_of_two (
    const unsigned big,
    const unsigned small ) [inline]
```

4.60.1.4 strip()

```
std::string MHAUtils::strip (
    const std::string & line ) [inline]
```

4.60.1.5 remove()

```
std::string MHAUtils::remove (
    const std::string & str_,
    char c ) [inline]
```

4.60.1.6 is_denormal() [1/3]

```
bool MHAUtils::is_denormal (
    mha_real_t x ) [inline]
```

Get the normal-ness of a mha_real_t.

Returns true iff x is not equal to zero and the absolute value of x is smaller than the minimum positive normalized value of mha_real_t.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

x	A mha_real_t floating point number

Returns

True if x is denormal, false otherwise

4.60.1.7 is_denormal() [2/3] `bool MHAUtils::is_denormal ( const mha_complex_t & x ) [inline]`

Get the normal-ness of a complex number.

Overload for **mha_complex_t** (p. 745). Returns true iff one or both of real and imaginary part are denormal

Parameters

x	[in] A **mha_complex_t** (p. 745) number

Returns

True if at least one component of x is denormal, false otherwise

4.60.1.8 is_denormal() [3/3] `bool MHAUtils::is_denormal ( const std::complex< mha_real_t > & x ) [inline]`

Get the normal-ness of a complex number.

Overload for std::complex Returns true iff one or both of real and imaginary part are denormal.

Parameters

x	[in] A **mha_complex_t** (p. 745) number

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

True if at least one component of x is denormal, false otherwise

4.60.1.9 spl2hl() `mha_real_t MHAUtils::spl2hl ( mha_real_t f )`

Get the offset of between dB(SPL) and dB(HL) for a given frequency according to ISO 389-7↩ :2005 (freefield); e.g.

an intensity of 22.1 dB(SPL) at 125 Hz is equivalent to 0 dB(HL), so spl2hl(125)=-22.1. Interpolation between mesh points is linear. The correction values for frequencies above 16 kHz are extrapolated."

Parameters

in	*f*	The frequency in Hz for which the offset shall be returned

Returns

The offet between dB(SPL) and dB(HL) at frequency f

Exceptions

MHA_Error *(p. 764)*	if f<0

4.61 MHAWindow Namespace Reference

Collection of Window types.

Classes

- class **bartlett_t**
  Bartlett window.
- class **base_t**
  Common base for window types.
- class **blackman_t**
  Blackman window.
- class **fun_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Generic window based on a generator function.

- class **hamming_t**

  Hamming window.
- class **hanning_t**

  von-Hann window
- class **rect_t**

  Rectangular window.
- class **user_t**

  User defined window.

Functions

- float **rect** (float)

  Rectangular window function.
- float **bartlett** (float)

  Bartlett window function.
- float **hanning** (float)

  Hanning window function.
- float **hamming** (float)

  Hamming window function.
- float **blackman** (float)

  Blackman window function.

4.61.1 Detailed Description

Collection of Window types.

4.61.2 Function Documentation

4.61.2.1 rect() `float MHAWindow::rect ( float x )`

Rectangular window function.

© 2005-2021 HörTech gGmbH, Oldenburg

4.61.2.2 bartlett() `float MHAWindow::bartlett ( float x )`

Bartlett window function.

4.61.2.3 hanning() `float MHAWindow::hanning ( float x )`

Hanning window function.

4.61.2.4 hamming() `float MHAWindow::hamming ( float x )`

Hamming window function.

4.61.2.5 blackman() `float MHAWindow::blackman ( float x )`

Blackman window function.

4.62 multibandcompressor Namespace Reference

Classes

- class **fftfb_plug_t**
- class **interface_t**
- class **plugin_signals_t**

4.63 noise_psd_estimator Namespace Reference

Classes

- class **noise_psd_estimator_if_t**
- class **noise_psd_estimator_t**

© 2005-2021 HörTech gGmbH, Oldenburg

4.64 overlapadd Namespace Reference

Classes

- class **overlapadd_if_t**
- class **overlapadd_t**

4.65 plingploing Namespace Reference

All classes for the plingploing music generator live in this namespace.

Classes

- class **if_t**
  Plugin class of the plingploing music generator.
- class **plingploing_t**
  Run-time configuration of the plingploing music generator.

Functions

- double **drand** (double a, double b)

4.65.1 Detailed Description

All classes for the plingploing music generator live in this namespace.

4.65.2 Function Documentation

4.65.2.1 drand()

```
double plingploing::drand (
    double a,
    double b )
```

© 2005-2021 HörTech gGmbH, Oldenburg

4.66 PluginLoader Namespace Reference

Classes

- class **config_file_splitter_t**
- class **fourway_processor_t**

  This abstract class defines the interface for classes that implement all types of signal domain processing supported by the MHA: wave2wave, spec2spec, wave2spec, and spec2wave.
- class **mhapluginloader_t**

Functions

- const char * **mhastrdomain** (**mha_domain_t**)
- void **mhaconfig_compare** (const **mhaconfig_t** &req, const **mhaconfig_t** &avail, const std::string &pref="")

  *Compare two **mhaconfig_t** (p. 851) structures, and report differences as an error.*

4.66.1 Function Documentation

4.66.1.1 mhastrdomain()

```
const char * PluginLoader::mhastrdomain (
    mha_domain_t d )
```

4.66.1.2 mhaconfig_compare()

```
void PluginLoader::mhaconfig_compare (
    const mhaconfig_t & req,
    const mhaconfig_t & avail,
    const std::string & pref = "" )
```

Compare two **mhaconfig_t** (p. 851) structures, and report differences as an error.

Parameters

req	Expected **mhaconfig_t** (p. 851) structure
avail	Available **mhaconfig_t** (p. 851) structure
pref	Prefix for error messages

© 2005-2021 HörTech gGmbH, Oldenburg

4.67 plugins Namespace Reference

Namespaces

- **hoertech**

4.68 plugins::hoertech Namespace Reference

Namespaces

- **acrec**

4.69 plugins::hoertech::acrec Namespace Reference

Classes

- class **acrec_t**
  Plugin interface class of plugin acrec.
- class **acwriter_t**
  acwriter_t *(p. 1382) decouples signal processing from writing to disk.*

Functions

- std::string **to_iso8601** (time_t tm)

4.69.1 Function Documentation

4.69.1.1 to_iso8601()

```
std::string plugins::hoertech::acrec::to_iso8601 (
    time_t tm )
```

4.70 rmslevel Namespace Reference

Classes

- class **mon_t**
- class **rmslevel_if_t**
  Interface class of the rmslevel plugin.
- class **rmslevel_t**
  Run-time configuration class of the rmslevel plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

Enumerations

- enum **UNIT** { **UNIT::SPL** =0, **UNIT::HL** =1 }

4.70.1 Enumeration Type Documentation

4.70.1.1 UNIT `enum rmslevel::UNIT [strong]`

Enumerator

SPL	
HL	

4.71 rohBeam Namespace Reference

Classes

- struct **configOptions**
- class **rohBeam**
- class **rohConfig**

Functions

- double **j0** (double x)

  Cylindrical bessel function of the first kind of order 0.

Variables

- auto **scalarify** =[ ](auto t){return t(0);}
- constexpr float **CONST_C** = 343.0115f
- constexpr int **refL** = 0
- constexpr int **refR** = 3

4.71.1 Function Documentation

4.71.1.1 j0() `double rohBeam::j0 ( double x )`

Cylindrical bessel function of the first kind of order 0.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

x	the argument of the function

Returns

j0(x)

4.71.2 Variable Documentation

4.71.2.1 scalarify `auto rohBeam::scalarify =[](auto t){return t(0);}`

4.71.2.2 CONST_C `constexpr float rohBeam::CONST_C = 343.0115f [constexpr]`

4.71.2.3 refL `constexpr int rohBeam::refL = 0 [constexpr]`

4.71.2.4 refR `constexpr int rohBeam::refR = 3 [constexpr]`

4.72 route Namespace Reference

Classes

- class **interface_t**
- class **process_t**

4.73 shadowfilter_begin Namespace Reference

Classes

- class **cfg_t**
- class **shadowfilter_begin_t**

© 2005-2021 HörTech gGmbH, Oldenburg

4.74 shadowfilter_end Namespace Reference

Classes

- class **cfg_t**
- class **shadowfilter_end_t**

4.75 smooth_cepstrum Namespace Reference

Classes

- class **smooth_cepstrum_if_t**
- class **smooth_cepstrum_t**
- class **smooth_params**

4.76 smoothgains_bridge Namespace Reference

Classes

- class **overlapadd_if_t**
- class **smoothspec_wrap_t**

4.77 testplugin Namespace Reference

Classes

- class **ac_parser_t**
- class **config_parser_t**
- class **if_t**
- class **signal_parser_t**

4.78 windnoise Namespace Reference

namespace for plugin windnoise which detects and cancels wind noise

Classes

- class **cfg_t**
  Runtime config class for windnoise plugin.
- class **if_t**
  interface class for windnoise plugin

© 2005-2021 HörTech gGmbH, Oldenburg

4.78.1 Detailed Description

namespace for plugin windnoise which detects and cancels wind noise

5 Class Documentation

5.1 ac2lsl::ac2lsl_t Class Reference

Plugin class of **ac2lsl** (p. 78).

Inheritance diagram for ac2lsl::ac2lsl_t:

Public Member Functions

- **ac2lsl_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)

  *Prepare constructs the vector of bridge variables and locks the configuration, then calls **update()*** (p. 165).
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s)

  Processing fct for waveforms.
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗s)

  Processing fct for spectra.
- void **process** ()

  Process function.
- void **release** ()

  Release fct.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- std::vector< std::string > **get_all_names_from_ac_space** (const **algo_comm_t** & **ac**) const

  Retrieves all variable names from the AC space.
- void **update** ()

  Construct new runtime configuration.

Private Attributes

- **MHAParser::vstring_t vars**
- **MHAParser::string_t source_id**
- **MHAParser::bool_t rt_strict**
- **MHAParser::bool_t activate**
- **MHAParser::int_t skip**
- **MHAEvents::patchbay_t< ac2lsl_t > patchbay**
- bool **is_first_run**

Additional Inherited Members

5.1.1 Detailed Description

Plugin class of **ac2lsl** (p. 78).

5.1.2 Constructor & Destructor Documentation

5.1.2.1 ac2lsl_t()

```
ac2lsl::ac2lsl_t::ac2lsl_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.1.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.1.3.1 prepare()

```
void ac2lsl::ac2lsl_t::prepare (
    mhaconfig_t & ) [virtual]
```

Prepare constructs the vector of bridge variables and locks the configuration, then calls **update()** (p. 165).

Implements **MHAPlugin::plugin_t< cfg_t >** (p. 1153).

5.1.3.2 process() [1/3]

```
mha_wave_t* ac2lsl::ac2lsl_t::process (
    mha_wave_t * s ) [inline]
```

Processing fct for waveforms.

Calls **process(void)** (p. 164).

5.1.3.3 process() [2/3]

```
mha_spec_t* ac2lsl::ac2lsl_t::process (
    mha_spec_t * s ) [inline]
```

Processing fct for spectra.

Calls **process(void)** (p. 164).

5.1.3.4 process() [3/3]

```
void ac2lsl::ac2lsl_t::process ( )
```

Process function.

Checks once if the plugin is run in a real-time thread and throws if rt_strict is true, then forwards to **cfg_t::process()** (p. 168).

5.1.3.5 release()

```
void ac2lsl::ac2lsl_t::release ( ) [virtual]
```

Release fct.

Unlocks variable name list

Reimplemented from **MHAPlugin::plugin_t< cfg_t >** (p. 1154).

5.1.3.6 get_all_names_from_ac_space()

```
std::vector< std::string > ac2lsl::ac2lsl↩
_t::get_all_names_from_ac_space (
    const algo_comm_t & ac ) const [private]
```

Retrieves all variable names from the AC space.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

ac	AC space

Returns

Vector of variable names

5.1.3.7 update() `void ac2lsl::ac2lsl_t::update ( ) [private]`

Construct new runtime configuration.

5.1.4 Member Data Documentation

5.1.4.1 vars `MHAParser::vstring_t ac2lsl::ac2lsl_t::vars [private]`

5.1.4.2 source_id `MHAParser::string_t ac2lsl::ac2lsl_t::source_id [private]`

5.1.4.3 rt_strict `MHAParser::bool_t ac2lsl::ac2lsl_t::rt_strict [private]`

5.1.4.4 activate `MHAParser::bool_t ac2lsl::ac2lsl_t::activate [private]`

5.1.4.5 skip `MHAParser::int_t ac2lsl::ac2lsl_t::skip [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.1.4.6 patchbay `MHAEvents::patchbay_t< ac2lsl_t> ac2lsl::ac2lsl_t::patchbay [private]`

5.1.4.7 is_first_run `bool ac2lsl::ac2lsl_t::is_first_run [private]`

The documentation for this class was generated from the following file:

- **ac2lsl.cpp**

5.2 ac2lsl::cfg_t Class Reference

Runtime configuration class of the **ac2lsl** (p. 78) plugin.

Public Member Functions

- **cfg_t** (const **algo_comm_t** &ac_, unsigned skip_, const std::string & **source_id**, const std::vector< std::string > &varnames_, double rate)

  *C'tor of **ac2lsl** (p. 78) run time configuration.*
- void **process** ()

Private Member Functions

- void **create_or_replace_var** (const std::string &name, const **comm_var_t** &v)
- void **check_vars** ()
- void **update_varlist** ()

Private Attributes

- std::map< std::string, std::unique_ptr< **save_var_base_t** > > **varlist**

  Maps variable name to unique ptr's of ac to lsl bridges.
- unsigned **skipcnt**

  Counter of frames to skip.
- const unsigned **skip**

  Number of frames to skip after each send.
- const double **srate**

  Sampling rate of the stream.
- const std::string **source_id**

  User configurable source id.
- const **algo_comm_t** & **ac**

  Handle to the ac space.

© 2005-2021 HörTech gGmbH, Oldenburg

5.2.1 Detailed Description

Runtime configuration class of the **ac2lsl** (p. 78) plugin.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 cfg_t()

```
cfg_t::cfg_t (
    const algo_comm_t & ac_,
    unsigned skip_,
    const std::string & source_id,
    const std::vector< std::string > & varnames_,
    double rate )
```

C'tor of **ac2lsl** (p. 78) run time configuration.

Parameters

*ac_*	AC space, source of data to send over LSL
*skip_*	Number of frames to skip after each send
*source_id_*	LSL identifier for this data stream
*varnames_*	Names of AC variables to send over LSL
rate	Rate with wich chunks of data are sent to the LSL stream. Usually the rate with which process calls happen, but may be lower due to the subsampling caused by skip_

5.2.3 Member Function Documentation

5.2.3.1 create_or_replace_var()

```
void cfg_t::create_or_replace_var (
    const std::string & name,
    const comm_var_t & v ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.2.3.2 check_vars() `void ac2lsl::cfg_t::check_vars ( ) [private]`

5.2.3.3 update_varlist() `void cfg_t::update_varlist ( ) [private]`

5.2.3.4 process() `void cfg_t::process ( )`

5.2.4 Member Data Documentation

5.2.4.1 varlist `std::map<std::string, std::unique_ptr< save_var_base_t> > ac2lsl↩ ::cfg_t::varlist [private]`

Maps variable name to unique ptr's of ac to lsl bridges.

5.2.4.2 skipcnt `unsigned ac2lsl::cfg_t::skipcnt [private]`

Counter of frames to skip.

5.2.4.3 skip `const unsigned ac2lsl::cfg_t::skip [private]`

Number of frames to skip after each send.

5.2.4.4 srate `const double ac2lsl::cfg_t::srate [private]`

Sampling rate of the stream.

© 2005-2021 HörTech gGmbH, Oldenburg

5.2.4.5 source_id `const std::string ac2lsl::cfg_t::source_id [private]`

User configurable source id.

5.2.4.6 ac `const algo_comm_t& ac2lsl::cfg_t::ac [private]`

Handle to the ac space.

The documentation for this class was generated from the following file:

- **ac2lsl.cpp**

5.3 ac2lsl::save_var_base_t Class Reference

Interface for ac to lsl bridge variable.

Inheritance diagram for ac2lsl::save_var_base_t:

Public Member Functions

- virtual void **send_frame** ()=0
- virtual void * **get_buf_address** () const noexcept=0
- virtual void **set_buf_address** (void *data)=0
- virtual lsl::stream_info **info** () const noexcept=0
- virtual unsigned **data_type** () const noexcept=0
- virtual unsigned **num_entries** () const noexcept=0
- virtual ~**save_var_base_t** ()=default

© 2005-2021 HörTech gGmbH, Oldenburg

5.3.1 Detailed Description

Interface for ac to lsl bridge variable.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 ~save_var_base_t() `virtual ac2lsl::save_var_base_t::~save_var_base_t ( ) [virtual], [default]`

5.3.3 Member Function Documentation

5.3.3.1 send_frame() `virtual void ac2lsl::save_var_base_t::send_frame ( ) [pure virtual]`

Implemented in **ac2lsl::save_var_t< mha_complex_t >** (p. 178), and **ac2lsl::save_var_t< T >** (p. 174).

5.3.3.2 get_buf_address() `virtual void* ac2lsl::save_var_base_t::get_buf_address ( ) const [pure virtual], [noexcept]`

Implemented in **ac2lsl::save_var_t< mha_complex_t >** (p. 177), and **ac2lsl::save_var_t< T >** (p. 173).

5.3.3.3 set_buf_address() `virtual void ac2lsl::save_var_base_t::set_buf_address ( void * data ) [pure virtual]`

Implemented in **ac2lsl::save_var_t< mha_complex_t >** (p. 177), and **ac2lsl::save_var_t< T >** (p. 173).

© 2005-2021 HörTech gGmbH, Oldenburg

5.3.3.4 info() `virtual lsl::stream_info ac2lsl::save_var_base_t::info ( ) const [pure virtual], [noexcept]`

Implemented in **ac2lsl::save_var_t< mha_complex_t >** (p. 177), and **ac2lsl::save_var_t< T >** (p. 174).

5.3.3.5 data_type() `virtual unsigned ac2lsl::save_var_base_t::data_type ( ) const [pure virtual], [noexcept]`

Implemented in **ac2lsl::save_var_t< mha_complex_t >** (p. 178), and **ac2lsl::save_var_t< T >** (p. 174).

5.3.3.6 num_entries() `virtual unsigned ac2lsl::save_var_base_t::num_entries ( ) const [pure virtual], [noexcept]`

Implemented in **ac2lsl::save_var_t< mha_complex_t >** (p. 177), and **ac2lsl::save_var_t< T >** (p. 174).

The documentation for this class was generated from the following file:

- **ac2lsl.cpp**

5.4 ac2lsl::save_var_t< T > Class Template Reference

Implementation for all ac to lsl bridges except complex types.

Inheritance diagram for ac2lsl::save_var_t< T >:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **save_var_t** (const std::string &name_, const std::string &type_, unsigned num_↩ entries_, const **mha_real_t** rate_, const lsl::channel_format_t format_, const std::string &source_id_, void ∗data_, const unsigned **data_type_**)

  C'tor of generic ac to lsl bridge.
- virtual void ∗ **get_buf_address** () const noexcept override

  Get buffer address as void pointer.
- virtual void **set_buf_address** (void ∗data) override

  Cast the input pointer to the appropriate type and set the buffer address.
- virtual lsl::stream_info **info** () const noexcept override

  Get stream info object from stream outlet.
- virtual unsigned **num_entries** () const noexcept override

  Get number of entries in the stream object.
- virtual unsigned **data_type** () const noexcept override

  Get data type id according MHA convention.
- virtual ∼**save_var_t** ()=default
- virtual void **send_frame** () override

  Send a frame to lsl.

Private Attributes

- lsl::stream_outlet **stream**

  LSL stream outlet.
- T ∗ **buf**

  Pointer to data buffer of the ac variable.
- const unsigned **data_type_**

  Data type id according to MHA convention.

5.4.1 Detailed Description

template<typename T>
class ac2lsl::save_var_t< T >

Implementation for all ac to lsl bridges except complex types.

5.4.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.4.2.1 save_var_t() template<typename T >

```
ac2lsl::save_var_t< T >::  save_var_t (
            const std::string & name_,
            const std::string & type_,
            unsigned num_entries_,
            const  mha_real_t rate_,
            const lsl::channel_format_t format_,
            const std::string & source_id_,
            void * data_,
            const unsigned data_type_ )  [inline]
```

C'tor of generic ac to lsl bridge.

Parameters

info	LSL stream info object containing metadata
data	Pointer to data buffer of the ac variable
data_type	Type id of the stream, in mha convention. Should be set to one if not a vector.

5.4.2.2 ~save_var_t() template<typename T >

```
virtual  ac2lsl::save_var_t< T >::~ save_var_t ( )  [virtual], [default]
```

5.4.3 Member Function Documentation

5.4.3.1 get_buf_address() template<typename T >

```
virtual void*  ac2lsl::save_var_t< T >::get_buf_address ( ) const  [inline], [override],
[virtual], [noexcept]
```

Get buffer address as void pointer.

Returns

Adress of the data buffer

Implements **ac2lsl::save_var_base_t** (p. 170).

5.4.3.2 set_buf_address() template<typename T >

```
virtual void  ac2lsl::save_var_t< T >::set_buf_address (
            void * data )  [inline], [override], [virtual]
```

Cast the input pointer to the appropriate type and set the buffer address.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

data	New buffer address

Implements **ac2lsl::save_var_base_t** (p. 170).

5.4.3.3 info() `template<typename T >`

`virtual lsl::stream_info` **`ac2lsl::save_var_t`**`< T >::info ( ) const [inline], [override], [virtual], [noexcept]`

Get stream info object from stream outlet.

Implements **ac2lsl::save_var_base_t** (p. 170).

5.4.3.4 num_entries() `template<typename T >`

`virtual unsigned` **`ac2lsl::save_var_t`**`< T >::num_entries ( ) const [inline], [override], [virtual], [noexcept]`

Get number of entries in the stream object.

Implements **ac2lsl::save_var_base_t** (p. 171).

5.4.3.5 data_type() `template<typename T >`

`virtual unsigned` **`ac2lsl::save_var_t`**`< T >::data_type ( ) const [inline], [override], [virtual], [noexcept]`

Get data type id according MHA convention.

Implements **ac2lsl::save_var_base_t** (p. 171).

5.4.3.6 send_frame() `template<typename T >`

`virtual void` **`ac2lsl::save_var_t`**`< T >::send_frame ( ) [inline], [override], [virtual]`

Send a frame to lsl.

Implements **ac2lsl::save_var_base_t** (p. 170).

© 2005-2021 HörTech gGmbH, Oldenburg

5.4.4 Member Data Documentation

5.4.4.1 stream `template<typename T >`

`lsl::stream_outlet` **`ac2lsl::save_var_t`**`< T >::stream [private]`

LSL stream outlet.

Interface to lsl

5.4.4.2 buf `template<typename T >`

`T*` **`ac2lsl::save_var_t`**`< T >::buf [private]`

Pointer to data buffer of the ac variable.

5.4.4.3 data_type_ `template<typename T >`

`const unsigned` **`ac2lsl::save_var_t`**`< T >::data_type_ [private]`

Data type id according to MHA convention.

The documentation for this class was generated from the following file:

- **ac2lsl.cpp**

5.5 ac2lsl::save_var_t< mha_complex_t > Class Reference

Template specialization of the **ac2lsl** (p. 78) bridge to take care of complex numbers.

Inheritance diagram for ac2lsl::save_var_t< mha_complex_t >:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **save_var_t** (const std::string &name_, const std::string &type_, const unsigned num_↩ entries_, const **mha_real_t** rate_, const lsl::channel_format_t format_, const std::string &source_id_, void ∗data_)

  C'tor of specialization for complex types.
- virtual void ∗ **get_buf_address** () const noexcept override
- virtual void **set_buf_address** (void ∗data) override
- virtual lsl::stream_info **info** () const noexcept override

  Get buffer address as void pointer.
- virtual unsigned **num_entries** () const noexcept override

  Get number of entries in the stream object.
- virtual unsigned **data_type** () const noexcept override

  Cast the input pointer to the appropriate type and set the buffer address.
- virtual ∼**save_var_t** ()=default
- virtual void **send_frame** () override

  Send a frame of complex types.

Private Attributes

- lsl::stream_outlet **stream**

  LSL stream outlet.
- **mha_complex_t** ∗ **buf**

  Pointer to data buffer of the ac variable.

5.5.1 Detailed Description

Template specialization of the **ac2lsl** (p. 78) bridge to take care of complex numbers.

This specialization is needed because lsl does not support complex numbers. Order is [re(0), im(0), re(1), im(1), ...]

5.5.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.5.2.1 save_var_t()

```
ac2lsl::save_var_t< mha_complex_t >::save_var_t (
    const std::string & name_,
    const std::string & type_,
    const unsigned num_entries_,
    const mha_real_t rate_,
    const lsl::channel_format_t format_,
    const std::string & source_id_,
    void * data_ ) [inline]
```

C'tor of specialization for complex types.

See generic c'tor for details.

5.5.2.2 ~save_var_t()

```
virtual ac2lsl::save_var_t< mha_complex_t >::~save_var_t
( ) [virtual], [default]
```

5.5.3 Member Function Documentation

5.5.3.1 get_buf_address()

```
virtual void* ac2lsl::save_var_t< mha_complex_t >↩
::get_buf_address ( ) const [inline], [override], [virtual], [noexcept]
```

Implements **ac2lsl::save_var_base_t** (p. 170).

5.5.3.2 set_buf_address()

```
virtual void ac2lsl::save_var_t< mha_complex_t >↩
::set_buf_address (
    void * data ) [inline], [override], [virtual]
```

Implements **ac2lsl::save_var_base_t** (p. 170).

5.5.3.3 info()

```
virtual lsl::stream_info ac2lsl::save_var_t< mha_complex_t >::info
( ) const [inline], [override], [virtual], [noexcept]
```

Get buffer address as void pointer.

Returns

Adress of the data buffer

Implements **ac2lsl::save_var_base_t** (p. 170).

© 2005-2021 HörTech gGmbH, Oldenburg

5.5.3.4 num_entries() `virtual unsigned` **`ac2lsl::save_var_t<`** **`mha_complex_t`** `>::num_entries ( ) const [inline], [override], [virtual], [noexcept]`

Get number of entries in the stream object.

Implements **ac2lsl::save_var_base_t** (p. 171).

5.5.3.5 data_type() `virtual unsigned` **`ac2lsl::save_var_t<`** **`mha_complex_t`** `>::data_type ( ) const [inline], [override], [virtual], [noexcept]`

Cast the input pointer to the appropriate type and set the buffer address.

Parameters

data	New buffer address

Implements **ac2lsl::save_var_base_t** (p. 171).

5.5.3.6 send_frame() `virtual void` **`ac2lsl::save_var_t<`** **`mha_complex_t`** `>::send_frame ( ) [inline], [override], [virtual]`

Send a frame of complex types.

Complex numbers are stored as alternating real and imaginary parts. An array of complex numbers in memory can be reinterpreted as a vector of real numbers that correspond to real and imaginary parts. LSL does not support complex types directly. Send one vector containing {buf[0].re,buf[0].im,buf[1].re,buf[1].im,...} instead.

Implements **ac2lsl::save_var_base_t** (p. 170).

5.5.4 Member Data Documentation

5.5.4.1 stream `lsl::stream_outlet` **`ac2lsl::save_var_t<`** **`mha_complex_t`** `>::stream [private]`

LSL stream outlet.

Interface to lsl

© 2005-2021 HörTech gGmbH, Oldenburg

5.5.4.2 buf `mha_complex_t* ac2lsl::save_var_t< mha_complex_t >::buf [private]`

Pointer to data buffer of the ac variable.

The documentation for this class was generated from the following file:

- **ac2lsl.cpp**

5.6 ac2lsl::type_info Struct Reference

Public Attributes

- const std::string **name**
- const lsl::channel_format_t **format**

5.6.1 Member Data Documentation

5.6.1.1 name `const std::string ac2lsl::type_info::name`

5.6.1.2 format `const lsl::channel_format_t ac2lsl::type_info::format`

The documentation for this struct was generated from the following file:

- **ac2lsl.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.7 ac2osc_t Class Reference

Plugin class of the ac2osc plugin.

Inheritance diagram for ac2osc_t:

Public Member Functions

- **ac2osc_t** (**algo_comm_t** iac, const std::string &configured_name)
  C'tor of plugin class.
- void **prepare** (**mhaconfig_t** &)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s)
  Processing fct for waveforms.
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗s)
  Processing fct for spectra.
- void **process** ()
  Process function.
- void **release** ()
  Release frees osc related memory, does cleanup.

Private Member Functions

- void **send_osc_float** ()
- void **update_mode** ()
  Start/Stop sending of messages.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::string_t host**
  OSC server host name.
- **MHAParser::string_t port**
  OSC server port.
- **MHAParser::int_t ttl**
  Time-to-live of UDP packages.
- **MHAParser::vstring_t vars**
  List of AC variables to be saved, empty for all.
- **MHAParser::kw_t mode**
  Record mode.
- **MHAParser::int_t skip**
  number of frames to skip after sending
- **MHAParser::bool_t rt_strict**
  abort if used in real-time thread?
- std::unique_ptr< **MHA_AC::acspace2matrix_t** > **acspace**
- **MHAEvents::patchbay_t**< **ac2osc_t** > **patchbay**
- bool **b_record**
- uint8_t * **rtmem**
- float **framerate**
- int **skipcnt**
- lo_address **lo_addr**
- bool **is_first_run**

Additional Inherited Members

5.7.1 Detailed Description

Plugin class of the ac2osc plugin.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 ac2osc_t()

```
ac2osc_t::ac2osc_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

C'tor of plugin class.

© 2005-2021 HörTech gGmbH, Oldenburg

5.7.3 Member Function Documentation

5.7.3.1 prepare() `void ac2osc_t::prepare ( mhaconfig_t & cf ) [virtual]`

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

5.7.3.2 process() [1/3] `mha_wave_t* ac2osc_t::process ( mha_wave_t * s ) [inline]`

Processing fct for waveforms.

Calls **process(void)** (p. 182).

5.7.3.3 process() [2/3] `mha_spec_t* ac2osc_t::process ( mha_spec_t * s ) [inline]`

Processing fct for spectra.

Calls **process(void)** (p. 182).

5.7.3.4 process() [3/3] `void ac2osc_t::process ( )`

Process function.

Checks once if the plugin is run in a real-time thread and throws if rt_strict is true, sends osc messages according to config.

5.7.3.5 release() `void ac2osc_t::release ( ) [virtual]`

Release frees osc related memory, does cleanup.

Reimplemented from **MHAPlugin::plugin_t< int >** (p. 1154).

© 2005-2021 HörTech gGmbH, Oldenburg

5.7.3.6 send_osc_float() `void ac2osc_t::send_osc_float ( ) [private]`

5.7.3.7 update_mode() `void ac2osc_t::update_mode ( ) [private]`

Start/Stop sending of messages.

5.7.4 Member Data Documentation

5.7.4.1 host `MHAParser::string_t ac2osc_t::host [private]`

OSC server host name.

5.7.4.2 port `MHAParser::string_t ac2osc_t::port [private]`

OSC server port.

5.7.4.3 ttl `MHAParser::int_t ac2osc_t::ttl [private]`

Time-to-live of UDP packages.

5.7.4.4 vars `MHAParser::vstring_t ac2osc_t::vars [private]`

List of AC variables to be saved, empty for all.

© 2005-2021 HörTech gGmbH, Oldenburg

5.7.4.5 mode `MHAParser::kw_t ac2osc_t::mode [private]`

Record mode.

5.7.4.6 skip `MHAParser::int_t ac2osc_t::skip [private]`

number of frames to skip after sending

5.7.4.7 rt_strict `MHAParser::bool_t ac2osc_t::rt_strict [private]`

abort if used in real-time thread?

5.7.4.8 acspace `std::unique_ptr< MHA_AC::acspace2matrix_t> ac2osc_t::acspace [private]`

5.7.4.9 patchbay `MHAEvents::patchbay_t< ac2osc_t> ac2osc_t::patchbay [private]`

5.7.4.10 b_record `bool ac2osc_t::b_record [private]`

5.7.4.11 rtmem `uint8_t* ac2osc_t::rtmem [private]`

5.7.4.12 framerate `float ac2osc_t::framerate [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.7.4.13 skipcnt `int ac2osc_t::skipcnt [private]`

5.7.4.14 lo_addr `lo_address ac2osc_t::lo_addr [private]`

5.7.4.15 is_first_run `bool ac2osc_t::is_first_run [private]`

The documentation for this class was generated from the following file:

- **ac2osc.cpp**

5.8 ac2wave_if_t Class Reference

Inheritance diagram for ac2wave_if_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **ac2wave_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_spec_t** ∗)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

Private Member Functions

- void **update** ()

Private Attributes

- **MHAParser::string_t name**
- **MHAParser::float_t gain_in**
- **MHAParser::float_t gain_ac**
- **MHAParser::int_t delay_in**
- **MHAParser::int_t delay_ac**
- **MHASignal::waveform_t** ∗ **zeros**
- bool **prepared**
- **MHAEvents::patchbay_t**< **ac2wave_if_t** > **patchbay**

Additional Inherited Members

5.8.1 Constructor & Destructor Documentation

5.8.1.1 ac2wave_if_t()

```
ac2wave_if_t::ac2wave_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.8.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.8.2.1 process() [1/2]

```
mha_wave_t * ac2wave_if_t::process (
    mha_spec_t * )
```

5.8.2.2 process() [2/2]

```
mha_wave_t * ac2wave_if_t::process (
    mha_wave_t * s )
```

5.8.2.3 prepare()

```
void ac2wave_if_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< ac2wave_t >** (p. 1153).

5.8.2.4 release()

```
void ac2wave_if_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< ac2wave_t >** (p. 1154).

5.8.2.5 update()

```
void ac2wave_if_t::update ( ) [private]
```

5.8.3 Member Data Documentation

5.8.3.1 name

```
MHAParser::string_t ac2wave_if_t::name [private]
```

5.8.3.2 gain_in

```
MHAParser::float_t ac2wave_if_t::gain_in [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.8.3.3 gain_ac `MHAParser::float_t ac2wave_if_t::gain_ac [private]`

5.8.3.4 delay_in `MHAParser::int_t ac2wave_if_t::delay_in [private]`

5.8.3.5 delay_ac `MHAParser::int_t ac2wave_if_t::delay_ac [private]`

5.8.3.6 zeros `MHASignal::waveform_t* ac2wave_if_t::zeros [private]`

5.8.3.7 prepared `bool ac2wave_if_t::prepared [private]`

5.8.3.8 patchbay `MHAEvents::patchbay_t< ac2wave_if_t> ac2wave_if_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **ac2wave.cpp**

5.9 ac2wave_t Class Reference

Public Member Functions

- **ac2wave_t** (unsigned int frames_, unsigned int channels_, **algo_comm_t** ac_, std↩ ::string name_, float gain_in_, float gain_ac_, unsigned int delay_in_, unsigned int delay↩ _ac_)
- **mha_wave_t** * **process** (**mha_wave_t** *)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- unsigned int **frames**
- unsigned int **channels**
- **mha_wave_t w**
- **algo_comm_t ac**
- std::string **name**
- **MHASignal::delay_wave_t delay_in**
- **MHASignal::delay_wave_t delay_ac**
- **mha_real_t gain_in**
- **mha_real_t gain_ac**

5.9.1 Constructor & Destructor Documentation

5.9.1.1 ac2wave_t()

```
ac2wave_t::ac2wave_t (
    unsigned int frames_,
    unsigned int channels_,
    algo_comm_t ac_,
    std::string name_,
    float gain_in_,
    float gain_ac_,
    unsigned int delay_in_,
    unsigned int delay_ac_ )
```

5.9.2 Member Function Documentation

5.9.2.1 process()

```
mha_wave_t * ac2wave_t::process (
    mha_wave_t * s )
```

5.9.3 Member Data Documentation

5.9.3.1 frames

```
unsigned int ac2wave_t::frames [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.9.3.2 channels `unsigned int ac2wave_t::channels [private]`

5.9.3.3 w `mha_wave_t ac2wave_t::w [private]`

5.9.3.4 ac `algo_comm_t ac2wave_t::ac [private]`

5.9.3.5 name `std::string ac2wave_t::name [private]`

5.9.3.6 delay_in `MHASignal::delay_wave_t ac2wave_t::delay_in [private]`

5.9.3.7 delay_ac `MHASignal::delay_wave_t ac2wave_t::delay_ac [private]`

5.9.3.8 gain_in `mha_real_t ac2wave_t::gain_in [private]`

5.9.3.9 gain_ac `mha_real_t ac2wave_t::gain_ac [private]`

The documentation for this class was generated from the following file:

- **ac2wave.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.10 ac_mul_t Class Reference

The class which implements the **ac_mul_t** (p. 191) plugin.

Inheritance diagram for ac_mul_t:

Public Member Functions

- **ac_mul_t** (**algo_comm_t** iac, const std::string &configured_name)

  Plugin constructor.
- void **prepare_** (**mhaconfig_t** &)

  *Prepare method, called **prepare_()** (p. 193) with trailing underscore because **ac_mul_t** (p. 191) does not inherit from plugin_t<>.*
- void **release_** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **scan_syntax** ()
- void **get_arg_type_and_dimension** ()
- void **get_arg_type_and_dimension** (const std::string &, **val_type_t** &, unsigned int &, unsigned int &)
- void **process** ()
- void **process_rr** ()
- void **process_rc** ()
- void **process_cr** ()
- void **process_cc** ()

Private Attributes

- **algo_comm_t ac**
- std::string **algo**
- **arg_type_t argt**
- std::string **str_a**
- std::string **str_b**
- **MHA_AC::waveform_t** ∗ **res_r**
- **MHA_AC::spectrum_t** ∗ **res_c**
- unsigned int **num_frames**
- unsigned int **num_channels**

Additional Inherited Members

5.10.1 Detailed Description

The class which implements the **ac_mul_t** (p. 191) plugin.

Different from most other plugins, the ac_mul plugin's interface class does not inherit from plugin_t<>, but from **MHAParser::string_t** (p. 1115). This way, it does not get inserted into the MHA configuration tree as a parser node wich can have multiple variables, but as a string variable.

The **ac_mul_t** (p. 191) variable multiplies two AC variables element-wise.

5.10.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.10.2.1 ac_mul_t()

```
ac_mul_t::ac_mul_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Plugin constructor.

5.10.3 Member Function Documentation

5.10.3.1 prepare_()

```
void ac_mul_t::prepare_ (
    mhaconfig_t & )
```

Prepare method, called **prepare_()** (p. 193) with trailing underscore because **ac_mul_t** (p. 191) does not inherit from plugin_t<>.

Leaves signal dimensions unchanged. The AC variables contained in the string expression must exist at this point.

5.10.3.2 release_()

```
void ac_mul_t::release_ ( )
```

5.10.3.3 process() [1/3]

```
mha_wave_t * ac_mul_t::process (
    mha_wave_t * s )
```

5.10.3.4 process() [2/3]

```
mha_spec_t * ac_mul_t::process (
    mha_spec_t * s )
```

5.10.3.5 scan_syntax()

```
void ac_mul_t::scan_syntax ( ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.10.3.6 get_arg_type_and_dimension() [1/2] void ac_mul_t::get_arg_type_and_↩ dimension ( ) [private]

5.10.3.7 get_arg_type_and_dimension() [2/2] void ac_mul_t::get_arg_type_and_↩ dimension (

```
const std::string & name,
 val_type_t & vt,
unsigned int & num_frames,
unsigned int & num_channels ) [private]
```

5.10.3.8 process() [3/3] void ac_mul_t::process ( ) [private]

5.10.3.9 process_rr() void ac_mul_t::process_rr ( ) [private]

5.10.3.10 process_rc() void ac_mul_t::process_rc ( ) [private]

5.10.3.11 process_cr() void ac_mul_t::process_cr ( ) [private]

5.10.3.12 process_cc() void ac_mul_t::process_cc ( ) [private]

5.10.4 Member Data Documentation

5.10.4.1 ac algo_comm_t ac_mul_t::ac [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.10.4.2 algo `std::string ac_mul_t::algo [private]`

5.10.4.3 argt `arg_type_t ac_mul_t::argt [private]`

5.10.4.4 str_a `std::string ac_mul_t::str_a [private]`

5.10.4.5 str_b `std::string ac_mul_t::str_b [private]`

5.10.4.6 res_r `MHA_AC::waveform_t* ac_mul_t::res_r [private]`

5.10.4.7 res_c `MHA_AC::spectrum_t* ac_mul_t::res_c [private]`

5.10.4.8 num_frames `unsigned int ac_mul_t::num_frames [private]`

5.10.4.9 num_channels `unsigned int ac_mul_t::num_channels [private]`

The documentation for this class was generated from the following files:

- **ac_mul.hh**
- **ac_mul.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.11 ac_proc::interface_t Class Reference

Inheritance diagram for ac_proc::interface_t:

Public Member Functions

- **interface_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- void **process** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Private Attributes

- std::string **algo**
- **MHAParser::mhapluginloader_t plug**
- **MHAParser::string_t input**
- **MHAParser::bool_t permute**
- **MHA_AC::waveform_t** ∗ **s_out**
- **MHASignal::waveform_t** ∗ **s_in_perm**
- bool **b_permute**
- **mha_wave_t s_in**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.11.1 Constructor & Destructor Documentation

5.11.1.1 interface_t()

```
ac_proc::interface_t::interface_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Default values are set and MHA configuration variables registered into the parser.

Parameters

iac	algorithm communication handle
configured_name	algorithm name

5.11.2 Member Function Documentation

5.11.2.1 prepare()

```
void ac_proc::interface_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

5.11.2.2 release()

```
void ac_proc::interface_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< int >** (p. 1154).

5.11.2.3 process() [1/3]

```
void ac_proc::interface_t::process ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.11.2.4 process() [2/3]

mha_spec_t * ac_proc::interface_t::process (**mha_spec_t** * *s*)

5.11.2.5 process() [3/3]

mha_wave_t * ac_proc::interface_t::process (**mha_wave_t** * *s*)

5.11.3 Member Data Documentation

5.11.3.1 algo

std::string ac_proc::interface_t::algo [private]

5.11.3.2 plug

MHAParser::mhapluginloader_t ac_proc::interface_t::plug [private]

5.11.3.3 input

MHAParser::string_t ac_proc::interface_t::input [private]

5.11.3.4 permute

MHAParser::bool_t ac_proc::interface_t::permute [private]

5.11.3.5 s_out

MHA_AC::waveform_t* ac_proc::interface_t::s_out [private]

5.11.3.6 s_in_perm

MHASignal::waveform_t* ac_proc::interface_t::s_in_perm [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.11.3.7 b_permute `bool ac_proc::interface_t::b_permute [private]`

5.11.3.8 s_in `mha_wave_t ac_proc::interface_t::s_in [private]`

The documentation for this class was generated from the following file:

- **ac_proc.cpp**

5.12 acConcat_wave Class Reference

Inheritance diagram for acConcat_wave:

Public Member Functions

- **acConcat_wave** (**algo_comm_t** iac, const std::string &configured_name)
  Constructs our plugin.
- ~**acConcat_wave** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
  Checks for the most recent configuration and defers processing to it.
- void **prepare** (**mhaconfig_t** &)
  Plugin preparation.
- void **release** (void)

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **MHAParser::int_t num_AC**
- **MHAParser::string_t prefix_names_AC**
- **MHAParser::vint_t samples_AC**
- **MHAParser::string_t name_con_AC**
- **MHAParser::int_t numchannels**

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t**< **acConcat_wave** > **patchbay**

Additional Inherited Members

5.12.1 Constructor & Destructor Documentation

5.12.1.1 acConcat_wave()

```
acConcat_wave::acConcat_wave (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructs our plugin.

5.12.1.2 ~acConcat_wave()

```
acConcat_wave::~acConcat_wave ( )
```

5.12.2 Member Function Documentation

5.12.2.1 process()

```
mha_wave_t * acConcat_wave::process (
    mha_wave_t * signal )
```

Checks for the most recent configuration and defers processing to it.

5.12.2.2 prepare()

```
void acConcat_wave::prepare (
    mhaconfig_t & ) [virtual]
```

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< acConcat_wave_config >** (p. 1153).

5.12.2.3 release()

```
void acConcat_wave::release (
    void ) [inline], [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< acConcat_wave_config >** (p. 1154).

5.12.2.4 update_cfg()

```
void acConcat_wave::update_cfg ( ) [private]
```

5.12.3 Member Data Documentation

5.12.3.1 num_AC

```
MHAParser::int_t acConcat_wave::num_AC
```

5.12.3.2 prefix_names_AC

```
MHAParser::string_t acConcat_wave::prefix_names_AC
```

5.12.3.3 samples_AC

```
MHAParser::vint_t acConcat_wave::samples_AC
```

5.12.3.4 name_con_AC

```
MHAParser::string_t acConcat_wave::name_con_AC
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.12.3.5 numchannels `MHAParser::int_t` `acConcat_wave::numchannels`

5.12.3.6 patchbay `MHAEvents::patchbay_t< acConcat_wave>` `acConcat_wave::patchbay` `[private]`

The documentation for this class was generated from the following files:

- **acConcat_wave.h**
- **acConcat_wave.cpp**

5.13 acConcat_wave_config Class Reference

Public Member Functions

- **acConcat_wave_config** (**algo_comm_t** & **ac**, **acConcat_wave** ∗_concat)
- ∼**acConcat_wave_config** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Public Attributes

- **algo_comm_t** & **ac**
- std::vector< std::string > **strNames_AC**
- std::vector< int > **numSamples_AC**
- **mha_wave_t vGCC**
- **MHA_AC::waveform_t** ∗ **vGCC_con**

5.13.1 Constructor & Destructor Documentation

5.13.1.1 acConcat_wave_config()

```
acConcat_wave_config::acConcat_wave_config (
    algo_comm_t & ac,
    acConcat_wave * _concat )
```

5.13.1.2 ∼acConcat_wave_config()

```
acConcat_wave_config::~acConcat_wave_config ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.13.2 Member Function Documentation

5.13.2.1 process() `mha_wave_t * acConcat_wave_config::process ( mha_wave_t * wave )`

5.13.3 Member Data Documentation

5.13.3.1 ac `algo_comm_t& acConcat_wave_config::ac`

5.13.3.2 strNames_AC `std::vector<std::string> acConcat_wave_config::strNames_AC`

5.13.3.3 numSamples_AC `std::vector<int> acConcat_wave_config::numSamples_AC`

5.13.3.4 vGCC `mha_wave_t acConcat_wave_config::vGCC`

5.13.3.5 vGCC_con `MHA_AC::waveform_t* acConcat_wave_config::vGCC_con`

The documentation for this class was generated from the following files:

- **acConcat_wave.h**
- **acConcat_wave.cpp**

5.14 acmon::ac_monitor_t Class Reference

A class for converting AC variables to Parser monitors of correct type.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **ac_monitor_t** (**MHAParser::parser_t** &parent, const std::string &name_, **algo_comm_t** ac, bool use_matrix)
  Converts AC variable to parser monitor.
- void **getvar** (**algo_comm_t** ac)
  Update values of monitor.

Public Attributes

- std::string **name**
  name of AC variable and parser monitor
- std::string **dimstr**
  columns x rows
- **MHAParser::vfloat_mon_t mon**
  Monitor used for real vectors.
- **MHAParser::mfloat_mon_t mon_mat**
  Monitor used for real matrices.
- **MHAParser::vcomplex_mon_t mon_complex**
  monitor used for complex vectors
- **MHAParser::mcomplex_mon_t mon_mat_complex**
  monitor used for complex matrices
- **MHAParser::parser_t** & **p_parser**
  parent parser to insert monitor into

Private Attributes

- bool **use_mat**
  if true, use matrix monitor, else use vector monitor

5.14.1 Detailed Description

A class for converting AC variables to Parser monitors of correct type.

5.14.2 Constructor & Destructor Documentation

5.14.2.1 ac_monitor_t()

```
acmon::ac_monitor_t::ac_monitor_t (
    MHAParser::parser_t & parent,
    const std::string & name_,
    algo_comm_t ac,
    bool use_matrix )
```

Converts AC variable to parser monitor.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

parent	The parser to insert a monitor into
*name_*	The name of the AC variable and the monitor variable
ac	Handle to algorithm communication space
use_matrix	Indicates if a matrix monitor type should be used.

5.14.3 Member Function Documentation

5.14.3.1 getvar() `void acmon::ac_monitor_t::getvar ( algo_comm_t ac )`

Update values of monitor.

Parameters

ac	Handle to algorithm communication space

5.14.4 Member Data Documentation

5.14.4.1 name `std::string acmon::ac_monitor_t::name`

name of AC variable and parser monitor

5.14.4.2 dimstr `std::string acmon::ac_monitor_t::dimstr`

columns x rows

© 2005-2021 HörTech gGmbH, Oldenburg

5.14.4.3 mon `MHAParser::vfloat_mon_t acmon::ac_monitor_t::mon`

Monitor used for real vectors.

5.14.4.4 mon_mat `MHAParser::mfloat_mon_t acmon::ac_monitor_t::mon_mat`

Monitor used for real matrices.

5.14.4.5 mon_complex `MHAParser::vcomplex_mon_t acmon::ac_monitor_t::mon_complex`

monitor used for complex vectors

5.14.4.6 mon_mat_complex `MHAParser::mcomplex_mon_t acmon::ac_monitor_t::mon_mat_complex`

monitor used for complex matrices

5.14.4.7 p_parser `MHAParser::parser_t& acmon::ac_monitor_t::p_parser`

parent parser to insert monitor into

5.14.4.8 use_mat `bool acmon::ac_monitor_t::use_mat [private]`

if true, use matrix monitor, else use vector monitor

The documentation for this class was generated from the following files:

- **ac_monitor_type.hh**
- **ac_monitor_type.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.15 acmon::acmon_t Class Reference

Inheritance diagram for acmon::acmon_t:

Public Member Functions

- **acmon_t** (**algo_comm_t**, const std::string &configured_name)
- ∼**acmon_t** ()
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Private Member Functions

- void **save_vars** ()
- void **update_recmode** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **algo_comm_t ac**
- **MHAParser::vstring_mon_t varlist**
- **MHAParser::vstring_mon_t dimensions**
- **MHAParser::kw_t dispmode**
- **MHAParser::kw_t recmode**
- std::vector< **ac_monitor_t** * > **vars**
- **MHAEvents::patchbay_t**< **acmon_t** > **patchbay**
- std::string **algo**
- bool **b_cont**
- bool **b_snapshot**

Additional Inherited Members

5.15.1 Constructor & Destructor Documentation

5.15.1.1 acmon_t()

```
acmon::acmon_t::acmon_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.15.1.2 ~acmon_t()

```
acmon::acmon_t::~acmon_t ( )
```

5.15.2 Member Function Documentation

5.15.2.1 prepare()

```
void acmon::acmon_t::prepare (
    mhaconfig_t & ) [virtual]
```

Implements **MHAPlugin::plugin_t**< **int** > (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.15.2.2 release() `void acmon::acmon_t::release ( ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t**< **int** > (p. 1154).

5.15.2.3 process() [1/2] `mha_spec_t * acmon::acmon_t::process ( mha_spec_t * s )`

5.15.2.4 process() [2/2] `mha_wave_t * acmon::acmon_t::process ( mha_wave_t * s )`

5.15.2.5 save_vars() `void acmon::acmon_t::save_vars ( ) [private]`

5.15.2.6 update_recmode() `void acmon::acmon_t::update_recmode ( ) [private]`

5.15.3 Member Data Documentation

5.15.3.1 ac `algo_comm_t acmon::acmon_t::ac [private]`

5.15.3.2 varlist `MHAParser::vstring_mon_t acmon::acmon_t::varlist [private]`

5.15.3.3 dimensions `MHAParser::vstring_mon_t acmon::acmon_t::dimensions [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.15.3.4 dispmode `MHAParser::kw_t acmon::acmon_t::dispmode [private]`

5.15.3.5 recmode `MHAParser::kw_t acmon::acmon_t::recmode [private]`

5.15.3.6 vars `std::vector< ac_monitor_t* > acmon::acmon_t::vars [private]`

5.15.3.7 patchbay `MHAEvents::patchbay_t< acmon_t > acmon::acmon_t::patchbay [private]`

5.15.3.8 algo `std::string acmon::acmon_t::algo [private]`

5.15.3.9 b_cont `bool acmon::acmon_t::b_cont [private]`

5.15.3.10 b_snapshot `bool acmon::acmon_t::b_snapshot [private]`

The documentation for this class was generated from the following file:

- **acmon.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.16 acPooling_wave Class Reference

Inheritance diagram for acPooling_wave:

Public Member Functions

- **acPooling_wave** (**algo_comm_t** iac, const std::string &configured_name)
  Constructs our plugin.
- ∼**acPooling_wave** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
  Checks for the most recent configuration and defers processing to it.
- void **prepare** (**mhaconfig_t** &)
  Plugin preparation.
- void **release** (void)

Public Attributes

- **MHAParser::int_t numsamples**
- **MHAParser::int_t pooling_wndlen**
- **MHAParser::kw_t pooling_type**
- **MHAParser::float_t upper_threshold**
- **MHAParser::float_t lower_threshold**
- **MHAParser::int_t neighbourhood**

© 2005-2021 HörTech gGmbH, Oldenburg

- **MHAParser::float_t alpha**
- **MHAParser::string_t p_name**
- **MHAParser::string_t p_biased_name**
- **MHAParser::string_t pool_name**
- **MHAParser::string_t max_pool_ind_name**
- **MHAParser::string_t like_ratio_name**
- **MHAParser::vfloat_t prob_bias**

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t**< **acPooling_wave** > **patchbay**

Additional Inherited Members

5.16.1 Constructor & Destructor Documentation

5.16.1.1 acPooling_wave()

```
acPooling_wave::acPooling_wave (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructs our plugin.

5.16.1.2 ~acPooling_wave()

```
acPooling_wave::~acPooling_wave ( )
```

5.16.2 Member Function Documentation

5.16.2.1 process()

```
mha_wave_t * acPooling_wave::process (
    mha_wave_t * signal )
```

Checks for the most recent configuration and defers processing to it.

5.16.2.2 prepare()

```
void acPooling_wave::prepare (
    mhaconfig_t & signal_info ) [virtual]
```

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< acPooling_wave_config >** (p. 1153).

5.16.2.3 release()

```
void acPooling_wave::release (
    void ) [inline], [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< acPooling_wave_config >** (p. 1154).

5.16.2.4 update_cfg()

```
void acPooling_wave::update_cfg ( ) [private]
```

5.16.3 Member Data Documentation

5.16.3.1 numsamples

```
MHAParser::int_t acPooling_wave::numsamples
```

5.16.3.2 pooling_wndlen

```
MHAParser::int_t acPooling_wave::pooling_wndlen
```

5.16.3.3 pooling_type

```
MHAParser::kw_t acPooling_wave::pooling_type
```

5.16.3.4 upper_threshold

```
MHAParser::float_t acPooling_wave::upper_threshold
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.16.3.5 lower_threshold `MHAParser::float_t acPooling_wave::lower_threshold`

5.16.3.6 neighbourhood `MHAParser::int_t acPooling_wave::neighbourhood`

5.16.3.7 alpha `MHAParser::float_t acPooling_wave::alpha`

5.16.3.8 p_name `MHAParser::string_t acPooling_wave::p_name`

5.16.3.9 p_biased_name `MHAParser::string_t acPooling_wave::p_biased_name`

5.16.3.10 pool_name `MHAParser::string_t acPooling_wave::pool_name`

5.16.3.11 max_pool_ind_name `MHAParser::string_t acPooling_wave::max_pool_ind_name`

5.16.3.12 like_ratio_name `MHAParser::string_t acPooling_wave::like_ratio_name`

5.16.3.13 prob_bias `MHAParser::vfloat_t acPooling_wave::prob_bias`

© 2005-2021 HörTech gGmbH, Oldenburg

5.16.3.14 patchbay `MHAEvents::patchbay_t< acPooling_wave> acPooling_wave::patchbay [private]`

The documentation for this class was generated from the following files:

- **acPooling_wave.h**
- **acPooling_wave.cpp**

5.17 acPooling_wave_config Class Reference

Public Member Functions

- **acPooling_wave_config** (**algo_comm_t** & **ac**, const **mhaconfig_t** in_cfg, **ac↩ Pooling_wave** ∗_pooling)
- ∼**acPooling_wave_config** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **insert** ()

Public Attributes

- **algo_comm_t** & **ac**
- std::string **raw_p_name**
- **MHA_AC::waveform_t p**
- **MHA_AC::waveform_t p_biased**
- **MHA_AC::waveform_t p_max**
- **MHA_AC::waveform_t like_ratio**
- **mha_wave_t c**
- unsigned int **pooling_ind**
- unsigned int **pooling_option**
- unsigned int **pooling_size**
- float **up_thresh**
- float **low_thresh**
- int **neigh**
- float **alpha**
- **MHASignal::waveform_t pool**
- **MHASignal::waveform_t prob_bias_func**

5.17.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.17.1.1 acPooling_wave_config()

```
acPooling_wave_config::acPooling_wave_config (
    algo_comm_t & ac,
    const mhaconfig_t in_cfg,
    acPooling_wave * _pooling )
```

5.17.1.2 ~acPooling_wave_config()

```
acPooling_wave_config::~acPooling_wave_config ( )
```

5.17.2 Member Function Documentation

5.17.2.1 process()

```
mha_wave_t * acPooling_wave_config::process (
    mha_wave_t * wave )
```

5.17.2.2 insert()

```
void acPooling_wave_config::insert ( )
```

5.17.3 Member Data Documentation

5.17.3.1 ac

```
algo_comm_t& acPooling_wave_config::ac
```

5.17.3.2 raw_p_name

```
std::string acPooling_wave_config::raw_p_name
```

5.17.3.3 p

```
MHA_AC::waveform_t acPooling_wave_config::p
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.17.3.4 p_biased

`MHA_AC::waveform_t acPooling_wave_config::p_biased`

5.17.3.5 p_max

`MHA_AC::waveform_t acPooling_wave_config::p_max`

5.17.3.6 like_ratio

`MHA_AC::waveform_t acPooling_wave_config::like_ratio`

5.17.3.7 c

`mha_wave_t acPooling_wave_config::c`

5.17.3.8 pooling_ind

`unsigned int acPooling_wave_config::pooling_ind`

5.17.3.9 pooling_option

`unsigned int acPooling_wave_config::pooling_option`

5.17.3.10 pooling_size

`unsigned int acPooling_wave_config::pooling_size`

5.17.3.11 up_thresh

`float acPooling_wave_config::up_thresh`

5.17.3.12 low_thresh

`float acPooling_wave_config::low_thresh`

© 2005-2021 HörTech gGmbH, Oldenburg

5.17.3.13 neigh `int acPooling_wave_config::neigh`

5.17.3.14 alpha `float acPooling_wave_config::alpha`

5.17.3.15 pool `MHASignal::waveform_t acPooling_wave_config::pool`

5.17.3.16 prob_bias_func `MHASignal::waveform_t acPooling_wave_config::prob_bias_func`

The documentation for this class was generated from the following files:

- **acPooling_wave.h**
- **acPooling_wave.cpp**

5.18 acsave::acsave_t Class Reference

Inheritance diagram for acsave::acsave_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **acsave_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **event_start_recording** ()
- void **event_stop_and_flush** ()

Private Types

- typedef std::vector< **save_var_t** ∗ > **varlist_t**

Private Member Functions

- void **process** ()

Private Attributes

- **MHAParser::bool_t bflush**
- **MHAParser::kw_t fileformat**
- **MHAParser::string_t fname**
- **MHAParser::float_t reclen**
- **MHAParser::vstring_t variables**
- **varlist_t varlist**
- std::string **algo**
- bool **b_prepared**
- bool **b_flushed**
- **MHAEvents::patchbay_t**< **acsave_t** > **patchbay**

Additional Inherited Members

5.18.1 Member Typedef Documentation

5.18.1.1 varlist_t `typedef std::vector< save_var_t*> acsave::acsave_t::varlist_t [private]`

5.18.2 Constructor & Destructor Documentation

5.18.2.1 acsave_t()

```
acsave::acsave_t::acsave_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.18.3 Member Function Documentation

5.18.3.1 prepare()

```
void acsave::acsave_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< cfg_t >** (p. 1153).

5.18.3.2 release()

```
void acsave::acsave_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< cfg_t >** (p. 1154).

5.18.3.3 process() [1/3]

```
mha_spec_t * acsave::acsave_t::process (
    mha_spec_t * s )
```

5.18.3.4 process() [2/3]

```
mha_wave_t * acsave::acsave_t::process (
    mha_wave_t * s )
```

5.18.3.5 event_start_recording()

```
void acsave::acsave_t::event_start_recording ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.18.3.6 event_stop_and_flush() `void acsave::acsave_t::event_stop_and_flush ( )`

5.18.3.7 process() [3/3] `void acsave::acsave_t::process ( ) [private]`

5.18.4 Member Data Documentation

5.18.4.1 bflush `MHAParser::bool_t acsave::acsave_t::bflush [private]`

5.18.4.2 fileformat `MHAParser::kw_t acsave::acsave_t::fileformat [private]`

5.18.4.3 fname `MHAParser::string_t acsave::acsave_t::fname [private]`

5.18.4.4 reclen `MHAParser::float_t acsave::acsave_t::reclen [private]`

5.18.4.5 variables `MHAParser::vstring_t acsave::acsave_t::variables [private]`

5.18.4.6 varlist `varlist_t acsave::acsave_t::varlist [private]`

5.18.4.7 algo `std::string acsave::acsave_t::algo [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.18.4.8 b_prepared `bool acsave::acsave_t::b_prepared [private]`

5.18.4.9 b_flushed `bool acsave::acsave_t::b_flushed [private]`

5.18.4.10 patchbay `MHAEvents::patchbay_t< acsave_t> acsave::acsave_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **acsave.cpp**

5.19 acsave::cfg_t Class Reference

Public Member Functions

- **cfg_t** (const **algo_comm_t** &iac, unsigned int imax_frames, std::vector< std::string > &var_names)
- ~**cfg_t** ()
- void **store_frame** ()
- void **flush_data** (const std::string &, unsigned int)

Private Attributes

- **algo_comm_t ac**
- unsigned int **nvars**
- **save_var_t** ** **varlist**
- unsigned int **rec_frames**
- unsigned int **max_frames**

5.19.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.19.1.1 cfg_t()

```
cfg_t::cfg_t (
    const algo_comm_t & iac,
    unsigned int imax_frames,
    std::vector< std::string > & var_names )
```

5.19.1.2 ~cfg_t()

```
cfg_t::~cfg_t ( )
```

5.19.2 Member Function Documentation

5.19.2.1 store_frame()

```
void cfg_t::store_frame ( )
```

This function is called in the processing thread.

5.19.2.2 flush_data()

```
void cfg_t::flush_data (
    const std::string & filename,
    unsigned int fmt )
```

This function is called in the configuration thread.

Parameters

filename	Output file name
fmt	Output file format

5.19.3 Member Data Documentation

5.19.3.1 ac

```
algo_comm_t acsave::cfg_t::ac [private]
```

5.19.3.2 nvars

```
unsigned int acsave::cfg_t::nvars [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.19.3.3 varlist `save_var_t** acsave::cfg_t::varlist [private]`

5.19.3.4 rec_frames `unsigned int acsave::cfg_t::rec_frames [private]`

5.19.3.5 max_frames `unsigned int acsave::cfg_t::max_frames [private]`

The documentation for this class was generated from the following file:

- **acsave.cpp**

5.20 acsave::mat4head_t Struct Reference

Public Attributes

- int32_t **t**
- int32_t **rows**
- int32_t **cols**
- int32_t **imag**
- int32_t **namelen**

5.20.1 Member Data Documentation

5.20.1.1 t `int32_t acsave::mat4head_t::t`

5.20.1.2 rows `int32_t acsave::mat4head_t::rows`

© 2005-2021 HörTech gGmbH, Oldenburg

5.20.1.3 cols `int32_t acsave::mat4head_t::cols`

5.20.1.4 imag `int32_t acsave::mat4head_t::imag`

5.20.1.5 namelen `int32_t acsave::mat4head_t::namelen`

The documentation for this struct was generated from the following file:

- **acsave.cpp**

5.21 acsave::save_var_t Class Reference

Public Member Functions

- **save_var_t** (const std::string &, int, const **algo_comm_t** &)
- ∼**save_var_t** ()
- void **store_frame** ()
- void **save_txt** (FILE ∗, unsigned int)
- void **save_mat4** (FILE ∗, unsigned int)
- void **save_m** (FILE ∗, unsigned int)

Public Attributes

- double ∗ **data**

Private Attributes

- std::string **name**
- unsigned int **nframes**
- unsigned int **ndim**
- unsigned int **maxframe**
- **algo_comm_t ac**
- unsigned int **framecnt**
- bool **b_complex**

© 2005-2021 HörTech gGmbH, Oldenburg

5.21.1 Constructor & Destructor Documentation

5.21.1.1 save_var_t()

```
acsave::save_var_t::save_var_t (
    const std::string & nm,
    int n,
    const  algo_comm_t & iac )
```

5.21.1.2 ∼save_var_t()

```
acsave::save_var_t::~save_var_t ( )
```

5.21.2 Member Function Documentation

5.21.2.1 store_frame()

```
void acsave::save_var_t::store_frame ( )
```

5.21.2.2 save_txt()

```
void acsave::save_var_t::save_txt (
    FILE * fh,
    unsigned int writeframes )
```

5.21.2.3 save_mat4()

```
void acsave::save_var_t::save_mat4 (
    FILE * fh,
    unsigned int writeframes )
```

5.21.2.4 save_m()

```
void acsave::save_var_t::save_m (
    FILE * fh,
    unsigned int writeframes )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.21.3 Member Data Documentation

5.21.3.1 data `double* acsave::save_var_t::data`

5.21.3.2 name `std::string acsave::save_var_t::name [private]`

5.21.3.3 nframes `unsigned int acsave::save_var_t::nframes [private]`

5.21.3.4 ndim `unsigned int acsave::save_var_t::ndim [private]`

5.21.3.5 maxframe `unsigned int acsave::save_var_t::maxframe [private]`

5.21.3.6 ac `algo_comm_t acsave::save_var_t::ac [private]`

5.21.3.7 framecnt `unsigned int acsave::save_var_t::framecnt [private]`

5.21.3.8 b_complex `bool acsave::save_var_t::b_complex [private]`

The documentation for this class was generated from the following file:

- **acsave.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.22 acSteer Class Reference

Inheritance diagram for acSteer:

Public Member Functions

- **acSteer** (**algo_comm_t** iac, const std::string &configured_name)
  Constructs our plugin.
- ∼**acSteer** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
  Thos method is a NOOP.
- void **prepare** (**mhaconfig_t** &)
  Plugin preparation.
- void **release** (void)

Public Attributes

- **MHAParser::string_t steerFile**
- **MHAParser::string_t acSteerName1**
- **MHAParser::string_t acSteerName2**
- **MHAParser::int_t nsteerchan**
- **MHAParser::int_t nrefmic**

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t< acSteer > patchbay**

Additional Inherited Members

5.22.1 Constructor & Destructor Documentation

5.22.1.1 acSteer()

```
acSteer::acSteer (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructs our plugin.

5.22.1.2 ~acSteer()

```
acSteer::~acSteer ( )
```

5.22.2 Member Function Documentation

5.22.2.1 process()

```
mha_spec_t * acSteer::process (
    mha_spec_t * signal )
```

Thos method is a NOOP.

5.22.2.2 prepare()

```
void acSteer::prepare (
    mhaconfig_t & signal_info ) [virtual]
```

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< acSteer_config >** (p. 1153).

5.22.2.3 release()

```
void acSteer::release (
    void ) [inline], [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< acSteer_config >** (p. 1154).

5.22.2.4 update_cfg()

```
void acSteer::update_cfg ( ) [private]
```

5.22.3 Member Data Documentation

5.22.3.1 steerFile

```
MHAParser::string_t acSteer::steerFile
```

5.22.3.2 acSteerName1

```
MHAParser::string_t acSteer::acSteerName1
```

5.22.3.3 acSteerName2

```
MHAParser::string_t acSteer::acSteerName2
```

5.22.3.4 nsteerchan

```
MHAParser::int_t acSteer::nsteerchan
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.22.3.5 nrefmic `MHAParser::int_t acSteer::nrefmic`

5.22.3.6 patchbay `MHAEvents::patchbay_t< acSteer> acSteer::patchbay [private]`

The documentation for this class was generated from the following files:

- **acSteer.h**
- **acSteer.cpp**

5.23 acSteer_config Class Reference

Public Member Functions

- **acSteer_config** (**algo_comm_t** &ac, const **mhaconfig_t** in_cfg, **acSteer** ∗ **acSteer**)
- ∼**acSteer_config** ()
- void **insert** ()

Public Attributes

- unsigned int **nchan**
- unsigned int **nfreq**
- unsigned int **nsteerchan**
- unsigned int **nrefmic**
- unsigned int **nangle**
- **MHA_AC::spectrum_t specSteer1**
- **MHA_AC::spectrum_t specSteer2**

5.23.1 Constructor & Destructor Documentation

5.23.1.1 acSteer_config()

```
acSteer_config::acSteer_config (
    algo_comm_t & ac,
    const mhaconfig_t in_cfg,
    acSteer * acSteer )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.23.1.2 ~acSteer_config() `acSteer_config::~acSteer_config ( )`

5.23.2 Member Function Documentation

5.23.2.1 insert() `void acSteer_config::insert ( )`

5.23.3 Member Data Documentation

5.23.3.1 nchan `unsigned int acSteer_config::nchan`

5.23.3.2 nfreq `unsigned int acSteer_config::nfreq`

5.23.3.3 nsteerchan `unsigned int acSteer_config::nsteerchan`

5.23.3.4 nrefmic `unsigned int acSteer_config::nrefmic`

5.23.3.5 nangle `unsigned int acSteer_config::nangle`

5.23.3.6 specSteer1 `MHA_AC::spectrum_t acSteer_config::specSteer1`

© 2005-2021 HörTech gGmbH, Oldenburg

5.23.3.7 specSteer2 `MHA_AC::spectrum_t acSteer_config::specSteer2`

The documentation for this class was generated from the following files:

- **acSteer.h**
- **acSteer.cpp**

5.24 acTransform_wave Class Reference

Inheritance diagram for acTransform_wave:

Public Member Functions

- **acTransform_wave** (**algo_comm_t** iac, const std::string &configured_name)

  Constructs our plugin.
- ∼**acTransform_wave** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

  Checks for the most recent configuration and defers processing to it.
- void **prepare** (**mhaconfig_t** &)

  Plugin preparation.
- void **release** (void)

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **MHAParser::string_t ang_name**
- **MHAParser::string_t raw_p_name**
- **MHAParser::string_t raw_p_max_name**
- **MHAParser::string_t rotated_p_name**
- **MHAParser::string_t rotated_p_max_name**
- **MHAParser::int_t numsamples**
- **MHAParser::bool_t to_from**

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t**< **acTransform_wave** > **patchbay**

Additional Inherited Members

5.24.1 Constructor & Destructor Documentation

5.24.1.1 acTransform_wave()

```
acTransform_wave::acTransform_wave (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructs our plugin.

5.24.1.2 ~acTransform_wave()

```
acTransform_wave::~acTransform_wave ( )
```

5.24.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.24.2.1 process()

```
mha_wave_t * acTransform_wave::process (
    mha_wave_t * signal )
```

Checks for the most recent configuration and defers processing to it.

5.24.2.2 prepare()

```
void acTransform_wave::prepare (
    mhaconfig_t & signal_info ) [virtual]
```

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< acTransform_wave_config >** (p. 1153).

5.24.2.3 release()

```
void acTransform_wave::release (
    void ) [inline], [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< acTransform_wave_config >** (p. 1154).

5.24.2.4 update_cfg()

```
void acTransform_wave::update_cfg ( ) [private]
```

5.24.3 Member Data Documentation

5.24.3.1 ang_name

```
MHAParser::string_t acTransform_wave::ang_name
```

5.24.3.2 raw_p_name

```
MHAParser::string_t acTransform_wave::raw_p_name
```

5.24.3.3 raw_p_max_name

```
MHAParser::string_t acTransform_wave::raw_p_max_name
```

5.24.3.4 rotated_p_name

```
MHAParser::string_t acTransform_wave::rotated_p_name
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.24.3.5 rotated_p_max_name

`MHAParser::string_t acTransform_wave::rotated_p_max_name`

5.24.3.6 numsamples

`MHAParser::int_t acTransform_wave::numsamples`

5.24.3.7 to_from

`MHAParser::bool_t acTransform_wave::to_from`

5.24.3.8 patchbay

`MHAEvents::patchbay_t< acTransform_wave> acTransform_wave::patchbay [private]`

The documentation for this class was generated from the following files:

- **acTransform_wave.h**
- **acTransform_wave.cpp**

5.25 acTransform_wave_config Class Reference

Public Member Functions

- **acTransform_wave_config** (**algo_comm_t** & **ac**, **acTransform_wave** *_transform)
- ~**acTransform_wave_config** ()
- **mha_wave_t** * **process** (**mha_wave_t** *)

Public Attributes

- **algo_comm_t** & **ac**
- std::string **ang_name**
- std::string **raw_p_name**
- std::string **raw_p_max_name**
- **MHA_AC::waveform_t rotated_p**
- **MHA_AC::int_t rotated_i**
- unsigned int **offset**
- unsigned int **resolution**
- unsigned int **to_from**

© 2005-2021 HörTech gGmbH, Oldenburg

5.25.1 Constructor & Destructor Documentation

5.25.1.1 acTransform_wave_config()

```
acTransform_wave_config::acTransform_wave_↩
config (
    algo_comm_t & ac,
    acTransform_wave * _transform )
```

5.25.1.2 ∼acTransform_wave_config()

```
acTransform_wave_config::∼acTransform_wave↩
_config ( )
```

5.25.2 Member Function Documentation

5.25.2.1 process()

```
mha_wave_t * acTransform_wave_config::process (
    mha_wave_t * wave )
```

5.25.3 Member Data Documentation

5.25.3.1 ac

```
algo_comm_t& acTransform_wave_config::ac
```

5.25.3.2 ang_name

```
std::string acTransform_wave_config::ang_name
```

5.25.3.3 raw_p_name

```
std::string acTransform_wave_config::raw_p_name
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.25.3.4 raw_p_max_name `std::string acTransform_wave_config::raw_p_max_name`

5.25.3.5 rotated_p `MHA_AC::waveform_t acTransform_wave_config::rotated_p`

5.25.3.6 rotated_i `MHA_AC::int_t acTransform_wave_config::rotated_i`

5.25.3.7 offset `unsigned int acTransform_wave_config::offset`

5.25.3.8 resolution `unsigned int acTransform_wave_config::resolution`

5.25.3.9 to_from `unsigned int acTransform_wave_config::to_from`

The documentation for this class was generated from the following files:

- **acTransform_wave.h**
- **acTransform_wave.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.26 adaptive_feedback_canceller Class Reference

Inheritance diagram for adaptive_feedback_canceller:

Public Member Functions

- **adaptive_feedback_canceller** (**algo_comm_t** **ac**, const std::string &configured_name)

  Constructs our plugin.
- ~**adaptive_feedback_canceller** ()
- **mha_wave_t** * **process** (**mha_wave_t** *)

  Checks for the most recent configuration and defers processing to it.
- void **prepare** (**mhaconfig_t** &)

  Plugin preparation.
- void **release** (void)

Public Attributes

- **MHAParser::float_t rho**
- **MHAParser::float_t c**
- **MHAParser::int_t ntaps**

© 2005-2021 HörTech gGmbH, Oldenburg

- **MHAParser::vfloat_t gains**
- **MHAParser::string_t name_e**
- **MHAParser::string_t name_f**
- **MHAParser::string_t name_lpc**
- **MHAParser::int_t lpc_order**
- **MHAParser::vint_t afc_delay**
- **MHAParser::vint_t delay_w**
- **MHAParser::vint_t delay_d**
- **MHAParser::int_t n_no_update**

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t< adaptive_feedback_canceller > patchbay**

Additional Inherited Members

5.26.1 Constructor & Destructor Documentation

5.26.1.1 adaptive_feedback_canceller()

```
adaptive_feedback_canceller::adaptive_↩
feedback_canceller (
            algo_comm_t ac,
           const std::string & configured_name )
```

Constructs our plugin.

5.26.1.2 ∼adaptive_feedback_canceller()

```
adaptive_feedback_canceller::∼adaptive_↩
feedback_canceller ( )
```

5.26.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.26.2.1 process() `mha_wave_t * adaptive_feedback_canceller::process ( mha_wave_t * signal )`

Checks for the most recent configuration and defers processing to it.

5.26.2.2 prepare() `void adaptive_feedback_canceller::prepare ( mhaconfig_t & signal_info ) [virtual]`

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< adaptive_feedback_canceller_config >** (p. 1153).

5.26.2.3 release() `void adaptive_feedback_canceller::release ( void ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t< adaptive_feedback_canceller_config >** (p. 1154).

5.26.2.4 update_cfg() `void adaptive_feedback_canceller::update_cfg ( ) [private]`

5.26.3 Member Data Documentation

5.26.3.1 rho `MHAParser::float_t adaptive_feedback_canceller::rho`

© 2005-2021 HörTech gGmbH, Oldenburg

5.26.3.2 c

`MHAParser::float_t` `adaptive_feedback_canceller::c`

5.26.3.3 ntaps

`MHAParser::int_t` `adaptive_feedback_canceller::ntaps`

5.26.3.4 gains

`MHAParser::vfloat_t` `adaptive_feedback_canceller::gains`

5.26.3.5 name_e

`MHAParser::string_t` `adaptive_feedback_canceller::name_e`

5.26.3.6 name_f

`MHAParser::string_t` `adaptive_feedback_canceller::name_f`

5.26.3.7 name_lpc

`MHAParser::string_t` `adaptive_feedback_canceller::name_lpc`

5.26.3.8 lpc_order

`MHAParser::int_t` `adaptive_feedback_canceller::lpc_order`

5.26.3.9 afc_delay

`MHAParser::vint_t` `adaptive_feedback_canceller::afc_delay`

5.26.3.10 delay_w

`MHAParser::vint_t` `adaptive_feedback_canceller::delay_w`

© 2005-2021 HörTech gGmbH, Oldenburg

5.26.3.11 delay_d `MHAParser::vint_t adaptive_feedback_canceller::delay_d`

5.26.3.12 n_no_update `MHAParser::int_t adaptive_feedback_canceller::n_no_update`

5.26.3.13 patchbay `MHAEvents::patchbay_t< adaptive_feedback_canceller> adaptive↩_feedback_canceller::patchbay [private]`

The documentation for this class was generated from the following files:

- **adaptive_feedback_canceller.h**
- **adaptive_feedback_canceller.cpp**

5.27 adaptive_feedback_canceller_config Class Reference

Public Member Functions

- **adaptive_feedback_canceller_config** (**algo_comm_t** & **ac**, const **mhaconfig_t** in_↩cfg, **adaptive_feedback_canceller** ∗afc)
- ∼**adaptive_feedback_canceller_config** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s_Y, **mha_real_t** rho, **mha_real_t** c)
- void **insert** ()

Private Attributes

- **algo_comm_t ac**
- unsigned int **ntaps**
- unsigned int **frames**
- unsigned int **channels**
- **MHA_AC::waveform_t s_E**
- **MHA_AC::waveform_t F**
- **MHASignal::waveform_t Pu**

  Power of input signal delayline.
- std::string **name_d_**
- std::string **name_lpc_**
- int **n_no_update_**
- int **no_iter**
- int **iter**
- double **PSD_val**

© 2005-2021 HörTech gGmbH, Oldenburg

- **MHASignal::waveform_t v_G**
- **MHASignal::waveform_t s_U**
- **MHASignal::delay_t s_E_afc_delay**
- **MHASignal::delay_t s_W**
- **MHASignal::ringbuffer_t s_Wflt**
- **MHASignal::delay_t s_U_delay**
- **MHASignal::ringbuffer_t s_U_delayflt**
- **MHASignal::waveform_t F_Uflt**
- **MHASignal::delay_t s_Y_delay**
- **MHASignal::ringbuffer_t s_Y_delayflt**
- **MHASignal::ringbuffer_t UbufferPrew**
- **mha_wave_t s_LPC**
- **mha_wave_t UPrew**
- **mha_wave_t YPrew**
- **mha_wave_t EPrew**
- **mha_wave_t UPrewW**
- **mha_wave_t smpl**
- **mha_wave_t * s_Usmpl**

5.27.1 Constructor & Destructor Documentation

5.27.1.1 adaptive_feedback_canceller_config()

```
adaptive_feedback_canceller_config↩
::adaptive_feedback_canceller_config (
            algo_comm_t & ac,
           const  mhaconfig_t in_cfg,
            adaptive_feedback_canceller * afc )
```

5.27.1.2 ~adaptive_feedback_canceller_config()

```
adaptive_feedback_canceller_↩
config::~adaptive_feedback_canceller_config ( )
```

5.27.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.27.2.1 process()

```
mha_wave_t * adaptive_feedback_canceller_config::process (
    mha_wave_t * s_Y,
    mha_real_t rho,
    mha_real_t c )
```

5.27.2.2 insert()

```
void adaptive_feedback_canceller_config::insert ( )
```

5.27.3 Member Data Documentation

5.27.3.1 ac

```
algo_comm_t adaptive_feedback_canceller_config::ac [private]
```

5.27.3.2 ntaps

```
unsigned int adaptive_feedback_canceller_config::ntaps [private]
```

5.27.3.3 frames

```
unsigned int adaptive_feedback_canceller_config::frames [private]
```

5.27.3.4 channels

```
unsigned int adaptive_feedback_canceller_config::channels [private]
```

5.27.3.5 s_E

```
MHA_AC::waveform_t adaptive_feedback_canceller_config::s_E [private]
```

5.27.3.6 F

```
MHA_AC::waveform_t adaptive_feedback_canceller_config::F [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.27.3.7 Pu `MHASignal::waveform_t adaptive_feedback_canceller_config::Pu [private]`

Power of input signal delayline.

5.27.3.8 name_d_ `std::string adaptive_feedback_canceller_config::name_d_ [private]`

5.27.3.9 name_lpc_ `std::string adaptive_feedback_canceller_config::name_lpc_↩ [private]`

5.27.3.10 n_no_update_ `int adaptive_feedback_canceller_config::n_no_update_↩ [private]`

5.27.3.11 no_iter `int adaptive_feedback_canceller_config::no_iter [private]`

5.27.3.12 iter `int adaptive_feedback_canceller_config::iter [private]`

5.27.3.13 PSD_val `double adaptive_feedback_canceller_config::PSD_val [private]`

5.27.3.14 v_G `MHASignal::waveform_t adaptive_feedback_canceller_config::v_G [private]`

5.27.3.15 s_U `MHASignal::waveform_t adaptive_feedback_canceller_config::s_U [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.27.3.16 s_E_afc_delay

MHASignal::delay_t adaptive_feedback_canceller_config↩ ::s_E_afc_delay [private]

5.27.3.17 s_W

MHASignal::delay_t adaptive_feedback_canceller_config::s_W [private]

5.27.3.18 s_Wflt

MHASignal::ringbuffer_t adaptive_feedback_canceller_config::s_↩ Wflt [private]

5.27.3.19 s_U_delay

MHASignal::delay_t adaptive_feedback_canceller_config::s_U_↩ delay [private]

5.27.3.20 s_U_delayflt

MHASignal::ringbuffer_t adaptive_feedback_canceller_config↩ ::s_U_delayflt [private]

5.27.3.21 F_Uflt

MHASignal::waveform_t adaptive_feedback_canceller_config::F_Uflt [private]

5.27.3.22 s_Y_delay

MHASignal::delay_t adaptive_feedback_canceller_config::s_Y_↩ delay [private]

5.27.3.23 s_Y_delayflt

MHASignal::ringbuffer_t adaptive_feedback_canceller_config↩ ::s_Y_delayflt [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.27.3.24 UbufferPrew

`MHASignal::ringbuffer_t adaptive_feedback_canceller_config::UbufferPrew [private]`

5.27.3.25 s_LPC

`mha_wave_t adaptive_feedback_canceller_config::s_LPC [private]`

5.27.3.26 UPrew

`mha_wave_t adaptive_feedback_canceller_config::UPrew [private]`

5.27.3.27 YPrew

`mha_wave_t adaptive_feedback_canceller_config::YPrew [private]`

5.27.3.28 EPrew

`mha_wave_t adaptive_feedback_canceller_config::EPrew [private]`

5.27.3.29 UPrewW

`mha_wave_t adaptive_feedback_canceller_config::UPrewW [private]`

5.27.3.30 smpl

`mha_wave_t adaptive_feedback_canceller_config::smpl [private]`

5.27.3.31 s_Usmpl

`mha_wave_t* adaptive_feedback_canceller_config::s_Usmpl [private]`

The documentation for this class was generated from the following files:

- **adaptive_feedback_canceller.h**
- **adaptive_feedback_canceller.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.28 addsndfile::addsndfile_if_t Class Reference

Inheritance diagram for addsndfile::addsndfile_if_t:

Public Member Functions

- **addsndfile_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

Private Member Functions

- void **update** ()
- void **change_mode** ()
- void **set_level** ()
- void **scan_dir** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::string_t filename**
- **MHAParser::string_t path**
- **MHAParser::bool_t loop**
- **MHAParser::float_t level**
- **MHAParser::kw_t levelmode**
- **MHAParser::kw_t resamplingmode**
- **MHAParser::vint_t channels**
- **MHAParser::kw_t mode**
- **MHAParser::float_t ramplen**
- **MHAParser::int_t startpos**
- **MHAParser::vint_mon_t mapping**
- **MHAParser::int_mon_t numchannels**
- **MHAParser::int_mon_t mhachannels**
- **MHAParser::int_mon_t active**
- **MHAParser::string_t search_pattern**
- **MHAParser::vstring_mon_t search_result**
- unsigned int **uint_mode**
- **MHAEvents::patchbay_t**< **addsndfile_if_t** > **patchbay**

Additional Inherited Members

5.28.1 Constructor & Destructor Documentation

5.28.1.1 addsndfile_if_t()

```
addsndfile::addsndfile_if_t::addsndfile_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.28.2 Member Function Documentation

5.28.2.1 process()

```
mha_wave_t * addsndfile::addsndfile_if_t::process (
    mha_wave_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.28.2.2 prepare() `void addsndfile::addsndfile_if_t::prepare ( mhaconfig_t & tf ) [virtual]`

Implements **MHAPlugin::plugin_t< runtime_cfg_t >** (p. 1153).

5.28.2.3 release() `void addsndfile::addsndfile_if_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t< runtime_cfg_t >** (p. 1154).

5.28.2.4 update() `void addsndfile::addsndfile_if_t::update ( ) [private]`

5.28.2.5 change_mode() `void addsndfile::addsndfile_if_t::change_mode ( ) [private]`

5.28.2.6 set_level() `void addsndfile::addsndfile_if_t::set_level ( ) [private]`

5.28.2.7 scan_dir() `void addsndfile::addsndfile_if_t::scan_dir ( ) [private]`

5.28.3 Member Data Documentation

5.28.3.1 filename `MHAParser::string_t addsndfile::addsndfile_if_t::filename [private]`

5.28.3.2 path `MHAParser::string_t addsndfile::addsndfile_if_t::path [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.28.3.3 loop

`MHAParser::bool_t addsndfile::addsndfile_if_t::loop [private]`

5.28.3.4 level

`MHAParser::float_t addsndfile::addsndfile_if_t::level [private]`

5.28.3.5 levelmode

`MHAParser::kw_t addsndfile::addsndfile_if_t::levelmode [private]`

5.28.3.6 resamplingmode

`MHAParser::kw_t addsndfile::addsndfile_if_t::resamplingmode [private]`

5.28.3.7 channels

`MHAParser::vint_t addsndfile::addsndfile_if_t::channels [private]`

5.28.3.8 mode

`MHAParser::kw_t addsndfile::addsndfile_if_t::mode [private]`

5.28.3.9 ramplen

`MHAParser::float_t addsndfile::addsndfile_if_t::ramplen [private]`

5.28.3.10 startpos

`MHAParser::int_t addsndfile::addsndfile_if_t::startpos [private]`

5.28.3.11 mapping

`MHAParser::vint_mon_t addsndfile::addsndfile_if_t::mapping [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.28.3.12 numchannels

MHAParser::int_mon_t addsndfile::addsndfile_if_t::numchannels [private]

5.28.3.13 mhachannels

MHAParser::int_mon_t addsndfile::addsndfile_if_t::mhachannels [private]

5.28.3.14 active

MHAParser::int_mon_t addsndfile::addsndfile_if_t::active [private]

5.28.3.15 search_pattern

MHAParser::string_t addsndfile::addsndfile_if_t::search↩_pattern [private]

5.28.3.16 search_result

MHAParser::vstring_mon_t addsndfile::addsndfile_if_t↩::search_result [private]

5.28.3.17 uint_mode

unsigned int addsndfile::addsndfile_if_t::uint_mode [private]

5.28.3.18 patchbay

MHAEvents::patchbay_t< addsndfile_if_t> addsndfile::addsndfile↩_if_t::patchbay [private]

The documentation for this class was generated from the following file:

- **addsndfile.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.29 addsndfile::level_adapt_t Class Reference

Inheritance diagram for addsndfile::level_adapt_t:

Public Member Functions

- **level_adapt_t** (**mhaconfig_t** cf, **mha_real_t** adapt_len, **mha_real_t** l_new_, **mha_real_t** l_old_)
- void **update_frame** ()
- **mha_real_t** **get_level** () const
- bool **can_update** () const

Private Attributes

- unsigned int **ilen**
- unsigned int **pos**
- **MHAWindow::fun_t** **wnd**
- **mha_real_t** **l_new**
- **mha_real_t** **l_old**

Additional Inherited Members

5.29.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.29.1.1 level_adapt_t()

```
addsndfile::level_adapt_t::level_adapt_t (
    mhaconfig_t cf,
    mha_real_t adapt_len,
    mha_real_t l_new_,
    mha_real_t l_old_ )
```

5.29.2 Member Function Documentation

5.29.2.1 update_frame()

```
void addsndfile::level_adapt_t::update_frame ( )
```

5.29.2.2 get_level()

```
mha_real_t addsndfile::level_adapt_t::get_level ( ) const [inline]
```

5.29.2.3 can_update()

```
bool addsndfile::level_adapt_t::can_update ( ) const [inline]
```

5.29.3 Member Data Documentation

5.29.3.1 ilen

```
unsigned int addsndfile::level_adapt_t::ilen [private]
```

5.29.3.2 pos

```
unsigned int addsndfile::level_adapt_t::pos [private]
```

5.29.3.3 wnd

```
MHAWindow::fun_t addsndfile::level_adapt_t::wnd [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.29.3.4 l_new `mha_real_t addsndfile::level_adapt_t::l_new [private]`

5.29.3.5 l_old `mha_real_t addsndfile::level_adapt_t::l_old [private]`

The documentation for this class was generated from the following file:

- **addsndfile.cpp**

5.30 addsndfile::resampled_soundfile_t Class Reference

Reads sound from file and resamples it if necessary and wanted.

Inheritance diagram for addsndfile::resampled_soundfile_t:

Public Member Functions

- **resampled_soundfile_t** (const std::string &name, float mha_sampling_rate, **addsndfile↩ _resampling_mode_t** resampling_mode)

  Reads sound from file and resamples if necessary and wanted.

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.30.1 Detailed Description

Reads sound from file and resamples it if necessary and wanted.

Sound data can then be used by addsndfile.

5.30.2 Constructor & Destructor Documentation

5.30.2.1 resampled_soundfile_t()

```
addsndfile::resampled_soundfile_t::resampled_↩
soundfile_t (
            const std::string & name,
            float mha_sampling_rate,
             addsndfile_resampling_mode_t resampling_mode )
```

Reads sound from file and resamples if necessary and wanted.

If the sound file does not specify a sampling rate, then the sound data is always used without resampling.

Parameters

name	Sound file name
mha_sampling_rate	The sampling rate of the MHA signal processing at the point of the addsndfile plugin
resampling_mode	DONT_RESAMPLE_STRICT: Do not resample, just use the samples from the sound file at the current sample rate, even if the sample rate of the sound file differs. DONT_RESAMPLE_PERMISSIVE: Do not resample, if the sample rate of the MHA differs from the sample rate of the sound file, raise an error. DO_RESAMPLE: Resample.

Exceptions

MHA_Error *(p. 764)*	If the sampling rate of the file does not match the sampling rate of the MHA and DONT_RESAMPLE_STRICT was requested. If resampling failed (e.g. due to non-rational quotient of MHA sampling rate and sound file sampling rate).

The documentation for this class was generated from the following file:

© 2005-2021 HörTech gGmbH, Oldenburg

- **addsndfile.cpp**

5.31 addsndfile::sndfile_t Class Reference

Inheritance diagram for addsndfile::sndfile_t:

Public Member Functions

- **sndfile_t** (const std::string &name, bool loop, unsigned int level_mode, std::vector< int > channels_, unsigned int nchannels, std::vector< int > &mapping, int &numchannels, unsigned int startpos, float mha_sampling_rate, **addsndfile_resampling_mode_t** resampling_mode)

Additional Inherited Members

5.31.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.31.1.1 sndfile_t()

```
addsndfile::sndfile_t::sndfile_t (
    const std::string & name,
    bool loop,
    unsigned int level_mode,
    std::vector< int > channels_,
    unsigned int nchannels,
    std::vector< int > & mapping,
    int & numchannels,
    unsigned int startpos,
    float mha_sampling_rate,
    addsndfile_resampling_mode_t resampling_mode )
```

The documentation for this class was generated from the following file:

- **addsndfile.cpp**

5.32 addsndfile::waveform_proxy_t Class Reference

Class helps to specify which instance of MHASignal_waveform_t parent instance is meant in **resampled_soundfile_t** (p. 257).

Inheritance diagram for addsndfile::waveform_proxy_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **waveform_proxy_t** (unsigned frames, unsigned **channels**)

Additional Inherited Members

5.32.1 Detailed Description

Class helps to specify which instance of MHASignal_waveform_t parent instance is meant in **resampled_soundfile_t** (p. 257).

5.32.2 Constructor & Destructor Documentation

5.32.2.1 waveform_proxy_t()

```
addsndfile::waveform_proxy_t::waveform_proxy_t (
    unsigned frames,
    unsigned channels ) [inline]
```

The documentation for this class was generated from the following file:

- **addsndfile.cpp**

5.33 ADM::ADM< F > Class Template Reference

Adaptive differential microphone, working for speech frequency range.

Public Member Functions

- **ADM** (F fs, F dist, unsigned lp_order, const F *lp_alphas, unsigned decomb_order, const F *decomb_alphas, F tau_beta=F(50e-3), F mu_beta=F(1e-4))

  Create Adaptive Differential Microphone.
- F **process** (const F &front, const F &back, const F &external_beta=F(-1), bool update↩_beta=true)

  ADM *(p. 261) processes one frame.*
- F **beta** () const

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **Delay**< F > **m_delay_front**
- **Delay**< F > **m_delay_back**
- **Linearphase_FIR**< F > **m_lp_bf**
- **Linearphase_FIR**< F > **m_lp_result**
- **Linearphase_FIR**< F > **m_decomb**
- F **m_beta**
- F **m_mu_beta**
- F **m_powerfilter_coeff**
- F **m_powerfilter_norm**
- F **m_powerfilter_state**

5.33.1 Detailed Description

template<**class F**>
class ADM::ADM< **F** >

Adaptive differential microphone, working for speech frequency range.

5.33.2 Constructor & Destructor Documentation

5.33.2.1 ADM() template<class F >

```
ADM::ADM< F >::  ADM (
            F fs,
            F dist,
            unsigned lp_order,
            const F * lp_alphas,
            unsigned decomb_order,
            const F * decomb_alphas,
            F tau_beta = F(50e-3),
            F mu_beta = F(1e-4) )
```

Create Adaptive Differential Microphone.

Parameters

fs	Sampling rate / Hz
dist	Distance between physical microphones / m
lp_order	Filter order of FIR lowpass filter used for adaptation

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

lp_alphas	Pointer to array of alpha coefficients for the lowpass filter used for adaptation. Since this class uses linear phase FIR filters only, only the first half (order/2 + 1) of the coefficients will be read (coefficients for linear-phase FIR filters are symmetric).
decomb_order	Filter order of FIR compensation filter (compensates for comb filter characteristic). decomb_order <= 1 deactivates filter.
decomb_alphas	Pointer to array of alpha coefficients for the compensation filter used to compensate for the comb filter characteristic. Since this class uses linear phase FIR filters only, only the first half (order/2 + 1)of the coefficients will be read (coefficients for linear-phase FIR filters are symmetric).
tau_beta	Time constant of the lowpass filter used for averaging the power of the output signal
mu_beta	adaptation speed

5.33.3 Member Function Documentation

5.33.3.1 process() `template<class F >`

```
F  ADM::ADM< F >::process (
            const F & front,
            const F & back,
            const F & external_beta = F(-1),
            bool update_beta = true )  [inline]
```

ADM (p. 261) processes one frame.

Parameters

front	The current front input signal sample
back	The current rear input signal sample
external_beta	If >= 0, this is used as the "beta" parameter for direction to filter out. Else, the beta parameter is adapted to filtered out a direction so that best reduction of signal intensity from the back hemisphere is achieved.
update_beta	Perform the beta adaptation step?

Returns

The computed output sample

© 2005-2021 HörTech gGmbH, Oldenburg

5.33.3.2 beta() template<class F >

F **ADM::ADM**< F >::beta ( ) const [inline]

5.33.4 Member Data Documentation

5.33.4.1 m_delay_front template<class F >

Delay<F> **ADM::ADM**< F >::m_delay_front [private]

5.33.4.2 m_delay_back template<class F >

Delay<F> **ADM::ADM**< F >::m_delay_back [private]

5.33.4.3 m_lp_bf template<class F >

Linearphase_FIR<F> **ADM::ADM**< F >::m_lp_bf [private]

5.33.4.4 m_lp_result template<class F >

Linearphase_FIR<F> **ADM::ADM**< F >::m_lp_result [private]

5.33.4.5 m_decomb template<class F >

Linearphase_FIR<F> **ADM::ADM**< F >::m_decomb [private]

5.33.4.6 m_beta template<class F >

F **ADM::ADM**< F >::m_beta [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.33.4.7 m_mu_beta template<class F >

F ADM::ADM< F >::m_mu_beta [private]

5.33.4.8 m_powerfilter_coeff template<class F >

F ADM::ADM< F >::m_powerfilter_coeff [private]

5.33.4.9 m_powerfilter_norm template<class F >

F ADM::ADM< F >::m_powerfilter_norm [private]

5.33.4.10 m_powerfilter_state template<class F >

F ADM::ADM< F >::m_powerfilter_state [private]

The documentation for this class was generated from the following file:

- **adm.hh**

5.34 ADM::Delay< F > Class Template Reference

A delay-line class.

Public Member Functions

- **Delay** (F samples, F f_design, F fs)

  Create a signal delay object.
- ~**Delay** ()
- F **process** (const F &in_sample)

  Apply delay to signal.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- unsigned **m_fullsamples**
  Integer part of delay.
- F **m_coeff**
  coefficient for 1st order IIR lowpass filter which does the subsample delay
- F **m_norm**
  normalization for the IIR subsample delay filter
- F * **m_state**
  Ringbuffer: Delayline.
- unsigned **m_now_in**
  current position for inserting new samples into m_state ringbuffer

5.34.1 Detailed Description

template<class F>
class ADM::Delay< F >

A delay-line class.

It can delay samples in a single audio channel. It stores samples while they are delayed until they have reached their target delay. It can also do subsample-delays for a limited frequency range below fs/4.

5.34.2 Constructor & Destructor Documentation

5.34.2.1 Delay() template<class F >

```
ADM::Delay< F >::  Delay (
            F samples,
            F f_design,
            F fs )
```

Create a signal delay object.

Parameters

samples	number of samples to delay (may be non-integer)
f_design	design frequency (in Hz). Subsampledelay is exact for this frequency and approximate for different frequencies
fs	sampling frequency (in Hz).

© 2005-2021 HörTech gGmbH, Oldenburg

5.34.2.2 ~Delay() template<class F >

ADM::Delay< F >::~ **Delay**

5.34.3 Member Function Documentation

5.34.3.1 process() template<class F >

F **ADM::Delay**< F >::process (
const F & *in_sample*) [inline]

Apply delay to signal.

Whenever a new audio sample enters the delay line, a previous audio sample, now delayed, is returned by this method. Sub-sample-delays are implemented by applying a first-order recursive lowpass filter. This method needs to be called repeatedly, once for each incoming audio sample in correct order for a block of audio with multiple samples (oldest first, newest last).

Parameters

in_sample	The current input signal sample

Returns

The output sample, which is one of the previously received input samples except for the sub-sample delay.

5.34.4 Member Data Documentation

5.34.4.1 m_fullsamples template<class F >

unsigned **ADM::Delay**< F >::m_fullsamples [private]

Integer part of delay.

© 2005-2021 HörTech gGmbH, Oldenburg

5.34.4.2 m_coeff `template<class F >`

`F ADM::Delay< F >::m_coeff [private]`

coefficient for 1st order IIR lowpass filter which does the subsample delay

5.34.4.3 m_norm `template<class F >`

`F ADM::Delay< F >::m_norm [private]`

normalization for the IIR subsample delay filter

5.34.4.4 m_state `template<class F >`

`F* ADM::Delay< F >::m_state [private]`

Ringbuffer: Delayline.

5.34.4.5 m_now_in `template<class F >`

`unsigned ADM::Delay< F >::m_now_in [private]`

current position for inserting new samples into m_state ringbuffer

The documentation for this class was generated from the following file:

- **adm.hh**

5.35 ADM::Linearphase_FIR< F > Class Template Reference

An efficient linear-phase fir filter implementation.

Public Member Functions

- **Linearphase_FIR** (unsigned order, const F *alphas)
  Create linear-phase FIR filter.
- ~**Linearphase_FIR** ()
- F **process** (const F &in_sample)
  Filter one sample with this linear-phase FIR filter.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- unsigned **m_order**
  The filter order of this linear-phase FIR filter.
- F * **m_alphas**
  FIR filter coefficients.
- F * **m_output**
  Ringbuffer for building future output.
- unsigned **m_now**
  current start of ringbuffer

5.35.1 Detailed Description

template<class F>
class ADM::Linearphase_FIR< F >

An efficient linear-phase fir filter implementation.

5.35.2 Constructor & Destructor Documentation

5.35.2.1 Linearphase_FIR() template<class F >

```
ADM::Linearphase_FIR< F >::  Linearphase_FIR (
            unsigned order,
            const F * alphas )
```

Create linear-phase FIR filter.

Parameters

order	filter order of this FIR filter. restriction: must be even.
alphas	pointer to Array of alpha coefficients. Since this class is for linear phase FIR filters only, only (order / 2 + 1) coefficients will be read. (Coefficients for linear-phase FIR filters are symmetric.)

5.35.2.2 ~Linearphase_FIR() template<class F >

```
ADM::Linearphase_FIR< F >::~ Linearphase_FIR
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.35.3 Member Function Documentation

5.35.3.1 process()

```
template<class F >
F ADM::Linearphase_FIR< F >::process (
        const F & in_sample ) [inline]
```

Filter one sample with this linear-phase FIR filter.

Parameters

in_sample	the current input sample

Returns

the computed output sample

5.35.4 Member Data Documentation

5.35.4.1 m_order

```
template<class F >
unsigned ADM::Linearphase_FIR< F >::m_order [private]
```

The filter order of this linear-phase FIR filter.

5.35.4.2 m_alphas

```
template<class F >
F* ADM::Linearphase_FIR< F >::m_alphas [private]
```

FIR filter coefficients.

Only m_order / 2 + 1 coefficients need to be stored since coefficients of linear-phase FIR filters are symmetric

5.35.4.3 m_output

```
template<class F >
F* ADM::Linearphase_FIR< F >::m_output [private]
```

Ringbuffer for building future output.

© 2005-2021 HörTech gGmbH, Oldenburg

5.35.4.4 m_now `template<class F >`
`unsigned` **ADM::Linearphase_FIR**`< F >::m_now [private]`

current start of ringbuffer

The documentation for this class was generated from the following file:

- **adm.hh**

5.36 adm_if_t Class Reference

Inheritance diagram for adm_if_t:

Public Member Functions

- **adm_if_t** (**algo_comm_t** **ac**, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗in)
- virtual void **prepare** (**mhaconfig_t** &)
- virtual void **release** ()

Private Member Functions

- void **update** ()
- bool **is_prepared** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHASignal::waveform_t** $*$ **out**
- **MHAParser::vint_t front_channels**
- **MHAParser::vint_t rear_channels**
- **MHAParser::vfloat_t distances**
- **MHAParser::int_t lp_order**
- **MHAParser::int_t decomb_order**
- **MHAParser::int_t bypass**
- **MHAParser::float_t beta**
- **MHAParser::vfloat_t mu_beta**
- **MHAParser::vfloat_t tau_beta**
- **MHAParser::vfloat_mon_t coeff_lp**
- **MHAParser::vfloat_mon_t coeff_decomb**
- **MHAParser::int_t adaptation_ratio**
- unsigned **input_channels**
- int **framecnt**
- **mha_real_t srate**
- **MHAEvents::patchbay_t**$<$ **adm_if_t** $>$ **patchbay**

Additional Inherited Members

5.36.1 Constructor & Destructor Documentation

5.36.1.1 adm_if_t()

```
adm_if_t::adm_if_t (
    algo_comm_t ac,
    const std::string & configured_name )
```

5.36.2 Member Function Documentation

5.36.2.1 process()

```
mha_wave_t * adm_if_t::process (
    mha_wave_t * in )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.36.2.2 prepare() `void adm_if_t::prepare ( mhaconfig_t & cfg ) [virtual]`

Implements **MHAPlugin::plugin_t< adm_rtconfig_t >** (p. 1153).

5.36.2.3 release() `void adm_if_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t< adm_rtconfig_t >** (p. 1154).

5.36.2.4 update() `void adm_if_t::update ( ) [private]`

5.36.2.5 is_prepared() `bool adm_if_t::is_prepared ( ) [inline], [private]`

5.36.3 Member Data Documentation

5.36.3.1 out `MHASignal::waveform_t* adm_if_t::out [private]`

5.36.3.2 front_channels `MHAParser::vint_t adm_if_t::front_channels [private]`

5.36.3.3 rear_channels `MHAParser::vint_t adm_if_t::rear_channels [private]`

5.36.3.4 distances `MHAParser::vfloat_t adm_if_t::distances [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.36.3.5 lp_order `MHAParser::int_t` adm_if_t::lp_order [private]

5.36.3.6 decomb_order `MHAParser::int_t` adm_if_t::decomb_order [private]

5.36.3.7 bypass `MHAParser::int_t` adm_if_t::bypass [private]

5.36.3.8 beta `MHAParser::float_t` adm_if_t::beta [private]

5.36.3.9 mu_beta `MHAParser::vfloat_t` adm_if_t::mu_beta [private]

5.36.3.10 tau_beta `MHAParser::vfloat_t` adm_if_t::tau_beta [private]

5.36.3.11 coeff_lp `MHAParser::vfloat_mon_t` adm_if_t::coeff_lp [private]

5.36.3.12 coeff_decomb `MHAParser::vfloat_mon_t` adm_if_t::coeff_decomb [private]

5.36.3.13 adaptation_ratio `MHAParser::int_t` adm_if_t::adaptation_ratio [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.36.3.14 input_channels `unsigned adm_if_t::input_channels [private]`

5.36.3.15 framecnt `int adm_if_t::framecnt [private]`

5.36.3.16 srate `mha_real_t adm_if_t::srate [private]`

5.36.3.17 patchbay `MHAEvents::patchbay_t< adm_if_t> adm_if_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **adm.cpp**

5.37 adm_rtconfig_t Class Reference

Public Types

- typedef **ADM::ADM**< **mha_real_t** > **adm_t**

Public Member Functions

- **adm_rtconfig_t** (unsigned nchannels_in, unsigned nchannels_out, int adaptation_↩ ratio_, const std::vector< int > & **front_channels**, const std::vector< int > & **rear_↩ channels**, const **mha_real_t** fs, const std::vector< **mha_real_t** > &distances, const int lp_order, const int decomb_order, const std::vector< **mha_real_t** > &tau_beta, const std::vector< **mha_real_t** > &mu_beta)

  Construct new ADMs.
- virtual ~**adm_rtconfig_t** ()
- size_t **num_adms** () const
- **adm_t** & **adm** (unsigned index)

  Returns adm object number index.
- int **front_channel** (unsigned index) const

  Returns index of front channel for adm number index.
- int **rear_channel** (unsigned index) const

  Returns index of rear channel for adm number index.
- int **get_adaptation_ratio** () const

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **check_index** (unsigned index) const
  Index checking for all internal arrays.

Private Attributes

- std::vector< int > **front_channels**
  Indices of channels containing the signals from the front microphones.
- std::vector< int > **rear_channels**
  Indices of channels containing the signals from the rear microphones.
- int **adaptation_ratio**
  Prescale.
- **MHASignal::waveform_t** * **lp_coeffs**
  Lowpass filter coefficients.
- std::vector< **MHASignal::waveform_t** * > **decomb_coeffs**
  Decomb-Filter coefficients.
- std::vector< **adm_t** * > **adms**
  ADMs.

5.37.1 Member Typedef Documentation

5.37.1.1 adm_t `typedef` **ADM::ADM< mha_real_t> adm_rtconfig_t::adm_t**

5.37.2 Constructor & Destructor Documentation

5.37.2.1 adm_rtconfig_t()

```
adm_rtconfig_t::adm_rtconfig_t (
    unsigned nchannels_in,
    unsigned nchannels_out,
    int adaptation_ratio_,
    const std::vector< int > & front_channels,
    const std::vector< int > & rear_channels,
    const mha_real_t fs,
    const std::vector< mha_real_t > & distances,
    const int lp_order,
    const int decomb_order,
    const std::vector< mha_real_t > & tau_beta,
    const std::vector< mha_real_t > & mu_beta )
```

Construct new ADMs.

Used when configuration changes.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

nchannels_in	Number of input channels
nchannels_out	Number of output channels
*adaptation_↩ ratio_*	Update beta every adaptation_ratio frames
front_channels	Parser's front_channels setting
rear_channels	Parser's front_channels setting
fs	Sampling rate / Hz
distances	Distances between microphones / m
lp_order	Filter order of FIR lowpass filter for adaptation
decomb_order	Filter order of FIR compensation filter (compensates for comb filter characteristic)
tau_beta	Time constants of the lowpass filter used for averaging the power of the output signal used for adaptation
mu_beta	Adaptation step sizes

5.37.2.2 ~adm_rtconfig_t() `adm_rtconfig_t::~adm_rtconfig_t ( ) [virtual]`

5.37.3 Member Function Documentation

5.37.3.1 check_index() `void adm_rtconfig_t::check_index ( unsigned index ) const [inline], [private]`

Index checking for all internal arrays.

Exceptions

MHA_Error *(p. 764)*	if index out of range.

5.37.3.2 num_adms() `size_t adm_rtconfig_t::num_adms ( ) const [inline]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.37.3.3 adm() `adm_t& adm_rtconfig_t::adm ( unsigned index ) [inline]`

Returns adm object number index.

5.37.3.4 front_channel() `int adm_rtconfig_t::front_channel ( unsigned index ) const [inline]`

Returns index of front channel for adm number index.

5.37.3.5 rear_channel() `int adm_rtconfig_t::rear_channel ( unsigned index ) const [inline]`

Returns index of rear channel for adm number index.

5.37.3.6 get_adaptation_ratio() `int adm_rtconfig_t::get_adaptation_ratio ( ) const [inline]`

5.37.4 Member Data Documentation

5.37.4.1 front_channels `std::vector<int> adm_rtconfig_t::front_channels [private]`

Indices of channels containing the signals from the front microphones.

5.37.4.2 rear_channels `std::vector<int> adm_rtconfig_t::rear_channels [private]`

Indices of channels containing the signals from the rear microphones.

© 2005-2021 HörTech gGmbH, Oldenburg

5.37.4.3 adaptation_ratio `int adm_rtconfig_t::adaptation_ratio [private]`

Prescale.

5.37.4.4 lp_coeffs `MHASignal::waveform_t* adm_rtconfig_t::lp_coeffs [private]`

Lowpass filter coefficients.

5.37.4.5 decomb_coeffs `std::vector< MHASignal::waveform_t*> adm_rtconfig_t↩ ::decomb_coeffs [private]`

Decomb-Filter coefficients.

5.37.4.6 adms `std::vector< adm_t *> adm_rtconfig_t::adms [private]`

ADMs.

The documentation for this class was generated from the following file:

- **adm.cpp**

5.38 algo_comm_t Struct Reference

A reference handle for algorithm communication variables.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- void * **handle**
  AC variable control handle.
- int(* **insert_var**)(void *, const char *, **comm_var_t**)
  Register an AC variable.
- int(* **insert_var_int**)(void *, const char *, int *)
  Register an int as an AC variable.
- int(* **insert_var_float**)(void *, const char *, float *)
  Register a float as an AC variable.
- int(* **insert_var_double**)(void *, const char *, double *)
- int(* **remove_var**)(void *, const char *)
  Remove an AC variable.
- int(* **remove_ref**)(void *, void *)
  Remove all AC variable which refer to address.
- int(* **is_var**)(void *, const char *)
  Test if an AC variable exists.
- int(* **get_var**)(void *, const char *, **comm_var_t** *)
  Get the variable handle of an AC variable.
- int(* **get_var_int**)(void *, const char *, int *)
  Get the value of an int AC variable.
- int(* **get_var_float**)(void *, const char *, float *)
  Get the value of a float AC variable.
- int(* **get_var_double**)(void *, const char *, double *)
- int(* **get_entries**)(void *, char *, unsigned int)
  Return a space separated list of all variable names.
- const char *(* **get_error**)(int)
  Convert AC error codes into human readable error messages.

5.38.1 Detailed Description

A reference handle for algorithm communication variables.

This structure contains a coontrol handle and a set of function pointers for sharing variables within one processing chain. See also section **Communication between algorithms** (p. 23).

5.38.2 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.38.2.1 handle `algo_comm_t::handle`

AC variable control handle.

5.38.2.2 insert_var `algo_comm_t::insert_var`

Register an AC variable.

This function can register a variable to be shared within one chain. If a variable of this name exists it will be overwritten.

Parameters

h	AC handle
n	name of variable. May not be empty. Must not contain space character. The name is copied, therefore it is allowed that the char array pointed to gets invalid after return.
v	variable handle of type **comm_var_t** (p. 359)

Returns

Error code or zero on success

5.38.2.3 insert_var_int `algo_comm_t::insert_var_int`

Register an int as an AC variable.

This function can register an int variable to be shared with other algorithms. It behaves similar to ac.insert_var.

Parameters

h	AC handle
n	name of variable
v	pointer on the variable

Returns

Error code or zero on success

© 2005-2021 HörTech gGmbH, Oldenburg

5.38.2.4 insert_var_float `algo_comm_t::insert_var_float`

Register a float as an AC variable.

This function can register a float variable to be shared with other algorithms. It behaves similar to ac.insert_var.

Parameters

h	AC handle
n	name of variable
v	pointer on the variable

Returns

Error code or zero on success

5.38.2.5 insert_var_double `int(* algo_comm_t::insert_var_double) (void *, const char *, double *)`

5.38.2.6 remove_var `algo_comm_t::remove_var`

Remove an AC variable.

Remove (unregister) an AC variable. After calling this function, the variable is not available to ac.is_var or ac.get_var. The data pointer is not affected.

Parameters

h	AC handle
n	name of variable to be removed

Returns

Error code or zero on success

© 2005-2021 HörTech gGmbH, Oldenburg

5.38.2.7 remove_ref `algo_comm_t::remove_ref`

Remove all AC variable which refer to address.

This function removes all AC variables whos data field points to the given address.

Parameters

h	AC handle
p	address which should not be referred to any more

Returns

Error code or zero on success

5.38.2.8 is_var `algo_comm_t::is_var`

Test if an AC variable exists.

This function tests if an AC variable of a given name exists. Use ac.get_var to get information about the variables type and dimension.

Parameters

h	AC handle
n	name of variable

Returns

1 if the variable exists, 0 otherwise

5.38.2.9 get_var `algo_comm_t::get_var`

Get the variable handle of an AC variable.

This function returns the variable handle **comm_var_t** (p. 359) of a variable of the given name. If no variable of that name exists, an error code is returned.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

h	AC handle
n	name of variable
v	pointer to a AC variable object

Returns

Error code or zero on success

5.38.2.10 get_var_int `algo_comm_t::get_var_int`

Get the value of an int AC variable.

This function returns the value of an int AC variable of the given name. If no variable exists, the variable type is mismatching or more than one entry is registered, a corresponding error code is returned. This is a special version of ac.get_var.

Parameters

h	AC handle
n	name of variable
v	pointer on an int variable to store the result

Returns

Error code or zero on success

5.38.2.11 get_var_float `algo_comm_t::get_var_float`

Get the value of a float AC variable.

This function returns the value of a float AC variable of the given name. If no variable exists, the variable type is mismatching or more than one entry is registered, a corresponding error code is returned. This is a special version of ac.get_var.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

h	AC handle
n	name of variable
v	pointer on a float variable to store the result

Returns

Error code or zero on success

5.38.2.12 get_var_double `int(* algo_comm_t::get_var_double) (void *, const char *, double *)`

5.38.2.13 get_entries `algo_comm_t::get_entries`

Return a space separated list of all variable names.

This function returns the names of all registered variables, separated by a single space.

Parameters

h	AC handle

Return values

ret	Character buffer for return value

Parameters

len	length of character buffer

Returns

Error code or zero on success. -1: invalid ac handle. -3: not enough room in character buffer to store all variable names.

© 2005-2021 HörTech gGmbH, Oldenburg

5.38.2.14 get_error `algo_comm_t::get_error`

Convert AC error codes into human readable error messages.

Parameters

e	Error code

Returns

Error message

The documentation for this struct was generated from the following files:

- **mha.hh**
- **mha_algo_comm.cpp**

5.39 alsa_base_t Class Reference

Inheritance diagram for alsa_base_t:

Public Member Functions

- **alsa_base_t** ()
- virtual ∼**alsa_base_t** ()=default
- virtual void **start** ()=0

  start puts alsa device in usable state
- virtual void **stop** ()=0

  stop informs alsa device that we do not need any more samples / will not provide any more samples
- virtual bool **read** (**mha_wave_t** ∗∗)=0

  read audio samples from the device into an internal ***mha_wave_t*** *(p. 840) buffer, then update the pointer given as parameter to point to the internal structure.*
- virtual bool **write** (**mha_wave_t** ∗)=0

  write audio samples from the given waveform buffer to the sound device.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- snd_pcm_t * **pcm**
  The underlying alsa handle to this sound card.

5.39.1 Constructor & Destructor Documentation

5.39.1.1 alsa_base_t() `alsa_base_t::alsa_base_t ( ) [inline]`

5.39.1.2 ~alsa_base_t() `virtual alsa_base_t::~alsa_base_t ( ) [virtual], [default]`

5.39.2 Member Function Documentation

5.39.2.1 start() `virtual void alsa_base_t::start ( ) [pure virtual]`

start puts alsa device in usable state

Implemented in **alsa_t**< **T** > (p. 293).

5.39.2.2 stop() `virtual void alsa_base_t::stop ( ) [pure virtual]`

stop informs alsa device that we do not need any more samples / will not provide any more samples

Implemented in **alsa_t**< **T** > (p. 293).

© 2005-2021 HörTech gGmbH, Oldenburg

5.39.2.3 read() `virtual bool alsa_base_t::read ( mha_wave_t ** ) [pure virtual]`

read audio samples from the device into an internal **mha_wave_t** (p. 840) buffer, then update the pointer given as parameter to point to the internal structure.

Converts sound samples from the integer data type provided by the sound card to floating-point values needed by the MHA in the range [-1.0,1.0]

Implemented in **alsa_t**< **T** > (p. 293).

5.39.2.4 write() `virtual bool alsa_base_t::write ( mha_wave_t * ) [pure virtual]`

write audio samples from the given waveform buffer to the sound device.

converts the floating point values coming from the MHA to the integer samples required by the sound card.

Implemented in **alsa_t**< **T** > (p. 293).

5.39.3 Member Data Documentation

5.39.3.1 pcm `snd_pcm_t* alsa_base_t::pcm`

The underlying alsa handle to this sound card.

The documentation for this class was generated from the following file:

- **MHAIOalsa.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.40 alsa_dev_par_parser_t Class Reference

Parser variables corresponding to one alsa device.

Inheritance diagram for alsa_dev_par_parser_t:

Public Member Functions

- **alsa_dev_par_parser_t** (snd_pcm_stream_t **stream_dir**)
  Constructor inserts the parser variables into this sub-parser.

Public Attributes

- **MHAParser::string_t device**
  Name of the device in the alsa world, like "hw:0.0", "default", etc.
- **MHAParser::int_t nperiods**
  Number of buffers of fragsize to hold in the alsa buffer.
- snd_pcm_stream_t **stream_dir**
  Remember the direction (capture/playback) of this device.

Additional Inherited Members

5.40.1 Detailed Description

Parser variables corresponding to one alsa device.

ALSA separates audio capture and audio playback into two different devices that have to be opened separately. This class encapsulates the parser variables that pertain to one such direction.

© 2005-2021 HörTech gGmbH, Oldenburg

5.40.2 Constructor & Destructor Documentation

5.40.2.1 alsa_dev_par_parser_t() `alsa_dev_par_parser_t::alsa_dev_par_parser_t ( snd_pcm_stream_t stream_dir )`

Constructor inserts the parser variables into this sub-parser.

Parameters

stream_dir	capture or playback

5.40.3 Member Data Documentation

5.40.3.1 device `MHAParser::string_t alsa_dev_par_parser_t::device`

Name of the device in the alsa world, like "hw:0.0", "default", etc.

5.40.3.2 nperiods `MHAParser::int_t alsa_dev_par_parser_t::nperiods`

Number of buffers of fragsize to hold in the alsa buffer.

Usually 2, the minimum possible.

5.40.3.3 stream_dir `snd_pcm_stream_t alsa_dev_par_parser_t::stream_dir`

Remember the direction (capture/playback) of this device.

The documentation for this class was generated from the following file:

- **MHAIOalsa.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.41 alsa_t< T > Class Template Reference

Our representation of one alsa device.

Inheritance diagram for alsa_t< T >:

Public Member Functions

- **alsa_t** (const **alsa_dev_par_parser_t** &par, unsigned int rate, unsigned int **fragsize**, unsigned int **channels**)

  Constructor receives the parameters for this device.
- ~**alsa_t** ()

  Destructor closes the sound device.
- void **start** () override

  start puts alsa device in usable state
- void **stop** () override

  stop informs alsa device that we do not need any more samples / will not provide any more samples
- bool **read** (**mha_wave_t** **) override

  *read audio samples from the device into an internal **mha_wave_t** (p. 840) buffer, then update the pointer given as parameter to point to the internal structure.*
- bool **write** (**mha_wave_t** *) override

  write audio samples from the given waveform buffer to the sound device.

Private Attributes

- unsigned int **channels**
- unsigned int **fragsize**
- T * **buffer**
- std::vector< **mha_real_t** > **frame_data**
- **MHASignal::waveform_t wave**

  internal buffer to store sound samples coming from the sound card.
- const **mha_real_t gain**
- const **mha_real_t invgain**
- snd_pcm_format_t **pcm_format**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.41.1 Detailed Description

template<typename T>
class alsa_t< T >

Our representation of one alsa device.

We can start and stop the device, and depending on the direction, read or write samples.

5.41.2 Constructor & Destructor Documentation

5.41.2.1 alsa_t() template<typename T >

```
alsa_t< T >::  alsa_t (
            const  alsa_dev_par_parser_t & par,
            unsigned int rate,
            unsigned int fragsize,
            unsigned int channels )
```

Constructor receives the parameters for this device.

It opens the sound device using the alsa library and selects the given parameters, but does not yet start the sound device to perform real I/O.

Parameters

par	our parser variable aggregator (containing direction, device name, and number of periods to place in alsa buffer)
rate	sampling rate in Hz
fragsize	samples per block per channel
channels	number of audio channels to open

5.41.2.2 ~alsa_t() template<typename T >

```
alsa_t< T >::~ alsa_t
```

Destructor closes the sound device.

© 2005-2021 HörTech gGmbH, Oldenburg

5.41.3 Member Function Documentation

5.41.3.1 start()

```
template<typename T >
void alsa_t< T >::start [override], [virtual]
```

start puts alsa device in usable state

Implements **alsa_base_t** (p. 287).

5.41.3.2 stop()

```
template<typename T >
void alsa_t< T >::stop [override], [virtual]
```

stop informs alsa device that we do not need any more samples / will not provide any more samples

Implements **alsa_base_t** (p. 287).

5.41.3.3 read()

```
template<typename T >
bool alsa_t< T >::read (
            mha_wave_t ** s ) [override], [virtual]
```

read audio samples from the device into an internal **mha_wave_t** (p. 840) buffer, then update the pointer given as parameter to point to the internal structure.

Converts sound samples from the integer data type provided by the sound card to floating-point values needed by the MHA in the range [-1.0,1.0]

Implements **alsa_base_t** (p. 287).

5.41.3.4 write()

```
template<typename T >
bool alsa_t< T >::write (
            mha_wave_t * s ) [override], [virtual]
```

write audio samples from the given waveform buffer to the sound device.

converts the floating point values coming from the MHA to the integer samples required by the sound card.

Implements **alsa_base_t** (p. 288).

© 2005-2021 HörTech gGmbH, Oldenburg

5.41.4 Member Data Documentation

5.41.4.1 channels `template<typename T >`

`unsigned int` **`alsa_t`**`< T >::channels [private]`

5.41.4.2 fragsize `template<typename T >`

`unsigned int` **`alsa_t`**`< T >::fragsize [private]`

5.41.4.3 buffer `template<typename T >`

`T*` **`alsa_t`**`< T >::buffer [private]`

5.41.4.4 frame_data `template<typename T >`

`std::vector<` **`mha_real_t`**`> ` **`alsa_t`**`< T >::frame_data [private]`

5.41.4.5 wave `template<typename T >`

`MHASignal::waveform_t` **`alsa_t`**`< T >::wave [private]`

internal buffer to store sound samples coming from the sound card.

5.41.4.6 gain `template<typename T >`

`const` **`mha_real_t`** **`alsa_t`**`< T >::gain [private]`

5.41.4.7 invgain `template<typename T >`

`const` **`mha_real_t`** **`alsa_t`**`< T >::invgain [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.41.4.8 pcm_format `template<typename T >`

`snd_pcm_format_t` **alsa_t**`< T >::pcm_format [private]`

The documentation for this class was generated from the following file:

- **MHAIOalsa.cpp**

5.42 altconfig_t Class Reference

Single class implementing plugin altconfig.

Inheritance diagram for altconfig_t:

Public Member Functions

- **altconfig_t** (**algo_comm_t** iac, const std::string &configured_name)

  Constructor initializes an instance of the altconfig plugin.
- void **prepare** (**mhaconfig_t** &cf)

  Invoked by MHA when this plugin should prepare for signal processing.
- void **release** ()

  Invoked by MHA when this plugin should call its release function.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **on_set_algos** ()
  Callback executed when the configuration variable "algos" is written to at run time.
- void **on_set_select** ()
  Callback executed when the configuration variable "select" is successfully written to at run time.
- void **event_select_all** ()
  Callback executed when the configuration variable "selectall" is written to at run time.
- std::map< std::string, **MHAParser::string_t** > **save_state** ()
  Save the old state of the user-defined sub-parsers for eventual restoration later.
- void **restore_state** (std::map< std::string, **MHAParser::string_t** > &state, std::map< std::string, **MHAParser::string_t** > &failed_state)
  Restore the old parser state from a saved state.

Private Attributes

- **MHAParser::vstring_t parser_algos**
- **MHAParser::kw_t select_plug**
- **MHAParser::bool_t selectall**
- std::map< std::string, **MHAParser::string_t** > **configs**
  Storage for alternative configuration commands.
- **MHAEvents::patchbay_t**< **altconfig_t** > **patchbay**
  Configuration event broker.

Additional Inherited Members

5.42.1 Detailed Description

Single class implementing plugin altconfig.

altconfig loads another plugin and can send configuration commands to that plugin. altconfig does not need a separate runtime configuration class, template parameter char is used as a placeholder. Uses parser variable names "algos" and "select" even though the alternatives that can be selected in this plugin are not algorithms or plugins but configuration commands in order to be interface-compatible with plugin altplugs. mhacontrol has a UI area that automatically fills with the alternatives found in either altplugs or altconfig if the complete MHA loads exactly one plugin with this interface.

5.42.2 Constructor & Destructor Documentation

5.42.2.1 altconfig_t()

```
altconfig_t::altconfig_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructor initializes an instance of the altconfig plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

iac	Algorithm communication variable space, not used by this plugin except to initialize base class.
configured_name	Configured name of this plugin.

5.42.3 Member Function Documentation

5.42.3.1 prepare() `void altconfig_t::prepare ( mhaconfig_t & cf ) [inline], [virtual]`

Invoked by MHA when this plugin should prepare for signal processing.

altconfig delegates to the loaded plugin and does not need to do more work to prepare.

Parameters

cf	signal dimensions, forwarded to loaded plugin which may change the signal dimensions.

Implements **MHAPlugin::plugin_t< char >** (p. 1153).

5.42.3.2 release() `void altconfig_t::release ( ) [inline], [virtual]`

Invoked by MHA when this plugin should call its release function.

altconfig delegates to the loaded plugin.

Reimplemented from **MHAPlugin::plugin_t< char >** (p. 1154).

5.42.3.3 on_set_algos() `void altconfig_t::on_set_algos ( ) [private]`

Callback executed when the configuration variable "algos" is written to at run time.

Adds the configuration variables that store the alternative configuration commands based on the new names and sets the allowed values of configuration variable "select"

© 2005-2021 HörTech gGmbH, Oldenburg

5.42.3.4 on_set_select() `void altconfig_t::on_set_select ( ) [private]`

Callback executed when the configuration variable "select" is successfully written to at run time.

Causes the execution of the stored command for the selected condition in the context of the loaded plugin.

5.42.3.5 event_select_all() `void altconfig_t::event_select_all ( ) [private]`

Callback executed when the configuration variable "selectall" is written to at run time.

When set to yes, iterates once through all stored configurations in the order they appear in configuration variable algos.

5.42.3.6 save_state() `std::map< std::string, MHAParser::string_t > altconfig_t↩ ::save_state ( ) [private]`

Save the old state of the user-defined sub-parsers for eventual restoration later.

operator= does not suffice to store/restore the old state because we need to re-insert the string_t's that were removed, but a copy of the string_t keeps a pointer to its parent and complains if we try to insert_item it again. So we just copy the relevant data into a new string_t.

Parameters

old_state	old parser state

Returns

Copy of the old state

5.42.3.7 restore_state() `void altconfig_t::restore_state ( std::map< std::string, MHAParser::string_t > & state, std::map< std::string, MHAParser::string_t > & failed_state ) [private]`

Restore the old parser state from a saved state.

Parameters

state	old parser state

© 2005-2021 HörTech gGmbH, Oldenburg

5.42.4 Member Data Documentation

5.42.4.1 parser_algos `MHAParser::vstring_t altconfig_t::parser_algos [private]`

5.42.4.2 select_plug `MHAParser::kw_t altconfig_t::select_plug [private]`

5.42.4.3 selectall `MHAParser::bool_t altconfig_t::selectall [private]`

5.42.4.4 configs `std::map<std::string, MHAParser::string_t> altconfig_t::configs [private]`

Storage for alternative configuration commands.

New entries are registered with the plugin's parser when parser_algos is updated.

5.42.4.5 patchbay `MHAEvents::patchbay_t< altconfig_t> altconfig_t::patchbay [private]`

Configuration event broker.

The documentation for this class was generated from the following files:

- **altconfig.hh**
- **altconfig.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.43 altplugs_t Class Reference

Inheritance diagram for altplugs_t:

Public Member Functions

- **altplugs_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- void **process** (**mha_wave_t** ∗, **mha_wave_t** ∗∗)
- void **process** (**mha_spec_t** ∗, **mha_wave_t** ∗∗)
- void **process** (**mha_wave_t** ∗, **mha_spec_t** ∗∗)
- void **process** (**mha_spec_t** ∗, **mha_spec_t** ∗∗)
- virtual std::string **parse** (const std::string &arg)
- virtual void **parse** (const char ∗a1, char ∗a2, unsigned int a3)

Private Member Functions

- void **event_set_plugs** ()
- void **event_add_plug** ()
- void **event_delete_plug** ()
- void **event_select_plug** ()
- void **update_selector_list** ()
- void **update_ramplen** ()
- void **proc_ramp** (**mha_wave_t** ∗s)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::bool_t use_own_ac**
- **MHAParser::vstring_t parser_plugs**
- **MHAParser::string_t add_plug**
- **MHAParser::string_t delete_plug**
- **MHAParser::float_t ramplen**
- **MHAParser::kw_t select_plug**
- **MHAParser::parser_t current**
- **MHAParser::vstring_mon_t nondefault_labels**
- std::vector< **mhaplug_cfg_t** * > **plugs**
- **mhaplug_cfg_t** * **selected_plug**
- **MHAEvents::patchbay_t**< **altplugs_t** > **patchbay**
- **MHASignal::waveform_t** * **fallback_wave**
- **MHASignal::spectrum_t** * **fallback_spec**
- **mhaconfig_t cfin**
- **mhaconfig_t cfout**
- bool **prepared**
- bool **added_via_plugs**
- unsigned int **ramp_counter**
- unsigned int **ramp_len**

Additional Inherited Members

5.43.1 Constructor & Destructor Documentation

5.43.1.1 altplugs_t()

```
altplugs_t::altplugs_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.43.2 Member Function Documentation

5.43.2.1 prepare()

```
void altplugs_t::prepare (
    mhaconfig_t & cf ) [virtual]
```

Implements **MHAPlugin::plugin_t**< **MHAWindow::fun_t** > (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.43.2.2 release() `void altplugs_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t**< **MHAWindow::fun_t** > (p. 1154).

5.43.2.3 process() [1/4]

```
void altplugs_t::process (
    mha_wave_t * sIn,
    mha_wave_t ** sOut )
```

5.43.2.4 process() [2/4]

```
void altplugs_t::process (
    mha_spec_t * sIn,
    mha_wave_t ** sOut )
```

5.43.2.5 process() [3/4]

```
void altplugs_t::process (
    mha_wave_t * sIn,
    mha_spec_t ** sOut )
```

5.43.2.6 process() [4/4]

```
void altplugs_t::process (
    mha_spec_t * sIn,
    mha_spec_t ** sOut )
```

5.43.2.7 parse() [1/2]

```
std::string altplugs_t::parse (
    const std::string & arg ) [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1035).

5.43.2.8 parse() [2/2]

```
virtual void altplugs_t::parse (
    const char * a1,
    char * a2,
    unsigned int a3 ) [inline], [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1036).

© 2005-2021 HörTech gGmbH, Oldenburg

5.43.2.9 event_set_plugs() `void altplugs_t::event_set_plugs ( ) [private]`

5.43.2.10 event_add_plug() `void altplugs_t::event_add_plug ( ) [private]`

5.43.2.11 event_delete_plug() `void altplugs_t::event_delete_plug ( ) [private]`

5.43.2.12 event_select_plug() `void altplugs_t::event_select_plug ( ) [private]`

5.43.2.13 update_selector_list() `void altplugs_t::update_selector_list ( ) [private]`

5.43.2.14 update_ramplen() `void altplugs_t::update_ramplen ( ) [private]`

5.43.2.15 proc_ramp() `void altplugs_t::proc_ramp ( mha_wave_t * s ) [private]`

5.43.3 Member Data Documentation

5.43.3.1 use_own_ac `MHAParser::bool_t altplugs_t::use_own_ac [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.43.3.2 parser_plugs `MHAParser::vstring_t altplugs_t::parser_plugs [private]`

5.43.3.3 add_plug `MHAParser::string_t altplugs_t::add_plug [private]`

5.43.3.4 delete_plug `MHAParser::string_t altplugs_t::delete_plug [private]`

5.43.3.5 ramplen `MHAParser::float_t altplugs_t::ramplen [private]`

5.43.3.6 select_plug `MHAParser::kw_t altplugs_t::select_plug [private]`

5.43.3.7 current `MHAParser::parser_t altplugs_t::current [private]`

5.43.3.8 nondefault_labels `MHAParser::vstring_mon_t altplugs_t::nondefault_labels [private]`

5.43.3.9 plugs `std::vector< mhaplug_cfg_t* > altplugs_t::plugs [private]`

5.43.3.10 selected_plug `mhaplug_cfg_t* altplugs_t::selected_plug [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.43.3.11 patchbay `MHAEvents::patchbay_t< altplugs_t> altplugs_t::patchbay [private]`

5.43.3.12 fallback_wave `MHASignal::waveform_t* altplugs_t::fallback_wave [private]`

5.43.3.13 fallback_spec `MHASignal::spectrum_t* altplugs_t::fallback_spec [private]`

5.43.3.14 cfin `mhaconfig_t altplugs_t::cfin [private]`

5.43.3.15 cfout `mhaconfig_t altplugs_t::cfout [private]`

5.43.3.16 prepared `bool altplugs_t::prepared [private]`

5.43.3.17 added_via_plugs `bool altplugs_t::added_via_plugs [private]`

5.43.3.18 ramp_counter `unsigned int altplugs_t::ramp_counter [private]`

5.43.3.19 ramp_len `unsigned int altplugs_t::ramp_len [private]`

The documentation for this class was generated from the following file:

- **altplugs.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.44 analysepath_t Class Reference

Public Member Functions

- **analysepath_t** (unsigned int nchannels_in, unsigned int inner_fragsize, int **priority**, **MHAProc_wave2wave_t** inner_proc_wave2wave, **MHAProc_wave2spec_t** inner_proc_wave2spec, void *ilibdata, **algo_comm_t** **outer_ac**, const **MHA_AC::acspace2matrix_t** &acspace_template, **mha_domain_t** **inner_out_domain**, unsigned int fifo_len_blocks)
- virtual ~**analysepath_t** ()
- void **rt_process** (**mha_wave_t** *)
- virtual int **svc** ()

Private Attributes

- **MHAProc_wave2wave_t** **inner_process_wave2wave**
- **MHAProc_wave2spec_t** **inner_process_wave2spec**
- **MHASignal::waveform_t** **inner_input**
- void * **libdata**
- **mha_fifo_lf_t**< **mha_real_t** > **wave_fifo**
- **mha_fifo_lf_t**< **MHA_AC::acspace2matrix_t** > **ac_fifo**
- **MHA_AC::acspace2matrix_t** **inner_ac_copy**
- **MHA_AC::acspace2matrix_t** **outer_ac_copy**
- **algo_comm_t** **outer_ac**
- **mha_domain_t** **inner_out_domain**
- **MHA_Error** **inner_error**
- bool **has_inner_error**
- bool **flag_terminate_inner_thread**
- int **input_to_process**
- pthread_mutex_t **ProcessMutex**
- pthread_attr_t **attr**
- struct sched_param **priority**
- int **scheduler**
- pthread_t **thread**
- pthread_cond_t **cond_to_process**

5.44.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.44.1.1 analysepath_t()

```
analysepath_t::analysepath_t (
    unsigned int nchannels_in,
    unsigned int inner_fragsize,
    int priority,
    MHAProc_wave2wave_t inner_proc_wave2wave,
    MHAProc_wave2spec_t inner_proc_wave2spec,
    void * ilibdata,
    algo_comm_t outer_ac,
    const MHA_AC::acspace2matrix_t & acspace_template,
    mha_domain_t inner_out_domain,
    unsigned int fifo_len_blocks )
```

5.44.1.2 ~analysepath_t()

```
analysepath_t::~analysepath_t ( ) [virtual]
```

5.44.2 Member Function Documentation

5.44.2.1 rt_process()

```
void analysepath_t::rt_process (
    mha_wave_t * outer_input )
```

5.44.2.2 svc()

```
int analysepath_t::svc ( ) [virtual]
```

5.44.3 Member Data Documentation

5.44.3.1 inner_process_wave2wave

```
MHAProc_wave2wave_t analysepath_t::inner_↩
process_wave2wave [private]
```

5.44.3.2 inner_process_wave2spec

```
MHAProc_wave2spec_t analysepath_t::inner_↩
process_wave2spec [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.44.3.3 inner_input

`MHASignal::waveform_t analysepath_t::inner_input [private]`

5.44.3.4 libdata

`void* analysepath_t::libdata [private]`

5.44.3.5 wave_fifo

`mha_fifo_lf_t< mha_real_t> analysepath_t::wave_fifo [private]`

5.44.3.6 ac_fifo

`mha_fifo_lf_t< MHA_AC::acspace2matrix_t> analysepath_t::ac_fifo [private]`

5.44.3.7 inner_ac_copy

`MHA_AC::acspace2matrix_t analysepath_t::inner_ac_copy [private]`

5.44.3.8 outer_ac_copy

`MHA_AC::acspace2matrix_t analysepath_t::outer_ac_copy [private]`

5.44.3.9 outer_ac

`algo_comm_t analysepath_t::outer_ac [private]`

5.44.3.10 inner_out_domain

`mha_domain_t analysepath_t::inner_out_domain [private]`

5.44.3.11 inner_error

`MHA_Error analysepath_t::inner_error [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.44.3.12 has_inner_error `bool analysepath_t::has_inner_error [private]`

5.44.3.13 flag_terminate_inner_thread `bool analysepath_t::flag_terminate_inner_thread [private]`

5.44.3.14 input_to_process `int analysepath_t::input_to_process [private]`

5.44.3.15 ProcessMutex `pthread_mutex_t analysepath_t::ProcessMutex [private]`

5.44.3.16 attr `pthread_attr_t analysepath_t::attr [private]`

5.44.3.17 priority `struct sched_param analysepath_t::priority [private]`

5.44.3.18 scheduler `int analysepath_t::scheduler [private]`

5.44.3.19 thread `pthread_t analysepath_t::thread [private]`

5.44.3.20 cond_to_process `pthread_cond_t analysepath_t::cond_to_process [private]`

The documentation for this class was generated from the following file:

- **analysispath.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.45 analysispath_if_t Class Reference

Inheritance diagram for analysispath_if_t:

Public Member Functions

- **analysispath_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- ∼**analysispath_if_t** ()

Private Member Functions

- void **loadlib** ()

Private Attributes

- **MHAEvents::patchbay_t**< **analysispath_if_t** > **patchbay**
- **MHAParser::string_t** **libname**
- **MHAParser::int_t** **fragsize**
- **MHAParser::int_t** **fifolen**
- **MHAParser::int_t** **priority**
- **MHAParser::vstring_t** **vars**
- **plug_t** ∗ **plug**
- std::string **algo**
- **MHA_AC::acspace2matrix_t** ∗ **acspace_template**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.45.1 Constructor & Destructor Documentation

5.45.1.1 analysispath_if_t()

```
analysispath_if_t::analysispath_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.45.1.2 ~analysispath_if_t()

```
analysispath_if_t::~analysispath_if_t ( )
```

5.45.2 Member Function Documentation

5.45.2.1 process()

```
mha_wave_t * analysispath_if_t::process (
    mha_wave_t * s )
```

5.45.2.2 prepare()

```
void analysispath_if_t::prepare (
    mhaconfig_t & conf ) [virtual]
```

Implements **MHAPlugin::plugin_t< analysepath_t >** (p. 1153).

5.45.2.3 release()

```
void analysispath_if_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< analysepath_t >** (p. 1154).

5.45.2.4 loadlib()

```
void analysispath_if_t::loadlib ( ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.45.3 Member Data Documentation

5.45.3.1 patchbay

MHAEvents::patchbay_t< **analysispath_if_t** > analysispath_if_t↩ ::patchbay [private]

5.45.3.2 libname

MHAParser::string_t analysispath_if_t::libname [private]

5.45.3.3 fragsize

MHAParser::int_t analysispath_if_t::fragsize [private]

5.45.3.4 fifolen

MHAParser::int_t analysispath_if_t::fifolen [private]

5.45.3.5 priority

MHAParser::int_t analysispath_if_t::priority [private]

5.45.3.6 vars

MHAParser::vstring_t analysispath_if_t::vars [private]

5.45.3.7 plug

plug_t* analysispath_if_t::plug [private]

5.45.3.8 algo

std::string analysispath_if_t::algo [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.45.3.9 acspace_template

`MHA_AC::acspace2matrix_t* analysispath_if_t::acspace_template [private]`

The documentation for this class was generated from the following file:

- **analysispath.cpp**

5.46 attenuate20_t Class Reference

Inheritance diagram for attenuate20_t:

Public Member Functions

- **attenuate20_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **release** (void) override
- void **prepare** (**mhaconfig_t** &signal_info) override
- **mha_wave_t** * **process** (**mha_wave_t** *signal)

Additional Inherited Members

5.46.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.46.1.1 attenuate20_t()

```
attenuate20_t::attenuate20_t (
    algo_comm_t iac,
    const std::string & configured_name ) [inline]
```

5.46.2 Member Function Documentation

5.46.2.1 release()

```
void attenuate20_t::release (
    void ) [inline], [override], [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< int >** (p. 1154).

5.46.2.2 prepare()

```
void attenuate20_t::prepare (
    mhaconfig_t & signal_info ) [inline], [override], [virtual]
```

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

5.46.2.3 process()

```
mha_wave_t* attenuate20_t::process (
    mha_wave_t * signal ) [inline]
```

The documentation for this class was generated from the following file:

- **attenuate20.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.47 audiometerbackend::audiometer_if_t Class Reference

Inheritance diagram for audiometerbackend::audiometer_if_t:

Public Member Functions

- **audiometer_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update** ()
- void **change_mode** ()
- void **set_level** ()

Private Attributes

- **MHAParser::int_t freq**
- **MHAParser::kw_t sigtype**
- **MHAParser::float_t level**
- **MHAParser::kw_t mode**
- **MHAParser::float_t ramplen**
- **MHASignal::loop_wavefragment_t::playback_mode_t pmode**
- std::vector< int > **outchannel**
- **MHAEvents::patchbay_t**< **audiometer_if_t** > **patchbay**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.47.1 Constructor & Destructor Documentation

5.47.1.1 audiometer_if_t()

```
audiometerbackend::audiometer_if_t::audiometer_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.47.2 Member Function Documentation

5.47.2.1 process()

```
mha_wave_t * audiometerbackend::audiometer_if_t::process (
    mha_wave_t * s )
```

5.47.2.2 prepare()

```
void audiometerbackend::audiometer_if_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t**< **runtime_cfg_t** > (p. 1153).

5.47.2.3 update()

```
void audiometerbackend::audiometer_if_t::update ( ) [private]
```

5.47.2.4 change_mode()

```
void audiometerbackend::audiometer_if_t::change_mode ( )
[private]
```

5.47.2.5 set_level()

```
void audiometerbackend::audiometer_if_t::set_level ( ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.47.3 Member Data Documentation

5.47.3.1 freq

MHAParser::int_t audiometerbackend::audiometer_if_t::freq [private]

5.47.3.2 sigtype

MHAParser::kw_t audiometerbackend::audiometer_if_t::sigtype [private]

5.47.3.3 level

MHAParser::float_t audiometerbackend::audiometer_if_t::level [private]

5.47.3.4 mode

MHAParser::kw_t audiometerbackend::audiometer_if_t::mode [private]

5.47.3.5 ramplen

MHAParser::float_t audiometerbackend::audiometer_if_t::ramplen [private]

5.47.3.6 pmode

MHASignal::loop_wavefragment_t::playback_mode_t audiometerbackend↩ ::audiometer_if_t::pmode [private]

5.47.3.7 outchannel

std::vector<int> audiometerbackend::audiometer_if_t::outchannel [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.47.3.8 patchbay `MHAEvents::patchbay_t< audiometer_if_t> audiometerbackend↩ ::audiometer_if_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **audiometerbackend.cpp**

5.48 audiometerbackend::level_adapt_t Class Reference

Inheritance diagram for audiometerbackend::level_adapt_t:

Public Member Functions

- **level_adapt_t** (**mhaconfig_t** cf, **mha_real_t** adapt_len, **mha_real_t** l_new_, **mha_↩ real_t** l_old_)
- void **update_frame** ()
- **mha_real_t get_level** () const
- bool **can_update** () const

Private Attributes

- unsigned int **ilen**
- unsigned int **pos**
- **MHAWindow::fun_t wnd**
- **mha_real_t l_new**
- **mha_real_t l_old**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.48.1 Constructor & Destructor Documentation

5.48.1.1 level_adapt_t()

```
audiometerbackend::level_adapt_t::level_adapt_t (
    mhaconfig_t cf,
    mha_real_t adapt_len,
    mha_real_t l_new_,
    mha_real_t l_old_ )
```

5.48.2 Member Function Documentation

5.48.2.1 update_frame()

```
void audiometerbackend::level_adapt_t::update_frame ( )
```

5.48.2.2 get_level()

```
mha_real_t audiometerbackend::level_adapt_t::get_level ( ) const [inline]
```

5.48.2.3 can_update()

```
bool audiometerbackend::level_adapt_t::can_update ( ) const [inline]
```

5.48.3 Member Data Documentation

5.48.3.1 ilen

```
unsigned int audiometerbackend::level_adapt_t::ilen [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.48.3.2 pos `unsigned int audiometerbackend::level_adapt_t::pos [private]`

5.48.3.3 wnd `MHAWindow::fun_t audiometerbackend::level_adapt_t::wnd [private]`

5.48.3.4 l_new `mha_real_t audiometerbackend::level_adapt_t::l_new [private]`

5.48.3.5 l_old `mha_real_t audiometerbackend::level_adapt_t::l_old [private]`

The documentation for this class was generated from the following file:

- **audiometerbackend.cpp**

5.49 audiometerbackend::lnn3rdoct_t Class Reference

Inheritance diagram for audiometerbackend::lnn3rdoct_t:

Public Member Functions

- **lnn3rdoct_t** (unsigned int fs, unsigned int f, float bw, unsigned int niter)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **iterate_lnn** ()
- void **bandpass** ()

Private Attributes

- unsigned int **_fmin**
- unsigned int **_fmax**
- std::random_device **dev**
- std::default_random_engine **rng**
- std::uniform_real_distribution< float > **random**

Additional Inherited Members

5.49.1 Constructor & Destructor Documentation

5.49.1.1 lnn3rdoct_t()

```
audiometerbackend::lnn3rdoct_t::lnn3rdoct_t (
    unsigned int fs,
    unsigned int f,
    float bw,
    unsigned int niter )
```

5.49.2 Member Function Documentation

5.49.2.1 iterate_lnn()

```
void audiometerbackend::lnn3rdoct_t::iterate_lnn ( ) [private]
```

5.49.2.2 bandpass()

```
void audiometerbackend::lnn3rdoct_t::bandpass ( ) [private]
```

5.49.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.49.3.1 _fmin `unsigned int audiometerbackend::lnn3rdoct_t::_fmin [private]`

5.49.3.2 _fmax `unsigned int audiometerbackend::lnn3rdoct_t::_fmax [private]`

5.49.3.3 dev `std::random_device audiometerbackend::lnn3rdoct_t::dev [private]`

5.49.3.4 rng `std::default_random_engine audiometerbackend::lnn3rdoct_t::rng [private]`

5.49.3.5 random `std::uniform_real_distribution<float> audiometerbackend::lnn3rdoct↩_t::random [private]`

The documentation for this class was generated from the following file:

- **audiometerbackend.cpp**

5.50 audiometerbackend::signal_gen_t Class Reference

Inheritance diagram for audiometerbackend::signal_gen_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **signal_gen_t** (int f, int fs, unsigned int sigtype)

Additional Inherited Members

5.50.1 Constructor & Destructor Documentation

5.50.1.1 signal_gen_t()

```
audiometerbackend::signal_gen_t::signal_gen_t (
    int f,
    int fs,
    unsigned int sigtype )
```

The documentation for this class was generated from the following file:

- **audiometerbackend.cpp**

5.51 audiometerbackend::sine_t Class Reference

Inheritance diagram for audiometerbackend::sine_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **sine_t** (unsigned int fs, unsigned int f)

Additional Inherited Members

5.51.1 Constructor & Destructor Documentation

5.51.1.1 sine_t()

```
sine_t::sine_t (
    unsigned int fs,
    unsigned int f )
```

The documentation for this class was generated from the following file:

- **audiometerbackend.cpp**

5.52 AuditoryProfile::fmap_t Class Reference

A class to store frequency dependent data (e.g., HTL and UCL).

Inherits map< mha_real_t, mha_real_t >.

Public Member Functions

- std::vector< **mha_real_t** > **get_frequencies** () const
  Return configured frequencies.
- std::vector< **mha_real_t** > **get_values** () const
  Return stored values corresponding to the frequencies.
- bool **isempty** () const

5.52.1 Detailed Description

A class to store frequency dependent data (e.g., HTL and UCL).

© 2005-2021 HörTech gGmbH, Oldenburg

5.52.2 Member Function Documentation

5.52.2.1 get_frequencies() `std::vector< mha_real_t > AuditoryProfile::fmap_t↩ ::get_frequencies ( ) const`

Return configured frequencies.

5.52.2.2 get_values() `std::vector< mha_real_t > AuditoryProfile::fmap_t::get_↩ values ( ) const`

Return stored values corresponding to the frequencies.

5.52.2.3 isempty() `bool AuditoryProfile::fmap_t::isempty ( ) const [inline]`

The documentation for this class was generated from the following files:

- **auditory_profile.h**
- **auditory_profile.cpp**

5.53 AuditoryProfile::parser_t Class Reference

Class to make the auditory profile accessible through the parser interface.

Inheritance diagram for AuditoryProfile::parser_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Classes

- class **ear_t**
- class **fmap_t**

Public Member Functions

- **parser_t** ()
- **AuditoryProfile::profile_t get_current_profile** ()

Private Attributes

- **AuditoryProfile::parser_t::ear_t L**
- **AuditoryProfile::parser_t::ear_t R**

Additional Inherited Members

5.53.1 Detailed Description

Class to make the auditory profile accessible through the parser interface.

5.53.2 Constructor & Destructor Documentation

5.53.2.1 parser_t() `AuditoryProfile::parser_t::parser_t ( )`

5.53.3 Member Function Documentation

5.53.3.1 get_current_profile() **`AuditoryProfile::profile_t`** `AuditoryProfile::parser↩_t::get_current_profile ( )`

© 2005-2021 HörTech gGmbH, Oldenburg

5.53.4 Member Data Documentation

5.53.4.1 L `AuditoryProfile::parser_t::ear_t` `AuditoryProfile::parser_t::L` `[private]`

5.53.4.2 R `AuditoryProfile::parser_t::ear_t` `AuditoryProfile::parser_t::R` `[private]`

The documentation for this class was generated from the following files:

- **auditory_profile.h**
- **auditory_profile.cpp**

5.54 AuditoryProfile::parser_t::ear_t Class Reference

Inheritance diagram for AuditoryProfile::parser_t::ear_t:

Public Member Functions

- **ear_t** ()
- **AuditoryProfile::profile_t::ear_t get_ear** () const

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **AuditoryProfile::parser_t::fmap_t HTL**
- **AuditoryProfile::parser_t::fmap_t UCL**

Additional Inherited Members

5.54.1 Constructor & Destructor Documentation

5.54.1.1 ear_t() `AuditoryProfile::parser_t::ear_t::ear_t ( )`

5.54.2 Member Function Documentation

5.54.2.1 get_ear() `AuditoryProfile::profile_t::ear_t AuditoryProfile::parser_t↩ ::ear_t::get_ear ( ) const`

5.54.3 Member Data Documentation

5.54.3.1 HTL `AuditoryProfile::parser_t::fmap_t AuditoryProfile::parser_t::ear_t↩ ::HTL [private]`

5.54.3.2 UCL `AuditoryProfile::parser_t::fmap_t AuditoryProfile::parser_t::ear_t↩ ::UCL [private]`

The documentation for this class was generated from the following files:

- **auditory_profile.h**
- **auditory_profile.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.55 AuditoryProfile::parser_t::fmap_t Class Reference

Inheritance diagram for AuditoryProfile::parser_t::fmap_t:

Public Member Functions

- **fmap_t** (const std::string &name, const std::string & **help**)
- **AuditoryProfile::fmap_t get_fmap** () const

Private Member Functions

- void **validate** ()

Private Attributes

- **MHAEvents::patchbay_t**< **AuditoryProfile::parser_t::fmap_t** > **patchbay**
- **MHAParser::vfloat_t f**
- **MHAParser::vfloat_t value**
- std::string **name_**

Additional Inherited Members

5.55.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.55.1.1 fmap_t()

```
AuditoryProfile::parser_t::fmap_t::fmap_t (
    const std::string & name,
    const std::string & help )
```

5.55.2 Member Function Documentation

5.55.2.1 get_fmap()

AuditoryProfile::fmap_t AuditoryProfile::parser_t::fmap_t↩ ::get_fmap ( ) const

5.55.2.2 validate()

void AuditoryProfile::parser_t::fmap_t::validate ( ) [private]

5.55.3 Member Data Documentation

5.55.3.1 patchbay

MHAEvents::patchbay_t< **AuditoryProfile::parser_t::fmap_t**> Auditory↩ Profile::parser_t::fmap_t::patchbay [private]

5.55.3.2 f

MHAParser::vfloat_t AuditoryProfile::parser_t::fmap_t::f [private]

5.55.3.3 value

MHAParser::vfloat_t AuditoryProfile::parser_t::fmap_t::value [private]

5.55.3.4 name_

std::string AuditoryProfile::parser_t::fmap_t::name_ [private]

The documentation for this class was generated from the following files:

- **auditory_profile.h**
- **auditory_profile.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.56 AuditoryProfile::profile_t Class Reference

The Auditory Profile class.

Classes

- class **ear_t**
  Class for ear-dependent parameters, e.g., audiograms or unilateral loudness scaling.

Public Member Functions

- **AuditoryProfile::profile_t::ear_t get_ear** (unsigned int channel) const
  Return ear information of channel number.

Public Attributes

- **AuditoryProfile::profile_t::ear_t L**
  Left ear data.
- **AuditoryProfile::profile_t::ear_t R**
  Right ear data.

5.56.1 Detailed Description

The Auditory Profile class.

See definition of auditory profile

Currently only the audiogram data is stored.

5.56.2 Member Function Documentation

5.56.2.1 get_ear()

```
AuditoryProfile::profile_t::ear_t AuditoryProfile::profile_t↩
::get_ear (
            unsigned int channel ) const [inline]
```

Return ear information of channel number.

© 2005-2021 HörTech gGmbH, Oldenburg

5.56.3 Member Data Documentation

5.56.3.1 L `AuditoryProfile::profile_t::ear_t` `AuditoryProfile::profile_t::L`

Left ear data.

5.56.3.2 R `AuditoryProfile::profile_t::ear_t` `AuditoryProfile::profile_t::R`

Right ear data.

The documentation for this class was generated from the following file:

- **auditory_profile.h**

5.57 AuditoryProfile::profile_t::ear_t Class Reference

Class for ear-dependent parameters, e.g., audiograms or unilateral loudness scaling.

Public Member Functions

- void **convert_empty2normal** ()

Public Attributes

- **AuditoryProfile::fmap_t HTL**
- **AuditoryProfile::fmap_t UCL**

5.57.1 Detailed Description

Class for ear-dependent parameters, e.g., audiograms or unilateral loudness scaling.

© 2005-2021 HörTech gGmbH, Oldenburg

5.57.2 Member Function Documentation

5.57.2.1 convert_empty2normal() `void AuditoryProfile::profile_t::ear_t::convert↩_empty2normal ( )`

5.57.3 Member Data Documentation

5.57.3.1 HTL `AuditoryProfile::fmap_t AuditoryProfile::profile_t::ear_t::HTL`

5.57.3.2 UCL `AuditoryProfile::fmap_t AuditoryProfile::profile_t::ear_t::UCL`

The documentation for this class was generated from the following files:

- **auditory_profile.h**
- **auditory_profile.cpp**

5.58 bbcalib_interface_t Class Reference

Inheritance diagram for bbcalib_interface_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **bbcalib_interface_t** (**algo_comm_t** iac, const std::string &configured_name)
- ∼**bbcalib_interface_t** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

Private Attributes

- **calibrator_t calib_in**
- **calibrator_t calib_out**
- **MHAParser::mhapluginloader_t plugloader**

Additional Inherited Members

5.58.1 Constructor & Destructor Documentation

5.58.1.1 bbcalib_interface_t()

```
bbcalib_interface_t::bbcalib_interface_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.58.1.2 ∼bbcalib_interface_t()

```
bbcalib_interface_t::~bbcalib_interface_t ( )
```

5.58.2 Member Function Documentation

5.58.2.1 process()

```
mha_wave_t * bbcalib_interface_t::process (
    mha_wave_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.58.2.2 prepare() `void bbcalib_interface_t::prepare ( mhaconfig_t & tf ) [virtual]`

Implements **MHAPlugin::plugin_t**< **int** > (p. 1153).

5.58.2.3 release() `void bbcalib_interface_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t**< **int** > (p. 1154).

5.58.3 Member Data Documentation

5.58.3.1 calib_in `calibrator_t bbcalib_interface_t::calib_in [private]`

5.58.3.2 calib_out `calibrator_t bbcalib_interface_t::calib_out [private]`

5.58.3.3 plugloader `MHAParser::mhapluginloader_t bbcalib_interface_t::plugloader [private]`

The documentation for this class was generated from the following file:

- **transducers.cpp**

5.59 calibrator_runtime_layer_t Class Reference

Public Member Functions

- **calibrator_runtime_layer_t** (bool is_input, const **mhaconfig_t** &tf, **calibrator_variables_t** &vars)
- **mha_real_t process** (**mha_wave_t** ∗∗)

© 2005-2021 HörTech gGmbH, Oldenburg

Static Private Member Functions

- static unsigned int **firfirlen** (const std::vector< std::vector< float > > &)
- static unsigned int **firfir2fftlen** (unsigned int, const std::vector< std::vector< float > > &)

Private Attributes

- **MHAFilter::fftfilter_t fir**
- **MHASignal::quantizer_t quant**
- **MHASignal::waveform_t gain**
- **softclipper_t softclip**
  The softclipper, only used when b_is_input == false.
- bool **b_is_input**
- bool **b_use_fir**
- bool **b_use_clipping**
- **MHASignal::loop_wavefragment_t speechnoise**
- **MHASignal::loop_wavefragment_t::playback_mode_t pmode**

5.59.1 Constructor & Destructor Documentation

5.59.1.1 calibrator_runtime_layer_t()

```
calibrator_runtime_layer_t::calibrator_runtime↩
_layer_t (
            bool is_input,
            const  mhaconfig_t & tf,
             calibrator_variables_t & vars )
```

5.59.2 Member Function Documentation

5.59.2.1 process()

```
mha_real_t calibrator_runtime_layer_t::process (
            mha_wave_t ** s )
```

5.59.2.2 firfirlen()

```
unsigned int calibrator_runtime_layer_t::firfirlen (
            const std::vector< std::vector< float > > & fir )  [static], [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.59.2.3 firfir2fftlen()

```
unsigned int calibrator_runtime_layer_t::firfir2fftlen (
    unsigned int fragsize,
    const std::vector< std::vector< float > > & fir ) [static], [private]
```

5.59.3 Member Data Documentation

5.59.3.1 fir

```
MHAFilter::fftfilter_t calibrator_runtime_layer_t::fir [private]
```

5.59.3.2 quant

```
MHASignal::quantizer_t calibrator_runtime_layer_t::quant [private]
```

5.59.3.3 gain

```
MHASignal::waveform_t calibrator_runtime_layer_t::gain [private]
```

5.59.3.4 softclip

```
softclipper_t calibrator_runtime_layer_t::softclip [private]
```

The softclipper, only used when b_is_input == false.

5.59.3.5 b_is_input

```
bool calibrator_runtime_layer_t::b_is_input [private]
```

5.59.3.6 b_use_fir

```
bool calibrator_runtime_layer_t::b_use_fir [private]
```

5.59.3.7 b_use_clipping

```
bool calibrator_runtime_layer_t::b_use_clipping [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.59.3.8 speechnoise `MHASignal::loop_wavefragment_t` `calibrator_runtime_layer_t::speechnoise [private]`

5.59.3.9 pmode `MHASignal::loop_wavefragment_t::playback_mode_t` `calibrator_runtime_layer_t::pmode [private]`

The documentation for this class was generated from the following file:

- **transducers.cpp**

5.60 calibrator_t Class Reference

Inheritance diagram for calibrator_t:

Public Member Functions

- **calibrator_t** (**algo_comm_t**, bool is_input)
- void **prepare** (**mhaconfig_t** &tf)
- void **release** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **update** ()
- void **update_tau_level** ()
- void **read_levels** ()

Private Attributes

- bool **b_is_input**
- **MHAEvents::patchbay_t**< **calibrator_t** > **patchbay**
- **calibrator_variables_t vars**
- bool **prepared**

Additional Inherited Members

5.60.1 Constructor & Destructor Documentation

5.60.1.1 calibrator_t()

```
calibrator_t::calibrator_t (
    algo_comm_t iac,
    bool is_input )
```

5.60.2 Member Function Documentation

5.60.2.1 prepare()

```
void calibrator_t::prepare (
    mhaconfig_t & tf ) [inline], [virtual]
```

Implements **MHAPlugin::plugin_t**< **runtime_cfg_t** > (p. 1153).

5.60.2.2 release()

```
void calibrator_t::release ( ) [inline], [virtual]
```

Reimplemented from **MHAPlugin::plugin_t**< **runtime_cfg_t** > (p. 1154).

© 2005-2021 HörTech gGmbH, Oldenburg

5.60.2.3 process() `mha_wave_t * calibrator_t::process ( mha_wave_t * s )`

5.60.2.4 update() `void calibrator_t::update ( ) [private]`

5.60.2.5 update_tau_level() `void calibrator_t::update_tau_level ( ) [private]`

5.60.2.6 read_levels() `void calibrator_t::read_levels ( ) [private]`

5.60.3 Member Data Documentation

5.60.3.1 b_is_input `bool calibrator_t::b_is_input [private]`

5.60.3.2 patchbay `MHAEvents::patchbay_t< calibrator_t> calibrator_t::patchbay [private]`

5.60.3.3 vars `calibrator_variables_t calibrator_t::vars [private]`

5.60.3.4 prepared `bool calibrator_t::prepared [private]`

The documentation for this class was generated from the following file:

- **transducers.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.61 calibrator_variables_t Class Reference

Public Member Functions

- **calibrator_variables_t** (bool is_input, **MHAParser::parser_t** &parent)

Public Attributes

- **MHAParser::vfloat_t peaklevel**
- **MHAParser::mfloat_t fir**
- **MHAParser::int_t nbits**
- **MHAParser::float_t tau_level**
- **MHAParser::kw_t spnoise_mode**
- **MHAParser::vint_t spnoise_channels**
- **MHAParser::float_t spnoise_level**
- **MHAParser::vfloat_mon_t rmslevel**
- **MHAParser::parser_t spnoise_parser**
- **MHAParser::float_mon_t srate**
- **MHAParser::int_mon_t fragsize**
- **MHAParser::int_mon_t num_channels**
- **MHAParser::parser_t config_parser**
- **softclipper_variables_t softclip**
- **MHAParser::bool_t do_clipping**

5.61.1 Constructor & Destructor Documentation

5.61.1.1 calibrator_variables_t()

```
calibrator_variables_t::calibrator_variables_t (
    bool is_input,
    MHAParser::parser_t & parent )
```

5.61.2 Member Data Documentation

5.61.2.1 peaklevel

```
MHAParser::vfloat_t calibrator_variables_t::peaklevel
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.61.2.2 fir `MHAParser::mfloat_t calibrator_variables_t::fir`

5.61.2.3 nbits `MHAParser::int_t calibrator_variables_t::nbits`

5.61.2.4 tau_level `MHAParser::float_t calibrator_variables_t::tau_level`

5.61.2.5 spnoise_mode `MHAParser::kw_t calibrator_variables_t::spnoise_mode`

5.61.2.6 spnoise_channels `MHAParser::vint_t calibrator_variables_t::spnoise_channels`

5.61.2.7 spnoise_level `MHAParser::float_t calibrator_variables_t::spnoise_level`

5.61.2.8 rmslevel `MHAParser::vfloat_mon_t calibrator_variables_t::rmslevel`

5.61.2.9 spnoise_parser `MHAParser::parser_t calibrator_variables_t::spnoise_parser`

5.61.2.10 srate `MHAParser::float_mon_t calibrator_variables_t::srate`

© 2005-2021 HörTech gGmbH, Oldenburg

5.61.2.11 fragsize `MHAParser::int_mon_t calibrator_variables_t::fragsize`

5.61.2.12 num_channels `MHAParser::int_mon_t calibrator_variables_t::num_channels`

5.61.2.13 config_parser `MHAParser::parser_t calibrator_variables_t::config_parser`

5.61.2.14 softclip `softclipper_variables_t calibrator_variables_t::softclip`

5.61.2.15 do_clipping `MHAParser::bool_t calibrator_variables_t::do_clipping`

The documentation for this class was generated from the following file:

- **transducers.cpp**

5.62 cfg_t Class Reference

Public Member Functions

- **cfg_t** (unsigned int ichannel, unsigned int numchannels, const **mha_complex_t** &iscale)
- **cfg_t** (unsigned int, unsigned int)
- **cfg_t** (**mhaconfig_t** chcfg, **mha_real_t** newlev, bool replace, **mha_real_t** len, int seed)
- void **process** (**mha_wave_t** ∗)
- void **process** (**mha_spec_t** ∗)
- **cfg_t** (**mha_real_t** tau_attack, **mha_real_t** tau_decay, unsigned int nch, **mha_real_t** start_limit, **mha_real_t** slope_db, **mha_real_t** fs)

Public Attributes

- unsigned int **channel**
- **mha_complex_t scale**
- **mha_real_t start_lin**
- **mha_real_t alpha**
- **MHAFilter::o1flt_lowpass_t attack**
- **MHAFilter::o1flt_maxtrack_t decay**

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **mha_real_t gain_wave_**
- **mha_real_t gain_spec_**
- bool **replace_**
- bool **use_frozen_**
- **MHASignal::waveform_t frozen_noise_**
- unsigned int **pos**
- std::default_random_engine **rng**
- std::uniform_real_distribution< **mha_real_t** > **rand_dist**

5.62.1 Constructor & Destructor Documentation

5.62.1.1 cfg_t() [1/4]

```
cfg_t::cfg_t (
    unsigned int ichannel,
    unsigned int numchannels,
    const mha_complex_t & iscale )
```

5.62.1.2 cfg_t() [2/4]

```
cfg_t::cfg_t (
    unsigned int ichannel,
    unsigned int numchannels )
```

5.62.1.3 cfg_t() [3/4]

```
cfg_t::cfg_t (
    mhaconfig_t chcfg,
    mha_real_t newlev,
    bool replace,
    mha_real_t len,
    int seed )
```

5.62.1.4 cfg_t() [4/4]

```
cfg_t::cfg_t (
    mha_real_t tau_attack,
    mha_real_t tau_decay,
    unsigned int nch,
    mha_real_t start_limit,
    mha_real_t slope_db,
    mha_real_t fs )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.62.2 Member Function Documentation

5.62.2.1 process() [1/2]

```
void cfg_t::process (
    mha_wave_t * s ) [inline]
```

5.62.2.2 process() [2/2]

```
void cfg_t::process (
    mha_spec_t * s ) [inline]
```

5.62.3 Member Data Documentation

5.62.3.1 channel

```
unsigned int cfg_t::channel
```

5.62.3.2 scale

```
mha_complex_t cfg_t::scale
```

5.62.3.3 gain_wave_

```
mha_real_t cfg_t::gain_wave_ [private]
```

5.62.3.4 gain_spec_

```
mha_real_t cfg_t::gain_spec_ [private]
```

5.62.3.5 replace_

```
bool cfg_t::replace_ [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.62.3.6 use_frozen_ `bool cfg_t::use_frozen_ [private]`

5.62.3.7 frozen_noise_ `MHASignal::waveform_t cfg_t::frozen_noise_ [private]`

5.62.3.8 pos `unsigned int cfg_t::pos [private]`

5.62.3.9 rng `std::default_random_engine cfg_t::rng [private]`

5.62.3.10 rand_dist `std::uniform_real_distribution< mha_real_t> cfg_t::rand_dist [private]`

5.62.3.11 start_lin `mha_real_t cfg_t::start_lin`

5.62.3.12 alpha `mha_real_t cfg_t::alpha`

5.62.3.13 attack `MHAFilter::o1flt_lowpass_t cfg_t::attack`

5.62.3.14 decay `MHAFilter::o1flt_maxtrack_t cfg_t::decay`

The documentation for this class was generated from the following files:

- **complex_scale_channel.cpp**
- **example6.cpp**
- **noise.cpp**
- **softclip.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.63 coherence::cohflt_if_t Class Reference

Inheritance diagram for coherence::cohflt_if_t:

Public Member Functions

- **cohflt_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

Private Member Functions

- void **update** ()

Private Attributes

- **MHAEvents::patchbay_t**< **cohflt_if_t** > **patchbay**
- **vars_t vars**
- const std::string **algo**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.63.1 Constructor & Destructor Documentation

5.63.1.1 cohflt_if_t()

```
coherence::cohflt_if_t::cohflt_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.63.2 Member Function Documentation

5.63.2.1 prepare()

```
void coherence::cohflt_if_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< cohflt_t >** (p. 1153).

5.63.2.2 release()

```
void coherence::cohflt_if_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< cohflt_t >** (p. 1154).

5.63.2.3 process()

```
mha_spec_t * coherence::cohflt_if_t::process (
    mha_spec_t * s )
```

5.63.2.4 update()

```
void coherence::cohflt_if_t::update ( ) [private]
```

5.63.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.63.3.1 patchbay `MHAEvents::patchbay_t< cohflt_if_t> coherence::cohflt_if_t↩ ::patchbay [private]`

5.63.3.2 vars `vars_t coherence::cohflt_if_t::vars [private]`

5.63.3.3 algo `const std::string coherence::cohflt_if_t::algo [private]`

The documentation for this class was generated from the following file:

- **coherence.cpp**

5.64 coherence::cohflt_t Class Reference

Inheritance diagram for coherence::cohflt_t:

Public Member Functions

- **cohflt_t** (**vars_t** &v, const **mhaconfig_t** &icf, **algo_comm_t** iac, const std::string &name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- void **insert** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- unsigned int **nbands**
- bool **avg_ipd**
- **mha_complex_t cg**
- float **g**
- float **c_scale**
- float **c_min**
- **MHASignal::waveform_t alpha**
- float **limit**
- **MHAFilter::o1flt_lowpass_t lp1r**
- **MHAFilter::o1flt_lowpass_t lp1i**
- **MHA_AC::spectrum_t coh_c**
- **MHA_AC::waveform_t coh_rlp**
- **MHASignal::waveform_t gain**
- **MHASignal::delay_wave_t gain_delay**
- **MHASignal::spectrum_t s_out**
- bool **bInvert**
- **MHAFilter::o1flt_lowpass_t lp1ltg**
- bool **b_ltg**
- std::vector< float > **staticgain**

Additional Inherited Members

5.64.1 Constructor & Destructor Documentation

5.64.1.1 cohflt_t()

```
coherence::cohflt_t::cohflt_t (
    vars_t & v,
    const mhaconfig_t & icf,
    algo_comm_t iac,
    const std::string & name )
```

5.64.2 Member Function Documentation

5.64.2.1 process()

```
mha_spec_t * coherence::cohflt_t::process (
    mha_spec_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.64.2.2 insert() `void coherence::cohflt_t::insert ( )`

5.64.3 Member Data Documentation

5.64.3.1 nbands `unsigned int coherence::cohflt_t::nbands [private]`

5.64.3.2 avg_ipd `bool coherence::cohflt_t::avg_ipd [private]`

5.64.3.3 cg `mha_complex_t coherence::cohflt_t::cg [private]`

5.64.3.4 g `float coherence::cohflt_t::g [private]`

5.64.3.5 c_scale `float coherence::cohflt_t::c_scale [private]`

5.64.3.6 c_min `float coherence::cohflt_t::c_min [private]`

5.64.3.7 alpha `MHASignal::waveform_t coherence::cohflt_t::alpha [private]`

5.64.3.8 limit `float coherence::cohflt_t::limit [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.64.3.9 lp1r

MHAFilter::o1flt_lowpass_t coherence::cohflt_t::lp1r [private]

5.64.3.10 lp1i

MHAFilter::o1flt_lowpass_t coherence::cohflt_t::lp1i [private]

5.64.3.11 coh_c

MHA_AC::spectrum_t coherence::cohflt_t::coh_c [private]

5.64.3.12 coh_rlp

MHA_AC::waveform_t coherence::cohflt_t::coh_rlp [private]

5.64.3.13 gain

MHASignal::waveform_t coherence::cohflt_t::gain [private]

5.64.3.14 gain_delay

MHASignal::delay_wave_t coherence::cohflt_t::gain_delay [private]

5.64.3.15 s_out

MHASignal::spectrum_t coherence::cohflt_t::s_out [private]

5.64.3.16 bInvert

bool coherence::cohflt_t::bInvert [private]

5.64.3.17 lp1ltg

MHAFilter::o1flt_lowpass_t coherence::cohflt_t::lp1ltg [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.64.3.18 b_ltg `bool coherence::cohflt_t::b_ltg [private]`

5.64.3.19 staticgain `std::vector<float> coherence::cohflt_t::staticgain [private]`

The documentation for this class was generated from the following file:

- **coherence.cpp**

5.65 coherence::vars_t Class Reference

Inheritance diagram for coherence::vars_t:

Public Member Functions

- **vars_t** (**MHAParser::parser_t** $*$)

Public Attributes

- **MHAParser::kw_t tau_unit**
- **MHAParser::vfloat_t tau**
- **MHAParser::vfloat_t alpha**
- **MHAParser::float_t limit**
- **MHAParser::vfloat_t mapping**
- **MHAParser::kw_t average**
- **MHAParser::bool_t invert**
- **MHAParser::bool_t ltgcomp**
- **MHAParser::vfloat_t ltgtau**
- **MHAParser::vfloat_t staticgain**
- **MHAParser::int_t delay**

© 2005-2021 HörTech gGmbH, Oldenburg

5.65.1 Constructor & Destructor Documentation

5.65.1.1 vars_t()

`coherence::vars_t::vars_t ( MHAParser::parser_t * p )`

5.65.2 Member Data Documentation

5.65.2.1 tau_unit

`MHAParser::kw_t coherence::vars_t::tau_unit`

5.65.2.2 tau

`MHAParser::vfloat_t coherence::vars_t::tau`

5.65.2.3 alpha

`MHAParser::vfloat_t coherence::vars_t::alpha`

5.65.2.4 limit

`MHAParser::float_t coherence::vars_t::limit`

5.65.2.5 mapping

`MHAParser::vfloat_t coherence::vars_t::mapping`

5.65.2.6 average

`MHAParser::kw_t coherence::vars_t::average`

© 2005-2021 HörTech gGmbH, Oldenburg

5.65.2.7 invert `MHAParser::bool_t coherence::vars_t::invert`

5.65.2.8 ltgcomp `MHAParser::bool_t coherence::vars_t::ltgcomp`

5.65.2.9 ltgtau `MHAParser::vfloat_t coherence::vars_t::ltgtau`

5.65.2.10 staticgain `MHAParser::vfloat_t coherence::vars_t::staticgain`

5.65.2.11 delay `MHAParser::int_t coherence::vars_t::delay`

The documentation for this class was generated from the following file:

- **coherence.cpp**

5.66 combc_if_t Class Reference

Inheritance diagram for combc_if_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **combc_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

Private Attributes

- **MHAParser::int_t outchannels**
- **MHAParser::bool_t interleaved**
- **MHAParser::string_t channel_gain_name**
- **MHAParser::string_t element_gain_name**

Additional Inherited Members

5.66.1 Constructor & Destructor Documentation

5.66.1.1 combc_if_t()

```
combc_if_t::combc_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.66.2 Member Function Documentation

5.66.2.1 prepare()

```
void combc_if_t::prepare (
    mhaconfig_t & chcfg ) [virtual]
```

Implements **MHAPlugin::plugin_t**< **combc_t** > (p. 1153).

5.66.2.2 process() [1/2]

```
mha_wave_t * combc_if_t::process (
    mha_wave_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.66.2.3 process() [2/2] `mha_spec_t * combc_if_t::process ( mha_spec_t * s )`

5.66.3 Member Data Documentation

5.66.3.1 outchannels `MHAParser::int_t combc_if_t::outchannels [private]`

5.66.3.2 interleaved `MHAParser::bool_t combc_if_t::interleaved [private]`

5.66.3.3 channel_gain_name `MHAParser::string_t combc_if_t::channel_gain_name [private]`

5.66.3.4 element_gain_name `MHAParser::string_t combc_if_t::element_gain_name [private]`

The documentation for this class was generated from the following file:

- **combinechannels.cpp**

5.67 combc_t Class Reference

Public Member Functions

- **combc_t** (**algo_comm_t** ac, **mhaconfig_t** cfg_input, **mhaconfig_t** cfg_output, std::vector< float > channel_gains, const std::string &element_gain_name, bool interleaved)
- **mha_wave_t** * **process** (**mha_wave_t** *s)
- **mha_spec_t** * **process** (**mha_spec_t** *s)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **algo_comm_t ac_**
- bool **interleaved_**
- unsigned int **nbands**
- **MHASignal::waveform_t w_out**
- **MHASignal::spectrum_t s_out**
- std::vector< **mha_real_t** > **channel_gains_**
- std::string **element_gain_name_**

5.67.1 Constructor & Destructor Documentation

5.67.1.1 combc_t()

```
combc_t::combc_t (
    algo_comm_t ac,
    mhaconfig_t cfg_input,
    mhaconfig_t cfg_output,
    std::vector< float > channel_gains,
    const std::string & element_gain_name,
    bool interleaved )
```

5.67.2 Member Function Documentation

5.67.2.1 process() [1/2]

```
mha_wave_t * combc_t::process (
    mha_wave_t * s )
```

5.67.2.2 process() [2/2]

```
mha_spec_t * combc_t::process (
    mha_spec_t * s )
```

5.67.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.67.3.1 ac_ `algo_comm_t combc_t::ac_ [private]`

5.67.3.2 interleaved_ `bool combc_t::interleaved_ [private]`

5.67.3.3 nbands `unsigned int combc_t::nbands [private]`

5.67.3.4 w_out `MHASignal::waveform_t combc_t::w_out [private]`

5.67.3.5 s_out `MHASignal::spectrum_t combc_t::s_out [private]`

5.67.3.6 channel_gains_ `std::vector< mha_real_t> combc_t::channel_gains_ [private]`

5.67.3.7 element_gain_name_ `std::string combc_t::element_gain_name_ [private]`

The documentation for this class was generated from the following file:

- **combinechannels.cpp**

5.68 comm_var_t Struct Reference

Algorithm communication variable structure.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- unsigned int **data_type**
  Type of data.
- unsigned int **num_entries**
  Number of entries.
- unsigned int **stride**
  length of one row (C interpretation) or of one column (Fortran interpretation)
- void * **data**
  Pointer to variable data.

5.68.1 Detailed Description

Algorithm communication variable structure.

Algorithm communication variables (AC variables) are objects of this type. The member data is a pointer to the variable 'data'. This pointer has to be valid for the lifetime of this AC variable. The member 'data_type' can be one of the predefined types or any user defined type. The member 'num_entries' describes the number of elements of this base type stored at the pointer address.

```
An AC variable can be registered with the \ref
algo_comm_t::insert_var "insert_var" function.
```

5.68.2 Member Data Documentation

5.68.2.1 data_type `comm_var_t::data_type`

Type of data.

This can be one of the predefined types

- MHA_AC_CHAR
- MHA_AC_INT
- MHA_AC_MHAREAL
- MHA_AC_FLOAT
- MHA_AC_DOUBLE
- MHA_AC_MHACOMPLEX
- MHA_AC_VEC_FLOAT or any user defined type with a value greater than
- MHA_AC_USER

© 2005-2021 HörTech gGmbH, Oldenburg

5.68.2.2 num_entries `comm_var_t::num_entries`

Number of entries.

5.68.2.3 stride `comm_var_t::stride`

length of one row (C interpretation) or of one column (Fortran interpretation)

5.68.2.4 data `comm_var_t::data`

Pointer to variable data.

The documentation for this struct was generated from the following files:

- **mha.hh**
- **mha_algo_comm.cpp**

5.69 complex_scale_channel_t Class Reference

Inheritance diagram for complex_scale_channel_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **complex_scale_channel_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** $*$ **process** (**mha_spec_t** $*$)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t**$<$ **complex_scale_channel_t** $>$ **patchbay**
- **MHAParser::int_t scale_ch**
- **MHAParser::complex_t factor**

Additional Inherited Members

5.69.1 Constructor & Destructor Documentation

5.69.1.1 complex_scale_channel_t()

```
complex_scale_channel_t::complex_scale_channel↩
_t (
            algo_comm_t iac,
           const std::string & configured_name )
```

5.69.2 Member Function Documentation

5.69.2.1 process()

```
mha_spec_t * complex_scale_channel_t::process (
            mha_spec_t * spec )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.69.2.2 prepare() `void complex_scale_channel_t::prepare ( mhaconfig_t & cfg ) [virtual]`

Implements **MHAPlugin::plugin_t**< **cfg_t** > (p. 1153).

5.69.2.3 update_cfg() `void complex_scale_channel_t::update_cfg ( ) [private]`

5.69.3 Member Data Documentation

5.69.3.1 patchbay `MHAEvents::patchbay_t< complex_scale_channel_t> complex_scale↩_channel_t::patchbay [private]`

5.69.3.2 scale_ch `MHAParser::int_t complex_scale_channel_t::scale_ch [private]`

5.69.3.3 factor `MHAParser::complex_t complex_scale_channel_t::factor [private]`

The documentation for this class was generated from the following file:

- **complex_scale_channel.cpp**

5.70 cpuload::cpuload_cfg_t Class Reference

Public Member Functions

- **cpuload_cfg_t** (**mha_real_t** factor_, size_t table_size_, bool use_sine_)
  Ctor of the runtime configuration class.
- void **process** (unsigned fac_)
  Process callback. Does not actually change signal.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **calc_sine** ()
- void **write_to_table** ()

Private Attributes

- float **phase** =0

  Phase of the sine.
- volatile float **result** =0

  Result of sin(phase). Volatile to prevent compiler from optimizing away the calculation.
- bool **use_sine** =false

  Use sine or do table operation.
- float **factor** =0

  cpu load factor. Values > 1 increase cpu load, values < 1 decrease it
- std::vector< float > **table**

  Table with arbitrary values to operate on. Unused if use_sine=true.

5.70.1 Constructor & Destructor Documentation

5.70.1.1 cpuload_cfg_t()

```
cpuload::cpuload_cfg_t::cpuload_cfg_t (
    mha_real_t factor_,
    size_t table_size_,
    bool use_sine_ )
```

Ctor of the runtime configuration class.

Parameters

*factor_*	cpu load factor. Values > 1 increase cpu load, values < 1 decrease it
*table_size_*	
*use_sine_*	

5.70.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.70.2.1 process() `void cpuload::cpuload_cfg_t::process ( unsigned fac_ )`

Process callback. Does not actually change signal.

5.70.2.2 calc_sine() `void cpuload::cpuload_cfg_t::calc_sine ( ) [private]`

5.70.2.3 write_to_table() `void cpuload::cpuload_cfg_t::write_to_table ( ) [private]`

5.70.3 Member Data Documentation

5.70.3.1 phase `float cpuload::cpuload_cfg_t::phase =0 [private]`

Phase of the sine.

5.70.3.2 result `volatile float cpuload::cpuload_cfg_t::result =0 [private]`

Result of sin(phase). Volatile to prevent compiler from optimizing away the calculation.

5.70.3.3 use_sine `bool cpuload::cpuload_cfg_t::use_sine =false [private]`

Use sine or do table operation.

5.70.3.4 factor `float cpuload::cpuload_cfg_t::factor =0 [private]`

cpu load factor. Values $>$ 1 increase cpu load, values $<$ 1 decrease it

© 2005-2021 HörTech gGmbH, Oldenburg

5.70.3.5 table `std::vector<float> cpuload::cpuload_cfg_t::table [private]`

Table with arbitrary values to operate on. Unused if use_sine=true.

The documentation for this class was generated from the following file:

- **cpuload.cpp**

5.71 cpuload::cpuload_if_t Class Reference

Inheritance diagram for cpuload::cpuload_if_t:

Public Member Functions

- **cpuload_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::float_t factor**
- **MHAParser::int_t table_size**
- **MHAParser::bool_t use_sine**
- **MHAEvents::patchbay_t< cpuload_if_t > patchbay**

Additional Inherited Members

5.71.1 Constructor & Destructor Documentation

5.71.1.1 cpuload_if_t()

```
cpuload::cpuload_if_t::cpuload_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.71.2 Member Function Documentation

5.71.2.1 process() [1/2]

```
mha_spec_t * cpuload::cpuload_if_t::process (
    mha_spec_t * s )
```

5.71.2.2 process() [2/2]

```
mha_wave_t * cpuload::cpuload_if_t::process (
    mha_wave_t * s )
```

5.71.2.3 prepare()

```
void cpuload::cpuload_if_t::prepare (
    mhaconfig_t & ) [virtual]
```

Implements **MHAPlugin::plugin_t< cpuload_cfg_t >** (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.71.2.4 update() `void cpuload::cpuload_if_t::update ( ) [private]`

5.71.3 Member Data Documentation

5.71.3.1 factor `MHAParser::float_t cpuload::cpuload_if_t::factor [private]`

5.71.3.2 table_size `MHAParser::int_t cpuload::cpuload_if_t::table_size [private]`

5.71.3.3 use_sine `MHAParser::bool_t cpuload::cpuload_if_t::use_sine [private]`

5.71.3.4 patchbay `MHAEvents::patchbay_t< cpuload_if_t> cpuload::cpuload_if_t↩ ::patchbay [private]`

The documentation for this class was generated from the following file:

- **cpuload.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.72 db_if_t Class Reference

Inheritance diagram for db_if_t:

Public Member Functions

- **db_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- ∼**db_if_t** ()

Private Attributes

- **MHAEvents::patchbay_t**< **db_if_t** > **patchbay**
- **MHAParser::int_t fragsize**
- **MHAParser::mhapluginloader_t plugloader**
- std::string **algo**
- bool **bypass**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.72.1 Constructor & Destructor Documentation

5.72.1.1 db_if_t()

```
db_if_t::db_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.72.1.2 ~db_if_t()

```
db_if_t::~db_if_t ( )
```

5.72.2 Member Function Documentation

5.72.2.1 process()

```
mha_wave_t * db_if_t::process (
    mha_wave_t * s )
```

5.72.2.2 prepare()

```
void db_if_t::prepare (
    mhaconfig_t & conf ) [virtual]
```

Implements **MHAPlugin::plugin_t< db_t >** (p. 1153).

5.72.2.3 release()

```
void db_if_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< db_t >** (p. 1154).

5.72.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.72.3.1 patchbay `MHAEvents::patchbay_t< db_if_t > db_if_t::patchbay [private]`

5.72.3.2 fragsize `MHAParser::int_t db_if_t::fragsize [private]`

5.72.3.3 plugloader `MHAParser::mhapluginloader_t db_if_t::plugloader [private]`

5.72.3.4 algo `std::string db_if_t::algo [private]`

5.72.3.5 bypass `bool db_if_t::bypass [private]`

The documentation for this class was generated from the following file:

- **db.cpp**

5.73 db_t Class Reference

Inheritance diagram for db_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **db_t** (unsigned int outer_fragsize, unsigned int inner_fragsize, unsigned int nch_in, unsigned int nch_out, **MHAParser::mhapluginloader_t** &plug)
- **mha_wave_t** ∗ **inner_process** (**mha_wave_t** ∗)

Private Attributes

- **MHAParser::mhapluginloader_t** & **plugloader**

Additional Inherited Members

5.73.1 Constructor & Destructor Documentation

5.73.1.1 db_t()

```
db_t::db_t (
    unsigned int outer_fragsize,
    unsigned int inner_fragsize,
    unsigned int nch_in,
    unsigned int nch_out,
    MHAParser::mhapluginloader_t & plug )
```

5.73.2 Member Function Documentation

5.73.2.1 inner_process()

```
mha_wave_t * db_t::inner_process (
    mha_wave_t * s ) [virtual]
```

Implements **MHASignal::doublebuffer_t** (p. 1209).

5.73.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.73.3.1 plugloader `MHAParser::mhapluginloader_t& db_t::plugloader [private]`

The documentation for this class was generated from the following file:

- **db.cpp**

5.74 dbasync_native::db_if_t Class Reference

Inheritance diagram for dbasync_native::db_if_t:

Public Member Functions

- **db_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- ∼**db_if_t** ()=default

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAKernel::algo_comm_class_t sub_ac**
- **MHAParser::mhapluginloader_t plugloader**
- **MHAParser::int_t fragsize**
- **MHAParser::int_t delay**
- **MHAParser::kw_t worker_thread_scheduler**
  Scheduler used for worker thread.
- **MHAParser::int_t worker_thread_priority**
  Priority of worker thread.
- **MHAParser::string_mon_t framework_thread_scheduler**
  Scheduler of the signal processing thread.
- **MHAParser::int_mon_t framework_thread_priority**
  Priority of signal processing thread.
- std::string **algo**

Additional Inherited Members

5.74.1 Constructor & Destructor Documentation

5.74.1.1 db_if_t()

```
db_if_t::db_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.74.1.2 ~db_if_t()

```
dbasync_native::db_if_t::~db_if_t ( ) [default]
```

5.74.2 Member Function Documentation

5.74.2.1 process()

```
mha_wave_t * db_if_t::process (
    mha_wave_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.74.2.2 prepare() `void db_if_t::prepare ( mhaconfig_t & conf ) [virtual]`

Implements **MHAPlugin::plugin_t**< **dbasync_t** > (p. 1153).

5.74.2.3 release() `void db_if_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t**< **dbasync_t** > (p. 1154).

5.74.3 Member Data Documentation

5.74.3.1 sub_ac `MHAKernel::algo_comm_class_t dbasync_native::db_if_t::sub_ac [private]`

5.74.3.2 plugloader `MHAParser::mhapluginloader_t dbasync_native::db_if_t::plugloader [private]`

5.74.3.3 fragsize `MHAParser::int_t dbasync_native::db_if_t::fragsize [private]`

5.74.3.4 delay `MHAParser::int_t dbasync_native::db_if_t::delay [private]`

5.74.3.5 worker_thread_scheduler `MHAParser::kw_t dbasync_native::db_if_t::worker_thread_scheduler [private]`

Scheduler used for worker thread.

© 2005-2021 HörTech gGmbH, Oldenburg

5.74.3.6 worker_thread_priority

`MHAParser::int_t dbasync_native::db_if_t::worker_thread_priority [private]`

Priority of worker thread.

5.74.3.7 framework_thread_scheduler

`MHAParser::string_mon_t dbasync_native::db_if_t::framework_thread_scheduler [private]`

Scheduler of the signal processing thread.

5.74.3.8 framework_thread_priority

`MHAParser::int_mon_t dbasync_native::db_if_t::framework_thread_priority [private]`

Priority of signal processing thread.

5.74.3.9 algo

`std::string dbasync_native::db_if_t::algo [private]`

The documentation for this class was generated from the following file:

- **dbasync.cpp**

5.75 dbasync_native::dbasync_t Class Reference

Inheritance diagram for dbasync_native::dbasync_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **dbasync_t** (unsigned int nchannels_in, unsigned int nchannels_out, unsigned int outer↩ _fragsize, unsigned int inner_fragsize, int **delay**, const std::string &thread_scheduler, int thread_priority, **MHAParser::mhapluginloader_t** &plug)
- ∼**dbasync_t** ()
- **mha_wave_t** ∗ **outer_process** (**mha_wave_t** ∗)
- int **svc** ()

Private Attributes

- **MHAParser::mhapluginloader_t** & **plugloader**
- **MHASignal::waveform_t** **inner_input**
- **MHASignal::waveform_t** **outer_output**
- pthread_attr_t **attr**
- struct sched_param **priority**
- int **scheduler**
- pthread_t **thread**

Additional Inherited Members

5.75.1 Constructor & Destructor Documentation

5.75.1.1 dbasync_t()

```
dbasync_native::dbasync_t::dbasync_t (
    unsigned int nchannels_in,
    unsigned int nchannels_out,
    unsigned int outer_fragsize,
    unsigned int inner_fragsize,
    int delay,
    const std::string & thread_scheduler,
    int thread_priority,
    MHAParser::mhapluginloader_t & plug )
```

5.75.1.2 ∼dbasync_t()

```
dbasync_native::dbasync_t::~dbasync_t ( )
```

5.75.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.75.2.1 outer_process()

`mha_wave_t * dbasync_native::dbasync_t::outer_process ( mha_wave_t * outer_input )`

5.75.2.2 svc()

`int dbasync_native::dbasync_t::svc ( )`

5.75.3 Member Data Documentation

5.75.3.1 plugloader

`MHAParser::mhapluginloader_t& dbasync_native::dbasync_t↩ ::plugloader [private]`

5.75.3.2 inner_input

`MHASignal::waveform_t dbasync_native::dbasync_t::inner_input [private]`

5.75.3.3 outer_output

`MHASignal::waveform_t dbasync_native::dbasync_t::outer_↩ output [private]`

5.75.3.4 attr

`pthread_attr_t dbasync_native::dbasync_t::attr [private]`

5.75.3.5 priority

`struct sched_param dbasync_native::dbasync_t::priority [private]`

5.75.3.6 scheduler

`int dbasync_native::dbasync_t::scheduler [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.75.3.7 thread `pthread_t dbasync_native::dbasync_t::thread [private]`

The documentation for this class was generated from the following file:

- **dbasync.cpp**

5.76 dbasync_native::delay_check_t Class Reference

Inheritance diagram for dbasync_native::delay_check_t:

Protected Member Functions

- **delay_check_t** (int delay, unsigned inner_fragsize, unsigned outer_fragsize)

5.76.1 Constructor & Destructor Documentation

5.76.1.1 delay_check_t()

```
dbasync_native::delay_check_t::delay_check_t (
    int delay,
    unsigned inner_fragsize,
    unsigned outer_fragsize ) [protected]
```

The documentation for this class was generated from the following file:

- **dbasync.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.77 dc::dc_if_t Class Reference

Inheritance diagram for dc::dc_if_t:

Public Member Functions

- **dc_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &tf)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

Private Member Functions

- void **update_monitors** ()

  Called from within the processing routines: updates the monitor variables.
- void **update** ()

  Called by MHA configuration change event mechanism: creates new runtime configuration.

Private Attributes

- std::string **algo**
- **MHAEvents::patchbay_t**< **dc_if_t** > **patchbay**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.77.1 Constructor & Destructor Documentation

5.77.1.1 dc_if_t()

```
dc_if_t::dc_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.77.2 Member Function Documentation

5.77.2.1 prepare()

```
void dc_if_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t**< **dc_t** > (p. 1153).

5.77.2.2 process() [1/2]

```
mha_wave_t * dc_if_t::process (
    mha_wave_t * s_in )
```

5.77.2.3 process() [2/2]

```
mha_spec_t * dc_if_t::process (
    mha_spec_t * s_in )
```

5.77.2.4 update_monitors()

```
void dc_if_t::update_monitors ( ) [private]
```

Called from within the processing routines: updates the monitor variables.

© 2005-2021 HörTech gGmbH, Oldenburg

5.77.2.5 update() `void dc_if_t::update ( ) [private]`

Called by MHA configuration change event mechanism: creates new runtime configuration.

5.77.3 Member Data Documentation

5.77.3.1 algo `std::string dc::dc_if_t::algo [private]`

5.77.3.2 patchbay `MHAEvents::patchbay_t< dc_if_t> dc::dc_if_t::patchbay [private]`

The documentation for this class was generated from the following files:

- **dc.hh**
- **dc.cpp**

5.78 dc::dc_t Class Reference

Inheritance diagram for dc::dc_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **dc_t** (**dc_vars_t** vars, **mha_real_t** filter_rate, unsigned int nch_, **algo_comm_t** ac, **mha_domain_t** domain, unsigned int **fftlen**, const std::string &algo, const std::vector< **mha_real_t** > &rmslevel_state={}, const std::vector< **mha_real_t** > &attack_state={}, const std::vector< **mha_real_t** > &decay_state={})
- **mha_wave_t** * **process** (**mha_wave_t** *)
- **mha_spec_t** * **process** (**mha_spec_t** *)
- void **explicit_insert** ()
- unsigned **get_nbands** () const

  Number of frequency bands accessor.
- unsigned **get_nch** () const

  Total number of channels accessor.
- const **MHASignal::waveform_t** & **get_level_in_db** () const
- const **MHASignal::waveform_t** & **get_level_in_db_adjusted** () const
- std::vector< **mha_real_t** > **get_rmslevel_filter_state** () const
- std::vector< **mha_real_t** > **get_attack_filter_state** () const
- std::vector< **mha_real_t** > **get_decay_filter_state** () const

Private Attributes

- std::vector< **MHATableLookup::linear_table_t** > **gt**
- std::vector< **mha_real_t** > **offset**
- **MHAFilter::o1flt_lowpass_t** **rmslevel**
- **MHAFilter::o1flt_lowpass_t** **attack**
- **MHAFilter::o1flt_maxtrack_t** **decay**
- bool **bypass**
- bool **log_interp**
- unsigned int **naudiochannels**
- unsigned int **nbands**
- unsigned int **nch**
- **MHA_AC::waveform_t** **level_in_db**
- **MHA_AC::waveform_t** **level_in_db_adjusted**
- unsigned int **fftlen**

Additional Inherited Members

5.78.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.78.1.1 dc_t()

```
dc_t::dc_t (
    dc_vars_t vars,
    mha_real_t filter_rate,
    unsigned int nch_,
    algo_comm_t ac,
    mha_domain_t domain,
    unsigned int fftlen,
    const std::string & algo,
    const std::vector< mha_real_t > & rmslevel_state = {},
    const std::vector< mha_real_t > & attack_state = {},
    const std::vector< mha_real_t > & decay_state = {} )
```

5.78.2 Member Function Documentation

5.78.2.1 process() [1/2]

```
mha_wave_t * dc_t::process (
    mha_wave_t * s )
```

5.78.2.2 process() [2/2]

```
mha_spec_t * dc_t::process (
    mha_spec_t * s )
```

5.78.2.3 explicit_insert()

```
void dc_t::explicit_insert ( )
```

5.78.2.4 get_nbands()

```
unsigned dc::dc_t::get_nbands ( ) const [inline]
```

Number of frequency bands accessor.

5.78.2.5 get_nch()

```
unsigned dc::dc_t::get_nch ( ) const [inline]
```

Total number of channels accessor.

© 2005-2021 HörTech gGmbH, Oldenburg

5.78.2.6 get_level_in_db() `const MHASignal::waveform_t& dc::dc_t::get_level_in_db ( ) const [inline]`

5.78.2.7 get_level_in_db_adjusted() `const MHASignal::waveform_t& dc::dc_t::get_level_in_db_adjusted ( ) const [inline]`

5.78.2.8 get_rmslevel_filter_state() `std::vector< mha_real_t> dc::dc_t::get_rmslevel_filter_state ( ) const [inline]`

5.78.2.9 get_attack_filter_state() `std::vector< mha_real_t> dc::dc_t::get_attack_filter_state ( ) const [inline]`

5.78.2.10 get_decay_filter_state() `std::vector< mha_real_t> dc::dc_t::get_decay_filter_state ( ) const [inline]`

5.78.3 Member Data Documentation

5.78.3.1 gt `std::vector< MHATableLookup::linear_table_t> dc::dc_t::gt [private]`

5.78.3.2 offset `std::vector< mha_real_t> dc::dc_t::offset [private]`

5.78.3.3 rmslevel `MHAFilter::o1flt_lowpass_t dc::dc_t::rmslevel [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.78.3.4 attack `MHAFilter::o1flt_lowpass_t` dc::dc_t::attack [private]

5.78.3.5 decay `MHAFilter::o1flt_maxtrack_t` dc::dc_t::decay [private]

5.78.3.6 bypass bool dc::dc_t::bypass [private]

5.78.3.7 log_interp bool dc::dc_t::log_interp [private]

5.78.3.8 naudiochannels unsigned int dc::dc_t::naudiochannels [private]

5.78.3.9 nbands unsigned int dc::dc_t::nbands [private]

5.78.3.10 nch unsigned int dc::dc_t::nch [private]

5.78.3.11 level_in_db `MHA_AC::waveform_t` dc::dc_t::level_in_db [private]

5.78.3.12 level_in_db_adjusted `MHA_AC::waveform_t` dc::dc_t::level_in_db_adjusted [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.78.3.13 fftlen `unsigned int dc::dc_t::fftlen [private]`

The documentation for this class was generated from the following files:

- **dc.hh**
- **dc.cpp**

5.79 dc::dc_vars_t Class Reference

Inheritance diagram for dc::dc_vars_t:

Public Member Functions

- **dc_vars_t** (**MHAParser::parser_t** &)

Public Attributes

- **MHAParser::mfloat_t gtdata**
- **MHAParser::vfloat_t gtmin**
- **MHAParser::vfloat_t gtstep**
- **MHAParser::vfloat_t taurmslevel**
- **MHAParser::vfloat_t tauattack**
- **MHAParser::vfloat_t taudecay**
- **MHAParser::vfloat_t offset**
- **MHAParser::string_t filterbank**
- std::string **cf_name**
- std::string **ef_name**
- std::string **bw_name**
- **MHAParser::string_t chname**
- **MHAParser::bool_t bypass**

© 2005-2021 HörTech gGmbH, Oldenburg

- **MHAParser::bool_t log_interp**
- **MHAParser::string_t clientid**
- **MHAParser::string_t gainrule**
- **MHAParser::string_t preset**
- **MHAParser::int_mon_t modified**
- **MHAParser::vfloat_mon_t input_level**
- **MHAParser::vfloat_mon_t filtered_level**
- **MHAParser::vfloat_mon_t center_frequencies**
- **MHAParser::vfloat_mon_t edge_frequencies**
- **MHAParser::vfloat_mon_t band_weights**

5.79.1 Constructor & Destructor Documentation

5.79.1.1 dc_vars_t()

`dc_vars_t::dc_vars_t ( MHAParser::parser_t & p ) [explicit]`

5.79.2 Member Data Documentation

5.79.2.1 gtdata

`MHAParser::mfloat_t dc::dc_vars_t::gtdata`

5.79.2.2 gtmin

`MHAParser::vfloat_t dc::dc_vars_t::gtmin`

5.79.2.3 gtstep

`MHAParser::vfloat_t dc::dc_vars_t::gtstep`

5.79.2.4 taurmslevel

`MHAParser::vfloat_t dc::dc_vars_t::taurmslevel`

© 2005-2021 HörTech gGmbH, Oldenburg

5.79.2.5 tauattack `MHAParser::vfloat_t dc::dc_vars_t::tauattack`

5.79.2.6 taudecay `MHAParser::vfloat_t dc::dc_vars_t::taudecay`

5.79.2.7 offset `MHAParser::vfloat_t dc::dc_vars_t::offset`

5.79.2.8 filterbank `MHAParser::string_t dc::dc_vars_t::filterbank`

5.79.2.9 cf_name `std::string dc::dc_vars_t::cf_name`

5.79.2.10 ef_name `std::string dc::dc_vars_t::ef_name`

5.79.2.11 bw_name `std::string dc::dc_vars_t::bw_name`

5.79.2.12 chname `MHAParser::string_t dc::dc_vars_t::chname`

5.79.2.13 bypass `MHAParser::bool_t dc::dc_vars_t::bypass`

© 2005-2021 HörTech gGmbH, Oldenburg

5.79.2.14 log_interp `MHAParser::bool_t` `dc::dc_vars_t::log_interp`

5.79.2.15 clientid `MHAParser::string_t` `dc::dc_vars_t::clientid`

5.79.2.16 gainrule `MHAParser::string_t` `dc::dc_vars_t::gainrule`

5.79.2.17 preset `MHAParser::string_t` `dc::dc_vars_t::preset`

5.79.2.18 modified `MHAParser::int_mon_t` `dc::dc_vars_t::modified`

5.79.2.19 input_level `MHAParser::vfloat_mon_t` `dc::dc_vars_t::input_level`

5.79.2.20 filtered_level `MHAParser::vfloat_mon_t` `dc::dc_vars_t::filtered_level`

5.79.2.21 center_frequencies `MHAParser::vfloat_mon_t` `dc::dc_vars_t::center_↩ frequencies`

5.79.2.22 edge_frequencies `MHAParser::vfloat_mon_t` `dc::dc_vars_t::edge_frequencies`

© 2005-2021 HörTech gGmbH, Oldenburg

5.79.2.23 band_weights `MHAParser::vfloat_mon_t dc::dc_vars_t::band_weights`

The documentation for this class was generated from the following files:

- **dc.hh**
- **dc.cpp**

5.80 dc::dc_vars_validator_t Class Reference

Inheritance diagram for dc::dc_vars_validator_t:

Public Member Functions

- **dc_vars_validator_t** (**dc_vars_t** &v, unsigned int s, **mha_domain_t** domain)

5.80.1 Constructor & Destructor Documentation

5.80.1.1 dc_vars_validator_t()

```
dc_vars_validator_t::dc_vars_validator_t (
    dc_vars_t & v,
    unsigned int s,
    mha_domain_t domain )
```

The documentation for this class was generated from the following files:

- **dc.hh**
- **dc.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.81 dc_simple::dc_if_t Class Reference

interface class for **dc_simple** (p. 86)

Inheritance diagram for dc_simple::dc_if_t:

Public Member Functions

- **dc_if_t** (**algo_comm_t** iac, const std::string &configured_name)
  *Constructor instantiates one **dc_simple** (p. 86) plugin.*
- void **prepare** (**mhaconfig_t** &tf)
  *Prepare **dc_simple** (p. 86) plugin for signal processing.*
- void **release** ()
  Sets prepared back to False.
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗s)
  Main process callback.
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s)
  Main process callback.

Private Member Functions

- void **update_dc** ()
  *Update **dc_t** (p. 397) runtime config when configuration parameters have changed.*
- void **update_level** ()
  *Update **level_smoother_t** (p. 407) runtime config when configuration parameters have changed.*

© 2005-2021 HörTech gGmbH, Oldenburg

- void **has_been_modified** ()
- void **read_modified** ()
- void **update_level_mon** ()

  *Updates the data of variable mon_l in **dc_t** (p. 397).*
- void **update_gain_mon** ()

  *Updates the data of variable mon_g in **dc_t** (p. 397).*

Private Attributes

- **MHAParser::string_t clientid**

  MHA Parser variables.
- **MHAParser::string_t gainrule**
- **MHAParser::string_t preset**
- **MHAParser::int_mon_t modified**
- **MHAParser::vfloat_mon_t mon_l**
- **MHAParser::vfloat_mon_t mon_g**
- **MHAParser::string_t filterbank**
- **MHAParser::vfloat_mon_t center_frequencies**
- **MHAParser::vfloat_mon_t edge_frequencies**
- **MHAEvents::patchbay_t**< **dc_if_t** > **patchbay**
- bool **prepared**

Additional Inherited Members

5.81.1 Detailed Description

interface class for **dc_simple** (p. 86)

5.81.2 Constructor & Destructor Documentation

5.81.2.1 dc_if_t()

```
dc_simple::dc_if_t::dc_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructor instantiates one **dc_simple** (p. 86) plugin.

5.81.3 Member Function Documentation

5.81.3.1 prepare()

```
void dc_simple::dc_if_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Prepare **dc_simple** (p. 86) plugin for signal processing.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

tf	signal_dimensions

Implements **MHAPlugin::plugin_t**< **dc_t** > (p. 1153).

5.81.3.2 release() `void dc_simple::dc_if_t::release ( ) [virtual]`

Sets prepared back to False.

Reimplemented from **MHAPlugin::plugin_t**< **dc_t** > (p. 1154).

5.81.3.3 process() [1/2] `mha_spec_t * dc_simple::dc_if_t::process ( mha_spec_t * s )`

Main process callback.

Takes mhatype spectrum input and calls type DC and LEVEL process methods, returning mhatype spectrum.

Parameters

s	input/output signal

5.81.3.4 process() [2/2] `mha_wave_t * dc_simple::dc_if_t::process ( mha_wave_t * s )`

Main process callback.

Takes mhatype wave input and calls type DC and LEVEL process methods, returning mhatype wave.

Parameters

s	input/output signal

© 2005-2021 HörTech gGmbH, Oldenburg

5.81.3.5 update_dc() `void dc_simple::dc_if_t::update_dc ( ) [private]`

Update **dc_t** (p. 397) runtime config when configuration parameters have changed.

5.81.3.6 update_level() `void dc_simple::dc_if_t::update_level ( ) [private]`

Update **level_smoother_t** (p. 407) runtime config when configuration parameters have changed.

5.81.3.7 has_been_modified() `void dc_simple::dc_if_t::has_been_modified ( ) [inline], [private]`

5.81.3.8 read_modified() `void dc_simple::dc_if_t::read_modified ( ) [inline], [private]`

5.81.3.9 update_level_mon() `void dc_simple::dc_if_t::update_level_mon ( ) [private]`

Updates the data of variable mon_l in **dc_t** (p. 397).

5.81.3.10 update_gain_mon() `void dc_simple::dc_if_t::update_gain_mon ( ) [private]`

Updates the data of variable mon_g in **dc_t** (p. 397).

5.81.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.81.4.1 clientid `MHAParser::string_t` dc_simple::dc_if_t::clientid [private]

MHA Parser variables.

5.81.4.2 gainrule `MHAParser::string_t` dc_simple::dc_if_t::gainrule [private]

5.81.4.3 preset `MHAParser::string_t` dc_simple::dc_if_t::preset [private]

5.81.4.4 modified `MHAParser::int_mon_t` dc_simple::dc_if_t::modified [private]

5.81.4.5 mon_l `MHAParser::vfloat_mon_t` dc_simple::dc_if_t::mon_l [private]

5.81.4.6 mon_g `MHAParser::vfloat_mon_t` dc_simple::dc_if_t::mon_g [private]

5.81.4.7 filterbank `MHAParser::string_t` dc_simple::dc_if_t::filterbank [private]

5.81.4.8 center_frequencies `MHAParser::vfloat_mon_t` dc_simple::dc_if_t::center_↩ frequencies [private]

5.81.4.9 edge_frequencies `MHAParser::vfloat_mon_t` dc_simple::dc_if_t::edge_↩ frequencies [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.81.4.10 patchbay `MHAEvents::patchbay_t< dc_if_t> dc_simple::dc_if_t::patchbay [private]`

5.81.4.11 prepared `bool dc_simple::dc_if_t::prepared [private]`

The documentation for this class was generated from the following files:

- **dc_simple.hh**
- **dc_simple.cpp**

5.82 dc_simple::dc_t Class Reference

Runtime config class for **dc_simple** (p. 86) plugin.

Inheritance diagram for dc_simple::dc_t:

Classes

- class **line_t**
  Helper class for usage in computing compression, expansion and limiting.

Public Member Functions

- **dc_t** (const **dc_vars_t** &vars, unsigned int nch)
  Constructor.
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗s, **mha_wave_t** ∗level_db)
  Process callback.
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s, **mha_wave_t** ∗level_db)
  Process callback.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- std::vector< float > **mon_l**
- std::vector< float > **mon_g**

Private Attributes

- std::vector< **mha_real_t** > **expansion_threshold**
  Threshold below which to apply expansion.
- std::vector< **mha_real_t** > **limiter_threshold**
  Threshold below which to compress.
- std::vector< **line_t** > **compression**
  The linear function for applying compression.
- std::vector< **line_t** > **expansion**
  The linear function for applying expansion.
- std::vector< **line_t** > **limiter**
  The linear function for applying limiting.
- std::vector< **mha_real_t** > **maxgain**
  Gain should not exceed this value.
- unsigned int **nbands**
  Number of bands.

Additional Inherited Members

5.82.1 Detailed Description

Runtime config class for **dc_simple** (p. 86) plugin.

5.82.2 Constructor & Destructor Documentation

5.82.2.1 dc_t()

```
dc_simple::dc_t::dc_t (
    const dc_vars_t & vars,
    unsigned int nch )
```

Constructor.

© 2005-2021 HörTech gGmbH, Oldenburg

5.82.3 Member Function Documentation

5.82.3.1 process() [1/2]

```
mha_spec_t * dc_simple::dc_t::process (
    mha_spec_t * s,
    mha_wave_t * level_db )
```

Process callback.

Compresses, expands or limits depending on the gain settings and filtered signal levels. Compresses the spectrum input signal in individual bands.

Parameters

s	input/output signal
level_db	smoothed levels of input signal in dB SPL

Returns

s. The input signal is modified in place.

5.82.3.2 process() [2/2]

```
mha_wave_t * dc_simple::dc_t::process (
    mha_wave_t * s,
    mha_wave_t * level_db )
```

Process callback.

Compresses, expands or limits depending on the gain settings and filtered signal levels. Compresses the waveform input signal in individual bands.

Parameters

s	input/output signal
level_db	smoothed levels of input signal in dB SPL

Returns

s. The input signal is modified in place.

© 2005-2021 HörTech gGmbH, Oldenburg

5.82.4 Member Data Documentation

5.82.4.1 expansion_threshold `std::vector< mha_real_t> dc_simple::dc_t::expansion↩ _threshold [private]`

Threshold below which to apply expansion.

5.82.4.2 limiter_threshold `std::vector< mha_real_t> dc_simple::dc_t::limiter_↩ threshold [private]`

Threshold below which to compress.

5.82.4.3 compression `std::vector< line_t> dc_simple::dc_t::compression [private]`

The linear function for applying compression.

5.82.4.4 expansion `std::vector< line_t> dc_simple::dc_t::expansion [private]`

The linear function for applying expansion.

5.82.4.5 limiter `std::vector< line_t> dc_simple::dc_t::limiter [private]`

The linear function for applying limiting.

5.82.4.6 maxgain `std::vector< mha_real_t> dc_simple::dc_t::maxgain [private]`

Gain should not exceed this value.

© 2005-2021 HörTech gGmbH, Oldenburg

5.82.4.7 nbands `unsigned int dc_simple::dc_t::nbands [private]`

Number of bands.

5.82.4.8 mon_l `std::vector<float> dc_simple::dc_t::mon_l`

5.82.4.9 mon_g `std::vector<float> dc_simple::dc_t::mon_g`

The documentation for this class was generated from the following files:

- **dc_simple.hh**
- **dc_simple.cpp**

5.83 dc_simple::dc_t::line_t Class Reference

Helper class for usage in computing compression, expansion and limiting.

Public Member Functions

- **line_t** (**mha_real_t** x1, **mha_real_t** y1, **mha_real_t** x2, **mha_real_t** y2)

  Constructor used for compression which takes two x and y coordinates each to find m and y0

- **line_t** (**mha_real_t** x1, **mha_real_t** y1, **mha_real_t** m_)

  Constructor used for expansion and limiting which takes x and y coordinates and a gradient, giving y0.
- **mha_real_t operator()** (**mha_real_t** x)

  Operator overload which returns.

Private Attributes

- **mha_real_t m**

  The gradient and y-intercept.
- **mha_real_t y0**

© 2005-2021 HörTech gGmbH, Oldenburg

5.83.1 Detailed Description

Helper class for usage in computing compression, expansion and limiting.

5.83.2 Constructor & Destructor Documentation

5.83.2.1 line_t() [1/2]

```
dc_simple::dc_t::line_t::line_t (
    mha_real_t x1,
    mha_real_t y1,
    mha_real_t x2,
    mha_real_t y2 )
```

Constructor used for compression which takes two x and y coordinates each to find m and y0

5.83.2.2 line_t() [2/2]

```
dc_simple::dc_t::line_t::line_t (
    mha_real_t x1,
    mha_real_t y1,
    mha_real_t m_ )
```

Constructor used for expansion and limiting which takes x and y coordinates and a gradient, giving y0.

5.83.3 Member Function Documentation

5.83.3.1 operator()()

```
mha_real_t dc_simple::dc_t::line_t::operator() (
    mha_real_t x ) [inline]
```

Operator overload which returns.

Parameters

x	

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

y values mapped to x by a linear equation with gradient m and intercept y0

5.83.4 Member Data Documentation

5.83.4.1 m `mha_real_t dc_simple::dc_t::line_t::m [private]`

The gradient and y-intercept.

5.83.4.2 y0 `mha_real_t dc_simple::dc_t::line_t::y0 [private]`

The documentation for this class was generated from the following files:

- **dc_simple.hh**
- **dc_simple.cpp**

5.84 dc_simple::dc_vars_t Class Reference

class for **dc_simple** (p. 86) plugin which registers variables to **MHAParser** (p. 122).

Inheritance diagram for dc_simple::dc_vars_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **dc_vars_t** (**MHAParser::parser_t** &p)

Public Attributes

- **MHAParser::vfloat_t g50**
- **MHAParser::vfloat_t g80**
- **MHAParser::vfloat_t maxgain**
- **MHAParser::vfloat_t expansion_threshold**
- **MHAParser::vfloat_t expansion_slope**
- **MHAParser::vfloat_t limiter_threshold**
- **MHAParser::vfloat_t tauattack**
- **MHAParser::vfloat_t taudecay**
- **MHAParser::bool_t bypass**

5.84.1 Detailed Description

class for **dc_simple** (p. 86) plugin which registers variables to **MHAParser** (p. 122).

5.84.2 Constructor & Destructor Documentation

5.84.2.1 dc_vars_t()

`dc_simple::dc_vars_t::dc_vars_t ( MHAParser::parser_t & p )`

5.84.3 Member Data Documentation

5.84.3.1 g50

`MHAParser::vfloat_t dc_simple::dc_vars_t::g50`

5.84.3.2 g80

`MHAParser::vfloat_t dc_simple::dc_vars_t::g80`

© 2005-2021 HörTech gGmbH, Oldenburg

5.84.3.3 maxgain `MHAParser::vfloat_t dc_simple::dc_vars_t::maxgain`

5.84.3.4 expansion_threshold `MHAParser::vfloat_t dc_simple::dc_vars_t::expansion↩_threshold`

5.84.3.5 expansion_slope `MHAParser::vfloat_t dc_simple::dc_vars_t::expansion_↩slope`

5.84.3.6 limiter_threshold `MHAParser::vfloat_t dc_simple::dc_vars_t::limiter_↩threshold`

5.84.3.7 tauattack `MHAParser::vfloat_t dc_simple::dc_vars_t::tauattack`

5.84.3.8 taudecay `MHAParser::vfloat_t dc_simple::dc_vars_t::taudecay`

5.84.3.9 bypass `MHAParser::bool_t dc_simple::dc_vars_t::bypass`

The documentation for this class was generated from the following files:

- **dc_simple.hh**
- **dc_simple.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.85 dc_simple::dc_vars_validator_t Class Reference

Helper class to check sizes of configuration variable vectors.

Inheritance diagram for dc_simple::dc_vars_validator_t:

Public Member Functions

- **dc_vars_validator_t** (const **dc_vars_t** &v, unsigned int s)
  Checks that all vectors in `v` *have size* `s` *or size 1.*

5.85.1 Detailed Description

Helper class to check sizes of configuration variable vectors.

5.85.2 Constructor & Destructor Documentation

5.85.2.1 dc_vars_validator_t() `dc_simple::dc_vars_validator_t::dc_vars_validator_t (`

```
const dc_vars_t & v,
unsigned int s )
```

Checks that all vectors in `v` have size `s` or size 1.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

v	Aggregation of vectors to check the sizes of.
s	Desired vector size for all vectors

Exceptions

MHA_Error *(p. 764)*	if `s` == 0.
MHA_Error *(p. 764)*	if the size of any vector in `v` is neither `s` nor 1.

The documentation for this class was generated from the following files:

- **dc_simple.hh**
- **dc_simple.cpp**

5.86 dc_simple::level_smoother_t Class Reference

Class which computes smoothed input levels on individual bands, using an attack and release filter, which are a first order low pass filter and a maximum tracker filter, respectively.

Inheritance diagram for dc_simple::level_smoother_t:

Public Member Functions

- **level_smoother_t** (const **dc_vars_t** &vars, **mha_real_t** filter_rate, **mhaconfig_t** buscfg)
- **mha_wave_t** ∗ **process** (**mha_spec_t** ∗s)
  Process callback.
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s)
  Process callback.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAFilter::o1flt_lowpass_t attack**
  first order low pass attack filter
- **MHAFilter::o1flt_maxtrack_t decay**
  maximum tracker decay filter
- unsigned int **nbands**
  Total number of frequency bands of this compressor.
- unsigned int **fftlen**
- **MHASignal::waveform_t level_wave**
- **MHASignal::waveform_t level_spec**

Additional Inherited Members

5.86.1 Detailed Description

Class which computes smoothed input levels on individual bands, using an attack and release filter, which are a first order low pass filter and a maximum tracker filter, respectively.

5.86.2 Constructor & Destructor Documentation

5.86.2.1 level_smoother_t()

```
dc_simple::level_smoother_t::level_smoother_t (
    const dc_vars_t & vars,
    mha_real_t filter_rate,
    mhaconfig_t buscfg )
```

5.86.3 Member Function Documentation

5.86.3.1 process() [1/2]

```
mha_wave_t * dc_simple::level_smoother_t::process (
    mha_spec_t * s )
```

Process callback.

Computes smoothed levels from the input mha type spectrum by applying a lowpass and maximum tracker filter

Returns

smoothed input levels in dB SPL

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s	input signal

5.86.3.2 process() [2/2] `mha_wave_t * dc_simple::level_smoother_t::process ( mha_wave_t * s )`

Process callback.

Computes smoothed levels from the input mha type waveform over individual bands by applying a lowpass and maximum tracker filter

Returns

smoothed input levels in dB SPL

Parameters

s	input/output signal

5.86.4 Member Data Documentation

5.86.4.1 attack `MHAFilter::o1flt_lowpass_t dc_simple::level_smoother_t::attack [private]`

first order low pass attack filter

5.86.4.2 decay `MHAFilter::o1flt_maxtrack_t dc_simple::level_smoother_t::decay [private]`

maximum tracker decay filter

© 2005-2021 HörTech gGmbH, Oldenburg

5.86.4.3 nbands `unsigned int dc_simple::level_smoother_t::nbands [private]`

Total number of frequency bands of this compressor.

5.86.4.4 fftlen `unsigned int dc_simple::level_smoother_t::fftlen [private]`

5.86.4.5 level_wave `MHASignal::waveform_t dc_simple::level_smoother_t::level_wave [private]`

5.86.4.6 level_spec `MHASignal::waveform_t dc_simple::level_smoother_t::level_spec [private]`

The documentation for this class was generated from the following files:

- **dc_simple.hh**
- **dc_simple.cpp**

5.87 delay::interface_t Class Reference

Inheritance diagram for delay::interface_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **interface_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Private Member Functions

- void **update** ()

Private Attributes

- **MHAParser::vint_t delays**
- **MHAEvents::patchbay_t**< **interface_t** > **patchbay**

Additional Inherited Members

5.87.1 Constructor & Destructor Documentation

5.87.1.1 interface_t()

```
delay::interface_t::interface_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.87.2 Member Function Documentation

5.87.2.1 prepare()

```
void delay::interface_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t**< **MHASignal::delay_t** > (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.87.2.2 process() `mha_wave_t * delay::interface_t::process ( mha_wave_t * s )`

5.87.2.3 update() `void delay::interface_t::update ( ) [private]`

5.87.3 Member Data Documentation

5.87.3.1 delays `MHAParser::vint_t delay::interface_t::delays [private]`

5.87.3.2 patchbay `MHAEvents::patchbay_t< interface_t> delay::interface_t::patchbay [private]`

The documentation for this class was generated from the following files:

- **delay.hh**
- **delay.cpp**

5.88 delaysum::delaysum_wave_if_t Class Reference

Interface class for the delaysum plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

Inheritance diagram for delaysum::delaysum_wave_if_t:

Public Member Functions

- **delaysum_wave_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAParser::vfloat_t weights**

  Linear weights to be multiplied with the audio signal, one factor for each channel.
- **MHAParser::vint_t delay**

  vector of channel-specific delays, in samples.
- **MHAEvents::patchbay_t**< **delaysum_wave_if_t** > **patchbay**

  The patchbay to react to config changes.

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.88.1 Detailed Description

Interface class for the delaysum plugin.

This plugin allows to delay and sum multiple input channels using individual delays and weights. After each channel gets delayed it is multiplied with the given weight and then added to the single outout channel.

5.88.2 Constructor & Destructor Documentation

5.88.2.1 delaysum_wave_if_t()

```
delaysum::delaysum_wave_if_t::delaysum_wave_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.88.3 Member Function Documentation

5.88.3.1 process()

```
mha_wave_t * delaysum::delaysum_wave_if_t::process (
    mha_wave_t * wave )
```

5.88.3.2 prepare()

```
void delaysum::delaysum_wave_if_t::prepare (
    mhaconfig_t & tfcfg ) [virtual]
```

Implements **MHAPlugin::plugin_t< delaysum_wave_t >** (p. 1153).

5.88.3.3 release()

```
void delaysum::delaysum_wave_if_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< delaysum_wave_t >** (p. 1154).

© 2005-2021 HörTech gGmbH, Oldenburg

5.88.3.4 update_cfg() `void delaysum::delaysum_wave_if_t::update_cfg ( ) [private]`

5.88.4 Member Data Documentation

5.88.4.1 weights `MHAParser::vfloat_t delaysum::delaysum_wave_if_t::weights [private]`

Linear weights to be multiplied with the audio signal, one factor for each channel.

Order is [chan0, chan1, ...]

5.88.4.2 delay `MHAParser::vint_t delaysum::delaysum_wave_if_t::delay [private]`

vector of channel-specific delays, in samples.

5.88.4.3 patchbay `MHAEvents::patchbay_t< delaysum_wave_if_t> delaysum::delaysum↩_wave_if_t::patchbay [private]`

The patchbay to react to config changes.

The documentation for this class was generated from the following file:

- **delaysum_wave.cpp**

5.89 delaysum::delaysum_wave_t Class Reference

Runtime configuration of the delaysum_wave plugin.

Inheritance diagram for delaysum::delaysum_wave_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **delaysum_wave_t** (unsigned int nch, unsigned int fragsize, const std::vector< **mha_real_t** > &weights_, const std::vector< int > &delays_)

  Constructor of the runtime configuration.
- **mha_wave_t** * **process** (**mha_wave_t** *)

Private Attributes

- std::vector< **mha_real_t** > **weights**

  Relative weights for each channel. Order is [chan0, chan1, ...].
- **MHASignal::waveform_t out**

  Output waveform.

5.89.1 Detailed Description

Runtime configuration of the delaysum_wave plugin.

Inherits from the already present delay_t class. The constructor initializes and validates the runtime configuration and forwards the delay vector to the delay_t class. The process function first calls delay_t::process and then multiplies every output channel with its weight and adds them into the output channel.

5.89.2 Constructor & Destructor Documentation

5.89.2.1 delaysum_wave_t()

```
delaysum::delaysum_wave_t::delaysum_wave_t (
    unsigned int nch,
    unsigned int fragsize,
    const std::vector< mha_real_t > & weights_,
    const std::vector< int > & delays_ )
```

Constructor of the runtime configuration.

Parameters

nch	Number of input channels.
fragsize	Size of one input fragment in frames.
*weights_*	Vector of weights for each channel.
*delays_*	Vector of delays, one entry per channel.

© 2005-2021 HörTech gGmbH, Oldenburg

5.89.3 Member Function Documentation

5.89.3.1 process()

```
mha_wave_t * delaysum::delaysum_wave_t::process (
    mha_wave_t * signal )
```

5.89.4 Member Data Documentation

5.89.4.1 weights

```
std::vector< mha_real_t> delaysum::delaysum_wave_t::weights [private]
```

Relative weights for each channel. Order is [chan0, chan1, ...].

5.89.4.2 out

```
MHASignal::waveform_t delaysum::delaysum_wave_t::out [private]
```

Output waveform.

The documentation for this class was generated from the following file:

- **delaysum_wave.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.90 delaysum_spec::delaysum_spec_if_t Class Reference

Inheritance diagram for delaysum_spec::delaysum_spec_if_t:

Public Member Functions

- **delaysum_spec_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAParser::vfloat_t groupdelay**
- **MHAParser::vfloat_t gain**
- **MHAEvents::patchbay_t**< **delaysum_spec_if_t** > **patchbay**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.90.1 Constructor & Destructor Documentation

5.90.1.1 delaysum_spec_if_t()

```
delaysum_spec::delaysum_spec_if_t::delaysum_spec_↩
if_t (
            algo_comm_t iac,
           const std::string & configured_name )
```

5.90.2 Member Function Documentation

5.90.2.1 process()

```
mha_spec_t * delaysum_spec::delaysum_spec_if_t::process (
            mha_spec_t * spec )
```

5.90.2.2 prepare()

```
void delaysum_spec::delaysum_spec_if_t::prepare (
            mhaconfig_t & signal_info ) [virtual]
```

Implements **MHAPlugin::plugin_t< delaysum_t >** (p. 1153).

5.90.2.3 update_cfg()

```
void delaysum_spec::delaysum_spec_if_t::update_cfg ( ) [private]
```

5.90.3 Member Data Documentation

5.90.3.1 groupdelay

```
MHAParser::vfloat_t delaysum_spec::delaysum_spec_if_t::groupdelay
[private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.90.3.2 gain `MHAParser::vfloat_t` `delaysum_spec::delaysum_spec_if_t::gain [private]`

5.90.3.3 patchbay `MHAEvents::patchbay_t< delaysum_spec_if_t>` `delaysum_spec↩ ::delaysum_spec_if_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **delaysum_spec.cpp**

5.91 delaysum_spec::delaysum_t Class Reference

Public Member Functions

- **delaysum_t** (std::vector< float > groupdelay, std::vector< float > gain, unsigned int n↩ Channels, unsigned int nFFT, float fs)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

Private Attributes

- **MHASignal::spectrum_t scale**
- **MHASignal::spectrum_t output**

5.91.1 Constructor & Destructor Documentation

5.91.1.1 delaysum_t()

```
delaysum_spec::delaysum_t::delaysum_t (
    std::vector< float > groupdelay,
    std::vector< float > gain,
    unsigned int nChannels,
    unsigned int nFFT,
    float fs )
```

5.91.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.91.2.1 process()

`mha_spec_t * delaysum_spec::delaysum_t::process ( mha_spec_t * spec )`

5.91.3 Member Data Documentation

5.91.3.1 scale

`MHASignal::spectrum_t delaysum_spec::delaysum_t::scale [private]`

5.91.3.2 output

`MHASignal::spectrum_t delaysum_spec::delaysum_t::output [private]`

The documentation for this class was generated from the following file:

- **delaysum_spec.cpp**

5.92 doasvm_classification Class Reference

Inheritance diagram for doasvm_classification:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **doasvm_classification** (**algo_comm_t** iac, const std::string &configured_name)
  Constructs our plugin.
- ~**doasvm_classification** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
  Checks for the most recent configuration and defers processing to it.
- void **prepare** (**mhaconfig_t** &)
  Plugin preparation.
- void **release** (void)

Public Attributes

- **MHAParser::vfloat_t angles**
- **MHAParser::mfloat_t w**
- **MHAParser::vfloat_t b**
- **MHAParser::vfloat_t x**
- **MHAParser::vfloat_t y**
- **MHAParser::string_t p_name**
- **MHAParser::string_t max_p_ind_name**
- **MHAParser::string_t vGCC_name**

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t**< **doasvm_classification** > **patchbay**

Additional Inherited Members

5.92.1 Constructor & Destructor Documentation

5.92.1.1 doasvm_classification()

```
doasvm_classification::doasvm_classification (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructs our plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

5.92.1.2 ~doasvm_classification() `doasvm_classification::~doasvm_classification ( )`

5.92.2 Member Function Documentation

5.92.2.1 process() `mha_wave_t * doasvm_classification::process ( mha_wave_t * signal )`

Checks for the most recent configuration and defers processing to it.

5.92.2.2 prepare() `void doasvm_classification::prepare ( mhaconfig_t & signal_info ) [virtual]`

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< doasvm_classification_config >** (p. 1153).

5.92.2.3 release() `void doasvm_classification::release ( void ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t< doasvm_classification_config >** (p. 1154).

5.92.2.4 update_cfg() `void doasvm_classification::update_cfg ( ) [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.92.3 Member Data Documentation

5.92.3.1 angles

MHAParser::vfloat_t doasvm_classification::angles

5.92.3.2 w

MHAParser::mfloat_t doasvm_classification::w

5.92.3.3 b

MHAParser::vfloat_t doasvm_classification::b

5.92.3.4 x

MHAParser::vfloat_t doasvm_classification::x

5.92.3.5 y

MHAParser::vfloat_t doasvm_classification::y

5.92.3.6 p_name

MHAParser::string_t doasvm_classification::p_name

5.92.3.7 max_p_ind_name

MHAParser::string_t doasvm_classification::max_p_ind_↩name

5.92.3.8 vGCC_name

MHAParser::string_t doasvm_classification::vGCC_name

© 2005-2021 HörTech gGmbH, Oldenburg

5.92.3.9 patchbay `MHAEvents::patchbay_t< doasvm_classification> doasvm_classification↩ ::patchbay [private]`

The documentation for this class was generated from the following files:

- **doasvm_classification.h**
- **doasvm_classification.cpp**

5.93 doasvm_classification_config Class Reference

Public Member Functions

- **doasvm_classification_config** (**algo_comm_t** & **ac**, **doasvm_classification** *_↩ doasvm)
- ∼**doasvm_classification_config** ()
- **mha_wave_t** * **process** (**mha_wave_t** *)

Public Attributes

- **algo_comm_t** & **ac**
- **doasvm_classification** * **doasvm**
- **MHA_AC::waveform_t p**
- **MHA_AC::int_t p_max**
- **mha_wave_t c**

5.93.1 Constructor & Destructor Documentation

5.93.1.1 doasvm_classification_config()

```
doasvm_classification_config::doasvm_↩ classification_config (
            algo_comm_t & ac,
            doasvm_classification * _doasvm )
```

5.93.1.2 ∼doasvm_classification_config()

```
doasvm_classification_config::∼doasvm_↩ classification_config ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.93.2 Member Function Documentation

5.93.2.1 process()

```
mha_wave_t * doasvm_classification_config::process (
    mha_wave_t * wave )
```

5.93.3 Member Data Documentation

5.93.3.1 ac

```
algo_comm_t& doasvm_classification_config::ac
```

5.93.3.2 doasvm

```
doasvm_classification* doasvm_classification_config::doasvm
```

5.93.3.3 p

```
MHA_AC::waveform_t doasvm_classification_config::p
```

5.93.3.4 p_max

```
MHA_AC::int_t doasvm_classification_config::p_max
```

5.93.3.5 c

```
mha_wave_t doasvm_classification_config::c
```

The documentation for this class was generated from the following files:

- **doasvm_classification.h**
- **doasvm_classification.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.94 doasvm_feature_extraction Class Reference

Inheritance diagram for doasvm_feature_extraction:

Public Member Functions

- **doasvm_feature_extraction** (**algo_comm_t** iac, const std::string &configured_name)

  Constructs our plugin.
- ~**doasvm_feature_extraction** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

  Checks for the most recent configuration and defers processing to it.
- void **prepare** (**mhaconfig_t** &)

  Plugin preparation.
- void **release** (void)

Public Attributes

- **MHAParser::int_t fftlen**
- **MHAParser::int_t max_lag**
- **MHAParser::int_t nupsample**
- **MHAParser::string_t vGCC_name**

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t< doasvm_feature_extraction > patchbay**

Additional Inherited Members

5.94.1 Constructor & Destructor Documentation

5.94.1.1 doasvm_feature_extraction()

```
doasvm_feature_extraction::doasvm_feature_extraction (
            algo_comm_t iac,
            const std::string & configured_name )
```

Constructs our plugin.

5.94.1.2 ~doasvm_feature_extraction()

```
doasvm_feature_extraction::~doasvm_feature_extraction ( )
```

5.94.2 Member Function Documentation

5.94.2.1 process()

```
mha_wave_t * doasvm_feature_extraction::process (
            mha_wave_t * signal )
```

Checks for the most recent configuration and defers processing to it.

5.94.2.2 prepare()

```
void doasvm_feature_extraction::prepare (
            mhaconfig_t & signal_info ) [virtual]
```

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t**< **doasvm_feature_extraction_config** > (p. 1153).

5.94.2.3 release()

```
void doasvm_feature_extraction::release (
    void ) [inline], [virtual]
```

Reimplemented from **MHAPlugin::plugin_t**< **doasvm_feature_extraction_config** > (p. 1154).

5.94.2.4 update_cfg()

```
void doasvm_feature_extraction::update_cfg ( ) [private]
```

5.94.3 Member Data Documentation

5.94.3.1 fftlen

```
MHAParser::int_t doasvm_feature_extraction::fftlen
```

5.94.3.2 max_lag

```
MHAParser::int_t doasvm_feature_extraction::max_lag
```

5.94.3.3 nupsample

```
MHAParser::int_t doasvm_feature_extraction::nupsample
```

5.94.3.4 vGCC_name

```
MHAParser::string_t doasvm_feature_extraction::vGCC_name
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.94.3.5 patchbay `MHAEvents::patchbay_t< doasvm_feature_extraction> doasvm_feature_extraction::patchbay [private]`

The documentation for this class was generated from the following files:

- **doasvm_feature_extraction.h**
- **doasvm_feature_extraction.cpp**

5.95 doasvm_feature_extraction_config Class Reference

Public Member Functions

- **doasvm_feature_extraction_config** (**algo_comm_t** &ac, const **mhaconfig_t** in_cfg, **doasvm_feature_extraction** ∗_doagcc)
- ∼**doasvm_feature_extraction_config** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Public Attributes

- **doasvm_feature_extraction** ∗ **doagcc**
- unsigned int **wndlen**
- unsigned int **fftlen**
- unsigned int **G_length**
- unsigned int **GCC_start**
- unsigned int **GCC_end**
- **MHA_AC::waveform_t vGCC_ac**
- **mha_fft_t fft**
- **mha_fft_t ifft**
- double **hifftwin_sum**
- **MHASignal::waveform_t proc_wave**
- **MHASignal::waveform_t hwin**
- **MHASignal::waveform_t hifftwin**
- **MHASignal::waveform_t vGCC**
- **MHASignal::spectrum_t in_spec**
- **MHASignal::spectrum_t G**

5.95.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.95.1.1 doasvm_feature_extraction_config()

```
doasvm_feature_extraction_config↩
::doasvm_feature_extraction_config (
            algo_comm_t & ac,
           const  mhaconfig_t in_cfg,
            doasvm_feature_extraction * _doagcc )
```

5.95.1.2 ∼doasvm_feature_extraction_config()

```
doasvm_feature_extraction_config↩
::∼doasvm_feature_extraction_config ( )
```

5.95.2 Member Function Documentation

5.95.2.1 process()

```
mha_wave_t * doasvm_feature_extraction_config::process (
            mha_wave_t * wave )
```

5.95.3 Member Data Documentation

5.95.3.1 doagcc

```
doasvm_feature_extraction* doasvm_feature_extraction_config↩
::doagcc
```

5.95.3.2 wndlen

```
unsigned int doasvm_feature_extraction_config::wndlen
```

5.95.3.3 fftlen

```
unsigned int doasvm_feature_extraction_config::fftlen
```

5.95.3.4 G_length

```
unsigned int doasvm_feature_extraction_config::G_length
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.95.3.5 GCC_start `unsigned int doasvm_feature_extraction_config::GCC_start`

5.95.3.6 GCC_end `unsigned int doasvm_feature_extraction_config::GCC_end`

5.95.3.7 vGCC_ac `MHA_AC::waveform_t doasvm_feature_extraction_config::vGCC_ac`

5.95.3.8 fft `mha_fft_t doasvm_feature_extraction_config::fft`

5.95.3.9 ifft `mha_fft_t doasvm_feature_extraction_config::ifft`

5.95.3.10 hifftwin_sum `double doasvm_feature_extraction_config::hifftwin_sum`

5.95.3.11 proc_wave `MHASignal::waveform_t doasvm_feature_extraction_config↩ ::proc_wave`

5.95.3.12 hwin `MHASignal::waveform_t doasvm_feature_extraction_config::hwin`

5.95.3.13 hifftwin `MHASignal::waveform_t doasvm_feature_extraction_config::hifftwin`

© 2005-2021 HörTech gGmbH, Oldenburg

5.95.3.14 vGCC `MHASignal::waveform_t` `doasvm_feature_extraction_config::vGCC`

5.95.3.15 in_spec `MHASignal::spectrum_t` `doasvm_feature_extraction_config::in_spec`

5.95.3.16 G `MHASignal::spectrum_t` `doasvm_feature_extraction_config::G`

The documentation for this class was generated from the following files:

- **doasvm_feature_extraction.h**
- **doasvm_feature_extraction.cpp**

5.96 double2acvar::double2acvar_t Class Reference

Plugin interface class for **double2acvar** (p. 90).

Inheritance diagram for double2acvar::double2acvar_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **double2acvar_t** (**algo_comm_t** iac, const std::string &configured_name)
  Standard plugin constructor.
- ~**double2acvar_t** ()=default
- template<class T >
  T * **process** (T *s)
  process() *(p. 435) does not alter the signal and has same implementation regardless of signal domain.*
- void **poll_latest_value_and_reinsert** ()
  *Called from **process()** (p. 435) and, when allowed, also from **on_configuration_update()** (p. 436).*
- void **prepare_** (**mhaconfig_t** &)
  Prepare method as expected by the plugin interface macros.
- void **release_** ()
  Release method as expected by the plugin interface macros.
- void **on_configuration_update** ()
  Callback function on write access to the string configuration value.

Private Attributes

- **MHA_AC::double_t ac_double**
  AC variable inserted by this plugin.
- **MHAEvents::patchbay_t< double2acvar_t > patchbay**
  Callback router.
- bool **is_prepared**
  Flag to keep track if we are currently prepared.

Additional Inherited Members

5.96.1 Detailed Description

Plugin interface class for **double2acvar** (p. 90).

5.96.2 Constructor & Destructor Documentation

5.96.2.1 double2acvar_t()

```
double2acvar::double2acvar_t::double2acvar_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Standard plugin constructor.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

iac	Algorithm communication variable space.
configured_name	Configured name of this plugin, also used as name of the AC variable.

5.96.2.2 ~double2acvar_t() `double2acvar::double2acvar_t::~double2acvar_t ( ) [default]`

5.96.3 Member Function Documentation

5.96.3.1 process() `template<class T >`

```
T * double2acvar::double2acvar_t::process (
            T * s )
```

process() (p. 435) does not alter the signal and has same implementation regardless of signal domain.

Parameters

s	Pointer to input signal structure, **mha_wave_t** (p. 840) or **mha_spec_t** (p. 794).

Returns

s, unaltered.

5.96.3.2 poll_latest_value_and_reinsert() `void double2acvar::double2acvar_t::poll_↩ latest_value_and_reinsert ( )`

Called from **process()** (p. 435) and, when allowed, also from **on_configuration_update()** (p. 436).

poll_latest_value_and_reinsert() (p. 435) retrieves the latest configured value and reinserts the AC variable into the AC space.

© 2005-2021 HörTech gGmbH, Oldenburg

5.96.3.3 prepare_() `void double2acvar::double2acvar_t::prepare_ ( mhaconfig_t & )`

Prepare method as expected by the plugin interface macros.

Parameter is not used nor altered. Sets is_prepared flag.

5.96.3.4 release_() `void double2acvar::double2acvar_t::release_ ( )`

Release method as expected by the plugin interface macros.

Resets is_prepared flag.

5.96.3.5 on_configuration_update() `void double2acvar::double2acvar_t::on_configuration↩ _update ( )`

Callback function on write access to the string configuration value.

5.96.4 Member Data Documentation

5.96.4.1 ac_double `MHA_AC::double_t double2acvar::double2acvar_t::ac_double [private]`

AC variable inserted by this plugin.

5.96.4.2 patchbay `MHAEvents::patchbay_t< double2acvar_t> double2acvar::double2acvar↩ _t::patchbay [private]`

Callback router.

© 2005-2021 HörTech gGmbH, Oldenburg

5.96.4.3 is_prepared `bool double2acvar::double2acvar_t::is_prepared [private]`

Flag to keep track if we are currently prepared.

If we are, then signal processing is active and AC variables may only be accessed when MHA is currently executing out **process()** (p. 435) method.

The documentation for this class was generated from the following file:

- **double2acvar.cpp**

5.97 dropgen_t Class Reference

Inheritance diagram for dropgen_t:

Public Member Functions

- **dropgen_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **MHAParser::float_t min_sleep_time**
- **MHAParser::float_t max_sleep_time**
- **MHAParser::float_t chance**
- **MHAEvents::patchbay_t< dropgen_t > patchbay**
- std::random_device **r**
- std::mt19937 **random_engine**
- std::uniform_real_distribution **dis**

Additional Inherited Members

5.97.1 Constructor & Destructor Documentation

5.97.1.1 dropgen_t()

```
dropgen_t::dropgen_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.97.2 Member Function Documentation

5.97.2.1 process() [1/2]

```
mha_wave_t * dropgen_t::process (
    mha_wave_t * s )
```

5.97.2.2 process() [2/2]

```
mha_spec_t * dropgen_t::process (
    mha_spec_t * s )
```

5.97.2.3 prepare()

```
void dropgen_t::prepare (
    mhaconfig_t & ) [virtual]
```

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.97.2.4 release() `void dropgen_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t**< **int** > (p. 1154).

5.97.3 Member Data Documentation

5.97.3.1 min_sleep_time `MHAParser::float_t dropgen_t::min_sleep_time`

5.97.3.2 max_sleep_time `MHAParser::float_t dropgen_t::max_sleep_time`

5.97.3.3 chance `MHAParser::float_t dropgen_t::chance`

5.97.3.4 patchbay `MHAEvents::patchbay_t< dropgen_t> dropgen_t::patchbay`

5.97.3.5 r `std::random_device dropgen_t::r`

5.97.3.6 random_engine `std::mt19937 dropgen_t::random_engine`

5.97.3.7 dis `std::uniform_real_distribution dropgen_t::dis`

The documentation for this class was generated from the following file:

- **dropgen.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.98 droptect_t Class Reference

Detect dropouts in a signal with a constant spectrum.

Inheritance diagram for droptect_t:

Public Member Functions

- **droptect_t** (**algo_comm_t** iac, const std::string &configured_name)

  This constructor initializes the configuration language variables and inserts them into the MHA configuration tree.
- void **prepare** (**mhaconfig_t** &signal_info)

  Allocates and initializes storage for this algorithm.
- void **release** (void)

  Deallocates storage.
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗signal)

  Compares current spectrum against history.

Private Attributes

- **MHAParser::vint_mon_t dropouts**
- **MHAParser::vint_mon_t consecutive_dropouts**
- **MHAParser::int_mon_t blocks**

© 2005-2021 HörTech gGmbH, Oldenburg

- **MHAParser::bool_t reset**
- **MHAParser::float_t threshold**
- **MHASignal::waveform_t * current_powspec**
- **MHASignal::waveform_t * filtered_powspec**
- **MHAParser::float_t tau**
- std::vector< bool > **filter_activated**
- float **period**

  The period of the process callback (duration of fragsize in seconds)
- **MHAParser::mfloat_mon_t filtered_powspec_mon**

  User access to filtered spectrum.
- **MHAParser::vfloat_mon_t level_mon**

  User access to broadband levels.

Additional Inherited Members

5.98.1 Detailed Description

Detect dropouts in a signal with a constant spectrum.

5.98.2 Constructor & Destructor Documentation

5.98.2.1 droptect_t()

```
droptect_t::droptect_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

This constructor initializes the configuration language variables and inserts them into the MHA configuration tree.

5.98.3 Member Function Documentation

5.98.3.1 prepare()

```
void droptect_t::prepare (
    mhaconfig_t & signal_info ) [virtual]
```

Allocates and initializes storage for this algorithm.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	contains fft length, number of channels, fft length and hop size.

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

5.98.3.2 release()

```
void droptect_t::release ( void ) [virtual]
```

Deallocates storage.

Reimplemented from **MHAPlugin::plugin_t< int >** (p. 1154).

5.98.3.3 process()

```
mha_spec_t * droptect_t::process ( mha_spec_t * signal )
```

Compares current spectrum against history.

If spectral power has changed or is below threshold, this is interpreted as dropout occurrence.

5.98.4 Member Data Documentation

5.98.4.1 dropouts

```
MHAParser::vint_mon_t droptect_t::dropouts [private]
```

5.98.4.2 consecutive_dropouts

```
MHAParser::vint_mon_t droptect_t::consecutive_↩
dropouts [private]
```

5.98.4.3 blocks

```
MHAParser::int_mon_t droptect_t::blocks [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.98.4.4 reset `MHAParser::bool_t droptect_t::reset [private]`

5.98.4.5 threshold `MHAParser::float_t droptect_t::threshold [private]`

5.98.4.6 current_powspec `MHASignal::waveform_t* droptect_t::current_powspec [private]`

5.98.4.7 filtered_powspec `MHASignal::waveform_t* droptect_t::filtered_powspec [private]`

5.98.4.8 tau `MHAParser::float_t droptect_t::tau [private]`

5.98.4.9 filter_activated `std::vector<bool> droptect_t::filter_activated [private]`

5.98.4.10 period `float droptect_t::period [private]`

The period of the process callback (duration of fragsize in seconds)

5.98.4.11 filtered_powspec_mon `MHAParser::mfloat_mon_t droptect_t::filtered_powspec_mon [private]`

User access to filtered spectrum.

© 2005-2021 HörTech gGmbH, Oldenburg

5.98.4.12 level_mon `MHAParser::vfloat_mon_t droptect_t::level_mon [private]`

User access to broadband levels.

The documentation for this class was generated from the following file:

- **droptect.cpp**

5.99 ds_t Class Reference

Inheritance diagram for ds_t:

Public Member Functions

- **ds_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

Private Attributes

- **MHAParser::int_t ratio**
- **MHAFilter::iir_filter_t antialias**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.99.1 Constructor & Destructor Documentation

5.99.1.1 ds_t()

```
ds_t::ds_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.99.2 Member Function Documentation

5.99.2.1 process()

```
mha_wave_t * ds_t::process (
    mha_wave_t * s )
```

5.99.2.2 prepare()

```
void ds_t::prepare (
    mhaconfig_t & cf ) [virtual]
```

Implements **MHAPlugin::plugin_t< MHASignal::waveform_t >** (p. 1153).

5.99.2.3 release()

```
void ds_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< MHASignal::waveform_t >** (p. 1154).

5.99.3 Member Data Documentation

5.99.3.1 ratio

```
MHAParser::int_t ds_t::ratio [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.99.3.2 antialias `MHAFilter::iir_filter_t ds_t::antialias [private]`

The documentation for this class was generated from the following file:

- **downsample.cpp**

5.100 dynamiclib_t Class Reference

Wrapper class around a shared library.

Inheritance diagram for dynamiclib_t:

Public Member Functions

- **dynamiclib_t** (const std::string &name_)

  C'tor of the wrapper class.
- virtual void * **resolve** (const std::string &name_)

  Resolves the function specified by name_ and returns a pointer to it or a nullptr if the function was not found in the wrapped library.
- virtual void * **resolve_checked** (const std::string &name_)

  Resolves the function specified by name_ and returns a pointer to it or throws an exception if the function was not found.
- virtual ~**dynamiclib_t** ()

  D'tor.
- virtual const std::string & **getmodulename** () const

  Returns unqualified filename of the wrapped library sans file suffix.
- virtual const std::string & **getname** () const

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Member Functions

- **dynamiclib_t** ()
  Default constructor.
- void **load_lib** (const std::string &name_)
  Loads the library specified in name_ and saves a handle in h.

Protected Attributes

- std::string **fullname**
  Fully qualified file name of the library.
- std::string **modulename**
  Unqualified file name of the library.
- **mha_libhandle_t h**
  Handle to the shared library.

5.100.1 Detailed Description

Wrapper class around a shared library.

Encapsulates the OS-specific stuff of loading the shared library, resolving functions, etc... Uses the dload API on Linux/macOS and the win32 API on Windows

5.100.2 Constructor & Destructor Documentation

5.100.2.1 dynamiclib_t() [1/2]

```
dynamiclib_t::dynamiclib_t (
    const std::string & name_ )
```

C'tor of the wrapper class.

Takes a the the file name of a shared libary w/o the suffix as argument, searches for the library in the system-dependent standard paths for libraries and in MHA_LIBRARY_PATH. Calls load_lib for the actual work.

Parameters

*name_*	File name of the shared library, without suffix

© 2005-2021 HörTech gGmbH, Oldenburg

Exceptions

MHA_Error *(p. 764)*	if the library can not be found or can not be loaded

5.100.2.2 ~dynamiclib_t() `dynamiclib_t::~dynamiclib_t ( ) [virtual]`

D'tor.

Closes the library handle.

5.100.2.3 dynamiclib_t() [2/2] `dynamiclib_t::dynamiclib_t ( ) [protected]`

Default constructor.

5.100.3 Member Function Documentation

5.100.3.1 resolve() `void * dynamiclib_t::resolve ( const std::string & name_ ) [virtual]`

Resolves the function specified by name_ and returns a pointer to it or a nullptr if the function was not found in the wrapped library.

Parameters

*name_*	Name of the function to be resolved

Returns

Pointer to the function

Reimplemented in **pluginlib_t** (p. 1362).

© 2005-2021 HörTech gGmbH, Oldenburg

5.100.3.2 resolve_checked() `void * dynamiclib_t::resolve_checked ( const std::string & name_ ) [virtual]`

Resolves the function specified by name_ and returns a pointer to it or throws an exception if the function was not found.

Parameters

*name_*	Name of the function to be resolved

Returns

Pointer to the function

5.100.3.3 getmodulename() `virtual const std::string& dynamiclib_t::getmodulename ( ) const [inline], [virtual]`

Returns unqualified filename of the wrapped library sans file suffix.

Returns

Unqualified filename of the wrapped library

5.100.3.4 getname() `virtual const std::string& dynamiclib_t::getname ( ) const [inline], [virtual]`

5.100.3.5 load_lib() `void dynamiclib_t::load_lib ( const std::string & name_ ) [protected]`

Loads the library specified in name_ and saves a handle in h.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

name↩ _	unqualified file name of the shared library w/o suffix

5.100.4 Member Data Documentation

5.100.4.1 fullname `std::string dynamiclib_t::fullname [protected]`

Fully qualified file name of the library.

5.100.4.2 modulename `std::string dynamiclib_t::modulename [protected]`

Unqualified file name of the library.

5.100.4.3 h `mha_libhandle_t dynamiclib_t::h [protected]`

Handle to the shared library.

The documentation for this class was generated from the following files:

- **mha_os.h**
- **mha_os.cpp**

5.101 DynComp::dc_afterburn_rt_t Class Reference

Real-time class for after burn effect.

Public Member Functions

- **dc_afterburn_rt_t** (const std::vector< float > &cf, unsigned int **channels**, float srate, const **dc_afterburn_vars_t** &vars)
- void **burn** (float &Gin, float Lin, unsigned int band, unsigned int channel)

  gain modifier method (afterburn).

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- std::vector< float > **drain_inv**
- std::vector< float > **conflux**
- std::vector< float > **maxgain**
- std::vector< float > **mpo_inv**
- std::vector< **MHAFilter::o1flt_lowpass_t** > **lp**

5.101.1 Detailed Description

Real-time class for after burn effect.

The constructor processes the parameters and creates pre-processed variables for efficient realtime processing.

5.101.2 Constructor & Destructor Documentation

5.101.2.1 dc_afterburn_rt_t()

```
DynComp::dc_afterburn_rt_t::dc_afterburn_rt_t (
    const std::vector< float > & cf,
    unsigned int channels,
    float srate,
    const dc_afterburn_vars_t & vars )
```

5.101.3 Member Function Documentation

5.101.3.1 burn()

```
void DynComp::dc_afterburn_rt_t::burn (
    float & Gin,
    float Lin,
    unsigned int band,
    unsigned int channel ) [inline]
```

gain modifier method (afterburn).

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

Gin	Linear gain.
Lin	Input level (Pascal).
band	Filter band number.
channel	Channel number.

Output level for MPO is estimated by Gin $*$ Lin.

5.101.4 Member Data Documentation

5.101.4.1 drain_inv `std::vector<float> DynComp::dc_afterburn_rt_t::drain_inv [private]`

5.101.4.2 conflux `std::vector<float> DynComp::dc_afterburn_rt_t::conflux [private]`

5.101.4.3 maxgain `std::vector<float> DynComp::dc_afterburn_rt_t::maxgain [private]`

5.101.4.4 mpo_inv `std::vector<float> DynComp::dc_afterburn_rt_t::mpo_inv [private]`

5.101.4.5 lp `std::vector< MHAFilter::o1flt_lowpass_t> DynComp::dc_afterburn_rt_t↩ ::lp [private]`

The documentation for this class was generated from the following files:

- **dc_afterburn.h**
- **dc_afterburn.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.102 DynComp::dc_afterburn_t Class Reference

Afterburn class, to be defined as a member of compressors.

Inheritance diagram for DynComp::dc_afterburn_t:

Public Member Functions

- **dc_afterburn_t** ()
- void **set_fb_pars** (const std::vector< float > &cf, unsigned int **channels**, float srate)
- void **unset_fb_pars** ()
- void **update_burner** ()
- void **burn** (float &Gin, float Lin, unsigned int band, unsigned int channel)

Private Member Functions

- void **update** ()

Private Attributes

- **MHAEvents::patchbay_t**< **dc_afterburn_t** > **patchbay**
- std::vector< float > **_cf**
- unsigned int **_channels**
- float **_srate**
- bool **commit_pending**
- bool **fb_pars_configured**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.102.1 Detailed Description

Afterburn class, to be defined as a member of compressors.

5.102.2 Constructor & Destructor Documentation

5.102.2.1 dc_afterburn_t() `DynComp::dc_afterburn_t::dc_afterburn_t ( )`

5.102.3 Member Function Documentation

5.102.3.1 set_fb_pars()

```
void DynComp::dc_afterburn_t::set_fb_pars (
    const std::vector< float > & cf,
    unsigned int channels,
    float srate )
```

5.102.3.2 unset_fb_pars() `void DynComp::dc_afterburn_t::unset_fb_pars ( )`

5.102.3.3 update_burner() `void DynComp::dc_afterburn_t::update_burner ( ) [inline]`

5.102.3.4 burn()

```
void DynComp::dc_afterburn_t::burn (
    float & Gin,
    float Lin,
    unsigned int band,
    unsigned int channel ) [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.102.3.5 update() `void DynComp::dc_afterburn_t::update ( ) [private]`

5.102.4 Member Data Documentation

5.102.4.1 patchbay `MHAEvents::patchbay_t< dc_afterburn_t> DynComp::dc_afterburn↩_t::patchbay [private]`

5.102.4.2 _cf `std::vector<float> DynComp::dc_afterburn_t::_cf [private]`

5.102.4.3 _channels `unsigned int DynComp::dc_afterburn_t::_channels [private]`

5.102.4.4 _srate `float DynComp::dc_afterburn_t::_srate [private]`

5.102.4.5 commit_pending `bool DynComp::dc_afterburn_t::commit_pending [private]`

5.102.4.6 fb_pars_configured `bool DynComp::dc_afterburn_t::fb_pars_configured [private]`

The documentation for this class was generated from the following files:

- **dc_afterburn.h**
- **dc_afterburn.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.103 DynComp::dc_afterburn_vars_t Class Reference

Variables for **dc_afterburn_t** (p. 453) class.

Inheritance diagram for DynComp::dc_afterburn_vars_t:

Public Member Functions

- **dc_afterburn_vars_t** ()

Public Attributes

- **MHAParser::vfloat_t f**
- **MHAParser::vfloat_t drain**
- **MHAParser::vfloat_t conflux**
- **MHAParser::vfloat_t maxgain**
- **MHAParser::vfloat_t mpo**
- **MHAParser::float_t taugain**
- **MHAParser::kw_t commit**
- **MHAParser::bool_t bypass**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.103.1 Detailed Description

Variables for **dc_afterburn_t** (p. 453) class.

5.103.2 Constructor & Destructor Documentation

5.103.2.1 dc_afterburn_vars_t() `DynComp::dc_afterburn_vars_t::dc_afterburn_vars_t ( )`

5.103.3 Member Data Documentation

5.103.3.1 f `MHAParser::vfloat_t DynComp::dc_afterburn_vars_t::f`

5.103.3.2 drain `MHAParser::vfloat_t DynComp::dc_afterburn_vars_t::drain`

5.103.3.3 conflux `MHAParser::vfloat_t DynComp::dc_afterburn_vars_t::conflux`

5.103.3.4 maxgain `MHAParser::vfloat_t DynComp::dc_afterburn_vars_t::maxgain`

5.103.3.5 mpo `MHAParser::vfloat_t DynComp::dc_afterburn_vars_t::mpo`

© 2005-2021 HörTech gGmbH, Oldenburg

5.103.3.6 taugain `MHAParser::float_t DynComp::dc_afterburn_vars_t::taugain`

5.103.3.7 commit `MHAParser::kw_t DynComp::dc_afterburn_vars_t::commit`

5.103.3.8 bypass `MHAParser::bool_t DynComp::dc_afterburn_vars_t::bypass`

The documentation for this class was generated from the following files:

- **dc_afterburn.h**
- **dc_afterburn.cpp**

5.104 DynComp::gaintable_t Class Reference

Gain table class.

Public Member Functions

- **gaintable_t** (const std::vector< **mha_real_t** > &LInput, const std::vector< **mha_real_t** > &FCenter, unsigned int **channels**)
  Constructor.
- ~**gaintable_t** ()
- void **update** (std::vector< std::vector< std::vector< **mha_real_t** > > > newGain)
  Update gains from an external table.
- **mha_real_t** **get_gain** (**mha_real_t** Lin, **mha_real_t** Fin, unsigned int channel)
  Read Gain from gain table.
- **mha_real_t** **get_gain** (**mha_real_t** Lin, unsigned int band, unsigned int channel)
  Read Gain from gain table.
- void **get_gain** (const **mha_wave_t** &Lin, **mha_wave_t** &Gain)
  Read Gains from gain table.
- unsigned int **nbands** () const
  Return number of frequency bands.
- unsigned int **nchannels** () const
  Return number of audio channels.
- std::vector< std::vector< **mha_real_t** > > **get_iofun** () const
  Return current input-output function.
- std::vector< **mha_real_t** > **get_vL** () const
- std::vector< **mha_real_t** > **get_vF** () const

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- unsigned int **num_L**
- unsigned int **num_F**
- unsigned int **num_channels**
- std::vector< **mha_real_t** > **vL**
- std::vector< **mha_real_t** > **vF**
- std::vector< **mha_real_t** > **vFlog**
- std::vector< std::vector< std::vector< **mha_real_t** > > > **data**

5.104.1 Detailed Description

Gain table class.

This gain table is intended to efficient table lookup, i.e, interpolation of levels, and optional interpolation of frequencies. Sample input levels and sample frequencies are given in the constructor. The gain entries can be updated with the **update()** (p. 460) member function via a gain prescription rule from an auditory profile.

5.104.2 Constructor & Destructor Documentation

5.104.2.1 gaintable_t()

```
gaintable_t::gaintable_t (
    const std::vector< mha_real_t > & LInput,
    const std::vector< mha_real_t > & FCenter,
    unsigned int channels )
```

Constructor.

Parameters

LInput	Input level samples, in equivalent LTASS_combined dB SPL.
FCenter	Frequency samples in Hz (e.g., center frequencies of filterbank).
channels	Number of audio channels (typically 2).

5.104.2.2 ~gaintable_t()

```
gaintable_t::~gaintable_t ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.104.3 Member Function Documentation

5.104.3.1 update()

```
void gaintable_t::update (
    std::vector< std::vector< std::vector< mha_real_t > > > newGain )
```

Update gains from an external table.

Parameters

newGain	New gain table entries.

Dimension change is not allowed. The number of entries are checked.

5.104.3.2 get_gain() [1/3]

```
mha_real_t gaintable_t::get_gain (
    mha_real_t Lin,
    mha_real_t Fin,
    unsigned int channel )
```

Read Gain from gain table.

Parameters

Lin	Input level
Fin	Input frequency (no match required)
channel	Audio channel

5.104.3.3 get_gain() [2/3]

```
mha_real_t gaintable_t::get_gain (
    mha_real_t Lin,
    unsigned int band,
    unsigned int channel )
```

Read Gain from gain table.

Parameters

Lin	Input level
band	Input frequency band
channel	Audio channel

© 2005-2021 HörTech gGmbH, Oldenburg

5.104.3.4 get_gain() [3/3]

```
void gaintable_t::get_gain (
    const mha_wave_t & Lin,
    mha_wave_t & Gain )
```

Read Gains from gain table.

Parameters

Lin	Input levels.
Gain	Output gain.

The number of channels in Lin and Gain must match the number of bands times number of channels in the gaintable.

5.104.3.5 nbands()

```
unsigned int DynComp::gaintable_t::nbands ( ) const [inline]
```

Return number of frequency bands.

5.104.3.6 nchannels()

```
unsigned int DynComp::gaintable_t::nchannels ( ) const [inline]
```

Return number of audio channels.

5.104.3.7 get_iofun()

```
std::vector< std::vector< mha_real_t > > gaintable_t::get↩
_iofun ( ) const
```

Return current input-output function.

5.104.3.8 get_vL()

```
std::vector< mha_real_t> DynComp::gaintable_t::get_vL ( ) const
[inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.104.3.9 get_vF() `std::vector< mha_real_t> DynComp::gaintable_t::get_vF ( ) const [inline]`

5.104.4 Member Data Documentation

5.104.4.1 num_L `unsigned int DynComp::gaintable_t::num_L [private]`

5.104.4.2 num_F `unsigned int DynComp::gaintable_t::num_F [private]`

5.104.4.3 num_channels `unsigned int DynComp::gaintable_t::num_channels [private]`

5.104.4.4 vL `std::vector< mha_real_t> DynComp::gaintable_t::vL [private]`

5.104.4.5 vF `std::vector< mha_real_t> DynComp::gaintable_t::vF [private]`

5.104.4.6 vFlog `std::vector< mha_real_t> DynComp::gaintable_t::vFlog [private]`

5.104.4.7 data `std::vector<std::vector<std::vector< mha_real_t> > > DynComp↩ ::gaintable_t::data [private]`

The documentation for this class was generated from the following files:

- **gaintable.h**
- **gaintable.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.105 equalize::cfg_t Class Reference

Public Member Functions

- **cfg_t** (int infft, int inchannels, std::vector< std::vector< float > > ifgains)
- **cfg_t** (const **cfg_t** &)=delete
- **cfg_t** & **operator=** (const **cfg_t** &)=delete
- ~**cfg_t** ()

Public Attributes

- int **num_bins**
- int **nchannels**
- **mha_real_t** * **fftgains**

5.105.1 Constructor & Destructor Documentation

5.105.1.1 cfg_t() [1/2]

```
cfg_t::cfg_t (
    int infft,
    int inchannels,
    std::vector< std::vector< float > > ifgains )
```

5.105.1.2 cfg_t() [2/2]

```
equalize::cfg_t::cfg_t (
    const cfg_t & ) [delete]
```

5.105.1.3 ~cfg_t()

```
cfg_t::~cfg_t ( )
```

5.105.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.105.2.1 operator=() `cfg_t& equalize::cfg_t::operator= ( const cfg_t & ) [delete]`

5.105.3 Member Data Documentation

5.105.3.1 num_bins `int equalize::cfg_t::num_bins`

5.105.3.2 nchannels `int equalize::cfg_t::nchannels`

5.105.3.3 fftgains `mha_real_t* equalize::cfg_t::fftgains`

The documentation for this class was generated from the following file:

- **equalize.cpp**

5.106 equalize::freqgains_t Class Reference

Inheritance diagram for equalize::freqgains_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **freqgains_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** $*$ **process** (**mha_spec_t** $*$)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update_gains** ()
- void **update_id** ()

Private Attributes

- **MHAParser::mfloat_t fftgains**
- **MHAParser::string_t id**
- **MHAEvents::patchbay_t**$<$ **freqgains_t** $>$ **patchbay**

Additional Inherited Members

5.106.1 Constructor & Destructor Documentation

5.106.1.1 freqgains_t()

```
equalize::freqgains_t::freqgains_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.106.2 Member Function Documentation

5.106.2.1 process()

```
mha_spec_t * equalize::freqgains_t::process (
    mha_spec_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.106.2.2 prepare() `void equalize::freqgains_t::prepare ( mhaconfig_t & tf ) [virtual]`

Implements **MHAPlugin::plugin_t**< **cfg_t** > (p. 1153).

5.106.2.3 update_gains() `void equalize::freqgains_t::update_gains ( ) [private]`

5.106.2.4 update_id() `void equalize::freqgains_t::update_id ( ) [private]`

5.106.3 Member Data Documentation

5.106.3.1 fftgains `MHAParser::mfloat_t equalize::freqgains_t::fftgains [private]`

5.106.3.2 id `MHAParser::string_t equalize::freqgains_t::id [private]`

5.106.3.3 patchbay `MHAEvents::patchbay_t< freqgains_t> equalize::freqgains_t↩ ::patchbay [private]`

The documentation for this class was generated from the following file:

- **equalize.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.107 example1_t Class Reference

This C++ class implements the simplest example plugin for the step-by-step tutorial.

Inheritance diagram for example1_t:

Public Member Functions

- **example1_t** (**algo_comm_t** iac, const std::string &configured_name)
  Do-nothing constructor.
- void **release** (void)
  Release may be empty.
- void **prepare** (**mhaconfig_t** &signal_info)
  Plugin preparation.
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗signal)
  Signal processing performed by the plugin.

Additional Inherited Members

5.107.1 Detailed Description

This C++ class implements the simplest example plugin for the step-by-step tutorial.

It inherits from **MHAPlugin::plugin_t** (p. 1151) for correct integration in the configuration language interface.

© 2005-2021 HörTech gGmbH, Oldenburg

5.107.2 Constructor & Destructor Documentation

5.107.2.1 example1_t()

```
example1_t::example1_t (
    algo_comm_t iac,
    const std::string & configured_name ) [inline]
```

Do-nothing constructor.

The constructor has to take these two arguments, but it does not have to use them. The base class has to be initialized.

5.107.3 Member Function Documentation

5.107.3.1 release()

```
void example1_t::release (
    void ) [inline], [virtual]
```

Release may be empty.

Reimplemented from **MHAPlugin::plugin_t< int >** (p. 1154).

5.107.3.2 prepare()

```
void example1_t::prepare (
    mhaconfig_t & signal_info ) [inline], [virtual]
```

Plugin preparation.

This plugin checks that the input signal has the waveform domain and contains at least one channel

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.107.3.3 process() `mha_wave_t* example1_t::process ( mha_wave_t * signal ) [inline]`

Signal processing performed by the plugin.

This plugin multiplies the signal in the first audio channel by a factor 0.1.

Parameters

signal	Pointer to the input signal structure.

Returns

Returns a pointer to the input signal structure, with a the signal modified by this plugin. (In-place processing)

The documentation for this class was generated from the following file:

- **example1.cpp**

5.108 example2_t Class Reference

This C++ class implements the second example plugin for the step-by-step tutorial.

Inheritance diagram for example2_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **example2_t** (**algo_comm_t** iac, const std::string &configured_name)

  This constructor initializes the configuration language variables and inserts them into the openMHA configuration tree.
- void **prepare** (**mhaconfig_t** &signal_info)

  Plugin preparation.
- void **release** (void)

  Undo restrictions posed in prepare.
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗signal)

  Signal processing performed by the plugin.

Private Attributes

- **MHAParser::int_t scale_ch**

  Index of audio channel to scale.
- **MHAParser::float_t factor**

  The scaling factor applied to the selected channel.

Additional Inherited Members

5.108.1 Detailed Description

This C++ class implements the second example plugin for the step-by-step tutorial.

It extends the first example by using configuration language variables to influence the processing.

5.108.2 Constructor & Destructor Documentation

5.108.2.1 example2_t()

```
example2_t::example2_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

This constructor initializes the configuration language variables and inserts them into the openMHA configuration tree.

© 2005-2021 HörTech gGmbH, Oldenburg

5.108.3 Member Function Documentation

5.108.3.1 prepare()

```
void example2_t::prepare (
    mhaconfig_t & signal_info ) [virtual]
```

Plugin preparation.

This plugin checks that the input signal has the waveform domain and contains enough channels.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

5.108.3.2 release()

`void example2_t::release ( void ) [virtual]`

Undo restrictions posed in prepare.

Reimplemented from **MHAPlugin::plugin_t< int >** (p. 1154).

5.108.3.3 process()

`mha_wave_t * example2_t::process ( mha_wave_t * signal )`

Signal processing performed by the plugin.

This plugin multiplies the signal in the selected audio channel by the configured factor.

Parameters

signal	Pointer to the input signal structure.

Returns

Returns a pointer to the input signal structure, with a the signal modified by this plugin. (In-place processing)

5.108.4 Member Data Documentation

5.108.4.1 scale_ch

`MHAParser::int_t example2_t::scale_ch [private]`

Index of audio channel to scale.

© 2005-2021 HörTech gGmbH, Oldenburg

5.108.4.2 factor `MHAParser::float_t example2_t::factor [private]`

The scaling factor applied to the selected channel.

The documentation for this class was generated from the following file:

- **example2.cpp**

5.109 example3_t Class Reference

A Plugin class using the openMHA Event mechanism.

Inheritance diagram for example3_t:

Public Member Functions

- **example3_t** (**algo_comm_t** iac, const std::string &configured_name)

  This constructor initializes the configuration language variables and inserts them into the openMHA configuration tree.
- void **prepare** (**mhaconfig_t** &signal_info)

  Plugin preparation.
- void **release** (void)

  Bookkeeping only.
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗signal)

  Signal processing performed by the plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **on_scale_ch_writeaccess** ()
- void **on_scale_ch_valuechanged** ()
- void **on_scale_ch_readaccess** ()
- void **on_prereadaccess** ()

Private Attributes

- **MHAParser::int_t scale_ch**
  Index of audio channel to scale.
- **MHAParser::float_t factor**
  The scaling factor applied to the selected channel.
- **MHAParser::int_mon_t prepared**
  Keep Track of the prepare/release calls.
- **MHAEvents::patchbay_t< example3_t > patchbay**
  The Event connector.

Additional Inherited Members

5.109.1 Detailed Description

A Plugin class using the openMHA Event mechanism.

This is the third example plugin for the step-by-step tutorial.

5.109.2 Constructor & Destructor Documentation

5.109.2.1 example3_t()

```
example3_t::example3_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

This constructor initializes the configuration language variables and inserts them into the openMHA configuration tree.

It connects the openMHA Events triggered by these configuration variables to the respective callbacks.

© 2005-2021 HörTech gGmbH, Oldenburg

5.109.3 Member Function Documentation

5.109.3.1 on_scale_ch_writeaccess() `void example3_t::on_scale_ch_writeaccess ( ) [private]`

5.109.3.2 on_scale_ch_valuechanged() `void example3_t::on_scale_ch_valuechanged ( ) [private]`

5.109.3.3 on_scale_ch_readaccess() `void example3_t::on_scale_ch_readaccess ( ) [private]`

5.109.3.4 on_prereadaccess() `void example3_t::on_prereadaccess ( ) [private]`

5.109.3.5 prepare() `void example3_t::prepare (` **mhaconfig_t** `&` *signal_info* `) [virtual]`

Plugin preparation.

This plugin checks that the input signal has the waveform domain and contains enough channels.

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t**< **int** > (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.109.3.6 release() `void example3_t::release ( void ) [virtual]`

Bookkeeping only.

Reimplemented from **MHAPlugin::plugin_t**< **int** > (p. 1154).

5.109.3.7 process() `mha_wave_t * example3_t::process ( mha_wave_t * signal )`

Signal processing performed by the plugin.

This plugin multiplies the signal in the selected audio channel by the configured factor.

Parameters

signal	Pointer to the input signal structure.

Returns

Returns a pointer to the input signal structure, with a the signal modified by this plugin. (In-place processing)

5.109.4 Member Data Documentation

5.109.4.1 scale_ch `MHAParser::int_t example3_t::scale_ch [private]`

Index of audio channel to scale.

5.109.4.2 factor `MHAParser::float_t example3_t::factor [private]`

The scaling factor applied to the selected channel.

© 2005-2021 HörTech gGmbH, Oldenburg

5.109.4.3 prepared `MHAParser::int_mon_t example3_t::prepared [private]`

Keep Track of the prepare/release calls.

5.109.4.4 patchbay `MHAEvents::patchbay_t< example3_t> example3_t::patchbay [private]`

The Event connector.

The documentation for this class was generated from the following file:

- **example3.cpp**

5.110 example4_t Class Reference

A Plugin class using the spectral signal.

Inheritance diagram for example4_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **example4_t** (**algo_comm_t** iac, const std::string &configured_name)

  This constructor initializes the configuration language variables and inserts them into the openMHA configuration tree.
- void **prepare** (**mhaconfig_t** &signal_info)

  Plugin preparation.
- void **release** (void)

  Bookkeeping only.
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗signal)

  Signal processing performed by the plugin.

Private Member Functions

- void **on_scale_ch_writeaccess** ()
- void **on_scale_ch_valuechanged** ()
- void **on_scale_ch_readaccess** ()
- void **on_prereadaccess** ()

Private Attributes

- **MHAParser::int_t scale_ch**

  Index of audio channel to scale.
- **MHAParser::float_t factor**

  The scaling factor applied to the selected channel.
- **MHAParser::int_mon_t prepared**

  Keep Track of the prepare/release calls.
- **MHAEvents::patchbay_t**< **example4_t** > **patchbay**

  The Event connector.

Additional Inherited Members

5.110.1 Detailed Description

A Plugin class using the spectral signal.

This is the fourth example plugin for the step-by-step tutorial.

5.110.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.110.2.1 example4_t()

```
example4_t::example4_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

This constructor initializes the configuration language variables and inserts them into the openMHA configuration tree.

It connects the openMHA Events triggered by these configuration variables to the respective callbacks.

5.110.3 Member Function Documentation

5.110.3.1 on_scale_ch_writeaccess()

```
void example4_t::on_scale_ch_writeaccess ( ) [private]
```

5.110.3.2 on_scale_ch_valuechanged()

```
void example4_t::on_scale_ch_valuechanged ( ) [private]
```

5.110.3.3 on_scale_ch_readaccess()

```
void example4_t::on_scale_ch_readaccess ( ) [private]
```

5.110.3.4 on_prereadaccess()

```
void example4_t::on_prereadaccess ( ) [private]
```

5.110.3.5 prepare()

```
void example4_t::prepare (
    mhaconfig_t & signal_info ) [virtual]
```

Plugin preparation.

This plugin checks that the input signal has the spectral domain and contains enough channels.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

5.110.3.6 release() `void example4_t::release ( void ) [virtual]`

Bookkeeping only.

Reimplemented from **MHAPlugin::plugin_t< int >** (p. 1154).

5.110.3.7 process() `mha_spec_t * example4_t::process ( mha_spec_t * signal )`

Signal processing performed by the plugin.

This plugin multiplies the spectral signal in the selected audio channel by the configured factor.

Parameters

signal	Pointer to the input signal structure.

Returns

Returns a pointer to the input signal structure, with a the signal modified by this plugin. (In-place processing)

5.110.4 Member Data Documentation

5.110.4.1 scale_ch `MHAParser::int_t example4_t::scale_ch [private]`

Index of audio channel to scale.

© 2005-2021 HörTech gGmbH, Oldenburg

5.110.4.2 factor `MHAParser::float_t` `example4_t::factor` `[private]`

The scaling factor applied to the selected channel.

5.110.4.3 prepared `MHAParser::int_mon_t` `example4_t::prepared` `[private]`

Keep Track of the prepare/release calls.

5.110.4.4 patchbay `MHAEvents::patchbay_t< example4_t>` `example4_t::patchbay` `[private]`

The Event connector.

The documentation for this class was generated from the following file:

- **example4.cpp**

5.111 example5_t Class Reference

Public Member Functions

- **example5_t** (unsigned int, unsigned int, **mha_real_t**)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

Private Attributes

- unsigned int **channel**
- **mha_real_t** **scale**

5.111.1 Constructor & Destructor Documentation

5.111.1.1 example5_t()

```
example5_t::example5_t (
    unsigned int ichannel,
    unsigned int numchannels,
    mha_real_t iscale )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.111.2 Member Function Documentation

5.111.2.1 process() `mha_spec_t * example5_t::process ( mha_spec_t * spec )`

5.111.3 Member Data Documentation

5.111.3.1 channel `unsigned int example5_t::channel [private]`

5.111.3.2 scale `mha_real_t example5_t::scale [private]`

The documentation for this class was generated from the following file:

- **example5.cpp**

5.112 example6_t Class Reference

Inheritance diagram for example6_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **example6_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** $*$ **process** (**mha_wave_t** $*$)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAParser::int_t channel_no**
- float **rmsdb**
- **MHAEvents::patchbay_t**$<$ **example6_t** $>$ **patchbay**

Additional Inherited Members

5.112.1 Constructor & Destructor Documentation

5.112.1.1 example6_t()

```
example6_t::example6_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.112.2 Member Function Documentation

5.112.2.1 process()

```
mha_wave_t * example6_t::process (
    mha_wave_t * wave )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.112.2.2 prepare() `void example6_t::prepare ( mhaconfig_t & tfcfg ) [virtual]`

Implements **MHAPlugin::plugin_t**< **cfg_t** > (p. 1153).

5.112.2.3 update_cfg() `void example6_t::update_cfg ( ) [private]`

5.112.3 Member Data Documentation

5.112.3.1 channel_no `MHAParser::int_t example6_t::channel_no [private]`

5.112.3.2 rmsdb `float example6_t::rmsdb [private]`

5.112.3.3 patchbay `MHAEvents::patchbay_t< example6_t> example6_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **example6.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.113 example7_t Class Reference

Inheritance diagram for example7_t:

Public Member Functions

- **example7_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **release** (void)
- void **prepare** (**mhaconfig_t** &)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Additional Inherited Members

5.113.1 Constructor & Destructor Documentation

5.113.1.1 example7_t()

```
example7_t::example7_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.113.2 Member Function Documentation

5.113.2.1 release() `void example7_t::release ( void ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t< int >** (p. 1154).

5.113.2.2 prepare() `void example7_t::prepare ( mhaconfig_t & signal_info ) [virtual]`

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

5.113.2.3 process() `mha_wave_t * example7_t::process ( mha_wave_t * signal )`

The documentation for this class was generated from the following files:

- **example7.hh**
- **example7.cpp**

5.114 expression_t Class Reference

Class for separating a string into a left hand value and a right hand value.

5.114.1 Detailed Description

Class for separating a string into a left hand value and a right hand value.

A list of valid operators can be provided. After construction, the class members lval, rval and op contain the apropriate contents.

The documentation for this class was generated from the following file:

- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.115 fader_if_t Class Reference

Inheritance diagram for fader_if_t:

Public Member Functions

- **fader_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t**< **fader_if_t** > **patchbay**
- **MHAParser::float_t tau**
- **MHAParser::vfloat_t newgains**
- **mha_real_t** ∗ **actgains**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.115.1 Constructor & Destructor Documentation

5.115.1.1 fader_if_t()

```
fader_if_t::fader_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.115.2 Member Function Documentation

5.115.2.1 process()

```
mha_spec_t * fader_if_t::process (
    mha_spec_t * s )
```

5.115.2.2 prepare()

```
void fader_if_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< spec_fader_t >** (p. 1153).

5.115.2.3 update_cfg()

```
void fader_if_t::update_cfg ( ) [private]
```

5.115.3 Member Data Documentation

5.115.3.1 patchbay

```
MHAEvents::patchbay_t< fader_if_t> fader_if_t::patchbay [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.115.3.2 tau `MHAParser::float_t fader_if_t::tau [private]`

5.115.3.3 newgains `MHAParser::vfloat_t fader_if_t::newgains [private]`

5.115.3.4 actgains `mha_real_t* fader_if_t::actgains [private]`

The documentation for this class was generated from the following file:

- **fader_spec.cpp**

5.116 fader_wave::fader_wave_if_t Class Reference

Inheritance diagram for fader_wave::fader_wave_if_t:

Public Member Functions

- **fader_wave_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** * **process** (**mha_wave_t** *)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **set_level** ()

Private Attributes

- **MHAParser::vfloat_t gain**
- **MHAParser::float_t ramplen**
- **MHAEvents::patchbay_t< fader_wave_if_t > patchbay**
- bool **prepared**

Additional Inherited Members

5.116.1 Constructor & Destructor Documentation

5.116.1.1 fader_wave_if_t()

```
fader_wave::fader_wave_if_t::fader_wave_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.116.2 Member Function Documentation

5.116.2.1 process()

```
mha_wave_t * fader_wave::fader_wave_if_t::process (
    mha_wave_t * s )
```

5.116.2.2 prepare()

```
void fader_wave::fader_wave_if_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< runtime_cfg_t >** (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.116.2.3 release() `void fader_wave::fader_wave_if_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t**< **runtime_cfg_t** > (p. 1154).

5.116.2.4 set_level() `void fader_wave::fader_wave_if_t::set_level ( ) [private]`

5.116.3 Member Data Documentation

5.116.3.1 gain `MHAParser::vfloat_t fader_wave::fader_wave_if_t::gain [private]`

5.116.3.2 ramplen `MHAParser::float_t fader_wave::fader_wave_if_t::ramplen [private]`

5.116.3.3 patchbay `MHAEvents::patchbay_t< fader_wave_if_t> fader_wave::fader_wave_if_t::patchbay [private]`

5.116.3.4 prepared `bool fader_wave::fader_wave_if_t::prepared [private]`

The documentation for this class was generated from the following file:

- **fader_wave.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.117 fader_wave::level_adapt_t Class Reference

Inheritance diagram for fader_wave::level_adapt_t:

Public Member Functions

- **level_adapt_t** (**mhaconfig_t** cf, **mha_real_t** adapt_len, std::vector< float > l_new_, std::vector< float > l_old_)
- void **update_frame** ()
- std::vector< float > **get_level** () const
- bool **can_update** () const

Private Attributes

- unsigned int **ilen**
- unsigned int **pos**
- **MHAWindow::fun_t wnd**
- std::vector< float > **l_new**
- std::vector< float > **l_old**

Additional Inherited Members

5.117.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.117.1.1 level_adapt_t()

```
fader_wave::level_adapt_t::level_adapt_t (
    mhaconfig_t cf,
    mha_real_t adapt_len,
    std::vector< float > l_new_,
    std::vector< float > l_old_ )
```

5.117.2 Member Function Documentation

5.117.2.1 update_frame()

```
void fader_wave::level_adapt_t::update_frame ( )
```

5.117.2.2 get_level()

```
std::vector<float> fader_wave::level_adapt_t::get_level ( ) const [inline]
```

5.117.2.3 can_update()

```
bool fader_wave::level_adapt_t::can_update ( ) const [inline]
```

5.117.3 Member Data Documentation

5.117.3.1 ilen

```
unsigned int fader_wave::level_adapt_t::ilen [private]
```

5.117.3.2 pos

```
unsigned int fader_wave::level_adapt_t::pos [private]
```

5.117.3.3 wnd

```
MHAWindow::fun_t fader_wave::level_adapt_t::wnd [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.117.3.4 l_new `std::vector<float> fader_wave::level_adapt_t::l_new [private]`

5.117.3.5 l_old `std::vector<float> fader_wave::level_adapt_t::l_old [private]`

The documentation for this class was generated from the following file:

- **fader_wave.cpp**

5.118 fftfbpow::fftfbpow_interface_t Class Reference

Interface class for fftfbpow plugin.

Inheritance diagram for fftfbpow::fftfbpow_interface_t:

Public Member Functions

- **fftfbpow_interface_t** (**algo_comm_t** iac, const std::string &configured_name)
  Constructor with standard MHA constructor parameters.
- void **prepare** (**mhaconfig_t** &tf)
  Standard MHA plugin prepare function.
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗s)
  Standard MHA plugin process fct.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **update_cfg** ()
  Constructs new runtime configuration in thread-safe manner.

Private Attributes

- std::string **name**
  Configured name of this plugin instance.
- **MHAEvents::patchbay_t< fftfbpow_interface_t > patchbay**
  Patchbay to connect to MHA configuration interface.

Additional Inherited Members

5.118.1 Detailed Description

Interface class for fftfbpow plugin.

5.118.2 Constructor & Destructor Documentation

5.118.2.1 fftfbpow_interface_t()

```
fftfbpow::fftfbpow_interface_t::fftfbpow_interface↩
_t (
            algo_comm_t iac,
           const std::string & configured_name )
```

Constructor with standard MHA constructor parameters.

Parameters

iac	Handle to algorithm communication variable space
configured_name	Configured name of this plugin instance

5.118.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.118.3.1 prepare() `void fftfbpow::fftfbpow_interface_t::prepare ( mhaconfig_t & tf ) [virtual]`

Standard MHA plugin prepare function.

Ensures that the input is in the frequency domain, calls **update_cfg()** (p. 496) and inserts fbpow into the AC space.

Parameters

tf	Incoming mha configuration structure, contains information about input signal

Implements **MHAPlugin::plugin_t**< **fftfbpow_t** > (p. 1153).

5.118.3.2 process() `mha_spec_t * fftfbpow::fftfbpow_interface_t::process ( mha_spec_t * s )`

Standard MHA plugin process fct.

Polls new config and calls **process()** (p. 496) of the runtime configuration.

Parameters

s	Input spectrum

Returns

Unchanged input spectrum

5.118.3.3 update_cfg() `void fftfbpow::fftfbpow_interface_t::update_cfg ( ) [private]`

Constructs new runtime configuration in thread-safe manner.

5.118.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.118.4.1 name `std::string fftfbpow::fftfbpow_interface_t::name [private]`

Configured name of this plugin instance.

5.118.4.2 patchbay `MHAEvents::patchbay_t< fftfbpow_interface_t> fftfbpow::fftfbpow↩_interface_t::patchbay [private]`

Patchbay to connect to MHA configuration interface.

The documentation for this class was generated from the following file:

- **fftfbpow.cpp**

5.119 fftfbpow::fftfbpow_t Class Reference

Run time configuration for the fftfbpow plugin.

Inheritance diagram for fftfbpow::fftfbpow_t:

Public Member Functions

- **fftfbpow_t** (**MHAOvlFilter::fftfb_vars_t** &vars, unsigned int nch, unsigned int nfft, **mha_real_t** fs, **algo_comm_t** ac, std::string name)

  Constructor of the run time configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **MHA_AC::waveform_t fbpow**
  AC variable containing the estimated power in each frequency band.

Additional Inherited Members

5.119.1 Detailed Description

Run time configuration for the fftfbpow plugin.

5.119.2 Constructor & Destructor Documentation

5.119.2.1 fftfbpow_t()

```
fftfbpow::fftfbpow_t::fftfbpow_t (
    MHAOvlFilter::fftfb_vars_t & vars,
    unsigned int nch,
    unsigned int nfft,
    mha_real_t fs,
    algo_comm_t ac,
    std::string name )
```

Constructor of the run time configuration.

Parameters

vars	Set of configuration variables for FFT-based overlapping filters
nch	Number of audio input channels
nfft	Length of FFT
fs	Sampling rate
ac	AC space
name	Configured name of plugin interface, used as prefix for AC variable names

5.119.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.119.3.1 fbpow `MHA_AC::waveform_t fftfbpow::fftfbpow_t::fbpow`

AC variable containing the estimated power in each frequency band.

The documentation for this class was generated from the following file:

- **fftfbpow.cpp**

5.120 fftfilter::fftfilter_t Class Reference

Public Member Functions

- **fftfilter_t** (const **MHAParser::mfloat_t** &irs, const unsigned int & **fragsize**, const unsigned int & **channels**, const unsigned int & **fftlen**)

  Initialization of new run-time configuration from channel-specific impulse responses.
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Private Attributes

- unsigned int **irslen**

  Length of the longest impulse response applied.
- unsigned int **fragsize**

  The block size (samples per channel) for waveform audio data.
- unsigned int **fftlen**

  FFT length used for filtering.
- unsigned int **channels**

  Number of prepared audio channels processed by this MHA plugin.
- **MHAFilter::fftfilter_t fftfilt**

  The filter object.

5.120.1 Detailed Description

Run-time configuration class for the fftfilter MHA plugin.

5.120.2 Constructor & Destructor Documentation

5.120.2.1 fftfilter_t()

```
fftfilter::fftfilter_t::fftfilter_t (
    const MHAParser::mfloat_t & irs,
    const unsigned int & fragsize_,
    const unsigned int & channels_,
    const unsigned int & fftlen_ )
```

Initialization of new run-time configuration from channel-specific impulse responses.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

irs	The matrix containing the impulse responses (one response per channel, or the same response for every channels) as set by the parser.
*fragsize_*	The block size (samples per channel) for waveform audio data
*channels_*	The number of prepared audio channels for this MHA plugin.
*fftlen_*	FFT length used for filtering

5.120.3 Member Function Documentation

5.120.3.1 process() `mha_wave_t * fftfilter::fftfilter_t::process ( mha_wave_t * s )`

Let fftfiler object handle the filtering

5.120.4 Member Data Documentation

5.120.4.1 irslen `unsigned int fftfilter::fftfilter_t::irslen [private]`

Length of the longest impulse response applied.

5.120.4.2 fragsize `unsigned int fftfilter::fftfilter_t::fragsize [private]`

The block size (samples per channel) for waveform audio data.

5.120.4.3 fftlen `unsigned int fftfilter::fftfilter_t::fftlen [private]`

FFT length used for filtering.

© 2005-2021 HörTech gGmbH, Oldenburg

5.120.4.4 channels `unsigned int fftfilter::fftfilter_t::channels [private]`

Number of prepared audio channels processed by this MHA plugin.

5.120.4.5 fftfilt `MHAFilter::fftfilter_t fftfilter::fftfilter_t::fftfilt [private]`

The filter object.

The documentation for this class was generated from the following file:

- **fftfilter.cpp**

5.121 fftfilter::interface_t Class Reference

Inheritance diagram for fftfilter::interface_t:

Public Member Functions

- **interface_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **update** ()

Private Attributes

- **MHAParser::mfloat_t irs**
- **MHAParser::int_t fftlen**
- **MHAParser::int_mon_t fftlen_final**
- **MHAEvents::patchbay_t< interface_t > patchbay**

Additional Inherited Members

5.121.1 Detailed Description

Implements the MHA plugin interface for FFTFilter

5.121.2 Constructor & Destructor Documentation

5.121.2.1 interface_t()

```
fftfilter::interface_t::interface_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.121.3 Member Function Documentation

5.121.3.1 process()

```
mha_wave_t * fftfilter::interface_t::process (
    mha_wave_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.121.3.2 prepare() `void fftfilter::interface_t::prepare ( mhaconfig_t & tf ) [virtual]`

Implements **MHAPlugin::plugin_t**< **fftfilter_t** > (p. 1153).

5.121.3.3 update() `void fftfilter::interface_t::update ( ) [private]`

5.121.4 Member Data Documentation

5.121.4.1 irs `MHAParser::mfloat_t fftfilter::interface_t::irs [private]`

5.121.4.2 fftlen `MHAParser::int_t fftfilter::interface_t::fftlen [private]`

5.121.4.3 fftlen_final `MHAParser::int_mon_t fftfilter::interface_t::fftlen_final [private]`

5.121.4.4 patchbay `MHAEvents::patchbay_t< interface_t> fftfilter::interface_t↩ ::patchbay [private]`

The documentation for this class was generated from the following file:

- **fftfilter.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.122 fftfilterbank::fftfb_interface_t Class Reference

Inheritance diagram for fftfilterbank::fftfb_interface_t:

Public Member Functions

- **fftfb_interface_t** (**algo_comm_t** iac, const std::string &configured_name)
  Default values are set and MHA configuration variables registered into the parser.
- void **prepare** (**mhaconfig_t** &)
  Prepare all variables for processing.
- void **release** ()
- **mha_spec_t** $*$ **process** (**mha_spec_t** $*$)
- **mha_wave_t** $*$ **process** (**mha_wave_t** $*$)

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAParser::bool_t return_imag**
- **MHAEvents::patchbay_t**$<$ **fftfb_interface_t** $>$ **patchbay**
- **MHA_AC::int_t nchannels**
- std::string **algo**
- bool **prepared**
- unsigned int **nbands**

Additional Inherited Members

5.122.1 Constructor & Destructor Documentation

5.122.1.1 fftfb_interface_t()

```
fftfilterbank::fftfb_interface_t::fftfb_interface_t ( algo_comm_t iac,
const std::string & configured_name )
```

Default values are set and MHA configuration variables registered into the parser.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

ac	algorithm communication handle
th	chain name
al	algorithm name

5.122.2 Member Function Documentation

5.122.2.1 prepare() `void fftfilterbank::fftfb_interface_t::prepare ( mhaconfig_t & tf ) [virtual]`

Prepare all variables for processing.

In this function, all variables are initialised and the filter shapes for each band are calculated. The filter shapes $W(f)$ are defined as

$$W(f) = W(T(S(f))) = W(x), \quad x = T(S(f)) = T(\hat{f}),$$

$W(x)$ beeing a symmetric window function in the interval $[-1, 1]$ and $S(f)$ the transformation from the linear scale to the given frequency scale (see functions in FreqScaleFun). The function $T(\hat{f})$ transforms the frequency range between the center frequencies $[\hat{f}_{k-1}, \hat{f}_k]$ and $[\hat{f}_k, \hat{f}_{k+1}]$ into the interval $[-1, 0]$ and $[0, 1]$, respectively. This function is realised by the function linscale().

Parameters

tf	Channel configuration

Implements **MHAPlugin::plugin_t< fftfb_plug_t >** (p. 1153).

5.122.2.2 release() `void fftfilterbank::fftfb_interface_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t< fftfb_plug_t >** (p. 1154).

© 2005-2021 HörTech gGmbH, Oldenburg

5.122.2.3 process() [1/2]

mha_spec_t * fftfilterbank::fftfb_interface_t::process (**mha_spec_t** * *s*)

5.122.2.4 process() [2/2]

mha_wave_t * fftfilterbank::fftfb_interface_t::process (**mha_wave_t** * *s*)

5.122.2.5 update_cfg()

void fftfilterbank::fftfb_interface_t::update_cfg ( ) [private]

5.122.3 Member Data Documentation

5.122.3.1 return_imag

MHAParser::bool_t fftfilterbank::fftfb_interface_t::return↩_imag [private]

5.122.3.2 patchbay

MHAEvents::patchbay_t< **fftfb_interface_t**> fftfilterbank↩::fftfb_interface_t::patchbay [private]

5.122.3.3 nchannels

MHA_AC::int_t fftfilterbank::fftfb_interface_t::nchannels [private]

5.122.3.4 algo

std::string fftfilterbank::fftfb_interface_t::algo [private]

5.122.3.5 prepared

bool fftfilterbank::fftfb_interface_t::prepared [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.122.3.6 nbands `unsigned int fftfilterbank::fftfb_interface_t::nbands [private]`

The documentation for this class was generated from the following file:

- **fftfilterbank.cpp**

5.123 fftfilterbank::fftfb_plug_t Class Reference

Inheritance diagram for fftfilterbank::fftfb_plug_t:

Public Member Functions

- **fftfb_plug_t** (**MHAOvlFilter::overlap_save_filterbank_t::vars_t** &, **mhaconfig_t** chcfg, **algo_comm_t** ac, std::string alg, bool return_imag)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **insert** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAOvlFilter::fftfb_ac_info_t fb_acinfo**
- **MHASignal::spectrum_t s_out**
- **MHA_AC::waveform_t imag**
- bool **return_imag_**

Additional Inherited Members

5.123.1 Constructor & Destructor Documentation

5.123.1.1 fftfb_plug_t()

```
fftfilterbank::fftfb_plug_t::fftfb_plug_t (
    MHAOvlFilter::overlap_save_filterbank_t::vars_t & vars,
    mhaconfig_t chcfg,
    algo_comm_t ac,
    std::string alg,
    bool return_imag )
```

5.123.2 Member Function Documentation

5.123.2.1 process() [1/2]

```
mha_spec_t * fftfilterbank::fftfb_plug_t::process (
    mha_spec_t * s )
```

5.123.2.2 process() [2/2]

```
mha_wave_t * fftfilterbank::fftfb_plug_t::process (
    mha_wave_t * s )
```

5.123.2.3 insert()

```
void fftfilterbank::fftfb_plug_t::insert ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.123.3 Member Data Documentation

5.123.3.1 fb_acinfo `MHAOvlFilter::fftfb_ac_info_t fftfilterbank::fftfb_plug_t↩ ::fb_acinfo [private]`

5.123.3.2 s_out `MHASignal::spectrum_t fftfilterbank::fftfb_plug_t::s_out [private]`

5.123.3.3 imag `MHA_AC::waveform_t fftfilterbank::fftfb_plug_t::imag [private]`

5.123.3.4 return_imag_ `bool fftfilterbank::fftfb_plug_t::return_imag_ [private]`

The documentation for this class was generated from the following file:

- **fftfilterbank.cpp**

5.124 fshift::fshift_config_t Class Reference

fshift runtime config class

Public Member Functions

- **fshift_config_t** (**fshift_t** const ∗const plug)
  C'tor of the fshift plugin runtime configuration class.
- ∼**fshift_config_t** ()=default
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- const unsigned int **kmin**
  FFT bin corresponding to fmin.
- const unsigned **kmax**
  FFT bin corresponding to fmax.
- const int **df**
  Frequency shift expressed in FFT bins.
- const **mha_complex_t delta_phi**
  Phase advance per fft frame.
- **mha_complex_t delta_phi_total**
  Sum of all phase advances.

5.124.1 Detailed Description

fshift runtime config class

5.124.2 Constructor & Destructor Documentation

5.124.2.1 fshift_config_t()

`fshift::fshift_config_t::fshift_config_t ( fshift_t const *const plug ) [explicit]`

C'tor of the fshift plugin runtime configuration class.

Parameters

plug	ptr to the plugin interface class. Configuration information is given this way to keep the argument list small.

5.124.2.2 ~fshift_config_t()

`fshift::fshift_config_t::~fshift_config_t ( ) [default]`

5.124.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.124.3.1 process() `mha_spec_t * fshift::fshift_config_t::process ( mha_spec_t * in )`

5.124.4 Member Data Documentation

5.124.4.1 kmin `const unsigned int fshift::fshift_config_t::kmin [private]`

FFT bin corresponding to fmin.

5.124.4.2 kmax `const unsigned fshift::fshift_config_t::kmax [private]`

FFT bin corresponding to fmax.

5.124.4.3 df `const int fshift::fshift_config_t::df [private]`

Frequency shift expressed in FFT bins.

5.124.4.4 delta_phi `const mha_complex_t fshift::fshift_config_t::delta_phi [private]`

Phase advance per fft frame.

5.124.4.5 delta_phi_total `mha_complex_t fshift::fshift_config_t::delta_phi_total [private]`

Sum of all phase advances.

The documentation for this class was generated from the following files:

- **fshift.hh**
- **fshift.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.125 fshift::fshift_t Class Reference

fshift plugin interface class

Inheritance diagram for fshift::fshift_t:

Public Member Functions

- **fshift_t** (**algo_comm_t** iac, const std::string &configured_name)
  Constructs our plugin.
- ∼**fshift_t** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
  Checks for the most recent configuration and defers processing to it.
- void **prepare** (**mhaconfig_t** &)
  Plugin preparation.
- void **release** (void)
- **mha_real_t** **fmin** () const
- **mha_real_t** **fmax** () const
- **mha_real_t** **df** () const

Private Member Functions

- void **update_cfg** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAEvents::patchbay_t< fshift_t > patchbay**
  patch bay for connecting configuration parser events with local member functions:
- **MHAParser::float_t m_fmin**
- **MHAParser::float_t m_fmax**
  upper boundary for frequency shifter
- **MHAParser::float_t m_df**
  Shift frequency in Hz.

Additional Inherited Members

5.125.1 Detailed Description

fshift plugin interface class

5.125.2 Constructor & Destructor Documentation

5.125.2.1 fshift_t()

```
fshift::fshift_t::fshift_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructs our plugin.

5.125.2.2 ~fshift_t()

```
fshift::fshift_t::~fshift_t ( )
```

5.125.3 Member Function Documentation

5.125.3.1 process()

```
mha_spec_t * fshift::fshift_t::process (
    mha_spec_t * signal )
```

Checks for the most recent configuration and defers processing to it.

5.125.3.2 prepare()

```
void fshift::fshift_t::prepare (
    mhaconfig_t & signal_info ) [virtual]
```

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< fshift_config_t >** (p. 1153).

5.125.3.3 release() `void fshift::fshift_t::release ( void ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t< fshift_config_t >** (p. 1154).

5.125.3.4 fmin() `mha_real_t fshift::fshift_t::fmin ( ) const [inline]`

5.125.3.5 fmax() `mha_real_t fshift::fshift_t::fmax ( ) const [inline]`

5.125.3.6 df() `mha_real_t fshift::fshift_t::df ( ) const [inline]`

5.125.3.7 update_cfg() `void fshift::fshift_t::update_cfg ( ) [private]`

5.125.4 Member Data Documentation

5.125.4.1 patchbay `MHAEvents::patchbay_t< fshift_t> fshift::fshift_t::patchbay [private]`

patch bay for connecting configuration parser events with local member functions:

© 2005-2021 HörTech gGmbH, Oldenburg

5.125.4.2 m_fmin `MHAParser::float_t fshift::fshift_t::m_fmin [private]`

5.125.4.3 m_fmax `MHAParser::float_t fshift::fshift_t::m_fmax [private]`

upper boundary for frequency shifter

5.125.4.4 m_df `MHAParser::float_t fshift::fshift_t::m_df [private]`

Shift frequency in Hz.

The documentation for this class was generated from the following files:

- **fshift.hh**
- **fshift.cpp**

5.126 fshift_hilbert::frequency_translator_t Class Reference

Inheritance diagram for fshift_hilbert::frequency_translator_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **frequency_translator_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

Private Member Functions

- void **update** ()

Private Attributes

- **MHAEvents::patchbay_t**< **frequency_translator_t** > **patchbay**
- **MHAParser::vfloat_t df**

  Vector containing the shift frequencies in Hz.
- **MHAParser::float_t fmin**

  Lower boundary for frequency shifter.
- **MHAParser::float_t fmax**

  Upper boundary for frequency shifter.
- **MHAParser::int_t irslen**

  Maximum length of cut off filter response.
- **MHAParser::kw_t phasemode**

  Mode of gain smoothing.

Additional Inherited Members

5.126.1 Constructor & Destructor Documentation

5.126.1.1 frequency_translator_t()

```
fshift_hilbert::frequency_translator_t::frequency↩
_translator_t (
            algo_comm_t iac,
           const std::string & configured_name )
```

5.126.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.126.2.1 process()

`mha_spec_t * fshift_hilbert::frequency_translator_t::process ( mha_spec_t * s )`

5.126.2.2 prepare()

`void fshift_hilbert::frequency_translator_t::prepare ( mhaconfig_t & tf ) [virtual]`

Implements **MHAPlugin::plugin_t< hilbert_shifter_t >** (p. 1153).

5.126.2.3 release()

`void fshift_hilbert::frequency_translator_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t< hilbert_shifter_t >** (p. 1154).

5.126.2.4 update()

`void fshift_hilbert::frequency_translator_t::update ( ) [private]`

5.126.3 Member Data Documentation

5.126.3.1 patchbay

`MHAEvents::patchbay_t< frequency_translator_t> fshift_hilbert↩ ::frequency_translator_t::patchbay [private]`

5.126.3.2 df

`MHAParser::vfloat_t fshift_hilbert::frequency_translator_t::df [private]`

Vector containing the shift frequencies in Hz.

5.126.3.3 fmin

`MHAParser::float_t fshift_hilbert::frequency_translator_t::fmin [private]`

Lower boundary for frequency shifter.

© 2005-2021 HörTech gGmbH, Oldenburg

5.126.3.4 fmax `MHAParser::float_t fshift_hilbert::frequency_translator_t::fmax [private]`

Upper boundary for frequency shifter.

5.126.3.5 irslen `MHAParser::int_t fshift_hilbert::frequency_translator_t::irslen [private]`

Maximum length of cut off filter response.

5.126.3.6 phasemode `MHAParser::kw_t fshift_hilbert::frequency_translator_t↩ ::phasemode [private]`

Mode of gain smoothing.

The documentation for this class was generated from the following file:

- **fshift_hilbert.cpp**

5.127 fshift_hilbert::hilbert_shifter_t Class Reference

Inheritance diagram for fshift_hilbert::hilbert_shifter_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **hilbert_shifter_t** (unsigned int fftlen, unsigned int **channels**, **mha_real_t** srate, unsigned int **kmin**, unsigned int **kmax**, std::vector< **mha_real_t** > dphi, unsigned int **frameshift**, unsigned int maxirslen, unsigned int phasemode)
- ~**hilbert_shifter_t** ()
- void **process** (**mha_spec_t** *)

Private Attributes

- **MHASignal::spectrum_t fullspec**

  Part of the spectrum to be frequency shifted.
- **MHASignal::spectrum_t analytic**

  *Analytic signal, defined as a(t)=x(t)+i*H(x(t))*
- **MHASignal::waveform_t shifted**

  The frequency shifted signal in the time domain.
- **MHASignal::spectrum_t mixw_shift**

  Helper varaible containing the coefficients used to split the spectrum.
- **MHASignal::spectrum_t mixw_ref**

  Helper varaible containing the coefficients used to split the spectrum.
- fftw_plan **plan_spec2analytic**

  FFT plan for the transformation of fullspec into the time domain.
- **mha_fft_t mhafft**

  MHA wrapper object for fftw.
- **MHASignal::waveform_t df**

  Vector holding one delta f value for every channel.
- unsigned int **kmin**

  FFT frame that corresponds to f_min.
- unsigned int **kmax**

  FFT frame that corresponds to f_max.
- unsigned int **frameshift**

  Total phase advance within one fragment.
- std::vector< **mha_complex_t** > **delta_phi**

  Phase advance per frame.
- std::vector< **mha_complex_t** > **delta_phi_total**

  Sum of all phase advances.

Additional Inherited Members

5.127.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.127.1.1 hilbert_shifter_t()

```
fshift_hilbert::hilbert_shifter_t::hilbert_shifter_t (
    unsigned int fftlen,
    unsigned int channels,
    mha_real_t srate,
    unsigned int kmin,
    unsigned int kmax,
    std::vector< mha_real_t > dphi,
    unsigned int frameshift,
    unsigned int maxirslen,
    unsigned int phasemode )
```

5.127.1.2 ~hilbert_shifter_t()

```
fshift_hilbert::hilbert_shifter_t::~hilbert_shifter↩
_t ( )
```

5.127.2 Member Function Documentation

5.127.2.1 process()

```
void fshift_hilbert::hilbert_shifter_t::process (
    mha_spec_t * s )
```

5.127.3 Member Data Documentation

5.127.3.1 fullspec

```
MHASignal::spectrum_t fshift_hilbert::hilbert_shifter_t::fullspec
[private]
```

Part of the spectrum to be frequency shifted.

5.127.3.2 analytic

```
MHASignal::spectrum_t fshift_hilbert::hilbert_shifter_t::analytic
[private]
```

Analytic signal, defined as a(t)=x(t)+i∗H(x(t))

© 2005-2021 HörTech gGmbH, Oldenburg

5.127.3.3 shifted

MHASignal::waveform_t fshift_hilbert::hilbert_shifter_t::shifted [private]

The frequency shifted signal in the time domain.

5.127.3.4 mixw_shift

MHASignal::spectrum_t fshift_hilbert::hilbert_shifter_t↩ ::mixw_shift [private]

Helper varaible containing the coefficients used to split the spectrum.

Contains 1 for every fft bin to be frequency shifted, 0 for all others

5.127.3.5 mixw_ref

MHASignal::spectrum_t fshift_hilbert::hilbert_shifter_t::mixw↩ _ref [private]

Helper varaible containing the coefficients used to split the spectrum.

Contains 0 for every fft bin to be frequency shifted, 1 for all others

5.127.3.6 plan_spec2analytic

fftw_plan fshift_hilbert::hilbert_shifter_t::plan_↩ spec2analytic [private]

FFT plan for the transformation of fullspec into the time domain.

5.127.3.7 mhafft

mha_fft_t fshift_hilbert::hilbert_shifter_t::mhafft [private]

MHA wrapper object for fftw.

5.127.3.8 df

MHASignal::waveform_t fshift_hilbert::hilbert_shifter_t::df [private]

Vector holding one delta f value for every channel.

© 2005-2021 HörTech gGmbH, Oldenburg

5.127.3.9 kmin `unsigned int fshift_hilbert::hilbert_shifter_t::kmin [private]`

FFT frame that corresponds to f_min.

5.127.3.10 kmax `unsigned int fshift_hilbert::hilbert_shifter_t::kmax [private]`

FFT frame that corresponds to f_max.

5.127.3.11 frameshift `unsigned int fshift_hilbert::hilbert_shifter_t::frameshift [private]`

Total phase advance within one fragment.

5.127.3.12 delta_phi `std::vector< mha_complex_t> fshift_hilbert::hilbert_shifter↩_t::delta_phi [private]`

Phase advance per frame.

5.127.3.13 delta_phi_total `std::vector< mha_complex_t> fshift_hilbert::hilbert_↩shifter_t::delta_phi_total [private]`

Sum of all phase advances.

The documentation for this class was generated from the following file:

- **fshift_hilbert.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.128 fw_t Class Reference

Inheritance diagram for fw_t:

Public Member Functions

- **fw_t** ()
- ∼**fw_t** ()
- bool **exit_request** () const

Protected Attributes

- **io_lib_t** ∗ **io_lib**
- int **proc_error**
- int **io_error**

Private Types

- enum **state_t** {
  fw_unprepared, **fw_stopped**, **fw_starting**, **fw_running**,
  fw_stopping, **fw_exiting** }

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **prepare** ()
  preparation for processing
- void **start** ()
  start of processing
- void **stop** ()
  stop/pause of processing
- void **release** ()
  release of IO device
- void **quit** ()
  controlled quit
- void **stopped** (int, int)
- void **started** ()
- int **process** (**mha_wave_t** ∗, **mha_wave_t** ∗∗)
- void **exec_fw_command** ()
- void **load_proc_lib** ()
- void **load_io_lib** ()
- void **fw_sleep_cmd** ()
- void **fw_until_cmd** ()
- void **get_input_signal_dimension** ()
- void **async_read** ()
- void **async_poll_msg** ()
- void **get_parserstate** ()

Static Private Member Functions

- static void **stopped** (void ∗h, int proc_err, int io_err)
- static void **started** (void ∗h)
- static int **process** (void ∗h, **mha_wave_t** ∗sIn, **mha_wave_t** ∗∗sOut)

Private Attributes

- **fw_vars_t prepare_vars**
- **MHAParser::int_mon_t nchannels_out**
- **MHAParser::string_t proc_name**
- **MHAParser::string_t io_name**
- **MHAParser::bool_t exit_on_stop**
- **MHAParser::int_t fw_sleep**
- **MHAParser::string_t fw_until**
- **MHAParser::kw_t fw_cmd**
- **MHAParser::string_mon_t parserstate**
- **MHAParser::string_t errorlog**
- **MHAParser::string_t fatallog**

© 2005-2021 HörTech gGmbH, Oldenburg

- **MHAParser::vstring_t plugins**
- **MHAParser::vstring_t plugin_paths**
- **MHAParser::bool_t dump_mha**
- **MHAParser::string_t inst_name**

  A variable for naming MHA instances.
- **MHAKernel::algo_comm_class_t ac**
- **PluginLoader::mhapluginloader_t * proc_lib**
- **mhaconfig_t cfin**
- **mhaconfig_t cfout**
- **state_t state**
- bool **b_exit_request**
- **MHAParser::string_mon_t proc_error_string**
- **MHAEvents::patchbay_t< fw_t > patchbay**

Additional Inherited Members

5.128.1 Member Enumeration Documentation

5.128.1.1 state_t `enum` **`fw_t::state_t`** `[private]`

Enumerator

fw_unprepared	
fw_stopped	
fw_starting	
fw_running	
fw_stopping	
fw_exiting	

5.128.2 Constructor & Destructor Documentation

5.128.2.1 fw_t() `fw_t::fw_t ( )`

© 2005-2021 HörTech gGmbH, Oldenburg

5.128.2.2 ~fw_t() `fw_t::~fw_t ( )`

5.128.3 Member Function Documentation

5.128.3.1 exit_request() `bool fw_t::exit_request ( ) const [inline]`

5.128.3.2 prepare() `void fw_t::prepare ( ) [private]`

preparation for processing

5.128.3.3 start() `void fw_t::start ( ) [private]`

start of processing

5.128.3.4 stop() `void fw_t::stop ( ) [private]`

stop/pause of processing

5.128.3.5 release() `void fw_t::release ( ) [private]`

release of IO device

5.128.3.6 quit() `void fw_t::quit ( ) [private]`

controlled quit

© 2005-2021 HörTech gGmbH, Oldenburg

5.128.3.7 stopped() [1/2]

```
static void fw_t::stopped (
    void * h,
    int proc_err,
    int io_err ) [inline], [static], [private]
```

5.128.3.8 started() [1/2]

```
static void fw_t::started (
    void * h ) [inline], [static], [private]
```

5.128.3.9 process() [1/2]

```
static int fw_t::process (
    void * h,
    mha_wave_t * sIn,
    mha_wave_t ** sOut ) [inline], [static], [private]
```

5.128.3.10 stopped() [2/2]

```
void fw_t::stopped (
    int proc_err,
    int io_err ) [private]
```

5.128.3.11 started() [2/2]

```
void fw_t::started ( ) [private]
```

5.128.3.12 process() [2/2]

```
int fw_t::process (
    mha_wave_t * s_in,
    mha_wave_t ** s_out ) [private]
```

5.128.3.13 exec_fw_command()

```
void fw_t::exec_fw_command ( ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.128.3.14 load_proc_lib() `void fw_t::load_proc_lib ( ) [private]`

5.128.3.15 load_io_lib() `void fw_t::load_io_lib ( ) [private]`

5.128.3.16 fw_sleep_cmd() `void fw_t::fw_sleep_cmd ( ) [private]`

5.128.3.17 fw_until_cmd() `void fw_t::fw_until_cmd ( ) [private]`

5.128.3.18 get_input_signal_dimension() `void fw_t::get_input_signal_dimension ( ) [private]`

5.128.3.19 async_read() `void fw_t::async_read ( ) [inline], [private]`

5.128.3.20 async_poll_msg() `void fw_t::async_poll_msg ( ) [private]`

5.128.3.21 get_parserstate() `void fw_t::get_parserstate ( ) [private]`

5.128.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.128.4.1 prepare_vars

fw_vars_t fw_t::prepare_vars [private]

5.128.4.2 nchannels_out

MHAParser::int_mon_t fw_t::nchannels_out [private]

5.128.4.3 proc_name

MHAParser::string_t fw_t::proc_name [private]

5.128.4.4 io_name

MHAParser::string_t fw_t::io_name [private]

5.128.4.5 exit_on_stop

MHAParser::bool_t fw_t::exit_on_stop [private]

5.128.4.6 fw_sleep

MHAParser::int_t fw_t::fw_sleep [private]

5.128.4.7 fw_until

MHAParser::string_t fw_t::fw_until [private]

5.128.4.8 fw_cmd

MHAParser::kw_t fw_t::fw_cmd [private]

5.128.4.9 parserstate

MHAParser::string_mon_t fw_t::parserstate [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.128.4.10 errorlog `MHAParser::string_t` `fw_t::errorlog` [private]

5.128.4.11 fatallog `MHAParser::string_t` `fw_t::fatallog` [private]

5.128.4.12 plugins `MHAParser::vstring_t` `fw_t::plugins` [private]

5.128.4.13 plugin_paths `MHAParser::vstring_t` `fw_t::plugin_paths` [private]

5.128.4.14 dump_mha `MHAParser::bool_t` `fw_t::dump_mha` [private]

5.128.4.15 inst_name `MHAParser::string_t` `fw_t::inst_name` [private]

A variable for naming MHA instances.

5.128.4.16 ac `MHAKernel::algo_comm_class_t` `fw_t::ac` [private]

5.128.4.17 proc_lib `PluginLoader::mhapluginloader_t*` `fw_t::proc_lib` [private]

5.128.4.18 cfin `mhaconfig_t` `fw_t::cfin` [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.128.4.19 cfout `mhaconfig_t fw_t::cfout [private]`

5.128.4.20 state `state_t fw_t::state [private]`

5.128.4.21 b_exit_request `bool fw_t::b_exit_request [private]`

5.128.4.22 io_lib `io_lib_t* fw_t::io_lib [protected]`

5.128.4.23 proc_error `int fw_t::proc_error [protected]`

5.128.4.24 io_error `int fw_t::io_error [protected]`

5.128.4.25 proc_error_string `MHAParser::string_mon_t fw_t::proc_error_string [private]`

5.128.4.26 patchbay `MHAEvents::patchbay_t< fw_t> fw_t::patchbay [private]`

The documentation for this class was generated from the following files:

- **mhafw_lib.h**
- **mhafw_lib.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.129 fw_vars_t Class Reference

Public Member Functions

- **fw_vars_t** (**MHAParser::parser_t** &)
- void **lock_srate_fragsize** ()
- void **lock_channels** ()
- void **unlock_srate_fragsize** ()
- void **unlock_channels** ()

Public Attributes

- **MHAParser::int_t pinchannels**
- **MHAParser::int_t pfragmentsize**
- **MHAParser::float_t psrate**

5.129.1 Constructor & Destructor Documentation

5.129.1.1 fw_vars_t() `fw_vars_t::fw_vars_t ( MHAParser::parser_t & p ) [explicit]`

5.129.2 Member Function Documentation

5.129.2.1 lock_srate_fragsize() `void fw_vars_t::lock_srate_fragsize ( )`

5.129.2.2 lock_channels() `void fw_vars_t::lock_channels ( )`

5.129.2.3 unlock_srate_fragsize() `void fw_vars_t::unlock_srate_fragsize ( )`

© 2005-2021 HörTech gGmbH, Oldenburg

5.129.2.4 unlock_channels() `void fw_vars_t::unlock_channels ( )`

5.129.3 Member Data Documentation

5.129.3.1 pinchannels `MHAParser::int_t fw_vars_t::pinchannels`

5.129.3.2 pfragmentsize `MHAParser::int_t fw_vars_t::pfragmentsize`

5.129.3.3 psrate `MHAParser::float_t fw_vars_t::psrate`

The documentation for this class was generated from the following files:

- **mhafw_lib.h**
- **mhafw_lib.cpp**

5.130 gain::gain_if_t Class Reference

Inheritance diagram for gain::gain_if_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **gain_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** $*$ **process** (**mha_wave_t** $*$)
- **mha_spec_t** $*$ **process** (**mha_spec_t** $*$)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

Private Member Functions

- void **update_gain** ()
- void **update_minmax** ()

Private Attributes

- **MHAEvents::patchbay_t**$<$ **gain_if_t** $>$ **patchbay**
- **MHAParser::vfloat_t gains**
- **MHAParser::float_t vmin**
- **MHAParser::float_t vmax**

Additional Inherited Members

5.130.1 Constructor & Destructor Documentation

5.130.1.1 gain_if_t()

```
gain::gain_if_t::gain_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.130.2 Member Function Documentation

5.130.2.1 process() [1/2]

```
mha_wave_t * gain::gain_if_t::process (
    mha_wave_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.130.2.2 process() [2/2]

`mha_spec_t * gain::gain_if_t::process ( mha_spec_t * s )`

5.130.2.3 prepare()

`void gain::gain_if_t::prepare ( mhaconfig_t & tf ) [virtual]`

Implements **MHAPlugin::plugin_t< scaler_t >** (p. 1153).

5.130.2.4 release()

`void gain::gain_if_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t< scaler_t >** (p. 1154).

5.130.2.5 update_gain()

`void gain::gain_if_t::update_gain ( ) [private]`

5.130.2.6 update_minmax()

`void gain::gain_if_t::update_minmax ( ) [private]`

5.130.3 Member Data Documentation

5.130.3.1 patchbay

`MHAEvents::patchbay_t< gain_if_t> gain::gain_if_t::patchbay [private]`

5.130.3.2 gains

`MHAParser::vfloat_t gain::gain_if_t::gains [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.130.3.3 vmin `MHAParser::float_t gain::gain_if_t::vmin [private]`

5.130.3.4 vmax `MHAParser::float_t gain::gain_if_t::vmax [private]`

The documentation for this class was generated from the following file:

- **gain.cpp**

5.131 gain::scaler_t Class Reference

Inheritance diagram for gain::scaler_t:

Public Member Functions

- **scaler_t** (const unsigned int & **channels**, const **MHAParser::vfloat_t** &gains)

Additional Inherited Members

5.131.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.131.1.1 scaler_t()

```
gain::scaler_t::scaler_t (
    const unsigned int & channels,
    const MHAParser::vfloat_t & gains )
```

The documentation for this class was generated from the following file:

- **gain.cpp**

5.132 gsc_adaptive_stage::gsc_adaptive_stage Class Reference

Public Member Functions

- **gsc_adaptive_stage** (**algo_comm_t** & **ac**, const **mhaconfig_t**, int **lenOldSamps**, bool **doCircularComp**, float **mu**, float **alp**, bool **useVAD**, const std::string &vad↩Name_)

  Ctor of the rt processing class.
- ∼**gsc_adaptive_stage** ()=default
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗wavin)

  Processing callback.

Private Member Functions

- void **insert** ()

  Re-insert all AC variables into the AC space.

Private Attributes

- **algo_comm_t ac**

  Handle to AC space.
- unsigned int **lenOldSamps**

  Number of old samples to buffer.
- unsigned int **lenNewSamps**

  Number of new samples.
- unsigned int **bufSize**

  Total buffer size.
- float **frac_old**

  Fraction of new samples to total buffer size.
- **mha_fft_t mha_fft**

  FFT handle.
- unsigned int **nfreq**

  Number of frequency bins.
- unsigned int **nchan**

© 2005-2021 HörTech gGmbH, Oldenburg

Number of channels in input signal.

- unsigned int **desired_chan**

  Channel index containing the desired response.
- bool **doCircularComp**

  Whether to compensate for circular convolution.
- float **mu**

  Linear coefficient for gradient.
- float **alp**

  Autoregressive coefficient for estimating PSD.
- bool **useVAD**

  Wether to use VAD.
- std::string **vadName**

  Name of VAD AC variable.
- **MHA_AC::waveform_t x**

  Buffered input signal.
- **MHA_AC::spectrum_t X**

  FFT of the buffered input signal.
- **MHA_AC::spectrum_t W**

  Time-varying filter.
- **MHA_AC::spectrum_t Y**

  Filter output, frequency domain.
- **MHA_AC::waveform_t y**

  Filter output, time domain.
- **MHA_AC::waveform_t d**

  Desired response.
- **MHA_AC::waveform_t e**

  Error signal, time domain.
- **MHA_AC::spectrum_t E**

  Error signal, frequency domain.
- **MHA_AC::spectrum_t E2**

  Error spectrum multiplied by input spectrum: E2=X∗E.
- **MHA_AC::waveform_t grad**

  Gradient.
- **MHA_AC::spectrum_t Grad**

  FT of the gradient.
- **MHA_AC::waveform_t e_out**

  Error signal.
- **MHA_AC::waveform_t P**

  Signal power estimate.
- **MHA_AC::waveform_t Psum**

  Signal power estimate, summed over all channels.

5.132.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.132.1.1 gsc_adaptive_stage()

```
gsc_adaptive_stage::gsc_adaptive_stage::gsc_adaptive↩
_stage (
            algo_comm_t & ac,
            const mhaconfig_t in_cfg,
            int lenOldSamps,
            bool doCircularComp,
            float mu,
            float alp,
            bool useVAD,
            const std::string & vadName_ )
```

Ctor of the rt processing class.

Parameters

ac	Handle to AC space
in_cfg	Input signal configuration
lenOldSamps	How many old samples to buffer
doCircularComp	Compensate for circular convolution?
mu	Scalar coefficient for gradient
alp	Autoregressive coefficient for estimating PSD
useVAD	Use voice activity detection?
*vadName_*	Name of the VAD AC variable

5.132.1.2 ~gsc_adaptive_stage()

```
gsc_adaptive_stage::gsc_adaptive_stage::~gsc_↩
adaptive_stage ( ) [default]
```

5.132.2 Member Function Documentation

5.132.2.1 process()

```
mha_wave_t * gsc_adaptive_stage::gsc_adaptive_stage::process (
            mha_wave_t * wavin )
```

Processing callback.

Parameters

wavin	input signal

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

Returns a pointer to the output signal

5.132.2.2 insert() `void gsc_adaptive_stage::gsc_adaptive_stage::insert ( ) [private]`

Re-insert all AC variables into the AC space.

5.132.3 Member Data Documentation

5.132.3.1 ac `algo_comm_t gsc_adaptive_stage::gsc_adaptive_stage::ac [private]`

Handle to AC space.

5.132.3.2 lenOldSamps `unsigned int gsc_adaptive_stage::gsc_adaptive_stage::lenOldSamps [private]`

Number of old samples to buffer.

5.132.3.3 lenNewSamps `unsigned int gsc_adaptive_stage::gsc_adaptive_stage::lenNewSamps [private]`

Number of new samples.

5.132.3.4 bufSize `unsigned int gsc_adaptive_stage::gsc_adaptive_stage::bufSize [private]`

Total buffer size.

Must be lenOldSamps+lenNewSamps

© 2005-2021 HörTech gGmbH, Oldenburg

5.132.3.5 frac_old `float gsc_adaptive_stage::gsc_adaptive_stage::frac_old [private]`

Fraction of new samples to total buffer size.

5.132.3.6 mha_fft `mha_fft_t gsc_adaptive_stage::gsc_adaptive_stage::mha_fft [private]`

FFT handle.

5.132.3.7 nfreq `unsigned int gsc_adaptive_stage::gsc_adaptive_stage::nfreq [private]`

Number of frequency bins.

5.132.3.8 nchan `unsigned int gsc_adaptive_stage::gsc_adaptive_stage::nchan [private]`

Number of channels in input signal.

5.132.3.9 desired_chan `unsigned int gsc_adaptive_stage::gsc_adaptive_stage::desired_chan [private]`

Channel index containing the desired response.

Always last channel by convention

5.132.3.10 doCircularComp `bool gsc_adaptive_stage::gsc_adaptive_stage::doCircularComp [private]`

Whether to compensate for circular convolution.

© 2005-2021 HörTech gGmbH, Oldenburg

5.132.3.11 mu `float gsc_adaptive_stage::gsc_adaptive_stage::mu [private]`

Linear coefficient for gradient.

5.132.3.12 alp `float gsc_adaptive_stage::gsc_adaptive_stage::alp [private]`

Autoregressive coefficient for estimating PSD.

5.132.3.13 useVAD `bool gsc_adaptive_stage::gsc_adaptive_stage::useVAD [private]`

Wether to use VAD.

5.132.3.14 vadName `std::string gsc_adaptive_stage::gsc_adaptive_stage::vadName [private]`

Name of VAD AC variable.

5.132.3.15 x `MHA_AC::waveform_t gsc_adaptive_stage::gsc_adaptive_stage::x [private]`

Buffered input signal.

5.132.3.16 X `MHA_AC::spectrum_t gsc_adaptive_stage::gsc_adaptive_stage::X [private]`

FFT of the buffered input signal.

5.132.3.17 W `MHA_AC::spectrum_t gsc_adaptive_stage::gsc_adaptive_stage::W [private]`

Time-varying filter.

© 2005-2021 HörTech gGmbH, Oldenburg

5.132.3.18 Y `MHA_AC::spectrum_t gsc_adaptive_stage::gsc_adaptive_stage::Y [private]`

Filter output, frequency domain.

5.132.3.19 y `MHA_AC::waveform_t gsc_adaptive_stage::gsc_adaptive_stage::y [private]`

Filter output, time domain.

5.132.3.20 d `MHA_AC::waveform_t gsc_adaptive_stage::gsc_adaptive_stage::d [private]`

Desired response.

5.132.3.21 e `MHA_AC::waveform_t gsc_adaptive_stage::gsc_adaptive_stage::e [private]`

Error signal, time domain.

5.132.3.22 E `MHA_AC::spectrum_t gsc_adaptive_stage::gsc_adaptive_stage::E [private]`

Error signal, frequency domain.

5.132.3.23 E2 `MHA_AC::spectrum_t gsc_adaptive_stage::gsc_adaptive_stage::E2 [private]`

Error spectrum multiplied by input spectrum: E2=X∗E.

5.132.3.24 grad `MHA_AC::waveform_t gsc_adaptive_stage::gsc_adaptive_stage::grad [private]`

Gradient.

© 2005-2021 HörTech gGmbH, Oldenburg

5.132.3.25 Grad `MHA_AC::spectrum_t gsc_adaptive_stage::gsc_adaptive_stage::Grad [private]`

FT of the gradient.

5.132.3.26 e_out `MHA_AC::waveform_t gsc_adaptive_stage::gsc_adaptive_stage::e_out [private]`

Error signal.

5.132.3.27 P `MHA_AC::waveform_t gsc_adaptive_stage::gsc_adaptive_stage::P [private]`

Signal power estimate.

5.132.3.28 Psum `MHA_AC::waveform_t gsc_adaptive_stage::gsc_adaptive_stage::Psum [private]`

Signal power estimate, summed over all channels.

The documentation for this class was generated from the following files:

- **gsc_adaptive_stage.hh**
- **gsc_adaptive_stage.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.133 gsc_adaptive_stage::gsc_adaptive_stage_if Class Reference

Plugin interface class.

Inheritance diagram for gsc_adaptive_stage::gsc_adaptive_stage_if:

Public Member Functions

- **gsc_adaptive_stage_if** (**algo_comm_t** iac, const std::string &configured_name)

  Constructs the interface to the adaptive filter plugin.
- ~**gsc_adaptive_stage_if** ()=default
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

  This plugin implements noise reduction using spectral subtraction: By nonnegative subtraction from the output magnitude of the estimated noise magnitude spectrum.
- void **prepare** (**mhaconfig_t** &)

  Plugin preparation.
- void **release** (void)

Private Member Functions

- void **update_cfg** ()

  Update the rt config.
- void **on_model_param_valuechanged** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAEvents::patchbay_t< gsc_adaptive_stage_if > patchbay**
- **MHAParser::int_t lenOldSamps**
  How many old samples should be buffered per filter block.
- **MHAParser::bool_t doCircularComp**
  Whether to compensate for circular convolution.
- **MHAParser::float_t mu**
  Linear coefficient for gradient used in filter adaption.
- **MHAParser::float_t alp**
  Autoregressive coefficient for PSD estimation.
- **MHAParser::bool_t useVAD**
  Wether to use VAD for conditional filter adaption.
- **MHAParser::string_t vadName**
  Name of VAD AC variable.

Additional Inherited Members

5.133.1 Detailed Description

Plugin interface class.

5.133.2 Constructor & Destructor Documentation

5.133.2.1 gsc_adaptive_stage_if()

```
gsc_adaptive_stage::gsc_adaptive_stage_if::gsc_↩
adaptive_stage_if (
            algo_comm_t iac,
           const std::string & configured_name )
```

Constructs the interface to the adaptive filter plugin.

Parameters

ac	Handle to the ac space

5.133.2.2 ~gsc_adaptive_stage_if()

```
gsc_adaptive_stage::gsc_adaptive_stage_if↩
```

```
::~gsc_adaptive_stage_if ( ) [default]
```

5.133.3 Member Function Documentation

5.133.3.1 process()

```
mha_wave_t * gsc_adaptive_stage::gsc_adaptive_stage_if::process (
    mha_wave_t * signal )
```

This plugin implements noise reduction using spectral subtraction: By nonnegative subtraction from the output magnitude of the estimated noise magnitude spectrum.

Parameters

signal	Pointer to the input signal structure.

Returns

Returns a pointer to the input signal structure, with a the signal modified by this plugin.

5.133.3.2 prepare()

```
void gsc_adaptive_stage::gsc_adaptive_stage_if::prepare (
    mhaconfig_t & signal_info ) [virtual]
```

Plugin preparation.

This plugin checks that the input signal has the spectral domain and contains at least one channel

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< gsc_adaptive_stage >** (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.133.3.3 release() `void gsc_adaptive_stage::gsc_adaptive_stage_if::release ( void ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t< gsc_adaptive_stage >** (p. 1154).

5.133.3.4 update_cfg() `void gsc_adaptive_stage::gsc_adaptive_stage_if::update_cfg ( ) [private]`

Update the rt config.

5.133.3.5 on_model_param_valuechanged() `void gsc_adaptive_stage::gsc_adaptive↩_stage_if::on_model_param_valuechanged ( ) [private]`

5.133.4 Member Data Documentation

5.133.4.1 patchbay **`MHAEvents::patchbay_t< gsc_adaptive_stage_if>`** `gsc_adaptive_↩stage::gsc_adaptive_stage_if::patchbay [private]`

5.133.4.2 lenOldSamps **`MHAParser::int_t`** `gsc_adaptive_stage::gsc_adaptive_stage_↩if::lenOldSamps [private]`

How many old samples should be buffered per filter block.

5.133.4.3 doCircularComp **`MHAParser::bool_t`** `gsc_adaptive_stage::gsc_adaptive_↩stage_if::doCircularComp [private]`

Whether to compensate for circular convolution.

© 2005-2021 HörTech gGmbH, Oldenburg

5.133.4.4 mu

`MHAParser::float_t gsc_adaptive_stage::gsc_adaptive_stage_if::mu [private]`

Linear coefficient for gradient used in filter adaption.

5.133.4.5 alp

`MHAParser::float_t gsc_adaptive_stage::gsc_adaptive_stage_if::alp [private]`

Autoregressive coefficient for PSD estimation.

5.133.4.6 useVAD

`MHAParser::bool_t gsc_adaptive_stage::gsc_adaptive_stage_if↩ ::useVAD [private]`

Wether to use VAD for conditional filter adaption.

5.133.4.7 vadName

`MHAParser::string_t gsc_adaptive_stage::gsc_adaptive_stage_↩ if::vadName [private]`

Name of VAD AC variable.

Ignored if useVAD=no

The documentation for this class was generated from the following files:

- **gsc_adaptive_stage_if.hh**
- **gsc_adaptive_stage_if.cpp**

5.134 gtfb_analyzer::gtfb_analyzer_cfg_t Struct Reference

Configuration for Gammatone Filterbank Analyzer.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- unsigned **bands** () const
  Each band is split into this number of bands.
- unsigned **channels** () const
  The number of separate audio channels.
- unsigned **frames** () const
  The number of frames in one chunk.
- **mha_complex_t** ∗ **states** (unsigned channel, unsigned band)
  Returns pointer to filter states for that band.
- **gtfb_analyzer_cfg_t** (unsigned ch, unsigned **frames**, unsigned ord, const std::vector< **mha_complex_t** > &_coeff, const std::vector< **mha_complex_t** > &_norm_phase)
  Create a configuration for Gammatone Filterbank Analyzer.
- ∼**gtfb_analyzer_cfg_t** ()
- **mha_complex_t** & **cvalue** (unsigned frame, unsigned channel, unsigned band)

Public Attributes

- unsigned **order**
  The order of the gammatone filters.
- std::vector< **mha_complex_t** > **coeff**
  The complex coefficients of the gammatone filter bands.
- std::vector< **mha_complex_t** > **norm_phase**
  Combination of normalization and phase correction factor.
- **mha_wave_t s_out**
  Storage for the (complex) output signal.
- std::vector< **mha_complex_t** > **state**
  Storage for Filter state.

5.134.1 Detailed Description

Configuration for Gammatone Filterbank Analyzer.

5.134.2 Constructor & Destructor Documentation

5.134.2.1 gtfb_analyzer_cfg_t()

```
gtfb_analyzer::gtfb_analyzer_cfg_t::gtfb_analyzer_↩
cfg_t (
            unsigned ch,
            unsigned frames,
            unsigned ord,
            const std::vector< mha_complex_t > & _coeff,
            const std::vector< mha_complex_t > & _norm_phase ) [inline]
```

Create a configuration for Gammatone Filterbank Analyzer.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

ch	Number of Audio channels.
frames	Number of Audio frames per chunk.
ord	The order of the gammatone filters.
_coeff	Complex gammatone filter coefficients.
_norm_phase	Normalization and phase correction factors.

5.134.2.2 ~gtfb_analyzer_cfg_t() `gtfb_analyzer::gtfb_analyzer_cfg_t::~gtfb_analyzer↩_cfg_t ( ) [inline]`

5.134.3 Member Function Documentation

5.134.3.1 bands() `unsigned gtfb_analyzer::gtfb_analyzer_cfg_t::bands ( ) const [inline]`

Each band is split into this number of bands.

5.134.3.2 channels() `unsigned gtfb_analyzer::gtfb_analyzer_cfg_t::channels ( ) const [inline]`

The number of separate audio channels.

5.134.3.3 frames() `unsigned gtfb_analyzer::gtfb_analyzer_cfg_t::frames ( ) const [inline]`

The number of frames in one chunk.

© 2005-2021 HörTech gGmbH, Oldenburg

5.134.3.4 states()

```
mha_complex_t* gtfb_analyzer::gtfb_analyzer_cfg_t::states (
    unsigned channel,
    unsigned band ) [inline]
```

Returns pointer to filter states for that band.

5.134.3.5 cvalue()

```
mha_complex_t& gtfb_analyzer::gtfb_analyzer_cfg_t::cvalue (
    unsigned frame,
    unsigned channel,
    unsigned band ) [inline]
```

5.134.4 Member Data Documentation

5.134.4.1 order

```
unsigned gtfb_analyzer::gtfb_analyzer_cfg_t::order
```

The order of the gammatone filters.

5.134.4.2 coeff

```
std::vector< mha_complex_t> gtfb_analyzer::gtfb_analyzer_cfg_t↩
::coeff
```

The complex coefficients of the gammatone filter bands.

5.134.4.3 norm_phase

```
std::vector< mha_complex_t> gtfb_analyzer::gtfb_analyzer_↩
cfg_t::norm_phase
```

Combination of normalization and phase correction factor.

© 2005-2021 HörTech gGmbH, Oldenburg

5.134.4.4 s_out `mha_wave_t` `gtfb_analyzer::gtfb_analyzer_cfg_t::s_out`

Storage for the (complex) output signal.

Each of the real input audio channels is split into frequency bands with complex time signal output. The split complex time signal is again stored in a **mha_wave_t** (p. 840) buffer. Each complex time signal is stored as adjacent real and imaginary channels. Complex output from one source channel is stored in adjacent complex output channels.

Example: If the input has 2 channels ch0 ch1, and **gtfb_analyzer** (p. 96) splits into 3 bands b0 b1 b2, then the order of output channels in s_out is: ch0_b0_real ch0_b0_imag ch0_b1_real ch0_b1_imag ch0_b2_real ch0_b2_imag ch1_b0_real ch1_b1_imag ch1_b1_real ch1_b1_↩ imag ch1_b2_real ch1_b2_imag

5.134.4.5 state `std::vector<` **`mha_complex_t`**`>` `gtfb_analyzer::gtfb_analyzer_cfg_t↩ ::state`

Storage for Filter state.

Holds **channels()** (p. 551) ∗ **bands()** (p. 551) ∗ order complex filter states. Layout↩ : state[(**bands()** (p. 551)∗channel+band)∗order+stage]

The documentation for this struct was generated from the following file:

- **gtfb_analyzer.cpp**

5.135 gtfb_analyzer::gtfb_analyzer_t Class Reference

Gammatone Filterbank Analyzer Plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

Inheritance diagram for gtfb_analyzer::gtfb_analyzer_t:

Public Member Functions

- **gtfb_analyzer_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t**< **gtfb_analyzer_t** > **patchbay**
- bool **prepared**
- **MHAParser::int_t** **order**
- **MHAParser::vcomplex_t** **coeff**
- **MHAParser::vcomplex_t** **norm_phase**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.135.1 Detailed Description

Gammatone Filterbank Analyzer Plugin.

5.135.2 Constructor & Destructor Documentation

5.135.2.1 gtfb_analyzer_t()

```
gtfb_analyzer::gtfb_analyzer_t::gtfb_analyzer_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.135.3 Member Function Documentation

5.135.3.1 process()

```
mha_wave_t * gtfb_analyzer::gtfb_analyzer_t::process (
    mha_wave_t * s )
```

5.135.3.2 prepare()

```
void gtfb_analyzer::gtfb_analyzer_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< gtfb_analyzer_cfg_t >** (p. 1153).

5.135.3.3 release()

```
void gtfb_analyzer::gtfb_analyzer_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< gtfb_analyzer_cfg_t >** (p. 1154).

5.135.3.4 update_cfg()

```
void gtfb_analyzer::gtfb_analyzer_t::update_cfg ( ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.135.4 Member Data Documentation

5.135.4.1 patchbay `MHAEvents::patchbay_t< gtfb_analyzer_t> gtfb_analyzer::gtfb↩_analyzer_t::patchbay [private]`

5.135.4.2 prepared `bool gtfb_analyzer::gtfb_analyzer_t::prepared [private]`

5.135.4.3 order `MHAParser::int_t gtfb_analyzer::gtfb_analyzer_t::order [private]`

5.135.4.4 coeff `MHAParser::vcomplex_t gtfb_analyzer::gtfb_analyzer_t::coeff [private]`

5.135.4.5 norm_phase `MHAParser::vcomplex_t gtfb_analyzer::gtfb_analyzer_t↩::norm_phase [private]`

The documentation for this class was generated from the following file:

- **gtfb_analyzer.cpp**

5.136 gtfb_simd_cfg_t Class Reference

Public Member Functions

- unsigned **get_bands** () const
  Each band is split into this number of bands.
- unsigned **get_channels** () const
  The number of separate audio channels.
- unsigned **get_frames** () const
  The number of frames in one chunk.
- **gtfb_simd_cfg_t** (**gtfb_simd_cfg_t** &&)=delete
- **gtfb_simd_cfg_t** & **operator=** (**gtfb_simd_cfg_t** &&)=delete
- **gtfb_simd_cfg_t** (const **gtfb_simd_cfg_t** &)=delete
- **gtfb_simd_cfg_t** & **operator=** (const **gtfb_simd_cfg_t** &)=delete
- **gtfb_simd_cfg_t** (unsigned ch, unsigned frames, unsigned ord, const std::vector< **mha_complex_t** > &_coeff, const std::vector< **mha_complex_t** > &_norm_phase)
- ∼**gtfb_simd_cfg_t** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s_in)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- unsigned **order**
  The order of the gammatone filters.
- unsigned **bands**
  Number of frequency bands per channel.
- unsigned **channels**
  Number of input audio channels.
- unsigned **bandsXchannels**
  Product of bands and channels.
- std::vector< **mha_complex_t** > **norm_phase**
  Combination of normalization and phase correction factor.
- float * **rinputs**
  input signal (1 sample) multiplied with norm_phase
- float * **iinputs**
- float * **rcoefficients**
  The complex coefficients of the gammatone filter bands.
- float * **icoefficients**
- float * **rstates**
- float * **istates**
- float * **sout_buf**
  output signal buffer, used by s_out.
- float * **large_array**
  Large float array.
- **mha_wave_t s_out**

5.136.1 Detailed Description

Configuration for Gammatone Filterbank SIMD Analyzer.

5.136.2 Constructor & Destructor Documentation

5.136.2.1 gtfb_simd_cfg_t() [1/3]

```
gtfb_simd_cfg_t::gtfb_simd_cfg_t (
    gtfb_simd_cfg_t && ) [delete]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.136.2.2 gtfb_simd_cfg_t() [2/3]

```
gtfb_simd_cfg_t::gtfb_simd_cfg_t (
    const gtfb_simd_cfg_t & ) [delete]
```

5.136.2.3 gtfb_simd_cfg_t() [3/3]

```
gtfb_simd_cfg_t::gtfb_simd_cfg_t (
    unsigned ch,
    unsigned frames,
    unsigned ord,
    const std::vector< mha_complex_t > & _coeff,
    const std::vector< mha_complex_t > & _norm_phase ) [inline]
```

Create a configuration for Gammatone Filterbank Analyzer.

Parameters

ch	Number of Audio channels.
frames	Number of Audio frames per chunk.
ord	The order of the gammatone filters.
_coeff	Complex gammatone filter coefficients.
_norm_phase	Normalization and phase correction factors.

5.136.2.4 ~gtfb_simd_cfg_t()

```
gtfb_simd_cfg_t::~gtfb_simd_cfg_t ( ) [inline]
```

5.136.3 Member Function Documentation

5.136.3.1 get_bands()

```
unsigned gtfb_simd_cfg_t::get_bands ( ) const [inline]
```

Each band is split into this number of bands.

5.136.3.2 get_channels()

```
unsigned gtfb_simd_cfg_t::get_channels ( ) const [inline]
```

The number of separate audio channels.

© 2005-2021 HörTech gGmbH, Oldenburg

5.136.3.3 get_frames() `unsigned gtfb_simd_cfg_t::get_frames ( ) const [inline]`

The number of frames in one chunk.

5.136.3.4 operator=() [1/2] `gtfb_simd_cfg_t& gtfb_simd_cfg_t::operator= ( gtfb_simd_cfg_t && ) [delete]`

5.136.3.5 operator=() [2/2] `gtfb_simd_cfg_t& gtfb_simd_cfg_t::operator= ( const gtfb_simd_cfg_t & ) [delete]`

5.136.3.6 process() `mha_wave_t* gtfb_simd_cfg_t::process ( mha_wave_t * s_in ) [inline]`

5.136.4 Member Data Documentation

5.136.4.1 order `unsigned gtfb_simd_cfg_t::order [private]`

The order of the gammatone filters.

5.136.4.2 bands `unsigned gtfb_simd_cfg_t::bands [private]`

Number of frequency bands per channel.

5.136.4.3 channels `unsigned gtfb_simd_cfg_t::channels [private]`

Number of input audio channels.

© 2005-2021 HörTech gGmbH, Oldenburg

5.136.4.4 bandsXchannels `unsigned gtfb_simd_cfg_t::bandsXchannels [private]`

Product of bands and channels.

5.136.4.5 norm_phase `std::vector< mha_complex_t> gtfb_simd_cfg_t::norm_phase [private]`

Combination of normalization and phase correction factor.

5.136.4.6 rinputs `float* gtfb_simd_cfg_t::rinputs [private]`

input signal (1 sample) multiplied with norm_phase

5.136.4.7 iinputs `float* gtfb_simd_cfg_t::iinputs [private]`

5.136.4.8 rcoefficients `float* gtfb_simd_cfg_t::rcoefficients [private]`

The complex coefficients of the gammatone filter bands.

5.136.4.9 icoefficients `float* gtfb_simd_cfg_t::icoefficients [private]`

5.136.4.10 rstates `float* gtfb_simd_cfg_t::rstates [private]`

Storage for Filter state. Holds **channels()** (p. 559) ∗ **bands()** (p. 559) ∗ order complex filter states. Layout: state[stage ∗ bandsXchannels + **bands()** (p. 559)∗channel+band]

© 2005-2021 HörTech gGmbH, Oldenburg

5.136.4.11 istates `float* gtfb_simd_cfg_t::istates [private]`

5.136.4.12 sout_buf `float* gtfb_simd_cfg_t::sout_buf [private]`

output signal buffer, used by s_out.

Contains bandsXchannels ∗ fragsize ∗2 entries. all real parts come before all complex parts. Order is: channel 0 band 0 real, channel 0 band 1 real, ..., channel 1 band 0 real channel 1 band 1 real, ... channel 0 band 0 imag ... then next sample

5.136.4.13 large_array `float* gtfb_simd_cfg_t::large_array [private]`

Large float array.

All previous pointers point into this array. Contains 3 unused entries to be able to adjust for alignment.

5.136.4.14 s_out `mha_wave_t gtfb_simd_cfg_t::s_out [private]`

Storage for the (complex) output signal. Each of the real input audio channels is split into frequency bands with complex time signal output. The split complex time signal is again stored in a **mha_wave_t** (p. 840) buffer. Each complex time signal is stored as adjacent real and imaginary channels. Complex output from one source channel is stored in adjacent complex output channels.

The documentation for this class was generated from the following file:

- **gtfb_simd.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.137 gtfb_simd_t Class Reference

Inheritance diagram for gtfb_simd_t:

Public Member Functions

- **gtfb_simd_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t**< **gtfb_simd_t** > **patchbay**
- bool **prepared**
- **MHAParser::int_t** **order**
- **MHAParser::vcomplex_t** **coeff**
- **MHAParser::vcomplex_t** **norm_phase**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.137.1 Constructor & Destructor Documentation

5.137.1.1 gtfb_simd_t()

```
gtfb_simd_t::gtfb_simd_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.137.2 Member Function Documentation

5.137.2.1 process()

```
mha_wave_t * gtfb_simd_t::process (
    mha_wave_t * s )
```

5.137.2.2 prepare()

```
void gtfb_simd_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< gtfb_simd_cfg_t >** (p. 1153).

5.137.2.3 update_cfg()

```
void gtfb_simd_t::update_cfg ( ) [private]
```

5.137.3 Member Data Documentation

5.137.3.1 patchbay

```
MHAEvents::patchbay_t< gtfb_simd_t> gtfb_simd_t::patchbay
[private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.137.3.2 prepared `bool gtfb_simd_t::prepared [private]`

5.137.3.3 order `MHAParser::int_t gtfb_simd_t::order [private]`

5.137.3.4 coeff `MHAParser::vcomplex_t gtfb_simd_t::coeff [private]`

5.137.3.5 norm_phase `MHAParser::vcomplex_t gtfb_simd_t::norm_phase [private]`

The documentation for this class was generated from the following file:

- **gtfb_simd.cpp**

5.138 gtfb_simple_rt_t Class Reference

Runtime configuration class of gtfb_simple_bridge plugin.

Public Member Functions

- **gtfb_simple_rt_t** (**algo_comm_t** ac, const std::string &name, **mhaconfig_t** &chcfg, std::vector< **mha_real_t** > cf, std::vector< **mha_real_t** > bw, unsigned int order, unsigned int pre_stages, unsigned int desired_delay, std::vector< **mha_real_t** > &vcltass, std::vector< **mha_real_t** > &resynthesis_gain, const std::string &element_gain_name)
- **mha_wave_t** ∗ **pre_plugin** (**mha_wave_t** ∗s)

  Filter real input signal s with the pre_stages filter orders in each gammatone filter band.
- **mha_wave_t** ∗ **post_plugin** (**mha_wave_t** ∗s)

  Post-filter the complex-valued filter-bank signal s after it has been processed by the loaded plugin.
- const **MHAFilter::gamma_flt_t** & **get_gf** () const

  Const-accessor to contained gammatone filterbank object.

Static Private Member Functions

- static std::vector< **mha_real_t** > **duplicate_vector** (const std::vector< **mha_real_t** > &src, unsigned int nchannels)

  Helper function to repeat the elements in a vector.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- unsigned int **_order**

  Total number of gammatone filter orders.
- unsigned int **_pre_stages**

  Number of filter orders applied before the loaded plugin is invoked.
- unsigned int **nbands**

  Number of frequency bands to produce = number of gammatone filters.
- **MHA_AC::waveform_t imag**

  Storage for the imaginary part of the filterbank signal.
- **MHA_AC::waveform_t accf**

  AC variable to publish the center frequencies of the gammatone filters.
- **MHA_AC::waveform_t acbw**

  AC variable to publish the bandwiths of the gammatone filters.
- **MHASignal::waveform_t input**

  Real part of the filterbank signal.
- **MHASignal::waveform_t output**

  Resynthesized broadband signal, used as the output signal of this plugin.
- **MHAFilter::gamma_flt_t gf**

  The gammatone filter bank implementation.
- **MHA_AC::waveform_t cLTASS**

  AC variable to publish band-specific LTASS level correction values.
- **MHA_AC::waveform_t ac_resynthesis_gain**

  AC variable to publish the configured per-frequency resyntesis gains.
- std::string **element_gain_name_**

  Either an empty string, or the name of an AC variable from which element-wise linear factors are read.
- **algo_comm_t _ac**

  Algorithm Communication Variable space.

5.138.1 Detailed Description

Runtime configuration class of gtfb_simple_bridge plugin.

5.138.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.138.2.1 gtfb_simple_rt_t()

```
gtfb_simple_rt_t::gtfb_simple_rt_t (
    algo_comm_t ac,
    const std::string & name,
    mhaconfig_t & chcfg,
    std::vector< mha_real_t > cf,
    std::vector< mha_real_t > bw,
    unsigned int order,
    unsigned int pre_stages,
    unsigned int desired_delay,
    std::vector< mha_real_t > & vcltass,
    std::vector< mha_real_t > & resynthesis_gain,
    const std::string & element_gain_name )
```

5.138.3 Member Function Documentation

5.138.3.1 pre_plugin()

```
mha_wave_t * gtfb_simple_rt_t::pre_plugin (
    mha_wave_t * s )
```

Filter real input signal s with the pre_stages filter orders in each gammatone filter band.

The real part of the complex output is returned in the return value of the method, the imaginary part is stored into the AC variable.

Parameters

s	real-valued, broad-band input signal

Returns

real part of complex-valued output signal after pre_stages gammatone filter orders have been applied in each band. Order of output bands in real and imaginary output are: (channel0,band0), (channel0,band1), ..., (cannel1,band0), ...

5.138.3.2 post_plugin()

```
mha_wave_t * gtfb_simple_rt_t::post_plugin (
    mha_wave_t * s )
```

Post-filter the complex-valued filter-bank signal s after it has been processed by the loaded plugin.

The remaining gammatone filter orders are applied to restrict the loaded plugin's output signal to the respective bands. After

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s	complex-valued, filter-bank signal. This signal is produced by letting the loaded plugin process the output signal of the pre_plugin method.

Returns

real part of complex-valued output signal after pre_stages gammatone filter orders have been applied in each band. Order of output bands in real and imaginary output are: (channel0,band0), (channel0,band1), ..., (cannel1,band0), ...

5.138.3.3 get_gf() `const MHAFilter::gamma_flt_t& gtfb_simple_rt_t::get_gf ( ) const [inline]`

Const-accessor to contained gammatone filterbank object.

5.138.3.4 duplicate_vector() `std::vector< mha_real_t > gtfb_simple_rt_t::duplicate_vector (`

```
const std::vector< mha_real_t > & src,
unsigned int nchannels ) [static], [private]
```

Helper function to repeat the elements in a vector.

Parameters

src	vector to repeat
nchannels	number of times to repeat input vector

Returns

a vector that returns nchannels repetitions of input vector.

5.138.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.138.4.1 _order `unsigned int gtfb_simple_rt_t::_order [private]`

Total number of gammatone filter orders.

5.138.4.2 _pre_stages `unsigned int gtfb_simple_rt_t::_pre_stages [private]`

Number of filter orders applied before the loaded plugin is invoked.

5.138.4.3 nbands `unsigned int gtfb_simple_rt_t::nbands [private]`

Number of frequency bands to produce = number of gammatone filters.

5.138.4.4 imag `MHA_AC::waveform_t gtfb_simple_rt_t::imag [private]`

Storage for the imaginary part of the filterbank signal.

It is used as the imaginary input signal for the loaded plugin. Furthermore, it is expected that the loaded plugin processes the imaginary part of the data in place.

5.138.4.5 accf `MHA_AC::waveform_t gtfb_simple_rt_t::accf [private]`

AC variable to publish the center frequencies of the gammatone filters.

5.138.4.6 acbw `MHA_AC::waveform_t gtfb_simple_rt_t::acbw [private]`

AC variable to publish the bandwiths of the gammatone filters.

5.138.4.7 input `MHASignal::waveform_t gtfb_simple_rt_t::input [private]`

Real part of the filterbank signal.

It is used as the real input signal to the loaded plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

5.138.4.8 output `MHASignal::waveform_t gtfb_simple_rt_t::output [private]`

Resynthesized broadband signal, used as the output signal of this plugin.

5.138.4.9 gf `MHAFilter::gamma_flt_t gtfb_simple_rt_t::gf [private]`

The gammatone filter bank implementation.

5.138.4.10 cLTASS `MHA_AC::waveform_t gtfb_simple_rt_t::cLTASS [private]`

AC variable to publish band-specific LTASS level correction values.

5.138.4.11 ac_resynthesis_gain `MHA_AC::waveform_t gtfb_simple_rt_t::ac_resynthesis↩_gain [private]`

AC variable to publish the configured per-frequency resyntesis gains.

5.138.4.12 element_gain_name_ `std::string gtfb_simple_rt_t::element_gain_name_ [private]`

Either an empty string, or the name of an AC variable from which element-wise linear factors are read.

5.138.4.13 _ac `algo_comm_t gtfb_simple_rt_t::_ac [private]`

Algorithm Communication Variable space.

The documentation for this class was generated from the following file:

- **gtfb_simple_bridge.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.139 gtfb_simple_t Class Reference

Interface class of gtfb_simple_bridge plugin.

Inheritance diagram for gtfb_simple_t:

Public Member Functions

- **gtfb_simple_t** (**algo_comm_t** iac, const std::string &configured_name)

  Constructor.
- void **prepare** (**mhaconfig_t** &chcfg)

  Prepare contained plugin for signal processing.
- void **release** ()

  Releases contained plugin.
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗sln)

  Process the input signal: Computes the filterbank with the pre-stages, calls the loaded plugin for processing of the filterbank signal, and resynthesizes the result to a modified broadband signal.
- void **setlock** (bool b)

  Locks / unlocks all configuration variables before / after signal processing.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::mhapluginloader_t plug**
  Handle to the loaded plugin.
- **MHAOvlFilter::fscale_bw_t fscale**
  Object determines the filterbank frequencies.
- **MHAParser::int_t order**
  total order of gammatone filter
- **MHAParser::int_t prestages**
  Number of gammatone filter order to process before the loaded plugin processes the filterbank signal.
- **MHAParser::int_t desired_delay**
  Desired group delay in audio samples.
- **MHAParser::string_t element_gain_name**
  Number of AC variable to take element-wise gain factors from for resynthesis.
- **MHAParser::vfloat_mon_t cLTASS**
  Monitoring of LTASS correction values / dB.
- **MHAParser::vfloat_mon_t resynthesis_gain**
  configured frequency-specific resynthesis gains
- **MHAParser::string_mon_t gf_internals**
  For tests and debugging: a serialization of the gammatone filter internals.
- std::string **name_**
  Configured algorithm name, used to name the AC variables.

Additional Inherited Members

5.139.1 Detailed Description

Interface class of gtfb_simple_bridge plugin.

5.139.2 Constructor & Destructor Documentation

5.139.2.1 gtfb_simple_t()

```
gtfb_simple_t::gtfb_simple_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructor.

Registers parser variables.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

ac	Algorithm Communication Variable space
chain	chain name
algo	configured name of this plugin instance

5.139.3 Member Function Documentation

5.139.3.1 prepare() `void gtfb_simple_t::prepare ( mhaconfig_t & chcfg ) [virtual]`

Prepare contained plugin for signal processing.

Allocates the runtime configuration instance. Locks all variables.

Implements **MHAPlugin::plugin_t< gtfb_simple_rt_t >** (p. 1153).

5.139.3.2 release() `void gtfb_simple_t::release ( ) [virtual]`

Releases contained plugin.

Unlocks all variables.

Reimplemented from **MHAPlugin::plugin_t< gtfb_simple_rt_t >** (p. 1154).

5.139.3.3 process() `mha_wave_t * gtfb_simple_t::process ( mha_wave_t * sIn )`

Process the input signal: Computes the filterbank with the pre-stages, calls the loaded plugin for processing of the filterbank signal, and resynthesizes the result to a modified broadband signal.

© 2005-2021 HörTech gGmbH, Oldenburg

5.139.3.4 setlock() `void gtfb_simple_t::setlock ( bool b ) [inline]`

Locks / unlocks all configuration variables before / after signal processing.

5.139.4 Member Data Documentation

5.139.4.1 plug `MHAParser::mhapluginloader_t gtfb_simple_t::plug [private]`

Handle to the loaded plugin.

5.139.4.2 fscale `MHAOvlFilter::fscale_bw_t gtfb_simple_t::fscale [private]`

Object determines the filterbank frequencies.

5.139.4.3 order `MHAParser::int_t gtfb_simple_t::order [private]`

total order of gammatone filter

5.139.4.4 prestages `MHAParser::int_t gtfb_simple_t::prestages [private]`

Number of gammatone filter order to process before the loaded plugin processes the filterbank signal.

5.139.4.5 desired_delay `MHAParser::int_t gtfb_simple_t::desired_delay [private]`

Desired group delay in audio samples.

© 2005-2021 HörTech gGmbH, Oldenburg

5.139.4.6 element_gain_name `MHAParser::string_t gtfb_simple_t::element_gain_name [private]`

Number of AC variable to take element-wise gain factors from for resynthesis.

5.139.4.7 cLTASS `MHAParser::vfloat_mon_t gtfb_simple_t::cLTASS [private]`

Monitoring of LTASS correction values / dB.

5.139.4.8 resynthesis_gain `MHAParser::vfloat_mon_t gtfb_simple_t::resynthesis_gain [private]`

configured frequency-specific resynthesis gains

5.139.4.9 gf_internals `MHAParser::string_mon_t gtfb_simple_t::gf_internals [private]`

For tests and debugging: a serialization of the gammatone filter internals.

5.139.4.10 name_ `std::string gtfb_simple_t::name_ [private]`

Configured algorithm name, used to name the AC variables.

The documentation for this class was generated from the following file:

- **gtfb_simple_bridge.cpp**

5.140 hanning_ramps_t Class Reference

Public Member Functions

- **hanning_ramps_t** (unsigned int, unsigned int)
- ~**hanning_ramps_t** ()
- void **operator()** (**MHASignal::waveform_t** &)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- unsigned int **len_a**
- unsigned int **len_b**
- **mha_real_t** ∗ **ramp_a**
- **mha_real_t** ∗ **ramp_b**

5.140.1 Constructor & Destructor Documentation

5.140.1.1 hanning_ramps_t()

```
hanning_ramps_t::hanning_ramps_t (
    unsigned int la,
    unsigned int lb )
```

5.140.1.2 ∼hanning_ramps_t()

```
hanning_ramps_t::~hanning_ramps_t ( )
```

5.140.2 Member Function Documentation

5.140.2.1 operator()()

```
void hanning_ramps_t::operator() (
    MHASignal::waveform_t & b )
```

5.140.3 Member Data Documentation

5.140.3.1 len_a

```
unsigned int hanning_ramps_t::len_a [private]
```

5.140.3.2 len_b

```
unsigned int hanning_ramps_t::len_b [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.140.3.3 ramp_a `mha_real_t* hanning_ramps_t::ramp_a [private]`

5.140.3.4 ramp_b `mha_real_t* hanning_ramps_t::ramp_b [private]`

The documentation for this class was generated from the following file:

- **spec2wave.cpp**

5.141 identity_t Class Reference

Inheritance diagram for identity_t:

Public Member Functions

- **identity_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** * **process** (**mha_wave_t** *)
- **mha_spec_t** * **process** (**mha_spec_t** *)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.141.1 Constructor & Destructor Documentation

5.141.1.1 identity_t()

```
identity_t::identity_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.141.2 Member Function Documentation

5.141.2.1 process() [1/2]

```
mha_wave_t * identity_t::process (
    mha_wave_t * s )
```

5.141.2.2 process() [2/2]

```
mha_spec_t * identity_t::process (
    mha_spec_t * s )
```

5.141.2.3 prepare()

```
void identity_t::prepare (
    mhaconfig_t & ) [virtual]
```

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

5.141.2.4 release()

```
void identity_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< int >** (p. 1154).

The documentation for this class was generated from the following file:

- **identity.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.142 iirfilter_t Class Reference

Inheritance diagram for iirfilter_t:

Public Member Functions

- **iirfilter_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare_** (**mhaconfig_t** &)
- void **release_** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Additional Inherited Members

5.142.1 Constructor & Destructor Documentation

5.142.1.1 iirfilter_t()

```
iirfilter_t::iirfilter_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.142.2 Member Function Documentation

5.142.2.1 prepare_() `void iirfilter_t::prepare_ ( mhaconfig_t & tf )`

5.142.2.2 release_() `void iirfilter_t::release_ ( ) [inline]`

5.142.2.3 process() `mha_wave_t * iirfilter_t::process ( mha_wave_t * s )`

The documentation for this class was generated from the following file:

- **iirfilter.cpp**

5.143 io_alsa_t Class Reference

MHA IO interface class for ALSA IO.

Inheritance diagram for io_alsa_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **io_alsa_t** (unsigned int fragsize, float samplerate, **IOProcessEvent_t proc_event**, void ∗ **proc_handle**, **IOStartedEvent_t start_event**, void ∗ **start_handle**, **IOStopped↩ Event_t stop_event**, void ∗ **stop_handle**)

  Constructor, receives the callback handles to interact with the MHA framework.
- template<typename T = void>
  void **prepare** (int, int)

  Called after the framework has perpared the processing plugins and the number of input and output channels are fixed.
- void **release** ()

  MHA framework leaves prepared state.
- void **start** ()

  MHA framework calls this function when signal processing should start.
- void **stop** ()

  MHA framework calls this function when signal processing should stop.
- template<> void **prepare** (int nch_in, int nch_out)

Static Public Member Functions

- static void ∗ **thread_start** (void ∗h)

  MHAIOAlsa uses a separate thread that calls the alsa read and write functions to read and write audio samples, these functions are blocking until samples can be read or written.

Private Member Functions

- void **process** ()

Private Attributes

- bool **b_process**
- unsigned int **fw_fragsize**
- unsigned int **fw_samplerate**
- **IOProcessEvent_t proc_event**
- void ∗ **proc_handle**
- **IOStartedEvent_t start_event**
- void ∗ **start_handle**
- **IOStoppedEvent_t stop_event**
- void ∗ **stop_handle**
- **alsa_base_t** ∗ **dev_in**
- **alsa_base_t** ∗ **dev_out**
- pthread_t **proc_thread**
- **alsa_dev_par_parser_t p_in**
- **alsa_dev_par_parser_t p_out**
- **MHAParser::bool_t pcmlink**
- **MHAParser::int_t priority**
- **MHAParser::kw_t format**
- **MHAParser::int_mon_t alsa_start_counter**
- **MHAEvents::patchbay_t**< **io_alsa_t** > **patchbay**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.143.1 Detailed Description

MHA IO interface class for ALSA IO.

5.143.2 Constructor & Destructor Documentation

5.143.2.1 io_alsa_t()

```
io_alsa_t::io_alsa_t (
    unsigned int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle )
```

Constructor, receives the callback handles to interact with the MHA framework.

5.143.3 Member Function Documentation

5.143.3.1 prepare() [1/2]

```
template<typename T >
void io_alsa_t::prepare (
    int nch_in,
    int nch_out )
```

Called after the framework has perpared the processing plugins and the number of input and output channels are fixed.

open pcm streams

5.143.3.2 release()

```
void io_alsa_t::release ( )
```

MHA framework leaves prepared state.

© 2005-2021 HörTech gGmbH, Oldenburg

5.143.3.3 start() `void io_alsa_t::start ( )`

MHA framework calls this function when signal processing should start.

5.143.3.4 stop() `void io_alsa_t::stop ( )`

MHA framework calls this function when signal processing should stop.

5.143.3.5 thread_start() `void * io_alsa_t::thread_start ( void * h ) [static]`

MHAIOAlsa uses a separate thread that calls the alsa read and write functions to read and write audio samples, these functions are blocking until samples can be read or written.

This is the start function of that thread.

5.143.3.6 process() `void io_alsa_t::process ( ) [private]`

5.143.3.7 prepare() [2/2] `template<> void io_alsa_t::prepare ( int nch_in, int nch_out )`

5.143.4 Member Data Documentation

5.143.4.1 b_process `bool io_alsa_t::b_process [private]`

5.143.4.2 fw_fragsize `unsigned int io_alsa_t::fw_fragsize [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.143.4.3 fw_samplerate `unsigned int io_alsa_t::fw_samplerate [private]`

5.143.4.4 proc_event `IOProcessEvent_t io_alsa_t::proc_event [private]`

5.143.4.5 proc_handle `void* io_alsa_t::proc_handle [private]`

5.143.4.6 start_event `IOStartedEvent_t io_alsa_t::start_event [private]`

5.143.4.7 start_handle `void* io_alsa_t::start_handle [private]`

5.143.4.8 stop_event `IOStoppedEvent_t io_alsa_t::stop_event [private]`

5.143.4.9 stop_handle `void* io_alsa_t::stop_handle [private]`

5.143.4.10 dev_in `alsa_base_t* io_alsa_t::dev_in [private]`

5.143.4.11 dev_out `alsa_base_t* io_alsa_t::dev_out [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.143.4.12 proc_thread `pthread_t io_alsa_t::proc_thread [private]`

5.143.4.13 p_in `alsa_dev_par_parser_t io_alsa_t::p_in [private]`

5.143.4.14 p_out `alsa_dev_par_parser_t io_alsa_t::p_out [private]`

5.143.4.15 pcmlink `MHAParser::bool_t io_alsa_t::pcmlink [private]`

5.143.4.16 priority `MHAParser::int_t io_alsa_t::priority [private]`

5.143.4.17 format `MHAParser::kw_t io_alsa_t::format [private]`

5.143.4.18 alsa_start_counter `MHAParser::int_mon_t io_alsa_t::alsa_start_counter [private]`

5.143.4.19 patchbay `MHAEvents::patchbay_t< io_alsa_t> io_alsa_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **MHAIOalsa.cpp**

5.144 io_asterisk_fwcb_t Class Reference

TCP sound-io library's interface to the framework callbacks.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **io_asterisk_fwcb_t** (**IOProcessEvent_t** **proc_event**, void ∗ **proc_handle**, **IO↩StartedEvent_t** **start_event**, void ∗ **start_handle**, **IOStoppedEvent_t** **stop_event**, void ∗ **stop_handle**)

  Constructor stores framework handles and initializes error numbers to 0.
- virtual ∼**io_asterisk_fwcb_t** ()

  Do-nothing destructor.
- virtual void **start** ()

  Call the framework's start callback.
- virtual int **process** (**mha_wave_t** ∗sIn, **mha_wave_t** ∗&sOut)

  Call the frameworks processing callback.
- virtual void **set_errnos** (int **proc_err**, int **io_err**)

  Save error numbers to use during.
- virtual void **stop** ()

  Call the frameworks stop callback.

Private Attributes

- **IOProcessEvent_t** **proc_event**

  Pointer to signal processing callback function.
- **IOStartedEvent_t** **start_event**

  Pointer to start notification callback function.
- **IOStoppedEvent_t** **stop_event**

  Pointer to stop notification callback function.
- void ∗ **proc_handle**

  Handles belonging to framework.
- void ∗ **start_handle**
- void ∗ **stop_handle**
- int **proc_err**

  Errors from the processing callback and from the TCP IO itself are stored here before closing Network handles.
- int **io_err**

5.144.1 Detailed Description

TCP sound-io library's interface to the framework callbacks.

5.144.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.144.2.1 io_asterisk_fwcb_t()

```
io_asterisk_fwcb_t::io_asterisk_fwcb_t (
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle )
```

Constructor stores framework handles and initializes error numbers to 0.

5.144.2.2 ~io_asterisk_fwcb_t()

```
virtual io_asterisk_fwcb_t::~io_asterisk_fwcb_t ( ) [inline], [virtual]
```

Do-nothing destructor.

5.144.3 Member Function Documentation

5.144.3.1 start()

```
void io_asterisk_fwcb_t::start ( ) [virtual]
```

Call the framework's start callback.

5.144.3.2 process()

```
int io_asterisk_fwcb_t::process (
    mha_wave_t * sIn,
    mha_wave_t *& sOut ) [virtual]
```

Call the frameworks processing callback.

Parameters

sIn	The input sound data just received from TCP.
sOut	A pointer to output sound data. Will point to the output sound data storage when the callback finishes.

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

Status, an error number from the signal processing callback. If this is != 0, then the connection should be closed.

5.144.3.3 set_errnos()

```
void io_asterisk_fwcb_t::set_errnos (
    int proc_err,
    int io_err ) [virtual]
```

Save error numbers to use during.

See also

stop (p. 587)

Parameters

proc_err	The error number from the

See also

process (p. 586) callback.

Parameters

io_err	The error number from the io library itself.

5.144.3.4 stop()

```
void io_asterisk_fwcb_t::stop ( ) [virtual]
```

Call the frameworks stop callback.

Uses the error numbers set previously with

See also

set_errnos (p. 587).

© 2005-2021 HörTech gGmbH, Oldenburg

5.144.4 Member Data Documentation

5.144.4.1 proc_event `IOProcessEvent_t io_asterisk_fwcb_t::proc_event [private]`

Pointer to signal processing callback function.

5.144.4.2 start_event `IOStartedEvent_t io_asterisk_fwcb_t::start_event [private]`

Pointer to start notification callback function.

Called when a new TCP connection is established or the user issues the start command while there is a connection.

5.144.4.3 stop_event `IOStoppedEvent_t io_asterisk_fwcb_t::stop_event [private]`

Pointer to stop notification callback function.

Called when the connection is closed.

5.144.4.4 proc_handle `void* io_asterisk_fwcb_t::proc_handle [private]`

Handles belonging to framework.

5.144.4.5 start_handle `void * io_asterisk_fwcb_t::start_handle [private]`

5.144.4.6 stop_handle `void * io_asterisk_fwcb_t::stop_handle [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.144.4.7 proc_err `int io_asterisk_fwcb_t::proc_err [private]`

Errors from the processing callback and from the TCP IO itself are stored here before closing Network handles.

MHAIOTCP is notified by the server when the connection has been taken down, and calls

See also

stop (p. 587) from that callback. Within **stop** (p. 587), these error numbers are read again and transmitted to the framework.

5.144.4.8 io_err `int io_asterisk_fwcb_t::io_err [private]`

The documentation for this class was generated from the following file:

- **MHAIOAsterisk.cpp**

5.145 io_asterisk_parser_t Class Reference

The parser interface of the IOAsterisk library.

Inheritance diagram for io_asterisk_parser_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- virtual const std::string & **get_local_address** () const

  Read parser variable local_address, this is the address of the network interface that should listen for incoming connections.
- virtual unsigned short **get_local_port** () const

  Read parser variable local_port, this is the TCP port that should be used for incoming connections.
- virtual void **set_local_port** (unsigned short port)

  Set parser variable local_port.
- virtual bool **get_server_port_open** () const

  Return the status of the server port as it is known to the parser.
- virtual void **set_server_port_open** (bool open)

  Inform the parser of the current status of the server socket.
- virtual bool **get_connected** () const

  Return the parser's knowledge concerning wether there currently exists an established sound data TCP connection or not.
- virtual void **set_connected** (bool **connected**)

  Inform the parser about the existance of a sound data connection.
- virtual void **set_new_peer** (unsigned short port, const std::string &host)

  Set parser monitor variables peer_port and peer_address, and calls set_connected(true).
- **io_asterisk_parser_t** ()

  Constructor initializes parser variables.
- virtual ∼**io_asterisk_parser_t** ()

  Do-nothing destructor.
- virtual void **debug** (const std::string &message)

Private Attributes

- **MHAParser::string_t local_address**

  Lets the user set the local network interface to listen on.
- **MHAParser::int_t local_port**

  Lets the user choose the local tcp port to listen on.
- **MHAParser::int_mon_t server_port_open**

  Indicates wether the TCP server socket is currently open.
- **MHAParser::int_mon_t connected**

  Indicator if there currently is a sound data connection over TCP.
- **MHAParser::string_mon_t peer_address**

  Display the ip address of the currently connected sound data client.
- **MHAParser::int_mon_t peer_port**

  Display the tcp port used by the current sound data client.
- **MHAParser::string_t debug_filename**

  filename to write debugging info to (if non-empty)
- FILE ∗ **debug_file**

  file handle to write debugging info to

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.145.1 Detailed Description

The parser interface of the IOAsterisk library.

5.145.2 Constructor & Destructor Documentation

5.145.2.1 io_asterisk_parser_t() `io_asterisk_parser_t::io_asterisk_parser_t ( )`

Constructor initializes parser variables.

5.145.2.2 ~io_asterisk_parser_t() `virtual io_asterisk_parser_t::~io_asterisk_parser_t ( ) [inline], [virtual]`

Do-nothing destructor.

5.145.3 Member Function Documentation

5.145.3.1 get_local_address() `virtual const std::string& io_asterisk_parser_t::get_local_address ( ) const [inline], [virtual]`

Read parser variable local_address, this is the address of the network interface that should listen for incoming connections.

Returns

A string containing the address of the local interface as it was set by the user.

© 2005-2021 HörTech gGmbH, Oldenburg

5.145.3.2 get_local_port() `unsigned short io_asterisk_parser_t::get_local_port ( ) const [virtual]`

Read parser variable local_port, this is the TCP port that should be used for incoming connections.

Returns

The local tcp port to listen on as it was chosen by the user. The port number is between MIN_TCP_PORT and MAX_TCP_PORT.

5.145.3.3 set_local_port() `void io_asterisk_parser_t::set_local_port ( unsigned short port ) [virtual]`

Set parser variable local_port.

This is needed when it was set to 0 before: In this case, the OS chooses a free port for the TCP server socket, and the port that it chose has to be published to the user over the parser interface.

Parameters

port	The TCP port number that is currently used. In the range [MIN_TCP_PORT, MAX_TCP_PORT], excluding 0.

Precondition

get_local_port() (p. 591) currently returns 0.

5.145.3.4 get_server_port_open() `bool io_asterisk_parser_t::get_server_port_open ( ) const [virtual]`

Return the status of the server port as it is known to the parser.

Returns

false after initialization, or the value most recently set via

See also

set_server_port_open (p. 592).

© 2005-2021 HörTech gGmbH, Oldenburg

5.145.3.5 set_server_port_open() `void io_asterisk_parser_t::set_server_port_open ( bool open ) [virtual]`

Inform the parser of the current status of the server socket.

Parameters

open	Indicates wether the server socket has just been opened or closed.

Precondition

open may only have the value true if **get_server_port_open()** (p. 592) currently returns false.

Postcondition

See also

get_server_port_open (p. 592) returns the **value** (p. 49) of open.

5.145.3.6 get_connected() `bool io_asterisk_parser_t::get_connected ( ) const [virtual]`

Return the parser's knowledge concerning wether there currently exists an established sound data TCP connection or not.

Returns

false after initialization, or the value most recently set via

See also

set_connected (p. 593).

5.145.3.7 set_connected() `void io_asterisk_parser_t::set_connected ( bool connected ) [virtual]`

Inform the parser about the existance of a sound data connection.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

connected	Indicates wether there currently is a connection or not.

Precondition

connected must not have the same value that is currently returned by

See also

get_connected (p. 593).

Postcondition

See also

get_connected (p. 593) returns the **value** (p. 49) of open.

5.145.3.8 set_new_peer()

```
void io_asterisk_parser_t::set_new_peer (
    unsigned short port,
    const std::string & host ) [virtual]
```

Set parser monitor variables peer_port and peer_address, and calls set_connected(true).

This method should be called when a new connection is established.

Parameters

port	The TCP port number used by the peer.
host	The Internet host where the peer is located.

Precondition

© 2005-2021 HörTech gGmbH, Oldenburg

See also

get_connected (p. 593) currently returns false.

Postcondition

See also

get_connected (p. 593) returns true.

5.145.3.9 debug()

`virtual void io_asterisk_parser_t::debug ( const std::string & message ) [inline], [virtual]`

5.145.4 Member Data Documentation

5.145.4.1 local_address

`MHAParser::string_t io_asterisk_parser_t::local_address [private]`

Lets the user set the local network interface to listen on.

5.145.4.2 local_port

`MHAParser::int_t io_asterisk_parser_t::local_port [private]`

Lets the user choose the local tcp port to listen on.

5.145.4.3 server_port_open

`MHAParser::int_mon_t io_asterisk_parser_t::server_port_open [private]`

Indicates wether the TCP server socket is currently open.

© 2005-2021 HörTech gGmbH, Oldenburg

5.145.4.4 connected `MHAParser::int_mon_t io_asterisk_parser_t::connected [private]`

Indicator if there currently is a sound data connection over TCP.

5.145.4.5 peer_address `MHAParser::string_mon_t io_asterisk_parser_t::peer_address [private]`

Display the ip address of the currently connected sound data client.

5.145.4.6 peer_port `MHAParser::int_mon_t io_asterisk_parser_t::peer_port [private]`

Display the tcp port used by the current sound data client.

5.145.4.7 debug_filename `MHAParser::string_t io_asterisk_parser_t::debug_filename [private]`

filename to write debugging info to (if non-empty)

5.145.4.8 debug_file `FILE* io_asterisk_parser_t::debug_file [private]`

file handle to write debugging info to

The documentation for this class was generated from the following file:

- **MHAIOAsterisk.cpp**

5.146 io_asterisk_sound_t Class Reference

Sound data handling of io tcp library.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **io_asterisk_sound_t** (int **fragsize**, float **samplerate**)
  Initialize sound data handling.
- virtual ∼**io_asterisk_sound_t** ()
  Do-nothing destructor.
- virtual void **prepare** (int **num_inchannels**, int **num_outchannels**)
  Called during prepare, sets number of audio channels and allocates sound data storage.
- virtual void **release** ()
  Called during release.
- virtual int **chunkbytes_in** () const
  Number of bytes that constitute one input sound chunk.
- virtual std::string **header** () const
  Create the tcp sound header lines.
- std::string & **hton** (const **mha_wave_t** ∗s_out)
  Serialize data for network transfer.
- **mha_wave_t** ∗ **ntoh** (const std::string &data)
  Deserialize data from network.

Private Attributes

- int **fragsize**
  Number of sound samples in each channel expected and returned from processing callback.
- float **samplerate**
  Sampling rate.
- int **num_inchannels**
  Number of input channels.
- int **num_outchannels**
- **MHASignal::waveform_t** ∗ **s_in**
  Storage for input signal.
- std::string **output_data**
  Serialized data for network trnasfer.

5.146.1 Detailed Description

Sound data handling of io tcp library.

5.146.2 Constructor & Destructor Documentation

5.146.2.1 io_asterisk_sound_t()

```
io_asterisk_sound_t::io_asterisk_sound_t (
    int fragsize,
    float samplerate )
```

Initialize sound data handling.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

fragsize	Number of sound samples in each channel expected and returned from processing callback.
samplerate	Number of samples per second in each channel.

5.146.2.2 ∼io_asterisk_sound_t() `virtual io_asterisk_sound_t::∼io_asterisk_sound↩_t ( ) [inline], [virtual]`

Do-nothing destructor.

5.146.3 Member Function Documentation

5.146.3.1 prepare() `void io_asterisk_sound_t::prepare ( int num_inchannels, int num_outchannels ) [virtual]`

Called during prepare, sets number of audio channels and allocates sound data storage.

Parameters

num_inchannels	Number of input audio channels.
num_outchannels	Number of output audio channels.

5.146.3.2 release() `void io_asterisk_sound_t::release ( ) [virtual]`

Called during release.

Deletes sound data storage.

© 2005-2021 HörTech gGmbH, Oldenburg

5.146.3.3 chunkbytes_in() `int io_asterisk_sound_t::chunkbytes_in ( ) const [virtual]`

Number of bytes that constitute one input sound chunk.

Returns

Number of bytes to read from TCP connection before invoking signal processing.

5.146.3.4 header() `std::string io_asterisk_sound_t::header ( ) const [virtual]`

Create the tcp sound header lines.

5.146.3.5 hton() `std::string & io_asterisk_sound_t::hton ( const mha_wave_t * s_out )`

Serialize data for network transfer.

5.146.3.6 ntoh() `mha_wave_t * io_asterisk_sound_t::ntoh ( const std::string & data )`

Deserialize data from network.

5.146.4 Member Data Documentation

5.146.4.1 fragsize `int io_asterisk_sound_t::fragsize [private]`

Number of sound samples in each channel expected and returned from processing callback.

© 2005-2021 HörTech gGmbH, Oldenburg

5.146.4.2 samplerate `float io_asterisk_sound_t::samplerate [private]`

Sampling rate.

Number of samples per second in each channel.

5.146.4.3 num_inchannels `int io_asterisk_sound_t::num_inchannels [private]`

Number of input channels.

Number of channels expected from and returned by signal processing callback.

5.146.4.4 num_outchannels `int io_asterisk_sound_t::num_outchannels [private]`

5.146.4.5 s_in `MHASignal::waveform_t* io_asterisk_sound_t::s_in [private]`

Storage for input signal.

5.146.4.6 output_data `std::string io_asterisk_sound_t::output_data [private]`

Serialized data for network trnasfer.

The documentation for this class was generated from the following file:

- **MHAIOAsterisk.cpp**

5.147 io_asterisk_t Class Reference

The tcp sound io library.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **io_asterisk_t** (int fragsize, float samplerate, **IOProcessEvent_t** proc_event, void *proc_handle, **IOStartedEvent_t** start_event, void *start_handle, **IOStoppedEvent_t** stop_event, void *stop_handle)
- void **prepare** (int num_inchannels, int num_outchannels)

  Allocate server socket and start thread waiting for sound data exchange.
- void **start** ()

  Call frameworks start callback if there is a sound data connection at the moment.
- void **stop** ()

  Close the current connection if there is one.
- void **release** ()

  Close the current connection and close the server socket.
- virtual void **accept_loop** ()

  IO thread executes this method.
- virtual void **connection_loop** (**MHA_TCP::Connection** *c)

  IO thread executes this method for each connection.
- virtual void **parse** (const char *cmd, char *retval, unsigned int len)

  Parser interface.
- virtual ~**io_asterisk_t** ()

Private Attributes

- **io_asterisk_parser_t parser**
- **io_asterisk_sound_t sound**
- **io_asterisk_fwcb_t fwcb**
- **MHA_TCP::Server * server**
- **MHA_TCP::Thread * thread**
- **MHA_TCP::Async_Notify notify_start**
- **MHA_TCP::Async_Notify notify_stop**
- **MHA_TCP::Async_Notify notify_release**

5.147.1 Detailed Description

The tcp sound io library.

5.147.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.147.2.1 io_asterisk_t()

```
io_asterisk_t::io_asterisk_t (
    int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle )
```

5.147.2.2 ~io_asterisk_t()

`virtual io_asterisk_t::~io_asterisk_t ( ) [inline], [virtual]`

5.147.3 Member Function Documentation

5.147.3.1 prepare()

```
void io_asterisk_t::prepare (
    int num_inchannels,
    int num_outchannels )
```

Allocate server socket and start thread waiting for sound data exchange.

prepare opens the tcp server socket and starts the io thread that listens for audio data on the tcp socket after doing some sanity checks

5.147.3.2 start()

`void io_asterisk_t::start ( )`

Call frameworks start callback if there is a sound data connection at the moment.

5.147.3.3 stop()

`void io_asterisk_t::stop ( )`

Close the current connection if there is one.

stop IO thread

© 2005-2021 HörTech gGmbH, Oldenburg

5.147.3.4 release() `void io_asterisk_t::release ( )`

Close the current connection and close the server socket.

Stop IO thread and close server socket.

5.147.3.5 accept_loop() `void io_asterisk_t::accept_loop ( ) [virtual]`

IO thread executes this method.

5.147.3.6 connection_loop() `void io_asterisk_t::connection_loop ( MHA_TCP::Connection * c ) [virtual]`

IO thread executes this method for each connection.

Parameters

c	pointer to connection. connection_loop deletes connection before exiting.

5.147.3.7 parse() `virtual void io_asterisk_t::parse ( const char * cmd, char * retval, unsigned int len ) [inline], [virtual]`

Parser interface.

5.147.4 Member Data Documentation

5.147.4.1 parser `io_asterisk_parser_t io_asterisk_t::parser [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.147.4.2 sound `io_asterisk_sound_t io_asterisk_t::sound [private]`

5.147.4.3 fwcb `io_asterisk_fwcb_t io_asterisk_t::fwcb [private]`

5.147.4.4 server `MHA_TCP::Server* io_asterisk_t::server [private]`

5.147.4.5 thread `MHA_TCP::Thread* io_asterisk_t::thread [private]`

5.147.4.6 notify_start `MHA_TCP::Async_Notify io_asterisk_t::notify_start [private]`

5.147.4.7 notify_stop `MHA_TCP::Async_Notify io_asterisk_t::notify_stop [private]`

5.147.4.8 notify_release `MHA_TCP::Async_Notify io_asterisk_t::notify_release [private]`

The documentation for this class was generated from the following file:

- **MHAIOAsterisk.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.148 io_file_t Class Reference

File IO.

Inheritance diagram for io_file_t:

Public Member Functions

- **io_file_t** (int **fragsize**, float **samplerate**, **IOProcessEvent_t** **proc_event**, void * **proc_handle**, **IOStartedEvent_t** **start_event**, void * **start_handle**, **IOStopped↩Event_t** **stop_event**, void * **stop_handle**)
- ~**io_file_t** ()
- void **prepare** (int, int)

  Allocate buffers, activate FILE client and install internal ports.
- void **start** ()
- void **stop** ()
- void **release** ()

  Remove FILE client and deallocate internal ports and buffers.

Private Member Functions

- void **stopped** (int, int)
- void **setlock** (bool locked)

  lock or unlock all parser variables.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- int **fragsize**
- float **samplerate**
- int **nchannels_in**
- int **nchannels_file_in**
- int **nchannels_out**
- **IOProcessEvent_t proc_event**
- void * **proc_handle**
- **IOStartedEvent_t start_event**
- void * **start_handle**
- **IOStoppedEvent_t stop_event**
- void * **stop_handle**
- **MHAParser::string_t filename_input**
- **MHAParser::string_t filename_output**
- **MHAParser::kw_t output_sample_format**
- **MHAParser::int_t startsample**
- **MHAParser::int_t length**
- **MHAParser::bool_t strict_channel_match**
- **MHAParser::bool_t strict_srate_match**
- **MHASignal::waveform_t * s_in**
- **MHASignal::waveform_t * s_file_in**
- **mha_wave_t * s_out**
- bool **b_prepared**
- SNDFILE * **sf_in**
- SNDFILE * **sf_out**
- SF_INFO **sfinf_in**
- SF_INFO **sfinf_out**
- sf_count_t **total_read**

Additional Inherited Members

5.148.1 Detailed Description

File IO.

5.148.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.148.2.1 io_file_t()

```
io_file_t::io_file_t (
    int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle )
```

5.148.2.2 ~io_file_t()

```
io_file_t::~io_file_t ( )
```

5.148.3 Member Function Documentation

5.148.3.1 prepare()

```
void io_file_t::prepare (
    int nch_in,
    int nch_out )
```

Allocate buffers, activate FILE client and install internal ports.

5.148.3.2 start()

```
void io_file_t::start ( )
```

5.148.3.3 stop()

```
void io_file_t::stop ( )
```

5.148.3.4 release()

```
void io_file_t::release ( )
```

Remove FILE client and deallocate internal ports and buffers.

© 2005-2021 HörTech gGmbH, Oldenburg

5.148.3.5 stopped()

```
void io_file_t::stopped (
    int proc_err,
    int io_err ) [private]
```

5.148.3.6 setlock()

```
void io_file_t::setlock (
    bool locked ) [private]
```

lock or unlock all parser variables.

Used in prepare/release.

Parameters

locked	When true, locks. When false, unlocks.

5.148.4 Member Data Documentation

5.148.4.1 fragsize

```
int io_file_t::fragsize [private]
```

5.148.4.2 samplerate

```
float io_file_t::samplerate [private]
```

5.148.4.3 nchannels_in

```
int io_file_t::nchannels_in [private]
```

5.148.4.4 nchannels_file_in

```
int io_file_t::nchannels_file_in [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.148.4.5 nchannels_out `int io_file_t::nchannels_out [private]`

5.148.4.6 proc_event `IOProcessEvent_t io_file_t::proc_event [private]`

5.148.4.7 proc_handle `void* io_file_t::proc_handle [private]`

5.148.4.8 start_event `IOStartedEvent_t io_file_t::start_event [private]`

5.148.4.9 start_handle `void* io_file_t::start_handle [private]`

5.148.4.10 stop_event `IOStoppedEvent_t io_file_t::stop_event [private]`

5.148.4.11 stop_handle `void* io_file_t::stop_handle [private]`

5.148.4.12 filename_input `MHAParser::string_t io_file_t::filename_input [private]`

5.148.4.13 filename_output `MHAParser::string_t io_file_t::filename_output [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.148.4.14 output_sample_format

`MHAParser::kw_t io_file_t::output_sample_format [private]`

5.148.4.15 startsample

`MHAParser::int_t io_file_t::startsample [private]`

5.148.4.16 length

`MHAParser::int_t io_file_t::length [private]`

5.148.4.17 strict_channel_match

`MHAParser::bool_t io_file_t::strict_channel_match [private]`

5.148.4.18 strict_srate_match

`MHAParser::bool_t io_file_t::strict_srate_match [private]`

5.148.4.19 s_in

`MHASignal::waveform_t* io_file_t::s_in [private]`

5.148.4.20 s_file_in

`MHASignal::waveform_t* io_file_t::s_file_in [private]`

5.148.4.21 s_out

`mha_wave_t* io_file_t::s_out [private]`

5.148.4.22 b_prepared

`bool io_file_t::b_prepared [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.148.4.23 sf_in `SNDFILE* io_file_t::sf_in [private]`

5.148.4.24 sf_out `SNDFILE* io_file_t::sf_out [private]`

5.148.4.25 sfinf_in `SF_INFO io_file_t::sfinf_in [private]`

5.148.4.26 sfinf_out `SF_INFO io_file_t::sfinf_out [private]`

5.148.4.27 total_read `sf_count_t io_file_t::total_read [private]`

The documentation for this class was generated from the following file:

- **MHAIOFile.cpp**

5.149 io_lib_t Class Reference

Class for loading MHA sound IO module.

Inheritance diagram for io_lib_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **io_lib_t** (int fragsize, float samplerate, **IOProcessEvent_t** proc_event, void ∗proc_↩handle, **IOStartedEvent_t** start_event, void ∗start_handle, **IOStoppedEvent_t** stop_↩event, void ∗stop_handle, std::string libname)

  load and initialize MHA sound io module.
- ∼**io_lib_t** ()

  Deinitialize and unload this MHA sound io module.
- void **prepare** (unsigned int inch, unsigned int outch)

  Prepare the sound io module.
- void **start** ()

  Tell the sound io module to start sound processing.
- void **stop** ()
- void **release** ()
- std::string **lib_str_error** (int err)
- std::string **get_documentation** () const
- std::vector< std::string > **get_categories** () const

Protected Member Functions

- void **test_error** ()

Protected Attributes

- int **lib_err**
- **pluginlib_t lib_handle**
- void ∗ **lib_data**
- **IOInit_t IOInit_cb**
- **IOPrepare_t IOPrepare_cb**
- **IOStart_t IOStart_cb**
- **IOStop_t IOStop_cb**
- **IORelease_t IORelease_cb**
- **IOSetVar_t IOSetVar_cb**
- **IOStrError_t IOStrError_cb**
- **IODestroy_t IODestroy_cb**
- std::string **plugin_documentation**
- std::vector< std::string > **plugin_categories**

Additional Inherited Members

5.149.1 Detailed Description

Class for loading MHA sound IO module.

© 2005-2021 HörTech gGmbH, Oldenburg

5.149.2 Constructor & Destructor Documentation

5.149.2.1 io_lib_t()

```
io_lib_t::io_lib_t (
    int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle,
    std::string libname )
```

load and initialize MHA sound io module.

5.149.2.2 ~io_lib_t()

```
io_lib_t::~io_lib_t ( )
```

Deinitialize and unload this MHA sound io module.

5.149.3 Member Function Documentation

5.149.3.1 prepare()

```
void io_lib_t::prepare (
    unsigned int inch,
    unsigned int outch )
```

Prepare the sound io module.

After preparation, the sound io module may start the sound processing at any time (external trigger). When the sound processing is started, the sound io module will call the start_event callback.

Parameters

inch	number of input channels
outch	number of output channels

© 2005-2021 HörTech gGmbH, Oldenburg

5.149.3.2 start() `void io_lib_t::start ( )`

Tell the sound io module to start sound processing.

Some io modules need this, for others that wait for external events this method might do nothing.

5.149.3.3 stop() `void io_lib_t::stop ( )`

5.149.3.4 release() `void io_lib_t::release ( )`

5.149.3.5 lib_str_error() `std::string io_lib_t::lib_str_error ( int err )`

5.149.3.6 get_documentation() `std::string io_lib_t::get_documentation ( ) const [inline]`

5.149.3.7 get_categories() `std::vector<std::string> io_lib_t::get_categories ( ) const [inline]`

5.149.3.8 test_error() `void io_lib_t::test_error ( ) [protected]`

5.149.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.149.4.1 lib_err `int io_lib_t::lib_err [protected]`

5.149.4.2 lib_handle `pluginlib_t io_lib_t::lib_handle [protected]`

5.149.4.3 lib_data `void* io_lib_t::lib_data [protected]`

5.149.4.4 IOInit_cb `IOInit_t io_lib_t::IOInit_cb [protected]`

5.149.4.5 IOPrepare_cb `IOPrepare_t io_lib_t::IOPrepare_cb [protected]`

5.149.4.6 IOStart_cb `IOStart_t io_lib_t::IOStart_cb [protected]`

5.149.4.7 IOStop_cb `IOStop_t io_lib_t::IOStop_cb [protected]`

5.149.4.8 IORelease_cb `IORelease_t io_lib_t::IORelease_cb [protected]`

5.149.4.9 IOSetVar_cb `IOSetVar_t io_lib_t::IOSetVar_cb [protected]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.149.4.10 IOStrError_cb `IOStrError_t io_lib_t::IOStrError_cb [protected]`

5.149.4.11 IODestroy_cb `IODestroy_t io_lib_t::IODestroy_cb [protected]`

5.149.4.12 plugin_documentation `std::string io_lib_t::plugin_documentation [protected]`

5.149.4.13 plugin_categories `std::vector<std::string> io_lib_t::plugin_categories [protected]`

The documentation for this class was generated from the following files:

- **mhafw_lib.h**
- **mhafw_lib.cpp**

5.150 io_parser_t Class Reference

Main class for Parser IO.

Inheritance diagram for io_parser_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **io_parser_t** (unsigned int **fragsize**, **IOProcessEvent_t** **proc_event**, void * **proc_handle**, **IOStartedEvent_t** **start_event**, void * **start_handle**, **IOStoppedEvent_t** **stop_event**, void * **stop_handle**)
- ~**io_parser_t** ()
- void **prepare** (int, int)

  Allocate buffers, activate JACK client and install internal ports.
- void **start** ()
- void **stop** ()
- void **release** ()

  Remove JACK client and deallocate internal ports and buffers.

Private Member Functions

- void **stopped** (int, int)
- void **started** ()
- void **process_frame** ()

Private Attributes

- unsigned int **fragsize**
- unsigned int **nchannels_in**
- unsigned int **nchannels_out**
- **IOProcessEvent_t** **proc_event**
- void * **proc_handle**
- **IOStartedEvent_t** **start_event**
- void * **start_handle**
- **IOStoppedEvent_t** **stop_event**
- void * **stop_handle**
- **MHAParser::mfloat_t** **input**
- **MHAParser::mfloat_mon_t** **output**
- **MHASignal::waveform_t** * **s_in**
- **mha_wave_t** * **s_out**
- bool **b_fw_started**
- bool **b_stopped**
- bool **b_prepared**
- bool **b_starting**
- **MHAEvents::patchbay_t**< **io_parser_t** > **patchbay**

Additional Inherited Members

5.150.1 Detailed Description

Main class for Parser IO.

© 2005-2021 HörTech gGmbH, Oldenburg

5.150.2 Constructor & Destructor Documentation

5.150.2.1 io_parser_t()

```
io_parser_t::io_parser_t (
 unsigned int fragsize,
  IOProcessEvent_t proc_event,
 void * proc_handle,
  IOStartedEvent_t start_event,
 void * start_handle,
  IOStoppedEvent_t stop_event,
 void * stop_handle )
```

5.150.2.2 ~io_parser_t()

```
io_parser_t::~io_parser_t ( )
```

5.150.3 Member Function Documentation

5.150.3.1 prepare()

```
void io_parser_t::prepare (
 int nch_in,
 int nch_out )
```

Allocate buffers, activate JACK client and install internal ports.

5.150.3.2 start()

```
void io_parser_t::start ( )
```

5.150.3.3 stop()

```
void io_parser_t::stop ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.150.3.4 release() `void io_parser_t::release ( )`

Remove JACK client and deallocate internal ports and buffers.

5.150.3.5 stopped() `void io_parser_t::stopped ( int proc_err, int io_err ) [private]`

5.150.3.6 started() `void io_parser_t::started ( ) [private]`

5.150.3.7 process_frame() `void io_parser_t::process_frame ( ) [private]`

5.150.4 Member Data Documentation

5.150.4.1 fragsize `unsigned int io_parser_t::fragsize [private]`

5.150.4.2 nchannels_in `unsigned int io_parser_t::nchannels_in [private]`

5.150.4.3 nchannels_out `unsigned int io_parser_t::nchannels_out [private]`

5.150.4.4 proc_event `IOProcessEvent_t io_parser_t::proc_event [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.150.4.5 proc_handle `void* io_parser_t::proc_handle [private]`

5.150.4.6 start_event `IOStartedEvent_t io_parser_t::start_event [private]`

5.150.4.7 start_handle `void* io_parser_t::start_handle [private]`

5.150.4.8 stop_event `IOStoppedEvent_t io_parser_t::stop_event [private]`

5.150.4.9 stop_handle `void* io_parser_t::stop_handle [private]`

5.150.4.10 input `MHAParser::mfloat_t io_parser_t::input [private]`

5.150.4.11 output `MHAParser::mfloat_mon_t io_parser_t::output [private]`

5.150.4.12 s_in `MHASignal::waveform_t* io_parser_t::s_in [private]`

5.150.4.13 s_out `mha_wave_t* io_parser_t::s_out [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.150.4.14 b_fw_started `bool io_parser_t::b_fw_started [private]`

5.150.4.15 b_stopped `bool io_parser_t::b_stopped [private]`

5.150.4.16 b_prepared `bool io_parser_t::b_prepared [private]`

5.150.4.17 b_starting `bool io_parser_t::b_starting [private]`

5.150.4.18 patchbay `MHAEvents::patchbay_t< io_parser_t> io_parser_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **MHAIOParser.cpp**

5.151 io_tcp_fwcb_t Class Reference

TCP sound-io library's interface to the framework callbacks.

Public Member Functions

- **io_tcp_fwcb_t** (**IOProcessEvent_t** **proc_event**, void * **proc_handle**, **IOStarted↩Event_t** **start_event**, void * **start_handle**, **IOStoppedEvent_t** **stop_event**, void * **stop_handle**)
  Constructor stores framework handles and initializes error numbers to 0.
- virtual ~**io_tcp_fwcb_t** ()
  Do-nothing destructor.
- virtual void **start** ()
  Call the framework's start callback.
- virtual int **process** (**mha_wave_t** *sIn, **mha_wave_t** *&sOut)
  Call the frameworks processing callback.
- virtual void **set_errnos** (int **proc_err**, int **io_err**)
  Save error numbers to use during.
- virtual void **stop** ()
  Call the frameworks stop callback.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **IOProcessEvent_t proc_event**
  Pointer to signal processing callback function.
- **IOStartedEvent_t start_event**
  Pointer to start notification callback function.
- **IOStoppedEvent_t stop_event**
  Pointer to stop notification callback function.
- void * **proc_handle**
  Handles belonging to framework.
- void * **start_handle**
- void * **stop_handle**
- int **proc_err**
  Errors from the processing callback and from the TCP IO itself are stored here before closing Network handles.
- int **io_err**

5.151.1 Detailed Description

TCP sound-io library's interface to the framework callbacks.

5.151.2 Constructor & Destructor Documentation

5.151.2.1 io_tcp_fwcb_t()

```
io_tcp_fwcb_t::io_tcp_fwcb_t (
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle )
```

Constructor stores framework handles and initializes error numbers to 0.

5.151.2.2 ~io_tcp_fwcb_t()

`virtual io_tcp_fwcb_t::~io_tcp_fwcb_t ( ) [inline], [virtual]`

Do-nothing destructor.

© 2005-2021 HörTech gGmbH, Oldenburg

5.151.3 Member Function Documentation

5.151.3.1 start() `void io_tcp_fwcb_t::start ( ) [virtual]`

Call the framework's start callback.

5.151.3.2 process() `int io_tcp_fwcb_t::process (`

```
mha_wave_t * sIn,
mha_wave_t *& sOut ) [virtual]
```

Call the frameworks processing callback.

Parameters

sIn	The input sound data just received from TCP.
sOut	A pointer to output sound data. Will point to the output sound data storage when the callback finishes.

Returns

Status, an error number from the signal processing callback. If this is != 0, then the connection should be closed.

5.151.3.3 set_errnos() `void io_tcp_fwcb_t::set_errnos (`

```
int proc_err,
int io_err ) [virtual]
```

Save error numbers to use during.

See also

stop (p. 624)

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

proc_err	The error number from the

See also

process (p. 623) callback.

Parameters

io_err	The error number from the io library itself.

5.151.3.4 stop() `void io_tcp_fwcb_t::stop ( ) [virtual]`

Call the frameworks stop callback.

Uses the error numbers set previously with

See also

set_errnos (p. 623).

5.151.4 Member Data Documentation

5.151.4.1 proc_event `IOProcessEvent_t io_tcp_fwcb_t::proc_event [private]`

Pointer to signal processing callback function.

5.151.4.2 start_event `IOStartedEvent_t io_tcp_fwcb_t::start_event [private]`

Pointer to start notification callback function.

Called when a new TCP connection is established or the user issues the start command while there is a connection.

© 2005-2021 HörTech gGmbH, Oldenburg

5.151.4.3 stop_event `IOStoppedEvent_t io_tcp_fwcb_t::stop_event [private]`

Pointer to stop notification callback function.

Called when the connection is closed.

5.151.4.4 proc_handle `void* io_tcp_fwcb_t::proc_handle [private]`

Handles belonging to framework.

5.151.4.5 start_handle `void * io_tcp_fwcb_t::start_handle [private]`

5.151.4.6 stop_handle `void * io_tcp_fwcb_t::stop_handle [private]`

5.151.4.7 proc_err `int io_tcp_fwcb_t::proc_err [private]`

Errors from the processing callback and from the TCP IO itself are stored here before closing Network handles.

MHAIOTCP is notified by the server when the connection has been taken down, and calls

See also

stop (p. 624) from that callback. Within **stop** (p. 624), these error numbers are read again and transmitted to the framework.

5.151.4.8 io_err `int io_tcp_fwcb_t::io_err [private]`

The documentation for this class was generated from the following file:

- **MHAIOTCP.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.152 io_tcp_parser_t Class Reference

The parser interface of the IOTCP library.

Inheritance diagram for io_tcp_parser_t:

Public Member Functions

- virtual const std::string & **get_local_address** () const

  Read parser variable local_address, this is the address of the network interface that should listen for incoming connections.
- virtual unsigned short **get_local_port** () const

  Read parser variable local_port, this is the TCP port that should be used for incoming connections.
- virtual void **set_local_port** (unsigned short port)

  Set parser variable local_port.
- virtual bool **get_server_port_open** () const

  Return the status of the server port as it is known to the parser.
- virtual void **set_server_port_open** (bool open)

  Inform the parser of the current status of the server socket.
- virtual bool **get_connected** () const

  Return the parser's knowledge concerning wether there currently exists an established sound data TCP connection or not.
- virtual void **set_connected** (bool **connected**)

  Inform the parser about the existance of a sound data connection.
- virtual void **set_new_peer** (unsigned short port, const std::string &host)

  Set parser monitor variables peer_port and peer_address, and calls set_connected(true).
- **io_tcp_parser_t** ()

  Constructor initializes parser variables.
- virtual ~**io_tcp_parser_t** ()

  Do-nothing destructor.
- virtual void **debug** (const std::string &message)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::string_t local_address**
  Lets the user set the local network interface to listen on.
- **MHAParser::int_t local_port**
  Lets the user choose the local tcp port to listen on.
- **MHAParser::int_mon_t server_port_open**
  Indicates wether the TCP server socket is currently open.
- **MHAParser::int_mon_t connected**
  Indicator if there currently is a sound data connection over TCP.
- **MHAParser::string_mon_t peer_address**
  Display the ip address of the currently connected sound data client.
- **MHAParser::int_mon_t peer_port**
  Display the tcp port used by the current sound data client.
- **MHAParser::string_t debug_filename**
  filename to write debugging info to (if non-empty)
- FILE * **debug_file**
  file handle to write debugging info to

Additional Inherited Members

5.152.1 Detailed Description

The parser interface of the IOTCP library.

5.152.2 Constructor & Destructor Documentation

5.152.2.1 io_tcp_parser_t() `io_tcp_parser_t::io_tcp_parser_t ( )`

Constructor initializes parser variables.

5.152.2.2 ~io_tcp_parser_t() `virtual io_tcp_parser_t::~io_tcp_parser_t ( ) [inline], [virtual]`

Do-nothing destructor.

© 2005-2021 HörTech gGmbH, Oldenburg

5.152.3 Member Function Documentation

5.152.3.1 get_local_address() `virtual const std::string& io_tcp_parser_t::get_local_address ( ) const [inline], [virtual]`

Read parser variable local_address, this is the address of the network interface that should listen for incoming connections.

Returns

A string containing the address of the local interface as it was set by the user.

5.152.3.2 get_local_port() `unsigned short io_tcp_parser_t::get_local_port ( ) const [virtual]`

Read parser variable local_port, this is the TCP port that should be used for incoming connections.

Returns

The local tcp port to listen on as it was chosen by the user. The port number is between MIN_TCP_PORT and MAX_TCP_PORT.

5.152.3.3 set_local_port() `void io_tcp_parser_t::set_local_port ( unsigned short port ) [virtual]`

Set parser variable local_port.

This is needed when it was set to 0 before: In this case, the OS chooses a free port for the TCP server socket, and the port that it chose has to be published to the user via the parser interface.

Parameters

port	The TCP port number that is currently used. In the range [MIN_TCP_PORT, MAX_TCP_PORT], excluding 0.

© 2005-2021 HörTech gGmbH, Oldenburg

Precondition

get_local_port() (p. 628) currently returns 0.

5.152.3.4 get_server_port_open() `bool io_tcp_parser_t::get_server_port_open ( ) const [virtual]`

Return the status of the server port as it is known to the parser.

Returns

false after initialization, or the value most recently set via

See also

set_server_port_open (p. 629).

5.152.3.5 set_server_port_open() `void io_tcp_parser_t::set_server_port_open ( bool open ) [virtual]`

Inform the parser of the current status of the server socket.

Parameters

open	Indicates wether the server socket has just been opened or closed.

Precondition

open may only have the value true if **get_server_port_open()** (p. 629) currently returns false.

Postcondition

See also

get_server_port_open (p. 629) returns the **value** (p. 49) of open.

© 2005-2021 HörTech gGmbH, Oldenburg

5.152.3.6 get_connected() `bool io_tcp_parser_t::get_connected ( ) const [virtual]`

Return the parser's knowledge concerning wether there currently exists an established sound data TCP connection or not.

Returns

false after initialization, or the value most recently set via

See also

set_connected (p. 630).

5.152.3.7 set_connected() `void io_tcp_parser_t::set_connected ( bool connected ) [virtual]`

Inform the parser about the existance of a sound data connection.

Parameters

connected	Indicates wether there currently is a connection or not.

Precondition

connected must not have the same value that is currently returned by

See also

get_connected (p. 629).

Postcondition

See also

get_connected (p. 629) returns the **value** (p. 49) of open.

© 2005-2021 HörTech gGmbH, Oldenburg

5.152.3.8 set_new_peer() `void io_tcp_parser_t::set_new_peer ( unsigned short port, const std::string & host ) [virtual]`

Set parser monitor variables peer_port and peer_address, and calls set_connected(true).

This method should be called when a new connection is established.

Parameters

port	The TCP port number used by the peer.
host	The Internet host where the peer is located.

Precondition

See also

get_connected (p. 629) currently returns false.

Postcondition

See also

get_connected (p. 629) returns true.

5.152.3.9 debug() `virtual void io_tcp_parser_t::debug ( const std::string & message ) [inline], [virtual]`

5.152.4 Member Data Documentation

5.152.4.1 local_address `MHAParser::string_t io_tcp_parser_t::local_address [private]`

Lets the user set the local network interface to listen on.

© 2005-2021 HörTech gGmbH, Oldenburg

5.152.4.2 local_port `MHAParser::int_t io_tcp_parser_t::local_port [private]`

Lets the user choose the local tcp port to listen on.

5.152.4.3 server_port_open `MHAParser::int_mon_t io_tcp_parser_t::server_port_open [private]`

Indicates wether the TCP server socket is currently open.

5.152.4.4 connected `MHAParser::int_mon_t io_tcp_parser_t::connected [private]`

Indicator if there currently is a sound data connection over TCP.

5.152.4.5 peer_address `MHAParser::string_mon_t io_tcp_parser_t::peer_address [private]`

Display the ip address of the currently connected sound data client.

5.152.4.6 peer_port `MHAParser::int_mon_t io_tcp_parser_t::peer_port [private]`

Display the tcp port used by the current sound data client.

5.152.4.7 debug_filename `MHAParser::string_t io_tcp_parser_t::debug_filename [private]`

filename to write debugging info to (if non-empty)

© 2005-2021 HörTech gGmbH, Oldenburg

5.152.4.8 debug_file `FILE* io_tcp_parser_t::debug_file [private]`

file handle to write debugging info to

The documentation for this class was generated from the following file:

- **MHAIOTCP.cpp**

5.153 io_tcp_sound_t Class Reference

Sound data handling of io tcp library.

Classes

- union **float_union**
  This union helps in conversion of floats from host byte order to network byte order and back again.

Public Member Functions

- **io_tcp_sound_t** (int **fragsize**, float **samplerate**)
  Initialize sound data handling.
- virtual ~**io_tcp_sound_t** ()
  Do-nothing destructor.
- virtual void **prepare** (int **num_inchannels**, int **num_outchannels**)
  Called during prepare, sets number of audio channels and allocates sound data storage.
- virtual void **release** ()
  Called during release.
- virtual int **chunkbytes_in** () const
  Number of bytes that constitute one input sound chunk.
- virtual std::string **header** () const
  Create the tcp sound header lines.
- virtual **mha_wave_t** * **ntoh** (const std::string &data)
  *Copy data received from tcp into **mha_wave_t** (p. 840) structure.*
- virtual std::string **hton** (const **mha_wave_t** *s_out)
  Copy sound data from the output sound structure to a string.

Static Private Member Functions

- static void **check_sound_data_type** ()
  Check if mha_real_t is a usable 32-bit floating point type.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- int **fragsize**
  Number of sound samples in each channel expected and returned from processing callback.
- float **samplerate**
  Sampling rate.
- int **num_inchannels**
  Number of input channels.
- int **num_outchannels**
- **MHASignal::waveform_t** * **s_in**
  Storage for input signal.

5.153.1 Detailed Description

Sound data handling of io tcp library.

5.153.2 Constructor & Destructor Documentation

5.153.2.1 io_tcp_sound_t()

```
io_tcp_sound_t::io_tcp_sound_t (
    int fragsize,
    float samplerate )
```

Initialize sound data handling.

Checks sound data type by calling

See also

check_sound_data_type (p. 635).

Parameters

fragsize	Number of sound samples in each channel expected and returned from processing callback.
samplerate	Number of samples per second in each channel.

© 2005-2021 HörTech gGmbH, Oldenburg

5.153.2.2 ~io_tcp_sound_t() `virtual io_tcp_sound_t::~io_tcp_sound_t ( ) [inline], [virtual]`

Do-nothing destructor.

5.153.3 Member Function Documentation

5.153.3.1 check_sound_data_type() `void io_tcp_sound_t::check_sound_data_type ( ) [static], [private]`

Check if mha_real_t is a usable 32-bit floating point type.

Exceptions

MHA_Error *(p. 764)*	if mha_real_t is not compatible to 32-bit float.

5.153.3.2 prepare() `void io_tcp_sound_t::prepare ( int num_inchannels, int num_outchannels ) [virtual]`

Called during prepare, sets number of audio channels and allocates sound data storage.

Parameters

num_inchannels	Number of input audio channels.
num_outchannels	Number of output audio channels.

5.153.3.3 release() `void io_tcp_sound_t::release ( ) [virtual]`

Called during release.

Deletes sound data storage.

© 2005-2021 HörTech gGmbH, Oldenburg

5.153.3.4 chunkbytes_in() `int io_tcp_sound_t::chunkbytes_in ( ) const [virtual]`

Number of bytes that constitute one input sound chunk.

Returns

Number of bytes to read from TCP connection before invoking signal processing.

5.153.3.5 header() `std::string io_tcp_sound_t::header ( ) const [virtual]`

Create the tcp sound header lines.

5.153.3.6 ntoh() `mha_wave_t * io_tcp_sound_t::ntoh ( const std::string & data ) [virtual]`

Copy data received from tcp into **mha_wave_t** (p. 840) structure.

Doing network-to-host byte order swapping in the process.

Parameters

data	One chunk (

See also

chunkbytes_in (p. 635)) of sound data to process.

Returns

Pointer to the sound data storage.

5.153.3.7 hton() `std::string io_tcp_sound_t::hton ( const mha_wave_t * s_out ) [virtual]`

Copy sound data from the output sound structure to a string.

Doing host-to-network byte order swapping while at it.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s_out	Pointer to the storage of the sound to put out.

Returns

The sound data in network byte order.

5.153.4 Member Data Documentation

5.153.4.1 fragsize `int io_tcp_sound_t::fragsize [private]`

Number of sound samples in each channel expected and returned from processing callback.

5.153.4.2 samplerate `float io_tcp_sound_t::samplerate [private]`

Sampling rate.

Number of samples per second in each channel.

5.153.4.3 num_inchannels `int io_tcp_sound_t::num_inchannels [private]`

Number of input channels.

Number of channels expected from and returned by signal processing callback.

5.153.4.4 num_outchannels `int io_tcp_sound_t::num_outchannels [private]`

5.153.4.5 s_in `MHASignal::waveform_t* io_tcp_sound_t::s_in [private]`

Storage for input signal.

The documentation for this class was generated from the following file:

- **MHAIOTCP.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.154 io_tcp_sound_t::float_union Union Reference

This union helps in conversion of floats from host byte order to network byte order and back again.

Public Attributes

- float **f**
- unsigned int **i**
- char **c** [4]

5.154.1 Detailed Description

This union helps in conversion of floats from host byte order to network byte order and back again.

5.154.2 Member Data Documentation

5.154.2.1 f `float io_tcp_sound_t::float_union::f`

5.154.2.2 i `unsigned int io_tcp_sound_t::float_union::i`

5.154.2.3 c `char io_tcp_sound_t::float_union::c[4]`

The documentation for this union was generated from the following file:

- **MHAIOTCP.cpp**

5.155 io_tcp_t Class Reference

The tcp sound io library.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **io_tcp_t** (int fragsize, float samplerate, **IOProcessEvent_t** proc_event, void *proc_handle, **IOStartedEvent_t** start_event, void *start_handle, **IOStoppedEvent_t** stop_event, void *stop_handle)
- void **prepare** (int num_inchannels, int num_outchannels)

  Allocate server socket and start thread waiting for sound data exchange.
- void **start** ()

  Call frameworks start callback if there is a sound data connection at the moment.
- void **stop** ()

  Close the current connection if there is one.
- void **release** ()

  Close the current connection and close the server socket.
- virtual void **accept_loop** ()

  IO thread executes this method.
- virtual void **connection_loop** (**MHA_TCP::Connection** *c)

  IO thread executes this method for each connection.
- virtual void **parse** (const char *cmd, char *retval, unsigned int len)

  Parser interface.
- virtual ~**io_tcp_t** ()

Private Attributes

- **io_tcp_parser_t parser**
- **io_tcp_sound_t sound**
- **io_tcp_fwcb_t fwcb**
- **MHA_TCP::Server * server**
- **MHA_TCP::Thread * thread**
- **MHA_TCP::Async_Notify notify_start**
- **MHA_TCP::Async_Notify notify_stop**
- **MHA_TCP::Async_Notify notify_release**

5.155.1 Detailed Description

The tcp sound io library.

5.155.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.155.2.1 io_tcp_t()

```
io_tcp_t::io_tcp_t (
    int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle )
```

5.155.2.2 ~io_tcp_t()

```
virtual io_tcp_t::~io_tcp_t ( ) [inline], [virtual]
```

5.155.3 Member Function Documentation

5.155.3.1 prepare()

```
void io_tcp_t::prepare (
    int num_inchannels,
    int num_outchannels )
```

Allocate server socket and start thread waiting for sound data exchange.

prepare opens the tcp server socket and starts the io thread that listens for audio data on the tcp socket after doing some sanity checks

5.155.3.2 start()

```
void io_tcp_t::start ( )
```

Call frameworks start callback if there is a sound data connection at the moment.

5.155.3.3 stop()

```
void io_tcp_t::stop ( )
```

Close the current connection if there is one.

stop IO thread

© 2005-2021 HörTech gGmbH, Oldenburg

5.155.3.4 release() `void io_tcp_t::release ( )`

Close the current connection and close the server socket.

Stop IO thread and close server socket.

5.155.3.5 accept_loop() `void io_tcp_t::accept_loop ( ) [virtual]`

IO thread executes this method.

5.155.3.6 connection_loop() `void io_tcp_t::connection_loop ( MHA_TCP::Connection * c ) [virtual]`

IO thread executes this method for each connection.

Parameters

c	pointer to connection. connection_loop deletes connection before exiting.

5.155.3.7 parse() `virtual void io_tcp_t::parse ( const char * cmd, char * retval, unsigned int len ) [inline], [virtual]`

Parser interface.

5.155.4 Member Data Documentation

5.155.4.1 parser `io_tcp_parser_t io_tcp_t::parser [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.155.4.2 sound `io_tcp_sound_t io_tcp_t::sound [private]`

5.155.4.3 fwcb `io_tcp_fwcb_t io_tcp_t::fwcb [private]`

5.155.4.4 server `MHA_TCP::Server* io_tcp_t::server [private]`

5.155.4.5 thread `MHA_TCP::Thread* io_tcp_t::thread [private]`

5.155.4.6 notify_start `MHA_TCP::Async_Notify io_tcp_t::notify_start [private]`

5.155.4.7 notify_stop `MHA_TCP::Async_Notify io_tcp_t::notify_stop [private]`

5.155.4.8 notify_release `MHA_TCP::Async_Notify io_tcp_t::notify_release [private]`

The documentation for this class was generated from the following file:

- **MHAIOTCP.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.156 io_wrapper Class Reference

Inheritance diagram for io_wrapper:

Public Member Functions

- **io_wrapper** (**algo_comm_t** iac, const std::string &libname)
- virtual ~**io_wrapper** ()=default
- virtual std::vector< std::string > **get_categories** ()
- virtual std::string **parse** (const std::string &str)
- virtual bool **has_parser** ()
- virtual std::string **get_documentation** ()
- virtual bool **has_process** (**mha_domain_t** in, **mha_domain_t** out)

Additional Inherited Members

5.156.1 Constructor & Destructor Documentation

5.156.1.1 io_wrapper()

```
io_wrapper::io_wrapper (
    algo_comm_t iac,
    const std::string & libname ) [inline]
```

5.156.1.2 ~io_wrapper() `virtual io_wrapper::~io_wrapper ( ) [virtual], [default]`

5.156.2 Member Function Documentation

5.156.2.1 get_categories() `virtual std::vector<std::string> io_wrapper::get_categories ( ) [inline], [virtual]`

Implements **plug_wrapperl** (p. 1353).

5.156.2.2 parse() `virtual std::string io_wrapper::parse ( const std::string & str ) [inline], [virtual]`

Implements **plug_wrapperl** (p. 1353).

5.156.2.3 has_parser() `virtual bool io_wrapper::has_parser ( ) [inline], [virtual]`

Implements **plug_wrapperl** (p. 1353).

5.156.2.4 get_documentation() `virtual std::string io_wrapper::get_documentation ( ) [inline], [virtual]`

Implements **plug_wrapperl** (p. 1353).

5.156.2.5 has_process() `virtual bool io_wrapper::has_process ( mha_domain_t in, mha_domain_t out ) [inline], [virtual]`

Implements **plug_wrapperl** (p. 1354).

The documentation for this class was generated from the following file:

- **generatemhaplugindoc.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.157 latex_doc_t Class Reference

Class to access the information stored in the plugin source code's MHAPLUGIN_DOCUME↩ NTATION macro.

Public Member Functions

- **latex_doc_t** (const std::string & **plugname**, const std::string & **plugin_macro**)
  Constructor loads the plugin into this process.
- std::string **get_latex_doc** ()
  This method accesses the compiled-in contents of the MHAPLUGIN_DOCUMENTATION macro and the exported interface functions of the loaded plugin to produce latex docuementation for the plugin.
- std::string **get_main_category** () const
- std::vector< std::string > **get_categories** () const

Private Member Functions

- std::string **strdom** (**mha_domain_t** d) const
- std::string **get_ac** (**MHAKernel::algo_comm_class_t** & **ac**, std::string txt) const
- std::string **parsername** (std::string s) const
- std::string **get_parser_var** (**MHAParser::base_t** *p, std::string name) const
- std::string **get_parser_tab** (**MHAParser::base_t** *p, const std::string &prefix, const std::string &latex_macro) const

Private Attributes

- const std::string **plugname**
- const std::string **latex_plugname**
- **MHAKernel::algo_comm_class_t ac**
- std::unique_ptr< **plug_wrapperl** > **loader**
- const std::string **plugin_macro**

5.157.1 Detailed Description

Class to access the information stored in the plugin source code's MHAPLUGIN_DOCUME↩ NTATION macro.

5.157.2 Constructor & Destructor Documentation

5.157.2.1 latex_doc_t()

```
latex_doc_t::latex_doc_t (
    const std::string & plugname,
    const std::string & plugin_macro )
```

Constructor loads the plugin into this process.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

plugname	Name of the MHA plugin to process
plugin_macro	name of the LaTeX section macro that documents a single plugin (e.g. "section", "subsection", "subsubsection", ...)

5.157.3 Member Function Documentation

5.157.3.1 get_latex_doc() `std::string latex_doc_t::get_latex_doc ( )`

This method accesses the compiled-in contents of the MHAPLUGIN_DOCUMENTATION macro and the exported interface functions of the loaded plugin to produce latex docuementation for the plugin.

It tentatively prepares the plugin for processing and checks the AC variables registered by the plugin.

Returns

the complete latex documentation for this plugin

5.157.3.2 get_main_category() `std::string latex_doc_t::get_main_category ( ) const`

Returns

the first word of the categories string in the MHAPLUGIN_DOCUMENTATION macro

5.157.3.3 get_categories() `std::vector< std::string > latex_doc_t::get_categories ( ) const`

Returns

a vector of all words in the categories string in the MHAPLUGIN_DOCUMENTATION macro

© 2005-2021 HörTech gGmbH, Oldenburg

5.157.3.4 strdom()

```
std::string latex_doc_t::strdom (
    mha_domain_t d ) const [private]
```

5.157.3.5 get_ac()

```
std::string latex_doc_t::get_ac (
    MHAKernel::algo_comm_class_t & ac,
    std::string txt ) const [private]
```

5.157.3.6 parsername()

```
std::string latex_doc_t::parsername (
    std::string s ) const [private]
```

5.157.3.7 get_parser_var()

```
std::string latex_doc_t::get_parser_var (
    MHAParser::base_t * p,
    std::string name ) const [private]
```

5.157.3.8 get_parser_tab()

```
std::string latex_doc_t::get_parser_tab (
    MHAParser::base_t * p,
    const std::string & prefix,
    const std::string & latex_macro ) const [private]
```

5.157.4 Member Data Documentation

5.157.4.1 plugname

```
const std::string latex_doc_t::plugname [private]
```

5.157.4.2 latex_plugname

```
const std::string latex_doc_t::latex_plugname [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.157.4.3 ac `MHAKernel::algo_comm_class_t latex_doc_t::ac [private]`

5.157.4.4 loader `std::unique_ptr< plug_wrapperI> latex_doc_t::loader [private]`

5.157.4.5 plugin_macro `const std::string latex_doc_t::plugin_macro [private]`

The documentation for this class was generated from the following file:

- **generatemhaplugindoc.cpp**

5.158 level_matching::channel_pair Class Reference

Public Member Functions

- **channel_pair** (const std::pair< int, int > &idx_, **mha_real_t** filter_rate_, **mha_real_t** mismatch_time_constant_)
  Channel pair ctor.
- **channel_pair** (int idx1_, int idx2_, **mha_real_t** filter_rate_, **mha_real_t** mismatch_↩time_constant_)
  Channel pair ctor.
- **mha_real_t update_mismatch** (const **mha_wave_t** &signal_)
  Calculates the filtered rms level mismatch.
- **mha_real_t update_mismatch** (const **mha_spec_t** &signal_, unsigned fftlen_)
  Calculates the filtered rms level mismatch.
- **mha_real_t get_mismatch** () const
  Get last filter result.
- const std::pair< int, int > & **get_idx** () const

Private Attributes

- std::pair< int, int > **idx**
  Indices of channels.
- **MHAFilter::o1flt_lowpass_t mismatch**
  Low-pass filtered level mismatch.

5.158.1 Constructor & Destructor Documentation

5.158.1.1 channel_pair() [1/2]

```
level_matching::channel_pair::channel_pair (
    const std::pair< int, int > & idx_,
    mha_real_t filter_rate_,
    mha_real_t mismatch_time_constant_ )
```

Channel pair ctor.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

*idx_*	Pair of channel indices
*filter_rate_*	Filter rate of low pass filter
*mismatch_time_↩ constant_*	Time constant of low pass filter

5.158.1.2 channel_pair() [2/2]

```
level_matching::channel_pair::channel_pair (
    int idx1_,
    int idx2_,
    mha_real_t filter_rate_,
    mha_real_t mismatch_time_constant_ )
```

Channel pair ctor.

Parameters

idx1	First channel index. Used as reference
idx2	Second channel index
*filter_rate_*	Filter rate of low pass filter
*mismatch_time_↩ constant_*	Time constant of low pass filter

5.158.2 Member Function Documentation

5.158.2.1 update_mismatch() [1/2]

```
mha_real_t level_matching::channel_pair::update↩
_mismatch (
    const mha_wave_t & signal_ )
```

Calculates the filtered rms level mismatch.

Parameters

signal	Input signal

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

Low-pass filtered mismatch ratio

5.158.2.2 update_mismatch() [2/2]

```
mha_real_t level_matching::channel_pair::update_mismatch (
    const mha_spec_t & signal_,
    unsigned fftlen_ )
```

Calculates the filtered rms level mismatch.

Parameters

signal	Input signal

Returns

Low-pass filtered mismatch ratio

5.158.2.3 get_mismatch()

```
mha_real_t level_matching::channel_pair::get_mismatch ( ) const
```

Get last filter result.

Returns

Last filter result

5.158.2.4 get_idx()

```
const std::pair<int,int>& level_matching::channel_pair::get_idx ( ) const [inline]
```

5.158.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.158.3.1 idx `std::pair<int,int> level_matching::channel_pair::idx [private]`

Indices of channels.

5.158.3.2 mismatch `MHAFilter::o1flt_lowpass_t level_matching::channel_pair↩ ::mismatch [private]`

Low-pass filtered level mismatch.

The documentation for this class was generated from the following files:

- **level_matching.hh**
- **level_matching.cpp**

5.159 level_matching::level_matching_config_t Class Reference

Public Member Functions

- **level_matching_config_t** (const **mhaconfig_t** &cfg_, const std::vector< std::vector< int >> &pairs_, **mha_real_t** signal_lp_fpass_, **mha_real_t** signal_fstop_, unsigned signal_lp_order_, **mha_real_t** level_tau_, **mha_real_t** range_)

  RT-config c'tor.
- ~**level_matching_config_t** ()=default
- **mha_wave_t** * **process** (**mha_wave_t** *)
- **mha_spec_t** * **process** (**mha_spec_t** *)

Private Attributes

- **MHASignal::waveform_t tmp**

  Temporary storage for input signal to use waveform_t helper functions.
- std::vector< **channel_pair** > **pairings**

  Channel pairings.
- **MHAFilter::iir_filter_state_t lp**

  Nth-order low pass filter used to exclude high frequencies from level matching.
- **mha_real_t range**

  Maximum matching range. Maximum adjustment done.
- unsigned **fftlen**

  fftlen in spectral domain

© 2005-2021 HörTech gGmbH, Oldenburg

5.159.1 Constructor & Destructor Documentation

5.159.1.1 level_matching_config_t()

```
level_matching::level_matching_config_t::level↩
_matching_config_t (
            const  mhaconfig_t & cfg_,
            const std::vector< std::vector< int >> & pairs_,
             mha_real_t signal_lp_fpass_,
             mha_real_t signal_lp_fstop_,
            unsigned signal_lp_order_,
             mha_real_t level_tau_,
             mha_real_t range_ )
```

RT-config c'tor.

Parameters

cfg↩_	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Precondition

Size of param pairs must be two

Parameters

*pairs_*	Matrix containing the channel indices of the microphone pairs. Two entries per row.
signal_tau	Time constant of the signal low pass filter in seconds.
level_tau	Time constant of the level mismatch averaging filter in seconds.

5.159.1.2 ∼level_matching_config_t()

```
level_matching::level_matching_config_t↩
::∼level_matching_config_t ( ) [default]
```

5.159.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.159.2.1 process() [1/2]

```
mha_wave_t * level_matching::level_matching_config_t↩
::process (
            mha_wave_t * s )
```

5.159.2.2 process() [2/2]

```
mha_spec_t * level_matching::level_matching_config_t↩
::process (
            mha_spec_t * s )
```

5.159.3 Member Data Documentation

5.159.3.1 tmp

```
MHASignal::waveform_t level_matching::level_matching_config_t::tmp
[private]
```

Temporary storage for input signal to use waveform_t helper functions.

5.159.3.2 pairings

```
std::vector< channel_pair> level_matching::level_matching_↩
config_t::pairings [private]
```

Channel pairings.

5.159.3.3 lp

```
MHAFilter::iir_filter_state_t level_matching::level_matching_config↩
_t::lp [private]
```

Nth-order low pass filter used to exclude high frequencies from level matching.

5.159.3.4 range

```
mha_real_t level_matching::level_matching_config_t::range [private]
```

Maximum matching range. Maximum adjustment done.

© 2005-2021 HörTech gGmbH, Oldenburg

5.159.3.5 fftlen `unsigned level_matching::level_matching_config_t::fftlen [private]`

fftlen in spectral domain

The documentation for this class was generated from the following files:

- **level_matching.hh**
- **level_matching.cpp**

5.160 level_matching::level_matching_t Class Reference

Inheritance diagram for level_matching::level_matching_t:

Public Member Functions

- **level_matching_t** (**algo_comm_t** iac, const std::string &configured_name)
  Plugin interface ctor.
- ~**level_matching_t** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
  Processing callback.
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
  Processing callback.
- void **prepare** (**mhaconfig_t** &)
  Plugin preparation callback.
- void **release** (void)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAParser::mint_t channels** ={"channels","[[0 1]]"}
- **MHAParser::float_t range** ={"Maximum matching range in dB","4","[0,["}
- **MHAParser::float_t lp_signal_fpass** ={"Upper edge of the lp pass band for the signal in Hz","4000","[0,["}
- **MHAParser::float_t lp_signal_fstop** ={"Stop band lower edge frequency for the signal in Hz","8000","[0,["}
- **MHAParser::float_t lp_signal_order** ={"Signal lp order","24","[1,["}
- **MHAParser::float_t lp_level_tau** ={"Low pass time constant for the mismatch in s","1","[0,["}
- **MHAEvents::patchbay_t**< **level_matching_t** > **patchbay**

Additional Inherited Members

5.160.1 Constructor & Destructor Documentation

5.160.1.1 level_matching_t()

`level_matching::level_matching_t::level_matching_t ( algo_comm_t iac, const std::string & configured_name )`

Plugin interface ctor.

Parameters

ac	

5.160.1.2 ~level_matching_t()

`level_matching::level_matching_t::~level_matching_t ( )`

5.160.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.160.2.1 process() [1/2] `mha_wave_t * level_matching::level_matching_t::process ( mha_wave_t * signal_ )`

Processing callback.

Parameters

signal	Input signal

5.160.2.2 process() [2/2] `mha_spec_t * level_matching::level_matching_t::process ( mha_spec_t * signal_ )`

Processing callback.

Parameters

signal	Input signal

5.160.2.3 prepare() `void level_matching::level_matching_t::prepare ( mhaconfig_t & ) [virtual]`

Plugin preparation callback.

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< level_matching_config_t >** (p. 1153).

5.160.2.4 release() `void level_matching::level_matching_t::release ( void ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t< level_matching_config_t >** (p. 1154).

© 2005-2021 HörTech gGmbH, Oldenburg

5.160.2.5 update_cfg() `void level_matching::level_matching_t::update_cfg ( ) [private]`

5.160.3 Member Data Documentation

5.160.3.1 channels `MHAParser::mint_t level_matching::level_matching_t::channels ={"channels","[[0 1]]"} [private]`

5.160.3.2 range `MHAParser::float_t level_matching::level_matching_t::range ={"Maximum matching range in dB","4","[0,["} [private]`

5.160.3.3 lp_signal_fpass `MHAParser::float_t level_matching::level_matching_t↩ ::lp_signal_fpass ={"Upper edge of the lp pass band for the signal in Hz","4000","[0,["} [private]`

5.160.3.4 lp_signal_fstop `MHAParser::float_t level_matching::level_matching_t↩ ::lp_signal_fstop ={"Stop band lower edge frequency for the signal in Hz","8000","[0,["} [private]`

5.160.3.5 lp_signal_order `MHAParser::float_t level_matching::level_matching_t↩ ::lp_signal_order ={"Signal lp order","24","[1,["} [private]`

5.160.3.6 lp_level_tau `MHAParser::float_t level_matching::level_matching_t::lp_↩ level_tau ={"Low pass time constant for the mismatch in s","1","[0,["} [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.160.3.7 patchbay `MHAEvents::patchbay_t< level_matching_t> level_matching↩ ::level_matching_t::patchbay [private]`

The documentation for this class was generated from the following files:

- **level_matching.hh**
- **level_matching.cpp**

5.161 levelmeter_t Class Reference

Inheritance diagram for levelmeter_t:

Public Member Functions

- **levelmeter_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update_tau** ()
- void **query_rms** ()
- void **query_peak** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::float_t tau**
- **MHAParser::kw_t mode**
- **MHAParser::vfloat_mon_t rms**
- **MHAParser::vfloat_mon_t peak**
- **MHAEvents::patchbay_t< levelmeter_t > patchbay**

Additional Inherited Members

5.161.1 Constructor & Destructor Documentation

5.161.1.1 levelmeter_t()

```
levelmeter_t::levelmeter_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.161.2 Member Function Documentation

5.161.2.1 process()

```
mha_wave_t * levelmeter_t::process (
    mha_wave_t * s )
```

5.161.2.2 prepare()

```
void levelmeter_t::prepare (
    mhaconfig_t & cf ) [virtual]
```

Implements **MHAPlugin::plugin_t< MHASignal::async_rmslevel_t >** (p. 1153).

5.161.2.3 update_tau()

```
void levelmeter_t::update_tau ( ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.161.2.4 query_rms() `void levelmeter_t::query_rms ( ) [private]`

5.161.2.5 query_peak() `void levelmeter_t::query_peak ( ) [private]`

5.161.3 Member Data Documentation

5.161.3.1 tau `MHAParser::float_t levelmeter_t::tau [private]`

5.161.3.2 mode `MHAParser::kw_t levelmeter_t::mode [private]`

5.161.3.3 rms `MHAParser::vfloat_mon_t levelmeter_t::rms [private]`

5.161.3.4 peak `MHAParser::vfloat_mon_t levelmeter_t::peak [private]`

5.161.3.5 patchbay `MHAEvents::patchbay_t< levelmeter_t> levelmeter_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **levelmeter.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.162 lpc Class Reference

Inheritance diagram for lpc:

Public Member Functions

- **lpc** (**algo_comm_t** iac, const std::string &configured_name)

  Constructs our plugin.
- ~**lpc** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

  Checks for the most recent configuration and defers processing to it.
- void **prepare** (**mhaconfig_t** &)

  Plugin preparation.
- void **release** (void)

Private Member Functions

- void **update_cfg** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- std::string **algo_name**
- **MHAParser::int_t lpc_order**
- **MHAParser::int_t lpc_buffer_size**
- **MHAParser::bool_t shift**
- **MHAParser::int_t comp_each_iter**
- **MHAParser::bool_t norm**
- **MHAEvents::patchbay_t< lpc > patchbay**

Additional Inherited Members

5.162.1 Constructor & Destructor Documentation

5.162.1.1 lpc()

```
lpc::lpc (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructs our plugin.

5.162.1.2 ~lpc()

```
lpc::~lpc ( )
```

5.162.2 Member Function Documentation

5.162.2.1 process()

```
mha_wave_t * lpc::process (
    mha_wave_t * signal )
```

Checks for the most recent configuration and defers processing to it.

5.162.2.2 prepare()

```
void lpc::prepare (
    mhaconfig_t & signal_info ) [virtual]
```

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< lpc_config >** (p. 1153).

5.162.2.3 release() `void lpc::release ( void ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t< lpc_config >** (p. 1154).

5.162.2.4 update_cfg() `void lpc::update_cfg ( ) [private]`

5.162.3 Member Data Documentation

5.162.3.1 algo_name `std::string lpc::algo_name [private]`

5.162.3.2 lpc_order `MHAParser::int_t lpc::lpc_order [private]`

5.162.3.3 lpc_buffer_size `MHAParser::int_t lpc::lpc_buffer_size [private]`

5.162.3.4 shift `MHAParser::bool_t lpc::shift [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.162.3.5 comp_each_iter `MHAParser::int_t lpc::comp_each_iter [private]`

5.162.3.6 norm `MHAParser::bool_t lpc::norm [private]`

5.162.3.7 patchbay `MHAEvents::patchbay_t< lpc> lpc::patchbay [private]`

The documentation for this class was generated from the following files:

- **lpc.h**
- **lpc.cpp**

5.163 lpc_bl_predictor Class Reference

Inheritance diagram for lpc_bl_predictor:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **lpc_bl_predictor** (**algo_comm_t** iac, const std::string &configured_name)
  Constructs our plugin.
- ∼**lpc_bl_predictor** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
  Checks for the most recent configuration and defers processing to it.
- void **prepare** (**mhaconfig_t** &)
  Plugin preparation.
- void **release** (void)

Public Attributes

- **MHAParser::int_t lpc_order**
- **MHAParser::string_t name_kappa**
- **MHAParser::string_t name_lpc_f**
- **MHAParser::string_t name_lpc_b**
- **MHAParser::string_t name_f**
- **MHAParser::string_t name_b**

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t**< **lpc_bl_predictor** > **patchbay**

Additional Inherited Members

5.163.1 Constructor & Destructor Documentation

5.163.1.1 lpc_bl_predictor()

```
lpc_bl_predictor::lpc_bl_predictor (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructs our plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

5.163.1.2 ~lpc_bl_predictor() `lpc_bl_predictor::~lpc_bl_predictor ( )`

5.163.2 Member Function Documentation

5.163.2.1 process() `mha_wave_t * lpc_bl_predictor::process ( mha_wave_t * signal )`

Checks for the most recent configuration and defers processing to it.

5.163.2.2 prepare() `void lpc_bl_predictor::prepare ( mhaconfig_t & signal_info ) [virtual]`

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< lpc_bl_predictor_config >** (p. 1153).

5.163.2.3 release() `void lpc_bl_predictor::release ( void ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t< lpc_bl_predictor_config >** (p. 1154).

5.163.2.4 update_cfg() `void lpc_bl_predictor::update_cfg ( ) [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.163.3 Member Data Documentation

5.163.3.1 lpc_order `MHAParser::int_t lpc_bl_predictor::lpc_order`

5.163.3.2 name_kappa `MHAParser::string_t lpc_bl_predictor::name_kappa`

5.163.3.3 name_lpc_f `MHAParser::string_t lpc_bl_predictor::name_lpc_f`

5.163.3.4 name_lpc_b `MHAParser::string_t lpc_bl_predictor::name_lpc_b`

5.163.3.5 name_f `MHAParser::string_t lpc_bl_predictor::name_f`

5.163.3.6 name_b `MHAParser::string_t lpc_bl_predictor::name_b`

5.163.3.7 patchbay `MHAEvents::patchbay_t< lpc_bl_predictor> lpc_bl_predictor↩ ::patchbay [private]`

The documentation for this class was generated from the following files:

- **lpc_bl_predictor.h**
- **lpc_bl_predictor.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.164 lpc_bl_predictor_config Class Reference

Public Member Functions

- **lpc_bl_predictor_config** (**algo_comm_t** &iac, const **mhaconfig_t** in_cfg, **lpc_bl_predictor** ∗_lpc)
- ∼**lpc_bl_predictor_config** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Private Attributes

- **algo_comm_t ac**
- **MHA_AC::waveform_t f_est**
- **MHA_AC::waveform_t b_est**
- **MHASignal::waveform_t forward**
- **MHASignal::waveform_t backward**
- int **lpc_order**
- std::string **name_km**
- std::string **name_f**
- std::string **name_b**
- **mha_wave_t km**
- **mha_wave_t s_f**
- **mha_wave_t s_b**

5.164.1 Constructor & Destructor Documentation

5.164.1.1 lpc_bl_predictor_config()

```
lpc_bl_predictor_config::lpc_bl_predictor_config (
            algo_comm_t & iac,
           const  mhaconfig_t in_cfg,
            lpc_bl_predictor * _lpc )
```

5.164.1.2 ∼lpc_bl_predictor_config()

```
lpc_bl_predictor_config::~lpc_bl_predictor_config ( )
```

5.164.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.164.2.1 process()

```
mha_wave_t * lpc_bl_predictor_config::process (
    mha_wave_t * wave )
```

5.164.3 Member Data Documentation

5.164.3.1 ac

```
algo_comm_t lpc_bl_predictor_config::ac [private]
```

5.164.3.2 f_est

```
MHA_AC::waveform_t lpc_bl_predictor_config::f_est [private]
```

5.164.3.3 b_est

```
MHA_AC::waveform_t lpc_bl_predictor_config::b_est [private]
```

5.164.3.4 forward

```
MHASignal::waveform_t lpc_bl_predictor_config::forward [private]
```

5.164.3.5 backward

```
MHASignal::waveform_t lpc_bl_predictor_config::backward [private]
```

5.164.3.6 lpc_order

```
int lpc_bl_predictor_config::lpc_order [private]
```

5.164.3.7 name_km

```
std::string lpc_bl_predictor_config::name_km [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.164.3.8 name_f `std::string lpc_bl_predictor_config::name_f [private]`

5.164.3.9 name_b `std::string lpc_bl_predictor_config::name_b [private]`

5.164.3.10 km `mha_wave_t lpc_bl_predictor_config::km [private]`

5.164.3.11 s_f `mha_wave_t lpc_bl_predictor_config::s_f [private]`

5.164.3.12 s_b `mha_wave_t lpc_bl_predictor_config::s_b [private]`

The documentation for this class was generated from the following files:

- **lpc_bl_predictor.h**
- **lpc_bl_predictor.cpp**

5.165 lpc_burglattice Class Reference

Inheritance diagram for lpc_burglattice:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **lpc_burglattice** (**algo_comm_t** iac, const std::string &configured_name)
  Constructs our plugin.
- ∼**lpc_burglattice** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
  Checks for the most recent configuration and defers processing to it.
- void **prepare** (**mhaconfig_t** &)
  Plugin preparation.
- void **release** (void)

Public Attributes

- **MHAParser::int_t lpc_order**
- **MHAParser::string_t name_kappa**
- **MHAParser::string_t name_f**
- **MHAParser::string_t name_b**
- **MHAParser::float_t lambda**

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t**< **lpc_burglattice** > **patchbay**

Additional Inherited Members

5.165.1 Constructor & Destructor Documentation

5.165.1.1 lpc_burglattice()

```
lpc_burglattice::lpc_burglattice (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructs our plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

5.165.1.2 ~lpc_burglattice() `lpc_burglattice::~lpc_burglattice ( )`

5.165.2 Member Function Documentation

5.165.2.1 process() `mha_wave_t * lpc_burglattice::process ( mha_wave_t * signal )`

Checks for the most recent configuration and defers processing to it.

5.165.2.2 prepare() `void lpc_burglattice::prepare ( mhaconfig_t & signal_info ) [virtual]`

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< lpc_burglattice_config >** (p. 1153).

5.165.2.3 release() `void lpc_burglattice::release ( void ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t< lpc_burglattice_config >** (p. 1154).

5.165.2.4 update_cfg() `void lpc_burglattice::update_cfg ( ) [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.165.3 Member Data Documentation

5.165.3.1 lpc_order `MHAParser::int_t lpc_burglattice::lpc_order`

5.165.3.2 name_kappa `MHAParser::string_t lpc_burglattice::name_kappa`

5.165.3.3 name_f `MHAParser::string_t lpc_burglattice::name_f`

5.165.3.4 name_b `MHAParser::string_t lpc_burglattice::name_b`

5.165.3.5 lambda `MHAParser::float_t lpc_burglattice::lambda`

5.165.3.6 patchbay `MHAEvents::patchbay_t< lpc_burglattice> lpc_burglattice↩ ::patchbay [private]`

The documentation for this class was generated from the following files:

- **lpc_burg-lattice.h**
- **lpc_burg-lattice.cpp**

5.166 lpc_burglattice_config Class Reference

Public Member Functions

- **lpc_burglattice_config** (**algo_comm_t** &iac, const **mhaconfig_t** in_cfg, **lpc_↩ burglattice** *_lpc)
- ~**lpc_burglattice_config** ()
- **mha_wave_t** * **process** (**mha_wave_t** *)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **algo_comm_t ac**
- **MHASignal::waveform_t forward**
- **MHASignal::waveform_t backward**
- **MHASignal::waveform_t kappa**
- **MHA_AC::waveform_t kappa_block**
- **MHASignal::waveform_t dm**
- **MHASignal::waveform_t nm**
- **mha_real_t lambda**
- int **lpc_order**
- std::string **name_f**
- std::string **name_b**
- **mha_wave_t s_f**
- **mha_wave_t s_b**

5.166.1 Constructor & Destructor Documentation

5.166.1.1 lpc_burglattice_config()

```
lpc_burglattice_config::lpc_burglattice_config (
    algo_comm_t & iac,
    const mhaconfig_t in_cfg,
    lpc_burglattice * _lpc )
```

5.166.1.2 ~lpc_burglattice_config()

```
lpc_burglattice_config::~lpc_burglattice_config
( )
```

5.166.2 Member Function Documentation

5.166.2.1 process()

```
mha_wave_t * lpc_burglattice_config::process (
    mha_wave_t * wave )
```

5.166.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.166.3.1 ac

`algo_comm_t` lpc_burglattice_config::ac [private]

5.166.3.2 forward

`MHASignal::waveform_t` lpc_burglattice_config::forward [private]

5.166.3.3 backward

`MHASignal::waveform_t` lpc_burglattice_config::backward [private]

5.166.3.4 kappa

`MHASignal::waveform_t` lpc_burglattice_config::kappa [private]

5.166.3.5 kappa_block

`MHA_AC::waveform_t` lpc_burglattice_config::kappa_block [private]

5.166.3.6 dm

`MHASignal::waveform_t` lpc_burglattice_config::dm [private]

5.166.3.7 nm

`MHASignal::waveform_t` lpc_burglattice_config::nm [private]

5.166.3.8 lambda

`mha_real_t` lpc_burglattice_config::lambda [private]

5.166.3.9 lpc_order

int lpc_burglattice_config::lpc_order [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.166.3.10 name_f `std::string lpc_burglattice_config::name_f [private]`

5.166.3.11 name_b `std::string lpc_burglattice_config::name_b [private]`

5.166.3.12 s_f `mha_wave_t lpc_burglattice_config::s_f [private]`

5.166.3.13 s_b `mha_wave_t lpc_burglattice_config::s_b [private]`

The documentation for this class was generated from the following files:

- **lpc_burg-lattice.h**
- **lpc_burg-lattice.cpp**

5.167 lpc_config Class Reference

Public Member Functions

- **lpc_config** (**algo_comm_t** &ac, const **mhaconfig_t** in_cfg, std::string &algo_name, unsigned int _order, unsigned int _lpc_buffer_size, bool _shift, unsigned int _comp_each_iter, bool _norm)
- ~**lpc_config** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **insert** ()

Private Attributes

- bool **norm**
- bool **shift**
- unsigned int **comp_each_iter**
- unsigned int **order**
- unsigned int **lpc_buffer_size**
- unsigned int **N**
- unsigned int **comp_iter**
- **mha_wave_t sample**
- std::vector< **mha_real_t** > **R**
- std::vector< **mha_real_t** > **A**
- **MHASignal::ringbuffer_t inwave**
- **MHA_AC::waveform_t lpc_out**
- **MHA_AC::waveform_t corr_out**

© 2005-2021 HörTech gGmbH, Oldenburg

5.167.1 Constructor & Destructor Documentation

5.167.1.1 lpc_config()

```
lpc_config::lpc_config (
    algo_comm_t & ac,
    const mhaconfig_t in_cfg,
    std::string & algo_name,
    unsigned int _order,
    unsigned int _lpc_buffer_size,
    bool _shift,
    unsigned int _comp_each_iter,
    bool _norm )
```

5.167.1.2 ~lpc_config()

```
lpc_config::~lpc_config ( )
```

5.167.2 Member Function Documentation

5.167.2.1 process()

```
mha_wave_t * lpc_config::process (
    mha_wave_t * wave )
```

5.167.2.2 insert()

```
void lpc_config::insert ( )
```

5.167.3 Member Data Documentation

5.167.3.1 norm

```
bool lpc_config::norm [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.167.3.2 shift `bool lpc_config::shift [private]`

5.167.3.3 comp_each_iter `unsigned int lpc_config::comp_each_iter [private]`

5.167.3.4 order `unsigned int lpc_config::order [private]`

5.167.3.5 lpc_buffer_size `unsigned int lpc_config::lpc_buffer_size [private]`

5.167.3.6 N `unsigned int lpc_config::N [private]`

5.167.3.7 comp_iter `unsigned int lpc_config::comp_iter [private]`

5.167.3.8 sample `mha_wave_t lpc_config::sample [private]`

5.167.3.9 R `std::vector< mha_real_t> lpc_config::R [private]`

5.167.3.10 A `std::vector< mha_real_t> lpc_config::A [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.167.3.11 inwave `MHASignal::ringbuffer_t lpc_config::inwave [private]`

5.167.3.12 lpc_out `MHA_AC::waveform_t lpc_config::lpc_out [private]`

5.167.3.13 corr_out `MHA_AC::waveform_t lpc_config::corr_out [private]`

The documentation for this class was generated from the following files:

- **lpc.h**
- **lpc.cpp**

5.168 lsl2ac::cfg_t Class Reference

Runtime configuration class of the **lsl2ac** (p. 96) plugin.

Public Member Functions

- **cfg_t** (const **algo_comm_t** &ac_, **overrun_behavior** overrun_, int bufsize_↩, int chunksize_, const std::vector< std::string > &streamnames_, int nchannels_, int nsamples_)

  *C'tor of **lsl2ac** (p. 96) run time configuration.*
- void **process** ()

Private Attributes

- std::map< std::string, std::unique_ptr< **save_var_t** > > **varlist**

  Maps variable name to unique ptr's of lsl to ac bridges.

5.168.1 Detailed Description

Runtime configuration class of the **lsl2ac** (p. 96) plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

5.168.2 Constructor & Destructor Documentation

5.168.2.1 cfg_t()

```
cfg_t::cfg_t (
const algo_comm_t & ac_,
 overrun_behavior overrun_,
int bufsize_,
int chunksize_,
const std::vector< std::string > & streamnames_,
int nchannels_,
int nsamples_ )
```

C'tor of **lsl2ac** (p. 96) run time configuration.

Parameters

*ac_*	AC space, data from LSL will be inserted as AC variables
*overrun_*	Overrun behavior
*bufsize_*	Buffer size of the LSL stream inlet
*chunksize_*	Chunk size of the LSL stream
*streamnames_*	Names of LSL streams to be subscribed to
*nchannels_*	Number of channels to expect in the the LSL streams. Zero means accept any.
*nsamples_*	Number of samples per channel in the AC variable. Zero means resize as needed.

5.168.3 Member Function Documentation

5.168.3.1 process()

```
void cfg_t::process ( )
```

5.168.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.168.4.1 varlist `std::map<std::string, std::unique_ptr< save_var_t> > lsl2ac↩ ::cfg_t::varlist [private]`

Maps variable name to unique ptr's of lsl to ac bridges.

The documentation for this class was generated from the following files:

- **lsl2ac.hh**
- **lsl2ac.cpp**

5.169 lsl2ac::lsl2ac_t Class Reference

Plugin class of **lsl2ac** (p. 96).

Inheritance diagram for lsl2ac::lsl2ac_t:

Public Member Functions

- **lsl2ac_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)

  *Prepare constructs the vector of bridge variables and locks the configuration, then calls **update()*** (p. 684).
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s)

© 2005-2021 HörTech gGmbH, Oldenburg

Processing fct for waveforms.

- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗s)

  Processing fct for spectra.
- void **process** ()

  Process function.
- void **release** ()

  Release fct.

Private Member Functions

- void **get_all_stream_names** ()

  Retrieves all stream names from LSL and fills them into available_streams.
- void **update** ()

  Construct new runtime configuration.
- void **setlock** (bool lock_)

  Convencience function lock/unlock all config variables.

Private Attributes

- **MHAParser::vstring_t streams** ={"List of LSL streams to be saved, empty for all.","[ ]"}

  Config variable for list of streams to be saved.
- **MHAParser::bool_t activate** ={"Receive from network?","yes"}

  Config variable for activation/deactivation of plugin.
- **MHAParser::kw_t overrun_behavior** ={"How to handle overrun","discard","[discard ignore]"}

  Config variable for overrun behavior.
- **MHAParser::int_t buffersize**

  Config variable for maximum buffer size of LSL.
- **MHAParser::int_t chunksize**

  Config variable for maximum chunk size of LSL.
- **MHAParser::int_t nchannels**

  Config variable for number of channels to expect from LSL stream.
- **MHAParser::int_t nsamples**

  Config variable for maximum chunk size of LSL.
- **MHAEvents::patchbay_t**< **lsl2ac_t** > **patchbay**

  Patchbay for configuration callbacks.
- **MHAParser::vstring_mon_t available_streams** ={"List of all available LSL **streams**"}

  Monitor variable containing all available streams.

Additional Inherited Members

5.169.1 Detailed Description

Plugin class of **lsl2ac** (p. 96).

© 2005-2021 HörTech gGmbH, Oldenburg

5.169.2 Constructor & Destructor Documentation

5.169.2.1 lsl2ac_t()

```
lsl2ac::lsl2ac_t::lsl2ac_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.169.3 Member Function Documentation

5.169.3.1 prepare()

```
void lsl2ac::lsl2ac_t::prepare (
    mhaconfig_t & ) [virtual]
```

Prepare constructs the vector of bridge variables and locks the configuration, then calls **update()** (p. 684).

Implements **MHAPlugin::plugin_t**< **cfg_t** > (p. 1153).

5.169.3.2 process() [1/3]

```
mha_wave_t* lsl2ac::lsl2ac_t::process (
    mha_wave_t * s ) [inline]
```

Processing fct for waveforms.

Calls **process(void)** (p. 683).

5.169.3.3 process() [2/3]

```
mha_spec_t* lsl2ac::lsl2ac_t::process (
    mha_spec_t * s ) [inline]
```

Processing fct for spectra.

Calls **process(void)** (p. 683).

5.169.3.4 process() [3/3]

```
void lsl2ac::lsl2ac_t::process ( )
```

Process function.

© 2005-2021 HörTech gGmbH, Oldenburg

5.169.3.5 release() `void lsl2ac::lsl2ac_t::release ( ) [virtual]`

Release fct.

Unlocks variable name list

Reimplemented from **MHAPlugin::plugin_t< cfg_t >** (p. 1154).

5.169.3.6 get_all_stream_names() `void lsl2ac::lsl2ac_t::get_all_stream_names ( ) [private]`

Retrieves all stream names from LSL and fills them into available_streams.

5.169.3.7 update() `void lsl2ac::lsl2ac_t::update ( ) [private]`

Construct new runtime configuration.

5.169.3.8 setlock() `void lsl2ac::lsl2ac_t::setlock ( bool lock_ ) [private]`

Convencience function lock/unlock all config variables.

Parameters

*lock_*	True to lock, False for unlock

5.169.4 Member Data Documentation

5.169.4.1 streams `MHAParser::vstring_t lsl2ac::lsl2ac_t::streams ={"List of LSL streams to be saved, empty for all.","[]"} [private]`

Config variable for list of streams to be saved.

© 2005-2021 HörTech gGmbH, Oldenburg

5.169.4.2 activate

`MHAParser::bool_t lsl2ac::lsl2ac_t::activate ={"Receive from network?","yes"} [private]`

Config variable for activation/deactivation of plugin.

5.169.4.3 overrun_behavior

`MHAParser::kw_t lsl2ac::lsl2ac_t::overrun_behavior ={"How to handle overrun","discard","[discard ignore]"} [private]`

Config variable for overrun behavior.

5.169.4.4 buffersize

`MHAParser::int_t lsl2ac::lsl2ac_t::buffersize [private]`

Config variable for maximum buffer size of LSL.

5.169.4.5 chunksize

`MHAParser::int_t lsl2ac::lsl2ac_t::chunksize [private]`

Config variable for maximum chunk size of LSL.

5.169.4.6 nchannels

`MHAParser::int_t lsl2ac::lsl2ac_t::nchannels [private]`

Config variable for number of channels to expect from LSL stream.

5.169.4.7 nsamples

`MHAParser::int_t lsl2ac::lsl2ac_t::nsamples [private]`

Config variable for maximum chunk size of LSL.

© 2005-2021 HörTech gGmbH, Oldenburg

5.169.4.8 patchbay `MHAEvents::patchbay_t< lsl2ac_t> lsl2ac::lsl2ac_t::patchbay [private]`

Patchbay for configuration callbacks.

5.169.4.9 available_streams `MHAParser::vstring_mon_t lsl2ac::lsl2ac_t::available↩_streams ={"List of all available LSL streams"} [private]`

Monitor variable containing all available streams.

The documentation for this class was generated from the following files:

- **lsl2ac.hh**
- **lsl2ac.cpp**

5.170 lsl2ac::save_var_t Class Reference

LSL to AC bridge variable.

Public Member Functions

- **save_var_t** (const **save_var_t** &)=delete
- **save_var_t** (**save_var_t** &&)=delete
- **save_var_t** & **operator=** (const **save_var_t** &)=delete
- **save_var_t** & **operator=** (**save_var_t** &&)=delete
- **save_var_t** (const lsl::stream_info &info_, const **algo_comm_t** &ac_, **overrun_↩behavior** ob_, int buffersize_, int chunksize_, int nchannels_, int nsamples_)

  C'tor of lsl to ac bridge.
- ~**save_var_t** ()=default
- lsl::stream_info **info** ()

  Get stream info object from stream inlet.
- void **receive_frame** ()

  Receive a samples from lsl and copy to AC space.

Private Member Functions

- void **pull_samples_ignore** ()

  Pull new samples, ignore overrun.
- void **pull_samples_discard** ()

  Pull new samples as long as there are samples ready for pickup in the LSL buffers.
- void **get_time_correction** ()

  Refresh time correction value every 5s.
- void **insert_vars** ()

  Insert stream value, time stamp and time offset into ac space.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- lsl::stream_inlet **stream**
  LSL stream outlet.
- std::vector< **mha_real_t** > **buf**
  Data buffer of the ac variable.
- std::vector< double > **ts_buf**
  Timestamp buffer.
- const **algo_comm_t** & **ac**
  Handle to AC space.
- **comm_var_t cv**
  Timeseries AC variable.
- **comm_var_t ts**
  Timestamp AC variable.
- double **tc** =0.0
  Current time correction.
- std::string **ts_name**
  Timestamp AC variable name.
- std::string **tc_name**
  Time correction AC variable name.
- std::string **new_name**
  Number of new samples AC variable name.
- std::chrono::time_point< std::chrono::steady_clock > **tic**
  time point of last time correction pull
- bool **skip** =false
  Should the variable be skipped in future process calls? Only true when error occured.
- **overrun_behavior ob**
  Behavior on stream overrun.
- const std::string **name**
  Name of stream.
- int32_t **nchannels**
  Number of channels.
- std::size_t **nsamples**
  Number of samples per channel in the AC variable.
- std::size_t **chunksize**
  Maximal chunk size of lsl stream.
- std::size_t **bufsize**
  Total buffer size of the ac variable buffer.
- int **n_new_samples**
  Number of most recently pulled samples per channel.

5.170.1 Detailed Description

LSL to AC bridge variable.

© 2005-2021 HörTech gGmbH, Oldenburg

5.170.2 Constructor & Destructor Documentation

5.170.2.1 save_var_t() [1/3]

```
lsl2ac::save_var_t::save_var_t (
    const save_var_t & ) [delete]
```

5.170.2.2 save_var_t() [2/3]

```
lsl2ac::save_var_t::save_var_t (
    save_var_t && ) [delete]
```

5.170.2.3 save_var_t() [3/3]

```
lsl2ac::save_var_t::save_var_t (
    const lsl::stream_info & info_,
    const algo_comm_t & ac_,
    overrun_behavior ob_,
    int buffersize_,
    int chunksize_,
    int nchannels_,
    int nsamples_ )
```

C'tor of lsl to ac bridge.

Parameters

*name_*	Name of LSL stream to be received
*info_*	LSL stream info object containing metadata
*ac_*	Handle to ac space
*ub_*	Underrun behavior. 0=Zero out, 1=Copy, 2=Abort
*ob_*	Overrun behavior. 0=Discard oldest, 1=Ignore
*buffersize_*	LSL buffer size
*chunksize_*	LSL chunk size
*nchannels_*	Number of channels in the AC variable must be zero or equal to number of channels in LSL stream
*nsamples_*	Number of samples per channel in the AC variable. Zero means resize as needed

© 2005-2021 HörTech gGmbH, Oldenburg

5.170.2.4 ~save_var_t() `lsl2ac::save_var_t::~save_var_t ( ) [default]`

5.170.3 Member Function Documentation

5.170.3.1 operator=() [1/2] `save_var_t& lsl2ac::save_var_t::operator= ( const save_var_t & ) [delete]`

5.170.3.2 operator=() [2/2] `save_var_t& lsl2ac::save_var_t::operator= ( save_var_t && ) [delete]`

5.170.3.3 info() `lsl::stream_info lsl2ac::save_var_t::info ( )`

Get stream info object from stream inlet.

5.170.3.4 receive_frame() `void lsl2ac::save_var_t::receive_frame ( )`

Receive a samples from lsl and copy to AC space.

Handling of underrun is configuration-dependent

5.170.3.5 pull_samples_ignore() `void lsl2ac::save_var_t::pull_samples_ignore ( ) [private]`

Pull new samples, ignore overrun.

If nsamples=0, leaves the buffers in a state where the newest samples are at the beginning of the buffers, the state of the older samples is undefined and n_new_samples contains the number of new samples per channel. If nsamples is non-zero, the buffers are rotated so the oldest samples are the first in the buffer and n_new_samples contains the number of new samples per channel.

© 2005-2021 HörTech gGmbH, Oldenburg

5.170.3.6 pull_samples_discard() `void lsl2ac::save_var_t::pull_samples_discard ( ) [private]`

Pull new samples as long as there are samples ready for pickup in the LSL buffers.

If nsamples=0, leaves the buffers in a state where the newest samples are at the beginning of the buffers, the state of the older samples is undefined and n_new_samples contains total number of new samples per channel. If nsamples is non-zero, the buffers are rotated so the oldest samples come first. n_samples_new then contains the total number of new samples per channel. If n_new_samples is larger than the number of samples in the AC variable that means samples had be discarded.

5.170.3.7 get_time_correction() `void lsl2ac::save_var_t::get_time_correction ( ) [private]`

Refresh time correction value every 5s.

5.170.3.8 insert_vars() `void lsl2ac::save_var_t::insert_vars ( ) [private]`

Insert stream value, time stamp and time offset into ac space.

5.170.4 Member Data Documentation

5.170.4.1 stream `lsl::stream_inlet lsl2ac::save_var_t::stream [private]`

LSL stream outlet.

Interface to lsl

5.170.4.2 buf `std::vector< mha_real_t> lsl2ac::save_var_t::buf [private]`

Data buffer of the ac variable.

© 2005-2021 HörTech gGmbH, Oldenburg

5.170.4.3 ts_buf `std::vector<double> lsl2ac::save_var_t::ts_buf [private]`

Timestamp buffer.

5.170.4.4 ac `const algo_comm_t& lsl2ac::save_var_t::ac [private]`

Handle to AC space.

5.170.4.5 cv `comm_var_t lsl2ac::save_var_t::cv [private]`

Timeseries AC variable.

5.170.4.6 ts `comm_var_t lsl2ac::save_var_t::ts [private]`

Timestamp AC variable.

5.170.4.7 tc `double lsl2ac::save_var_t::tc =0.0 [private]`

Current time correction.

5.170.4.8 ts_name `std::string lsl2ac::save_var_t::ts_name [private]`

Timestamp AC variable name.

5.170.4.9 tc_name `std::string lsl2ac::save_var_t::tc_name [private]`

Time correction AC variable name.

© 2005-2021 HörTech gGmbH, Oldenburg

5.170.4.10 new_name `std::string lsl2ac::save_var_t::new_name [private]`

Number of new samples AC variable name.

5.170.4.11 tic `std::chrono::time_point<std::chrono::steady_clock> lsl2ac::save_var_t::tic [private]`

time point of last time correction pull

5.170.4.12 skip `bool lsl2ac::save_var_t::skip =false [private]`

Should the variable be skipped in future process calls? Only true when error occured.

5.170.4.13 ob `overrun_behavior lsl2ac::save_var_t::ob [private]`

Behavior on stream overrun.

5.170.4.14 name `const std::string lsl2ac::save_var_t::name [private]`

Name of stream.

Must be saved separately because the stream info might be unrecoverable in error cases

5.170.4.15 nchannels `int32_t lsl2ac::save_var_t::nchannels [private]`

Number of channels.

5.170.4.16 nsamples `std::size_t lsl2ac::save_var_t::nsamples [private]`

Number of samples per channel in the AC variable.

Zero means resize as needed.

© 2005-2021 HörTech gGmbH, Oldenburg

5.170.4.17 chunksize `std::size_t lsl2ac::save_var_t::chunksize [private]`

Maximal chunk size of lsl stream.

5.170.4.18 bufsize `std::size_t lsl2ac::save_var_t::bufsize [private]`

Total buffer size of the ac variable buffer.

nsamples∗nchannels if nsamples is set, chunksize∗nchannels otherwise if chunksize is set, nchannels if neither nsamples and chunksize are set.

5.170.4.19 n_new_samples `int lsl2ac::save_var_t::n_new_samples [private]`

Number of most recently pulled samples per channel.

The documentation for this class was generated from the following files:

- **lsl2ac.hh**
- **lsl2ac.cpp**

5.171 matlab_wrapper::callback Class Reference

Utility class connecting a user_config_t instance to its corresponding configuration parser.

Public Member Functions

- **callback** (**matlab_wrapper_t** ∗parent_, user_config_t ∗user_config_, **MHAParser↩ ::mfloat_t** ∗var_)

  Ctor.
- void **on_writeaccess** ()

  Write access callback.

Private Attributes

- user_config_t ∗ **user_config**

  Ptr to user_config_t.
- **MHAParser::mfloat_t** ∗ **var**

  Ptr to parser.
- **matlab_wrapper_t** ∗ **parent**

  Ptr to parent plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

5.171.1 Detailed Description

Utility class connecting a user_config_t instance to its corresponding configuration parser.

Provides the callback. Every time the a user defined configuration parser is changed, this class makes sure that the corresponding 'matlab-side' struct is also changed and a new real time config is created and pushed.

5.171.2 Constructor & Destructor Documentation

5.171.2.1 callback()

```
matlab_wrapper::callback::callback (
    matlab_wrapper_t * parent_,
    user_config_t * user_config_,
    MHAParser::mfloat_t * var_ )
```

Ctor.

Parameters

*parent_*	Ptr to the parent parser. Used to signal finished change.
*user_config_*	Ptr to user_config_t struct holding the 'matlab side' of the configuration variable
*var_*	Ptr to the configuration parser corresponding to user_config_

5.171.3 Member Function Documentation

5.171.3.1 on_writeaccess()

```
void matlab_wrapper::callback::on_writeaccess ( )
```

Write access callback.

To be called on every writeaccess of ∗var. Synchronises ∗var and ∗user_config.

5.171.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.171.4.1 user_config `user_config_t* matlab_wrapper::callback::user_config [private]`

Ptr to user_config_t.

5.171.4.2 var `MHAParser::mfloat_t* matlab_wrapper::callback::var [private]`

Ptr to parser.

5.171.4.3 parent `matlab_wrapper_t* matlab_wrapper::callback::parent [private]`

Ptr to parent plugin.

The documentation for this class was generated from the following files:

- **matlab_wrapper.hh**
- **matlab_wrapper.cpp**

5.172 matlab_wrapper::matlab_wrapper_rt_cfg_t Class Reference

Thin wrapper around the emxArray containing the user defined configuration variables.

Public Member Functions

- **matlab_wrapper_rt_cfg_t** (emxArray_user_config_t *user_config_)
  Ctor of the real time configuration class.
- ~**matlab_wrapper_rt_cfg_t** ()
  Dtor.

Public Attributes

- emxArray_user_config_t * **user_config**
  User configuration to be used in the process callback.

© 2005-2021 HörTech gGmbH, Oldenburg

5.172.1 Detailed Description

Thin wrapper around the emxArray containing the user defined configuration variables.

This wrapper holds a copy of the user configuration which is used by the process callback. On write access to a variable, the user configuration is changed and a new copy is created and exchanged in a real time safe manner.

5.172.2 Constructor & Destructor Documentation

5.172.2.1 matlab_wrapper_rt_cfg_t()

```
matlab_wrapper::matlab_wrapper_rt_cfg_t::matlab↩
_wrapper_rt_cfg_t (
            emxArray_user_config_t * user_config_ ) [explicit]
```

Ctor of the real time configuration class.

Creates a local copy of the user config

Parameters

*user_↩ config_*	Pointer to the array containing the user config

5.172.2.2 ~matlab_wrapper_rt_cfg_t()

```
matlab_wrapper::matlab_wrapper_rt_cfg_t↩
::~matlab_wrapper_rt_cfg_t ( )
```

Dtor.

Calls the appropriate c-style destructors of the emx_ types

5.172.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.172.3.1 user_config `emxArray_user_config_t* matlab_wrapper::matlab_wrapper_rt_cfg_t::user_config`

User configuration to be used in the process callback.

The documentation for this class was generated from the following files:

- **matlab_wrapper.hh**
- **matlab_wrapper.cpp**

5.173 matlab_wrapper::matlab_wrapper_t Class Reference

Matlab wraper plugin interface class.

Inheritance diagram for matlab_wrapper::matlab_wrapper_t:

Classes

- class **wrapped_plugin_t**

  Wrapper class around the matlab-generated library.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **matlab_wrapper_t** (const **algo_comm_t** iac, const std::string &configured_name)
  Ctor of the plugin interface class.
- void **process** (**mha_wave_t** ∗sin, **mha_wave_t** ∗∗sout)
  Pure waveform processing.
- void **process** (**mha_spec_t** ∗sin, **mha_spec_t** ∗∗sout)
  Pure spectrum processing.
- void **process** (**mha_wave_t** ∗sin, **mha_spec_t** ∗∗sout)
  Signal processing with domain transformation from waveform to spectrum.
- void **process** (**mha_spec_t** ∗sin, **mha_wave_t** ∗∗sout)
  Signal processing with domain transformation from spectrum to waveform.
- void **prepare** (**mhaconfig_t** &signal_info)
  Prepare callback.
- void **release** ()
  Release callback.

Private Member Functions

- void **insert_monitors** ()
- void **update_monitors** ()
- void **insert_config_vars** ()
- void **load_lib** ()
  Create new library wrapper and load library.
- void **update** ()
  Create new real time safe user config from user config.

Private Attributes

- friend **callback**
- **MHAParser::string_t library_name** {"Name of matlab generated library",""}
  Configuration variable holding the file name of the matlab generated library.
- **MHAEvents::patchbay_t**< **matlab_wrapper_t** > **patchbay**
  Patchbay for the interface plugins.
- **MHAEvents::patchbay_t**< **callback** > **cb_patchbay**
  Patchbay for the custom callbacks.
- std::unique_ptr< **wrapped_plugin_t** > **plug**
  Unique ptr holding the instance of the plugin wrapper class.
- std::deque< **MHAParser::mfloat_t** > **vars**
  Deque holding the user defined configuration variables.
- std::deque< **callback** > **callbacks**
  Deque holding the callbacks for the user defined variables' write access.
- std::deque< **MHAParser::mfloat_mon_t** > **monitors**
  Deque holding the monitors corresponding to user state.

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.173.1 Detailed Description

Matlab wraper plugin interface class.

5.173.2 Constructor & Destructor Documentation

5.173.2.1 matlab_wrapper_t()

```
matlab_wrapper::matlab_wrapper_t::matlab_wrapper_t (
    const algo_comm_t iac,
    const std::string & configured_name )
```

Ctor of the plugin interface class.

Parameters

iac	Handle to ac space
configured_name	Configured name of this plugin

5.173.3 Member Function Documentation

5.173.3.1 process() [1/4]

```
void matlab_wrapper::matlab_wrapper_t::process (
    mha_wave_t * sin,
    mha_wave_t ** sout )
```

Pure waveform processing.

Parameters

sin	input waveform signal
sout	output waveform signal

© 2005-2021 HörTech gGmbH, Oldenburg

5.173.3.2 process() [2/4]

```
void matlab_wrapper::matlab_wrapper_t::process (
    mha_spec_t * sin,
    mha_spec_t ** sout )
```

Pure spectrum processing.

Parameters

sin	input spectrum signal
sout	output spectrum signal

5.173.3.3 process() [3/4]

```
void matlab_wrapper::matlab_wrapper_t::process (
    mha_wave_t * sin,
    mha_spec_t ** sout )
```

Signal processing with domain transformation from waveform to spectrum.

Parameters

sin	input waveform signal
sout	output spectrum signal

5.173.3.4 process() [4/4]

```
void matlab_wrapper::matlab_wrapper_t::process (
    mha_spec_t * sin,
    mha_wave_t ** sout )
```

Signal processing with domain transformation from spectrum to waveform.

Parameters

sin	input spectrum signal
sout	output waveform signal

5.173.3.5 prepare()

```
void matlab_wrapper::matlab_wrapper_t::prepare (
    mhaconfig_t & signal_info ) [virtual]
```

Prepare callback.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	struct holding the input/output signal dimensions

Implements **MHAPlugin::plugin_t**< **matlab_wrapper_rt_cfg_t** > (p. 1153).

5.173.3.6 release() `void matlab_wrapper::matlab_wrapper_t::release ( ) [virtual]`

Release callback.

Reimplemented from **MHAPlugin::plugin_t**< **matlab_wrapper_rt_cfg_t** > (p. 1154).

5.173.3.7 insert_monitors() `void matlab_wrapper::matlab_wrapper_t::insert_monitors ( ) [private]`

5.173.3.8 update_monitors() `void matlab_wrapper::matlab_wrapper_t::update_monitors ( ) [private]`

5.173.3.9 insert_config_vars() `void matlab_wrapper::matlab_wrapper_t::insert_↩ config_vars ( ) [private]`

5.173.3.10 load_lib() `void matlab_wrapper::matlab_wrapper_t::load_lib ( ) [private]`

Create new library wrapper and load library.

To be called write access to library_name.

5.173.3.11 update() `void matlab_wrapper::matlab_wrapper_t::update ( ) [private]`

Create new real time safe user config from user config.

Called by the custom callbacks.

© 2005-2021 HörTech gGmbH, Oldenburg

5.173.4 Member Data Documentation

5.173.4.1 callback `friend matlab_wrapper::matlab_wrapper_t::callback [private]`

5.173.4.2 library_name `MHAParser::string_t matlab_wrapper::matlab_wrapper_t↩ ::library_name {"Name of matlab generated library",""} [private]`

Configuration variable holding the file name of the matlab generated library.

5.173.4.3 patchbay `MHAEvents::patchbay_t< matlab_wrapper_t> matlab_wrapper↩ ::matlab_wrapper_t::patchbay [private]`

Patchbay for the interface plugins.

5.173.4.4 cb_patchbay `MHAEvents::patchbay_t< callback> matlab_wrapper::matlab_↩ wrapper_t::cb_patchbay [private]`

Patchbay for the custom callbacks.

Can use normal patchbay bc/ of the interface of patchbay_t

5.173.4.5 plug `std::unique_ptr< wrapped_plugin_t> matlab_wrapper::matlab_wrapper↩ _t::plug [private]`

Unique ptr holding the instance of the plugin wrapper class.

5.173.4.6 vars `std::deque< MHAParser::mfloat_t> matlab_wrapper::matlab_wrapper_t↩ ::vars [private]`

Deque holding the user defined configuration variables.

Deque is used because we need an indexable container that does not invalidate pointers

© 2005-2021 HörTech gGmbH, Oldenburg

5.173.4.7 callbacks `std::deque< callback> matlab_wrapper::matlab_wrapper_t::callbacks [private]`

Deque holding the callbacks for the user defined variables' write access.

Deque is used because we need an indexable container that does not invalidate pointers

5.173.4.8 monitors `std::deque< MHAParser::mfloat_mon_t> matlab_wrapper::matlab_wrapper_t::monitors [private]`

Deque holding the monitors corresponding to user state.

Deque is used because we need an indexable container that does not invalidate pointers.

The documentation for this class was generated from the following files:

- **matlab_wrapper.hh**
- **matlab_wrapper.cpp**

5.174 matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t Class Reference

Wrapper class around the matlab-generated library.

Public Member Functions

- **wrapped_plugin_t** ()=delete
- **wrapped_plugin_t** (const char *name_)
  Ctor.
- virtual ~**wrapped_plugin_t** ()
  Dtor.
- **mha_wave_t** * **process_ww** (**mha_wave_t** *s, emxArray_user_config_t *user_config_)
  Process callback.
- **mha_spec_t** * **process_ss** (**mha_spec_t** *s, emxArray_user_config_t *user_config_)
  Process callback.
- **mha_spec_t** * **process_ws** (**mha_wave_t** *s, emxArray_user_config_t *user_config_)
  Process callback.
- **mha_wave_t** * **process_sw** (**mha_spec_t** *s, emxArray_user_config_t *user_config_)
  Process callback.
- void **prepare** (**mhaconfig_t** &config)
  Prepare callback.
- void **release** ()
  Release callback.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- emxArray_user_config_t * **user_config** =nullptr
  Ptr to user config array.
- emxArray_user_config_t * **state** =nullptr
  Ptr to state array.

Private Attributes

- **dynamiclib_t library_handle**
  Handle to matlab generated shared library.
- void(* **fcn_terminate**)() =nullptr
  Handle to matlab generated cleanup function.
- void(* **fcn_init**)(emxArray_user_config_t *, emxArray_user_config_t *) =nullptr
  Handle to matlab generated init function.
- void(* **fcn_process_ww**)(const emxArray_real_T *, const signal_dimensions_t *, const emxArray_user_config_t *, emxArray_user_config_t *, emxArray_real_T *) =nullptr
  Handle to matlab generated wave to wave process function.
- void(* **fcn_process_ss**)(const emxArray_creal_T *, const signal_dimensions_t *, const emxArray_user_config_t *, emxArray_user_config_t *, emxArray_creal_T *) =nullptr
  Handle to matlab generated spectrum to spectrum process function.
- void(* **fcn_process_ws**)(const emxArray_real_T *, const signal_dimensions_t *, const emxArray_user_config_t *, emxArray_user_config_t *, emxArray_creal_T *) =nullptr
  Handle to matlab generated wave to spectrum process function.
- void(* **fcn_process_sw**)(const emxArray_creal_T *, const signal_dimensions_t *, const emxArray_user_config_t *, emxArray_user_config_t *, emxArray_real_T *) =nullptr
  Handle to matlab generated spectrum to wave process function.
- void(* **fcn_prepare**)(signal_dimensions_t *, emxArray_user_config_t *, emxArray_↩user_config_t *) =nullptr
  Handle to matlab generated prepare function.
- void(* **fcn_release**)() =nullptr
- signal_dimensions_t **signal_dimensions** {0,'U',0,0,0,0}
  Signal dimensions.
- emxArray_real_T * **wave_in** =nullptr
  Ptr to emxArray holding the input signal.
- emxArray_real_T * **wave_out** =nullptr
  Ptr to emxArray holding the output signal.
- emxArray_creal_T * **spec_in** =nullptr
  Ptr to emxArray holding the input signal.
- emxArray_creal_T * **spec_out** =nullptr
  Ptr to emxArray holding the output signal.
- std::unique_ptr< **MHASignal::waveform_t** > **mha_wave_out**
  MHA waveform holding the output signal.
- std::unique_ptr< **MHASignal::spectrum_t** > **mha_spec_out**
  MHA waveform holding the output signal.

© 2005-2021 HörTech gGmbH, Oldenburg

5.174.1 Detailed Description

Wrapper class around the matlab-generated library.

5.174.2 Constructor & Destructor Documentation

5.174.2.1 wrapped_plugin_t() [1/2] matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t::wrapped_plugin_t ( ) [delete]

5.174.2.2 wrapped_plugin_t() [2/2] matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t::wrapped_plugin_t (const char * *name_*) [explicit]

Ctor.

Opens matlab-generated library and resolves functions

Parameters

*name_*	file name of shared library

5.174.2.3 ~wrapped_plugin_t() matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t::~wrapped_plugin_t ( ) [virtual]

Dtor.

5.174.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.174.3.1 process_ww()

```
mha_wave_t * matlab_wrapper::matlab_wrapper_t::wrapped_↩
plugin_t::process_ww (
            mha_wave_t * s,
            emxArray_user_config_t * user_config_ )
```

Process callback.

Converts mha-type wave to matlab array and calls wrapped process callback, then convert output to mha type

Parameters

s	input/output signal
*user_↩ config_*	Ptr to user configuration array

5.174.3.2 process_ss()

```
mha_spec_t * matlab_wrapper::matlab_wrapper_t::wrapped_↩
plugin_t::process_ss (
            mha_spec_t * s,
            emxArray_user_config_t * user_config_ )
```

Process callback.

Converts mha-type spectrum to matlab array and calls wrapped process callback, then convert output to mha type

Parameters

s	input/output signal
*user_↩ config_*	Ptr to user configuration array

5.174.3.3 process_ws()

```
mha_spec_t * matlab_wrapper::matlab_wrapper_t::wrapped_↩
plugin_t::process_ws (
            mha_wave_t * s,
            emxArray_user_config_t * user_config_ )
```

Process callback.

Converts mha-type wave to matlab array and calls wrapped process callback, then convert output to mha type

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s	input/output signal
*user_↩ config_*	Ptr to user configuration array

5.174.3.4 process_sw()

```
mha_wave_t * matlab_wrapper::matlab_wrapper_t::wrapped_↩
plugin_t::process_sw (
            mha_spec_t * s,
            emxArray_user_config_t * user_config_ )
```

Process callback.

Converts mha-type spectrum to matlab array and calls wrapped process callback, then convert output to mha type

Parameters

s	input/output signal
*user_↩ config_*	Ptr to user configuration array

5.174.3.5 prepare()

```
void matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t::prepare
(
            mhaconfig_t & config )
```

Prepare callback.

Calls wrapped prepare function if necessary and determines output signal dimensions

5.174.3.6 release()

```
void matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t::release
( )
```

Release callback.

Cleans up io arrays and calls wrapped release if necessary.

© 2005-2021 HörTech gGmbH, Oldenburg

5.174.4 Member Data Documentation

5.174.4.1 user_config `emxArray_user_config_t* matlab_wrapper::matlab_wrapper_t↩ ::wrapped_plugin_t::user_config =nullptr`

Ptr to user config array.

5.174.4.2 state `emxArray_user_config_t* matlab_wrapper::matlab_wrapper_t::wrapped↩ _plugin_t::state =nullptr`

Ptr to state array.

5.174.4.3 library_handle `dynamiclib_t matlab_wrapper::matlab_wrapper_t::wrapped_↩ plugin_t::library_handle [private]`

Handle to matlab generated shared library.

5.174.4.4 fcn_terminate `void(* matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t↩ ::fcn_terminate) () =nullptr [private]`

Handle to matlab generated cleanup function.

5.174.4.5 fcn_init `void(* matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t::fcn_↩ init) (emxArray_user_config_t *, emxArray_user_config_t *) =nullptr [private]`

Handle to matlab generated init function.

© 2005-2021 HörTech gGmbH, Oldenburg

5.174.4.6 fcn_process_ww

`void(* matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t::fcn_process_ww) (const emxArray_real_T *, const signal_dimensions_t *, const emxArray_user_config_t *, emxArray_user_config_t *, emxArray_real_T *) =nullptr [private]`

Handle to matlab generated wave to wave process function.

5.174.4.7 fcn_process_ss

`void(* matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t::fcn_process_ss) (const emxArray_creal_T *, const signal_dimensions_t *, const emxArray_user_config_t *, emxArray_user_config_t *, emxArray_creal_T *) =nullptr [private]`

Handle to matlab generated spectrum to spectrum process function.

5.174.4.8 fcn_process_ws

`void(* matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t::fcn_process_ws) (const emxArray_real_T *, const signal_dimensions_t *, const emxArray_user_config_t *, emxArray_user_config_t *, emxArray_creal_T *) =nullptr [private]`

Handle to matlab generated wave to spectrum process function.

5.174.4.9 fcn_process_sw

`void(* matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t::fcn_process_sw) (const emxArray_creal_T *, const signal_dimensions_t *, const emxArray_user_config_t *, emxArray_user_config_t *, emxArray_real_T *) =nullptr [private]`

Handle to matlab generated spectrum to wave process function.

5.174.4.10 fcn_prepare

`void(* matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t::fcn_prepare) (signal_dimensions_t *, emxArray_user_config_t *, emxArray_user_config_t *) =nullptr [private]`

Handle to matlab generated prepare function.

© 2005-2021 HörTech gGmbH, Oldenburg

5.174.4.11 fcn_release `void(* matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t↩ ::fcn_release) () =nullptr [private]`

5.174.4.12 signal_dimensions `signal_dimensions_t matlab_wrapper::matlab_wrapper_↩ t::wrapped_plugin_t::signal_dimensions {0,'U',0,0,0,0} [private]`

Signal dimensions.

Matlab-equivalent to mha_config_t

5.174.4.13 wave_in `emxArray_real_T* matlab_wrapper::matlab_wrapper_t::wrapped_↩ plugin_t::wave_in =nullptr [private]`

Ptr to emxArray holding the input signal.

5.174.4.14 wave_out `emxArray_real_T* matlab_wrapper::matlab_wrapper_t::wrapped_↩ plugin_t::wave_out =nullptr [private]`

Ptr to emxArray holding the output signal.

5.174.4.15 spec_in `emxArray_creal_T* matlab_wrapper::matlab_wrapper_t::wrapped_↩ plugin_t::spec_in =nullptr [private]`

Ptr to emxArray holding the input signal.

5.174.4.16 spec_out `emxArray_creal_T* matlab_wrapper::matlab_wrapper_t::wrapped_↩ plugin_t::spec_out =nullptr [private]`

Ptr to emxArray holding the output signal.

© 2005-2021 HörTech gGmbH, Oldenburg

5.174.4.17 mha_wave_out `std::unique_ptr< MHASignal::waveform_t> matlab_wrapper↩ ::matlab_wrapper_t::wrapped_plugin_t::mha_wave_out [private]`

MHA waveform holding the output signal.

5.174.4.18 mha_spec_out `std::unique_ptr< MHASignal::spectrum_t> matlab_wrapper↩ ::matlab_wrapper_t::wrapped_plugin_t::mha_spec_out [private]`

MHA waveform holding the output signal.

The documentation for this class was generated from the following files:

- **matlab_wrapper.hh**
- **matlab_wrapper.cpp**

5.175 matlab_wrapper::types< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- **matlab_wrapper.hh**

5.176 matlab_wrapper::types< MHA_SPECTRUM > Struct Reference

Public Types

- typedef emxArray_creal_T **array_type**
- typedef **mha_spec_t signal_type**
- typedef **MHASignal::spectrum_t class_signal_type**

5.176.1 Member Typedef Documentation

5.176.1.1 array_type `typedef emxArray_creal_T matlab_wrapper::types< MHA_SPECT↩ RUM >:: array_type`

© 2005-2021 HörTech gGmbH, Oldenburg

5.176.1.2 signal_type `typedef mha_spec_t matlab_wrapper::types< MHA_SPECTRUM >::signal_type`

5.176.1.3 class_signal_type `typedef MHASignal::spectrum_t matlab_wrapper::types< MHA_SPECTRUM >::class_signal_type`

The documentation for this struct was generated from the following file:

- **matlab_wrapper.hh**

5.177 matlab_wrapper::types< MHA_WAVEFORM > Struct Reference

Public Types

- typedef emxArray_real_T **array**
- typedef **mha_wave_t signal_type**
- typedef **MHASignal::waveform_t class_signal_type**

5.177.1 Member Typedef Documentation

5.177.1.1 array `typedef emxArray_real_T matlab_wrapper::types< MHA_WAVEFORM >::array`

5.177.1.2 signal_type `typedef mha_wave_t matlab_wrapper::types< MHA_WAVEFORM >::signal_type`

5.177.1.3 class_signal_type `typedef MHASignal::waveform_t matlab_wrapper::types< MHA_WAVEFORM >::class_signal_type`

The documentation for this struct was generated from the following file:

- **matlab_wrapper.hh**

© 2005-2021 HörTech gGmbH, Oldenburg

5.178 matrixmixer::cfg_t Class Reference

Public Member Functions

- **cfg_t** (std::vector< std::vector< float > > imixer, unsigned int ci, unsigned int co, unsigned int fragsize, unsigned int nfft)
- **mha_wave_t** * **process** (**mha_wave_t** *)
- **mha_spec_t** * **process** (**mha_spec_t** *)

Private Attributes

- **MHASignal::waveform_t m**
- **MHASignal::waveform_t wout**
- **MHASignal::spectrum_t sout**

5.178.1 Constructor & Destructor Documentation

5.178.1.1 cfg_t()

```
cfg_t::cfg_t (
    std::vector< std::vector< float > > imixer,
    unsigned int ci,
    unsigned int co,
    unsigned int fragsize,
    unsigned int nfft )
```

5.178.2 Member Function Documentation

5.178.2.1 process() [1/2]

```
mha_wave_t * cfg_t::process (
    mha_wave_t * s )
```

5.178.2.2 process() [2/2]

```
mha_spec_t * cfg_t::process (
    mha_spec_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.178.3 Member Data Documentation

5.178.3.1 m `MHASignal::waveform_t matrixmixer::cfg_t::m [private]`

5.178.3.2 wout `MHASignal::waveform_t matrixmixer::cfg_t::wout [private]`

5.178.3.3 sout `MHASignal::spectrum_t matrixmixer::cfg_t::sout [private]`

The documentation for this class was generated from the following file:

- **matrixmixer.cpp**

5.179 matrixmixer::matmix_t Class Reference

Inheritance diagram for matrixmixer::matmix_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **matmix_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

Private Member Functions

- void **update_m** ()

Private Attributes

- **MHAEvents::patchbay_t**< **matmix_t** > **patchbay**
- **MHAParser::mfloat_t** **mixer**
- unsigned int **ci**
- unsigned int **co**

Additional Inherited Members

5.179.1 Constructor & Destructor Documentation

5.179.1.1 matmix_t()

```
matrixmixer::matmix_t::matmix_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.179.2 Member Function Documentation

5.179.2.1 prepare()

```
void matrixmixer::matmix_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t**< **cfg_t** > (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.179.2.2 process() [1/2] `mha_wave_t * matrixmixer::matmix_t::process ( mha_wave_t * s )`

5.179.2.3 process() [2/2] `mha_spec_t * matrixmixer::matmix_t::process ( mha_spec_t * s )`

5.179.2.4 update_m() `void matrixmixer::matmix_t::update_m ( ) [private]`

5.179.3 Member Data Documentation

5.179.3.1 patchbay `MHAEvents::patchbay_t< matmix_t> matrixmixer::matmix_t::patchbay [private]`

5.179.3.2 mixer `MHAParser::mfloat_t matrixmixer::matmix_t::mixer [private]`

5.179.3.3 ci `unsigned int matrixmixer::matmix_t::ci [private]`

5.179.3.4 co `unsigned int matrixmixer::matmix_t::co [private]`

The documentation for this class was generated from the following file:

- **matrixmixer.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.180 mconv::MConv Class Reference

Inheritance diagram for mconv::MConv:

Public Member Functions

- **MConv** (const **algo_comm_t** &iac, const std::string &algoname)
  Plugin constructor.
- void **prepare** (**mhaconfig_t** &mhaconfig)
  Prepare this plugin for processing.
- void **release** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Private Member Functions

- void **update** ()
  *Update is needed only once, since this plugin allows only change of irs after **prepare()*** (p. 719).
- void **update_irs** ()
  *This function updates the irs without allowing a change of its size after **prepare()*** (p. 719).

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::int_t nchannels_out**
  Number of output channels to produce.
- **MHAParser::vint_t inch**
  Vector of input channel indices.
- **MHAParser::vint_t outch**
  Vector of output channel indices.
- **MHAParser::mfloat_t irs**
  Impulse responses, one per row.
- unsigned int **nchannels_in**
  Number of input channels, set during prepare.
- unsigned int **fragsize**
  Fragsize, set during prepare, is used as the partition length in the partitioned convolution.
- **MHAEvents::patchbay_t< MConv > patchbay**

Additional Inherited Members

5.180.1 Detailed Description

class implements plugin for partitioned convolution. A matrix of impulse responses, filtering n input channels to m output channels, is supported.

5.180.2 Constructor & Destructor Documentation

5.180.2.1 MConv()

```
mconv::MConv::MConv (
    const algo_comm_t & iac,
    const std::string & algoname )
```

Plugin constructor.

Parameters

iac	handle and function pointers for algorithm communication
configured_name	The name by which e.g an mhachain refers to this instance of the mconv plugin

© 2005-2021 HörTech gGmbH, Oldenburg

5.180.3 Member Function Documentation

5.180.3.1 prepare() `void mconv::MConv::prepare ( mhaconfig_t & mhaconfig ) [virtual]`

Prepare this plugin for processing.

Parameters

mhaconfig	Configuration for this plugin (Input/Output parameter) Sample rate, fragment size, number of channels are detailed here.

Implements **MHAPlugin::plugin_t**< **MHAFilter::partitioned_convolution_t** > (p. 1153).

5.180.3.2 release() `void mconv::MConv::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t**< **MHAFilter::partitioned_convolution_t** > (p. 1154).

5.180.3.3 process() `mha_wave_t * mconv::MConv::process ( mha_wave_t * s_in )`

5.180.3.4 update() `void mconv::MConv::update ( ) [private]`

Update is needed only once, since this plugin allows only change of irs after **prepare()** (p. 719).

5.180.3.5 update_irs() `void mconv::MConv::update_irs ( ) [private]`

This function updates the irs without allowing a change of its size after **prepare()** (p. 719).

© 2005-2021 HörTech gGmbH, Oldenburg

5.180.4 Member Data Documentation

5.180.4.1 nchannels_out `MHAParser::int_t mconv::MConv::nchannels_out [private]`

Number of output channels to produce.

5.180.4.2 inch `MHAParser::vint_t mconv::MConv::inch [private]`

Vector of input channel indices.

Each element in this vector identifies the input channel to which to apply the corresponding impulse response in irs.

5.180.4.3 outch `MHAParser::vint_t mconv::MConv::outch [private]`

Vector of output channel indices.

Each element in this vector identifies the output channel to which the result of filtering with the corresponding impulse response in irs is mixed.

5.180.4.4 irs `MHAParser::mfloat_t mconv::MConv::irs [private]`

Impulse responses, one per row.

For each row, the corresponding element of inch identifies the source channel, and the corresponding element of outch identifies the target channel.

5.180.4.5 nchannels_in `unsigned int mconv::MConv::nchannels_in [private]`

Number of input channels, set during prepare.

5.180.4.6 fragsize `unsigned int mconv::MConv::fragsize [private]`

Fragsize, set during prepare, is used as the partition length in the partitioned convolution.

© 2005-2021 HörTech gGmbH, Oldenburg

5.180.4.7 patchbay `MHAEvents::patchbay_t< MConv> mconv::MConv::patchbay [private]`

The documentation for this class was generated from the following file:

- **mconv.cpp**

5.181 MHA_AC::ac2matrix_helper_t Class Reference

Inheritance diagram for MHA_AC::ac2matrix_helper_t:

Public Member Functions

- **ac2matrix_helper_t** (**algo_comm_t**, const std::string &)
- void **getvar** ()

Public Attributes

- **algo_comm_t ac**
- std::string **name**
- std::string **username**
- **MHASignal::uint_vector_t size**
- bool **is_complex**

Protected Attributes

- **comm_var_t acvar**

© 2005-2021 HörTech gGmbH, Oldenburg

5.181.1 Constructor & Destructor Documentation

5.181.1.1 ac2matrix_helper_t()

```
MHA_AC::ac2matrix_helper_t::ac2matrix_helper_t (
    algo_comm_t iac,
    const std::string & iname )
```

5.181.2 Member Function Documentation

5.181.2.1 getvar()

```
void MHA_AC::ac2matrix_helper_t::getvar ( )
```

5.181.3 Member Data Documentation

5.181.3.1 ac

```
algo_comm_t MHA_AC::ac2matrix_helper_t::ac
```

5.181.3.2 name

```
std::string MHA_AC::ac2matrix_helper_t::name
```

5.181.3.3 username

```
std::string MHA_AC::ac2matrix_helper_t::username
```

5.181.3.4 size

```
MHASignal::uint_vector_t MHA_AC::ac2matrix_helper_t::size
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.181.3.5 is_complex `bool MHA_AC::ac2matrix_helper_t::is_complex`

5.181.3.6 acvar `comm_var_t MHA_AC::ac2matrix_helper_t::acvar [protected]`

The documentation for this class was generated from the following files:

- **mha_algo_comm.h**
- **mha_algo_comm.cpp**

5.182 MHA_AC::ac2matrix_t Class Reference

Copy AC variable to a matrix.

Inheritance diagram for MHA_AC::ac2matrix_t:

Public Member Functions

- **ac2matrix_t** (**algo_comm_t ac**, const std::string & **name**)
  Constructor.
- void **update** ()
  Update contents of the matrix from the AC space.
- const std::string & **getname** () const
  Return name of AC variable/matrix.
- const std::string & **getusername** () const
  Return user specified name of AC variable/matrix.
- void **insert** (**algo_comm_t ac**)
  Insert matrix into an AC space (other than source AC space)

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.182.1 Detailed Description

Copy AC variable to a matrix.

This class constructs a matrix of same size as an AC variable and can copy the AC variable to itself. The **update()** (p. 724) function is real-time safe.

5.182.2 Constructor & Destructor Documentation

5.182.2.1 ac2matrix_t()

```
MHA_AC::ac2matrix_t::ac2matrix_t (
    algo_comm_t ac,
    const std::string & name )
```

Constructor.

Parameters

ac	AC handle
name	Name of AC variable to be copied

5.182.3 Member Function Documentation

5.182.3.1 update()

```
void MHA_AC::ac2matrix_t::update ( )
```

Update contents of the matrix from the AC space.

This function is real-time safe. The copy operation performance is of the order of the number of elements in the matrix.

5.182.3.2 getname()

```
const std::string& MHA_AC::ac2matrix_t::getname ( ) const [inline]
```

Return name of AC variable/matrix.

© 2005-2021 HörTech gGmbH, Oldenburg

5.182.3.3 getusername() `const std::string& MHA_AC::ac2matrix_t::getusername ( ) const [inline]`

Return user specified name of AC variable/matrix.

5.182.3.4 insert() `void MHA_AC::ac2matrix_t::insert ( algo_comm_t ac )`

Insert matrix into an AC space (other than source AC space)

Parameters

ac	AC space handle to insert data

Note

The AC variable data buffer points to the data of the matrix. Modifications of the AC variable directly modify the data of the matrix; after deletion of the matrix, the data buffer is invalid.

The documentation for this class was generated from the following files:

- **mha_algo_comm.h**
- **mha_algo_comm.cpp**

5.183 MHA_AC::acspace2matrix_t Class Reference

Copy all or a subset of all numeric AC variables into an array of matrixes.

Public Member Functions

- **acspace2matrix_t** (**algo_comm_t** ac, const std::vector< std::string > &names)
  Constructor.
- **acspace2matrix_t** (const **MHA_AC::acspace2matrix_t** &src)
  Constructor with initialization from an instance.
- ~**acspace2matrix_t** ()
- **MHA_AC::acspace2matrix_t** & **operator=** (const **MHA_AC::acspace2matrix_t** &src)
  Copy all contents (deep copy).
- **MHA_AC::ac2matrix_t** & **operator[ ]** (unsigned int k)

© 2005-2021 HörTech gGmbH, Oldenburg

Access operator.

- const **MHA_AC::ac2matrix_t** & **operator[ ]** (unsigned int k) const

  Constant access operator.
- void **update** ()

  Update function.
- unsigned int **size** () const

  Number of matrixes in AC space.
- unsigned int **frame** () const

  Actual frame number.
- void **insert** (**algo_comm_t** ac)

  Insert AC space copy into an AC space (other than source AC space)

Private Attributes

- unsigned int **len**
- **MHA_AC::ac2matrix_t** ** **data**
- unsigned int **frameno**

5.183.1 Detailed Description

Copy all or a subset of all numeric AC variables into an array of matrixes.

5.183.2 Constructor & Destructor Documentation

5.183.2.1 acspace2matrix_t() [1/2]

```
MHA_AC::acspace2matrix_t::acspace2matrix_t (
    algo_comm_t ac,
    const std::vector< std::string > & names )
```

Constructor.

Scan all given AC variables and allocate corresponding matrixes.

Parameters

ac	AC handle.
names	Names of AC variables, or empty for all.

© 2005-2021 HörTech gGmbH, Oldenburg

5.183.2.2 acspace2matrix_t() [2/2]

```
MHA_AC::acspace2matrix_t::acspace2matrix_t (
    const MHA_AC::acspace2matrix_t & src )
```

Constructor with initialization from an instance.

Parameters

src	Instance to be copied.

5.183.2.3 ~acspace2matrix_t()

```
MHA_AC::acspace2matrix_t::~acspace2matrix_t ( )
```

5.183.3 Member Function Documentation

5.183.3.1 operator=()

```
MHA_AC::acspace2matrix_t & MHA_AC::acspace2matrix_t::operator=
(
    const MHA_AC::acspace2matrix_t & src )
```

Copy all contents (deep copy).

Parameters

src	Array of matrixes to be copied.

5.183.3.2 operator[]() [1/2]

```
MHA_AC::ac2matrix_t& MHA_AC::acspace2matrix_t::operator[]
(
    unsigned int k ) [inline]
```

Access operator.

Parameters

k	index into array; should not exceed **size()** (p. 728)-1.

© 2005-2021 HörTech gGmbH, Oldenburg

Return values

Reference	to matrix.

5.183.3.3 operator[]() [2/2] const MHA_AC::ac2matrix_t& MHA_AC::acspace2matrix_t↩ ::operator[] (unsigned int *k*) const [inline]

Constant access operator.

Parameters

k	index into array; should not exceed **size()** (p. 728)-1.

Return values

Constant	reference to matrix.

5.183.3.4 update() void MHA_AC::acspace2matrix_t::update ( ) [inline]

Update function.

This function updates all matrixes from their corresponding AC variables. It can be called from the MHA Framework prepare function or in the processing callback.

5.183.3.5 size() unsigned int MHA_AC::acspace2matrix_t::size ( ) const [inline]

Number of matrixes in AC space.

5.183.3.6 frame() unsigned int MHA_AC::acspace2matrix_t::frame ( ) const [inline]

Actual frame number.

5.183.3.7 insert() void MHA_AC::acspace2matrix_t::insert (**algo_comm_t** *ac*)

Insert AC space copy into an AC space (other than source AC space)

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

ac	AC space handle to insert data

5.183.4 Member Data Documentation

5.183.4.1 len `unsigned int MHA_AC::acspace2matrix_t::len [private]`

5.183.4.2 data `MHA_AC::ac2matrix_t** MHA_AC::acspace2matrix_t::data [private]`

5.183.4.3 frameno `unsigned int MHA_AC::acspace2matrix_t::frameno [private]`

The documentation for this class was generated from the following files:

- **mha_algo_comm.h**
- **mha_algo_comm.cpp**

5.184 MHA_AC::double_t Class Reference

Insert a double precision floating point variable into the AC space.

Public Member Functions

- **double_t** (**algo_comm_t**, std::string, double=0)
- ~**double_t** ()
- void **insert** ()

Public Attributes

- double **data**

  Floating point (double precision) value variable.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **algo_comm_t ac**
- std::string **name**

5.184.1 Detailed Description

Insert a double precision floating point variable into the AC space.

The variable is automatically removed on destruction.

5.184.2 Constructor & Destructor Documentation

5.184.2.1 double_t()

```
MHA_AC::double_t::double_t (
    algo_comm_t ac,
    std::string name,
    double val = 0 )
```

5.184.2.2 ~double_t()

```
MHA_AC::double_t::~double_t ( )
```

5.184.3 Member Function Documentation

5.184.3.1 insert()

```
void MHA_AC::double_t::insert ( )
```

5.184.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.184.4.1 data `double MHA_AC::double_t::data`

Floating point (double precision) value variable.

5.184.4.2 ac `algo_comm_t MHA_AC::double_t::ac [private]`

5.184.4.3 name `std::string MHA_AC::double_t::name [private]`

The documentation for this class was generated from the following files:

- **mha_algo_comm.h**
- **mha_algo_comm.cpp**

5.185 MHA_AC::float_t Class Reference

Insert a float point variable into the AC space.

Public Member Functions

- **float_t** (**algo_comm_t**, std::string, float=0)
  Constructor.
- ~**float_t** ()
- void **insert** ()

Public Attributes

- float **data**
  Floating point value variable.

Private Attributes

- **algo_comm_t ac**
- std::string **name**

© 2005-2021 HörTech gGmbH, Oldenburg

5.185.1 Detailed Description

Insert a float point variable into the AC space.

The variable is automatically removed on destruction.

5.185.2 Constructor & Destructor Documentation

5.185.2.1 float_t()

```
MHA_AC::float_t::float_t (
    algo_comm_t ac,
    std::string name,
    float val = 0 )
```

Constructor.

5.185.2.2 ~float_t()

```
MHA_AC::float_t::~float_t ( )
```

5.185.3 Member Function Documentation

5.185.3.1 insert()

```
void MHA_AC::float_t::insert ( )
```

5.185.4 Member Data Documentation

5.185.4.1 data

```
float MHA_AC::float_t::data
```

Floating point value variable.

© 2005-2021 HörTech gGmbH, Oldenburg

5.185.4.2 ac `algo_comm_t MHA_AC::float_t::ac [private]`

5.185.4.3 name `std::string MHA_AC::float_t::name [private]`

The documentation for this class was generated from the following files:

- **mha_algo_comm.h**
- **mha_algo_comm.cpp**

5.186 MHA_AC::int_t Class Reference

Insert a integer variable into the AC space.

Public Member Functions

- **int_t** (**algo_comm_t ac**, std::string **name**, int val=0)
- ~**int_t** ()
- void **insert** ()

Public Attributes

- int **data**
  Integer value variable.

Private Attributes

- **algo_comm_t ac**
- std::string **name**

5.186.1 Detailed Description

Insert a integer variable into the AC space.

The variable is automatically removed on destruction.

© 2005-2021 HörTech gGmbH, Oldenburg

5.186.2 Constructor & Destructor Documentation

5.186.2.1 int_t()

```
MHA_AC::int_t::int_t (
    algo_comm_t ac,
    std::string name,
    int val = 0 )
```

5.186.2.2 ∼int_t()

```
MHA_AC::int_t::∼int_t ( )
```

5.186.3 Member Function Documentation

5.186.3.1 insert()

```
void MHA_AC::int_t::insert ( )
```

5.186.4 Member Data Documentation

5.186.4.1 data

```
int MHA_AC::int_t::data
```

Integer value variable.

5.186.4.2 ac

```
algo_comm_t MHA_AC::int_t::ac [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.186.4.3 name `std::string MHA_AC::int_t::name [private]`

The documentation for this class was generated from the following files:

- **mha_algo_comm.h**
- **mha_algo_comm.cpp**

5.187 MHA_AC::spectrum_t Class Reference

Insert a **MHASignal::spectrum_t** (p. 1248) class into the AC space.

Inheritance diagram for MHA_AC::spectrum_t:

Public Member Functions

- **spectrum_t** (**algo_comm_t ac**, std::string **name**, unsigned int bins, unsigned int **channels**, bool insert_now)

  Create the AC variable.
- ~**spectrum_t** ()
- void **insert** ()

  Insert AC variable into AC space.

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Attributes

- **algo_comm_t ac**
- std::string **name**

Additional Inherited Members

5.187.1 Detailed Description

Insert a **MHASignal::spectrum_t** (p. 1248) class into the AC space.

The variable is automatically removed on destruction.

5.187.2 Constructor & Destructor Documentation

5.187.2.1 spectrum_t()

```
MHA_AC::spectrum_t::spectrum_t (
    algo_comm_t ac,
    std::string name,
    unsigned int bins,
    unsigned int channels,
    bool insert_now )
```

Create the AC variable.

Parameters

ac	AC handle
name	Name of variable in AC space
bins	Number of FFT bins in the **waveform_t** (p. 739) class
channels	Number of audio channels in the **waveform_t** (p. 739) class
insert_now	Insert implicitely in the constructor (true) or explicitely in the **insert()** (p. 737) function (false)

5.187.2.2 ~spectrum_t()

```
MHA_AC::spectrum_t::~spectrum_t ( ) [virtual]
```

Reimplemented from **MHASignal::spectrum_t** (p. 1250).

© 2005-2021 HörTech gGmbH, Oldenburg

5.187.3 Member Function Documentation

5.187.3.1 insert() `void MHA_AC::spectrum_t::insert ( )`

Insert AC variable into AC space.

5.187.4 Member Data Documentation

5.187.4.1 ac `algo_comm_t MHA_AC::spectrum_t::ac [protected]`

5.187.4.2 name `std::string MHA_AC::spectrum_t::name [protected]`

The documentation for this class was generated from the following files:

- **mha_algo_comm.h**
- **mha_algo_comm.cpp**

5.188 MHA_AC::stat_t Class Reference

Inheritance diagram for MHA_AC::stat_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **stat_t** (**algo_comm_t** ac, const std::string &name, const unsigned int &frames, const unsigned int & **channels**, bool insert_now)
- void **update** ()
- void **insert** ()

Private Attributes

- **MHA_AC::waveform_t mean**
- **MHA_AC::waveform_t std**

5.188.1 Constructor & Destructor Documentation

5.188.1.1 stat_t()

```
MHA_AC::stat_t::stat_t (
    algo_comm_t ac,
    const std::string & name,
    const unsigned int & frames,
    const unsigned int & channels,
    bool insert_now )
```

5.188.2 Member Function Documentation

5.188.2.1 update()

```
void MHA_AC::stat_t::update ( )
```

5.188.2.2 insert()

```
void MHA_AC::stat_t::insert ( )
```

5.188.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.188.3.1 mean `MHA_AC::waveform_t MHA_AC::stat_t::mean [private]`

5.188.3.2 std `MHA_AC::waveform_t MHA_AC::stat_t::std [private]`

The documentation for this class was generated from the following files:

- **mha_algo_comm.h**
- **mha_algo_comm.cpp**

5.189 MHA_AC::waveform_t Class Reference

Insert a **MHASignal::waveform_t** (p. 1263) class into the AC space.

Inheritance diagram for MHA_AC::waveform_t:

Public Member Functions

- **waveform_t** (**algo_comm_t** **ac**, std::string **name**, unsigned int frames, unsigned int **channels**, bool insert_now)

  Create the AC variable.
- ~**waveform_t** ()
- void **insert** ()

  Insert AC variable into AC space.

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Attributes

- **algo_comm_t ac**
- std::string **name**

Additional Inherited Members

5.189.1 Detailed Description

Insert a **MHASignal::waveform_t** (p. 1263) class into the AC space.

The variable is automatically removed on destruction.

5.189.2 Constructor & Destructor Documentation

5.189.2.1 waveform_t()

```
MHA_AC::waveform_t::waveform_t (
    algo_comm_t ac,
    std::string name,
    unsigned int frames,
    unsigned int channels,
    bool insert_now )
```

Create the AC variable.

Parameters

ac	AC handle
name	Name of variable in AC space
frames	Number of frames in the **waveform_t** (p. 739) class
channels	Number of audio channels in the **waveform_t** (p. 739) class
insert_now	Insert implicitely in the constructor (true) or explicitely in the **insert()** (p. 741) function (false)

5.189.2.2 ~waveform_t()

```
MHA_AC::waveform_t::~waveform_t ( ) [virtual]
```

Reimplemented from **MHASignal::waveform_t** (p. 1266).

© 2005-2021 HörTech gGmbH, Oldenburg

5.189.3 Member Function Documentation

5.189.3.1 insert() `void MHA_AC::waveform_t::insert ( )`

Insert AC variable into AC space.

5.189.4 Member Data Documentation

5.189.4.1 ac `algo_comm_t MHA_AC::waveform_t::ac [protected]`

5.189.4.2 name `std::string MHA_AC::waveform_t::name [protected]`

The documentation for this class was generated from the following files:

- **mha_algo_comm.h**
- **mha_algo_comm.cpp**

5.190 mha_audio_descriptor_t Struct Reference

Description of an audio fragment (planned as a replacement of **mhaconfig_t** (p. 851)).

Public Attributes

- unsigned int **n_samples**
  Number of samples.
- unsigned int **n_channels**
  Number of audio channels.
- unsigned int **n_freqs**
  Number of frequency bands.
- unsigned int **is_complex**
  Flag about sample type.
- **mha_real_t dt**
  Time distance between samples (only equidistant samples allowed)
- **mha_real_t** ∗ **cf**
  Center frequencies of frequency bands.
- **mha_real_t** ∗ **chdir**
  Hint on source direction of channel, values below zero is left, values above zero is right, zero means unknown.

© 2005-2021 HörTech gGmbH, Oldenburg

5.190.1 Detailed Description

Description of an audio fragment (planned as a replacement of **mhaconfig_t** (p. 851)).

5.190.2 Member Data Documentation

5.190.2.1 n_samples `unsigned int mha_audio_descriptor_t::n_samples`

Number of samples.

5.190.2.2 n_channels `unsigned int mha_audio_descriptor_t::n_channels`

Number of audio channels.

5.190.2.3 n_freqs `unsigned int mha_audio_descriptor_t::n_freqs`

Number of frequency bands.

5.190.2.4 is_complex `unsigned int mha_audio_descriptor_t::is_complex`

Flag about sample type.

5.190.2.5 dt `mha_real_t mha_audio_descriptor_t::dt`

Time distance between samples (only equidistant samples allowed)

© 2005-2021 HörTech gGmbH, Oldenburg

5.190.2.6 cf `mha_real_t* mha_audio_descriptor_t::cf`

Center frequencies of frequency bands.

5.190.2.7 chdir `mha_real_t* mha_audio_descriptor_t::chdir`

Hint on source direction of channel, values below zero is left, values above zero is right, zero means unknown.

The documentation for this struct was generated from the following file:

- **mha.hh**

5.191 mha_audio_t Struct Reference

An audio fragment in the openMHA (planned as a replacement of **mha_wave_t** (p. 840) and **mha_spec_t** (p. 794)).

Public Attributes

- **mha_audio_descriptor_t descriptor**
  Dimension and description of the data.
- **mha_real_t * rdata**
  *Data pointer if flag **mha_audio_descriptor_t::is_complex** (p. 742) is unset.*
- **mha_complex_t * cdata**
  *Data pointer if flag **mha_audio_descriptor_t::is_complex** (p. 742) is set.*

5.191.1 Detailed Description

An audio fragment in the openMHA (planned as a replacement of **mha_wave_t** (p. 840) and **mha_spec_t** (p. 794)).

The data alignment is $(t_0, c_0, f_0), (t_0, c_0, f_1), \ldots, (t_0, c_0, f_{freqs}), (t_0, c_1, f_0), \ldots$. This allows a direct cast of the current **mha_wave_t** (p. 840) and **mha_spec_t** (p. 794) data pointers into corresponding **mha_audio_t** (p. 743) objects.

5.191.2 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.191.2.1 descriptor `mha_audio_descriptor_t mha_audio_t::descriptor`

Dimension and description of the data.

5.191.2.2 rdata `mha_real_t* mha_audio_t::rdata`

Data pointer if flag **mha_audio_descriptor_t::is_complex** (p. 742) is unset.

5.191.2.3 cdata `mha_complex_t* mha_audio_t::cdata`

Data pointer if flag **mha_audio_descriptor_t::is_complex** (p. 742) is set.

The documentation for this struct was generated from the following file:

- **mha.hh**

5.192 mha_channel_info_t Struct Reference

Channel information structure.

Public Attributes

- int **id**
  channel id
- char **idstr** [32]
  channel id
- unsigned int **side**
  side (left/right)
- **mha_direction_t dir**
  source direction
- **mha_real_t peaklevel**
  Peak level corresponds to this SPL (dB) level.

5.192.1 Detailed Description

Channel information structure.

© 2005-2021 HörTech gGmbH, Oldenburg

5.192.2 Member Data Documentation

5.192.2.1 id `int mha_channel_info_t::id`

channel id

5.192.2.2 idstr `char mha_channel_info_t::idstr[32]`

channel id

5.192.2.3 side `unsigned int mha_channel_info_t::side`

side (left/right)

5.192.2.4 dir `mha_direction_t mha_channel_info_t::dir`

source direction

5.192.2.5 peaklevel `mha_real_t mha_channel_info_t::peaklevel`

Peak level corresponds to this SPL (dB) level.

The documentation for this struct was generated from the following file:

- **mha.hh**

5.193 mha_complex_t Struct Reference

Type for complex floating point values.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **mha_real_t re**
  Real part.
- **mha_real_t im**
  Imaginary part.

5.193.1 Detailed Description

Type for complex floating point values.

5.193.2 Member Data Documentation

5.193.2.1 re `mha_real_t mha_complex_t::re`

Real part.

5.193.2.2 im `mha_real_t mha_complex_t::im`

Imaginary part.

The documentation for this struct was generated from the following file:

- **mha.hh**

5.194 mha_complex_test_array_t Struct Reference

Several places in MHA rely on the fact that you can cast an array of **mha_complex_t** (p. 745) c[] to an array of mha_real_t r[] with r[0] == c[0].re r[1] == c[0].im r[2] == c[1].re ...

Public Attributes

- **mha_complex_t c** [2]

© 2005-2021 HörTech gGmbH, Oldenburg

5.194.1 Detailed Description

Several places in MHA rely on the fact that you can cast an array of **mha_complex_t** (p. 745) c[] to an array of mha_real_t r[] with r[0] == c[0].re r[1] == c[0].im r[2] == c[1].re ...

Check these expectations in static asserts.

5.194.2 Member Data Documentation

5.194.2.1 c `mha_complex_t mha_complex_test_array_t::c[2]`

The documentation for this struct was generated from the following file:

- **mha.hh**

5.195 mha_dblbuf_t< FIFO > Class Template Reference

The doublebuffer adapts blocksizes between an outer process, which provides input data and takes output data, and an inner process, which processes the input signal and generates output data using a different block size than the outer process.

Public Types

- typedef FIFO::value_type **value_type**

  The datatype exchanged by the FIFO and this doublebuffer.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- virtual unsigned **get_inner_size** () const
- virtual unsigned **get_outer_size** () const
- virtual unsigned **get_delay** () const
- virtual unsigned **get_fifo_size** () const
- virtual unsigned **get_input_channels** () const
- virtual unsigned **get_output_channels** () const
- virtual unsigned **get_input_fifo_fill_count** () const
- virtual unsigned **get_output_fifo_fill_count** () const
- virtual unsigned **get_input_fifo_space** () const
- virtual unsigned **get_output_fifo_space** () const
- virtual **MHA_Error** ∗ **get_inner_error** () const
- virtual void **provoke_inner_error** (const **MHA_Error** &)
- virtual void **provoke_outer_error** (const **MHA_Error** &)
- **mha_dblbuf_t** (unsigned **outer_size**, unsigned **inner_size**, unsigned **delay**, unsigned **input_channels**, unsigned **output_channels**, const **value_type** &delay_data)

  Constructor creates FIFOs with specified delay.
- virtual ∼**mha_dblbuf_t** ()
- virtual void **process** (const **value_type** ∗input_signal, **value_type** ∗output_signal, unsigned count)

  The outer process has to call this method to propagate the input signal to the inner process, and receives back the output signal.
- virtual void **input** (**value_type** ∗input_signal)

  The inner process has to call this method to receive its input signal.
- virtual void **output** (const **value_type** ∗output_signal)

  The outer process has to call this method to deliver its output signal.

Private Attributes

- unsigned **outer_size**

  The block size used by the outer process.
- unsigned **inner_size**

  The block size used by the inner process.
- unsigned **delay**

  The delay introduced by bidirectional buffer size adaptation.
- unsigned **fifo_size**

  The size of each of the FIFOs.
- unsigned **input_channels**

  The number of input channels.
- unsigned **output_channels**

  The number of output channels.
- FIFO **input_fifo**

  The FIFO for transporting the input signal from the outer process to the inner process.
- FIFO **output_fifo**

© 2005-2021 HörTech gGmbH, Oldenburg

The FIFO for transporting the output signal from the inner process to the outer process.

- **MHA_Error * inner_error**

  Owned copy of exception to be thrown in inner thread.

- **MHA_Error * outer_error**

  Owned copy of exception to be thrown in outer thread.

5.195.1 Detailed Description

template<class FIFO>
class mha_dblbuf_t< FIFO >

The doublebuffer adapts blocksizes between an outer process, which provides input data and takes output data, and an inner process, which processes the input signal and generates output data using a different block size than the outer process.

This class introduces the channels concept. Input and output may have different channel counts.

5.195.2 Member Typedef Documentation

5.195.2.1 value_type `template<class FIFO >`

`typedef FIFO::value_type` **mha_dblbuf_t**`< FIFO >::` **value_type**

The datatype exchanged by the FIFO and this doublebuffer.

5.195.3 Constructor & Destructor Documentation

5.195.3.1 mha_dblbuf_t() `template<class FIFO >`

```
mha_dblbuf_t< FIFO >::  mha_dblbuf_t (
            unsigned outer_size,
            unsigned inner_size,
            unsigned delay,
            unsigned input_channels,
            unsigned output_channels,
            const  value_type & delay_data )
```

Constructor creates FIFOs with specified delay.

Warning

The doublebuffer may block or raise an exception if the delay is too small. To avoid this, the delay should be

$$delay >= (inner_size - gcd(inner_size, outer_size))$$

.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

outer_size	The block size used by the outer process.
inner_size	The block size used by the inner process.
delay	The total delay
input_channels	Number of input channels
output_channels	Number of output channels
delay_data	The delay consists of copies of this value.

5.195.3.2 ~mha_dblbuf_t() `template<class FIFO >`

`mha_dblbuf_t< FIFO >::~ mha_dblbuf_t [virtual]`

5.195.4 Member Function Documentation

5.195.4.1 get_inner_size() `template<class FIFO >`

`virtual unsigned mha_dblbuf_t< FIFO >::get_inner_size ( ) const [inline], [virtual]`

5.195.4.2 get_outer_size() `template<class FIFO >`

`virtual unsigned mha_dblbuf_t< FIFO >::get_outer_size ( ) const [inline], [virtual]`

5.195.4.3 get_delay() `template<class FIFO >`

`virtual unsigned mha_dblbuf_t< FIFO >::get_delay ( ) const [inline], [virtual]`

5.195.4.4 get_fifo_size() `template<class FIFO >`

`virtual unsigned mha_dblbuf_t< FIFO >::get_fifo_size ( ) const [inline], [virtual]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.195.4.5 get_input_channels() template<class FIFO >

virtual unsigned **mha_dblbuf_t**< FIFO >::get_input_channels ( ) const [inline], [virtual]

5.195.4.6 get_output_channels() template<class FIFO >

virtual unsigned **mha_dblbuf_t**< FIFO >::get_output_channels ( ) const [inline], [virtual]

5.195.4.7 get_input_fifo_fill_count() template<class FIFO >

virtual unsigned **mha_dblbuf_t**< FIFO >::get_input_fifo_fill_count ( ) const [inline], [virtual]

5.195.4.8 get_output_fifo_fill_count() template<class FIFO >

virtual unsigned **mha_dblbuf_t**< FIFO >::get_output_fifo_fill_count ( ) const [inline], [virtual]

5.195.4.9 get_input_fifo_space() template<class FIFO >

virtual unsigned **mha_dblbuf_t**< FIFO >::get_input_fifo_space ( ) const [inline], [virtual]

5.195.4.10 get_output_fifo_space() template<class FIFO >

virtual unsigned **mha_dblbuf_t**< FIFO >::get_output_fifo_space ( ) const [inline], [virtual]

5.195.4.11 get_inner_error() template<class FIFO >

virtual **MHA_Error*** **mha_dblbuf_t**< FIFO >::get_inner_error ( ) const [inline], [virtual]

© 2005-2021 HörTech gGmbH, Oldenburg

5.195.4.12 provoke_inner_error() template<class FIFO >

```
void mha_dblbuf_t< FIFO >::provoke_inner_error (
            const MHA_Error & error ) [virtual]
```

5.195.4.13 provoke_outer_error() template<class FIFO >

```
void mha_dblbuf_t< FIFO >::provoke_outer_error (
            const MHA_Error & error ) [virtual]
```

5.195.4.14 process() template<class FIFO >

```
void mha_dblbuf_t< FIFO >::process (
            const value_type * input_signal,
             value_type * output_signal,
            unsigned count ) [virtual]
```

The outer process has to call this method to propagate the input signal to the inner process, and receives back the output signal.

Parameters

input_signal	Pointer to the input signal array.
output_signal	Pointer to the output signal array.
count	The number of data instances provided and expected, lower or equal to inner_size given to constructor.

Exceptions

MHA_Error *(p. 764)*	When count is > outer_size as given to constructor or the underlying fifo implementation detects an error.

5.195.4.15 input() template<class FIFO >

```
void mha_dblbuf_t< FIFO >::input (
             value_type * input_signal ) [virtual]
```

The inner process has to call this method to receive its input signal.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

input_signal	Array where the doublebuffer can store the signal.

Exceptions

MHA_Error *(p. 764)*	When the underlying fifo implementation detects an error.

5.195.4.16 output()

```
template<class FIFO >
void mha_dblbuf_t< FIFO >::output (
            const value_type * output_signal ) [virtual]
```

The outer process has to call this method to deliver its output signal.

Parameters

output_signal	Array from which doublebuffer reads outputsignal.

Exceptions

MHA_Error *(p. 764)*	When the underlying fifo implementation detects an error.

5.195.5 Member Data Documentation

5.195.5.1 outer_size

```
template<class FIFO >
unsigned mha_dblbuf_t< FIFO >::outer_size [private]
```

The block size used by the outer process.

5.195.5.2 inner_size

```
template<class FIFO >
unsigned mha_dblbuf_t< FIFO >::inner_size [private]
```

The block size used by the inner process.

© 2005-2021 HörTech gGmbH, Oldenburg

5.195.5.3 delay `template<class FIFO >`

`unsigned` **`mha_dblbuf_t`**`< FIFO >::delay [private]`

The delay introduced by bidirectional buffer size adaptation.

5.195.5.4 fifo_size `template<class FIFO >`

`unsigned` **`mha_dblbuf_t`**`< FIFO >::fifo_size [private]`

The size of each of the FIFOs.

5.195.5.5 input_channels `template<class FIFO >`

`unsigned` **`mha_dblbuf_t`**`< FIFO >::input_channels [private]`

The number of input channels.

5.195.5.6 output_channels `template<class FIFO >`

`unsigned` **`mha_dblbuf_t`**`< FIFO >::output_channels [private]`

The number of output channels.

5.195.5.7 input_fifo `template<class FIFO >`

`FIFO` **`mha_dblbuf_t`**`< FIFO >::input_fifo [private]`

The FIFO for transporting the input signal from the outer process to the inner process.

5.195.5.8 output_fifo `template<class FIFO >`

`FIFO` **`mha_dblbuf_t`**`< FIFO >::output_fifo [private]`

The FIFO for transporting the output signal from the inner process to the outer process.

© 2005-2021 HörTech gGmbH, Oldenburg

5.195.5.9 inner_error `template<class FIFO >`

`MHA_Error* mha_dblbuf_t< FIFO >::inner_error [private]`

Owned copy of exception to be thrown in inner thread.

5.195.5.10 outer_error `template<class FIFO >`

`MHA_Error* mha_dblbuf_t< FIFO >::outer_error [private]`

Owned copy of exception to be thrown in outer thread.

The documentation for this class was generated from the following files:

- **mha_fifo.h**
- **mha_fifo.cpp**

5.196 mha_direction_t Struct Reference

Channel source direction structure.

Public Attributes

- **mha_real_t azimuth**
  azimuth in radiants
- **mha_real_t elevation**
  elevation in radiants
- **mha_real_t distance**
  distance in meters

5.196.1 Detailed Description

Channel source direction structure.

5.196.2 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.196.2.1 azimuth `mha_real_t mha_direction_t::azimuth`

azimuth in radiants

5.196.2.2 elevation `mha_real_t mha_direction_t::elevation`

elevation in radiants

5.196.2.3 distance `mha_real_t mha_direction_t::distance`

distance in meters

The documentation for this struct was generated from the following file:

- **mha.hh**

5.197 mha_drifter_fifo_t< T > Class Template Reference

A FIFO class for blocksize adaptation without Synchronization.

Inheritance diagram for mha_drifter_fifo_t< T >:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- virtual void **write** (const T *data, unsigned count)

  write data to fifo
- virtual void **read** (T * **buf**, unsigned count)

  Read data from fifo.
- virtual unsigned **get_fill_count** () const

  *Return fill_count, adding **mha_drifter_fifo_t<T>::startup_zeros** (p. 764) to the number of samples actually in the fifo's buffer.*
- virtual unsigned **get_available_space** () const

  *Return available space, subtracting number of **mha_drifter_fifo_t<T>::startup_zeros** (p. 764) from the available_space actually present in the fifo's buffer.*
- virtual unsigned **get_des_fill_count** () const

  The desired fill count of this fifo.
- virtual unsigned **get_min_fill_count** () const

  The minimum fill count of this fifo.
- virtual void **stop** ()

  *Called by **mha_drifter_fifo_t<T>::read** (p. 759) or **mha_drifter_fifo_t<T>::write** (p. 759) when their xrun in succession counter exceeds its limit.*
- virtual void **starting** ()

  *Called by **mha_drifter_fifo_t<T>::read** (p. 759) or **mha_drifter_fifo_t<T>::write** (p. 759) when the respective flag (**mha_drifter_fifo_t<T>::reader_started** (p. 762) or **mha_drifter_fifo_t<T>::writer_started** (p. 762)) is about to be toggled from false to true.*
- **mha_drifter_fifo_t** (unsigned min_fill_count, unsigned **desired_fill_count**, unsigned max_fill_count)

  Create drifter FIFO.
- **mha_drifter_fifo_t** (unsigned min_fill_count, unsigned **desired_fill_count**, unsigned max_fill_count, const T &t)

  Create drifter FIFO where all (initially unused) copies of T are initialized as copies of t.

Private Attributes

- const unsigned **minimum_fill_count**

  The minimum fill count of this fifo.
- const unsigned **desired_fill_count**

  The desired fill count of the fifo.
- bool **writer_started**

  Flag set to true when write is called the first time.
- bool **reader_started**

  Flag set to true when read is called for the first time.
- unsigned **writer_xruns_total**

  The number of xruns seen by the writer since object instantiation.
- unsigned **reader_xruns_total**

  The number of xruns seen by the reader since object instantiation.
- unsigned **writer_xruns_since_start**

© 2005-2021 HörTech gGmbH, Oldenburg

The number of xruns seen by the writer since the last start of processing.

- unsigned **reader_xruns_since_start**

  The number of xruns seen by the reader since the last start of processing.
- unsigned **writer_xruns_in_succession**

  The number of xruns seen by the writer in succession.
- unsigned **reader_xruns_in_succession**

  The number of xruns seen by the reader in succession.
- unsigned **maximum_writer_xruns_in_succession_before_stop**

  A limit to the number of xruns seen in succession during write before the data transmission through the FIFO is stopped.
- unsigned **maximum_reader_xruns_in_succession_before_stop**

  A limit to the number of xruns seen in succession during read before the data transmission through the FIFO is stopped.
- **mha_fifo_t**< T >:: **value_type null_data**

  The value used in place of missing data.
- unsigned **startup_zeros**

  *When processing starts, that is when both **mha_drifter_fifo_t<T>::reader_started** (p. 762) and **mha_drifter_fifo_t<T>::writer_started** (p. 762) are true, then first **mha_drifter_fifo_t<T>::desired_fill_count** (p. 762) instances of **mha_drifter_fifo_t<T>::null_data** (p. 763) are delivered to the reader.*

Additional Inherited Members

5.197.1 Detailed Description

template<class T>
class mha_drifter_fifo_t< T >

A FIFO class for blocksize adaptation without Synchronization.

Features: delay concept (desired, minimum and maximum delay), drifting support by throwing away data or inserting zeroes.

5.197.2 Constructor & Destructor Documentation

5.197.2.1 mha_drifter_fifo_t() [1/2] template<class T >

```
mha_drifter_fifo_t< T >::  mha_drifter_fifo_t (
            unsigned min_fill_count,
            unsigned desired_fill_count,
            unsigned max_fill_count )
```

Create drifter FIFO.

© 2005-2021 HörTech gGmbH, Oldenburg

5.197.2.2 mha_drifter_fifo_t() [2/2] template<class T >

```
mha_drifter_fifo_t< T >::  mha_drifter_fifo_t (
            unsigned min_fill_count,
            unsigned desired_fill_count,
            unsigned max_fill_count,
            const T & t )
```

Create drifter FIFO where all (initially unused) copies of T are initialized as copies of t.

5.197.3 Member Function Documentation

5.197.3.1 write() template<class T >

```
void  mha_drifter_fifo_t< T >::write (
            const T * data,
            unsigned count )  [virtual]
```

write data to fifo

Sets **writer_started** (p. 762) to true.

When processing has started, i.e. both **reader_started** (p. 762) and **writer_started** (p. 762) are true, write specified ammount of data to the fifo. If there is not enough space available, then the exceeding data is lost and the writer xrun counters are increased.

Processing is stopped when **writer_xruns_in_succession** (p. 763) exceeds **maximum_writer_xruns_in_succession_before_stop** (p. 763).

Parameters

data	Pointer to source data.
count	Number of instances to copy

Reimplemented from **mha_fifo_t< T >** (p. 779).

5.197.3.2 read() template<class T >

```
void  mha_drifter_fifo_t< T >::read (
            T * buf,
            unsigned count )  [virtual]
```

© 2005-2021 HörTech gGmbH, Oldenburg

Read data from fifo.

Sets **reader_started** (p. 762) to true.

When processing has started, i.e. both **reader_started** (p. 762) and **writer_started** (p. 762) are true, then read specified ammount of data from the fifo. As long as **startup_zeros** (p. 764) is > 0, **null_data** (p. 763) is delivered to the reader and **startup_zeros** (p. 764) is diminished. Only when **startup_zeros** (p. 764) has reached 0, data is actually read from the fifo's buffer.

If the read would cause the fifo's fill count to drop below **minimum_fill_count** (p. 761), then only so much data are read that **minimum_fill_count** (p. 761) entries remain in the fifo, the missing data is replaced with **null_data** (p. 763), and the reader xrun counters are increased.

Processing is stopped when **reader_xruns_in_succession** (p. 763) exceeds **maximum_reader_xruns_in_succession_before_stop** (p. 763).

Parameters

buf	Pointer to the target buffer
count	Number of instances to copy

Reimplemented from **mha_fifo_t< T >** (p. 780).

5.197.3.3 get_fill_count() `template<class T >`

`unsigned` **`mha_drifter_fifo_t`**`< T >::get_fill_count [virtual]`

Return fill_count, adding **mha_drifter_fifo_t<T>::startup_zeros** (p. 764) to the number of samples actually in the fifo's buffer.

Reimplemented from **mha_fifo_t< T >** (p. 780).

5.197.3.4 get_available_space() `template<class T >`

`unsigned` **`mha_drifter_fifo_t`**`< T >::get_available_space [virtual]`

Return available space, subtracting number of **mha_drifter_fifo_t<T>::startup_zeros** (p. 764) from the available_space actually present in the fifo's buffer.

TODO: uncertain if this is a good idea.

Reimplemented from **mha_fifo_t< T >** (p. 780).

© 2005-2021 HörTech gGmbH, Oldenburg

5.197.3.5 get_des_fill_count() template<class T >

```
virtual unsigned mha_drifter_fifo_t< T >::get_des_fill_count ( ) const [inline],
[virtual]
```

The desired fill count of this fifo.

5.197.3.6 get_min_fill_count() template<class T >

```
virtual unsigned mha_drifter_fifo_t< T >::get_min_fill_count ( ) const [inline],
[virtual]
```

The minimum fill count of this fifo.

5.197.3.7 stop() template<class T >

```
void mha_drifter_fifo_t< T >::stop [virtual]
```

Called by **mha_drifter_fifo_t<T>::read** (p. 759) or **mha_drifter_fifo_t<T>::write** (p. 759) when their xrun in succession counter exceeds its limit.

Called by **read** (p. 759) or **write** (p. 759) when their xrun in succession counter exceeds its limit.

May also be called explicitly.

5.197.3.8 starting() template<class T >

```
void mha_drifter_fifo_t< T >::starting [virtual]
```

Called by **mha_drifter_fifo_t<T>::read** (p. 759) or **mha_drifter_fifo_t<T>::write** (p. 759) when the respective flag (**mha_drifter_fifo_t<T>::reader_started** (p. 762) or **mha_drifter_↩ fifo_t<T>::writer_started** (p. 762)) is about to be toggled from false to true.

The fifo's buffer is emptied, this method resets **startup_zeros** (p. 764) to **desired_fill_count** (p. 762), and it also resets **reader_xruns_since_start** (p. 763) and **writer_xruns_since_start** (p. 762) to 0.

5.197.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.197.4.1 minimum_fill_count `template<class T >`

`const unsigned` **`mha_drifter_fifo_t`**`< T >::minimum_fill_count [private]`

The minimum fill count of this fifo.

5.197.4.2 desired_fill_count `template<class T >`

`const unsigned` **`mha_drifter_fifo_t`**`< T >::desired_fill_count [private]`

The desired fill count of the fifo.

The fifo is initialized with this ammount of data when data transmission starts.

5.197.4.3 writer_started `template<class T >`

`bool` **`mha_drifter_fifo_t`**`< T >::writer_started [private]`

Flag set to true when write is called the first time.

5.197.4.4 reader_started `template<class T >`

`bool` **`mha_drifter_fifo_t`**`< T >::reader_started [private]`

Flag set to true when read is called for the first time.

5.197.4.5 writer_xruns_total `template<class T >`

`unsigned` **`mha_drifter_fifo_t`**`< T >::writer_xruns_total [private]`

The number of xruns seen by the writer since object instantiation.

5.197.4.6 reader_xruns_total `template<class T >`

`unsigned` **`mha_drifter_fifo_t`**`< T >::reader_xruns_total [private]`

The number of xruns seen by the reader since object instantiation.

© 2005-2021 HörTech gGmbH, Oldenburg

5.197.4.7 writer_xruns_since_start

`template<class T >`
`unsigned` **`mha_drifter_fifo_t`**`< T >::writer_xruns_since_start [private]`

The number of xruns seen by the writer since the last start of processing.

5.197.4.8 reader_xruns_since_start

`template<class T >`
`unsigned` **`mha_drifter_fifo_t`**`< T >::reader_xruns_since_start [private]`

The number of xruns seen by the reader since the last start of processing.

5.197.4.9 writer_xruns_in_succession

`template<class T >`
`unsigned` **`mha_drifter_fifo_t`**`< T >::writer_xruns_in_succession [private]`

The number of xruns seen by the writer in succession.

Reset to 0 every time a write succeeds without xrun.

5.197.4.10 reader_xruns_in_succession

`template<class T >`
`unsigned` **`mha_drifter_fifo_t`**`< T >::reader_xruns_in_succession [private]`

The number of xruns seen by the reader in succession.

Reset to 0 every time a read succeeds without xrun.

5.197.4.11 maximum_writer_xruns_in_succession_before_stop

`template<class T >`
`unsigned` **`mha_drifter_fifo_t`**`< T >::maximum_writer_xruns_in_succession_before_stop [private]`

A limit to the number of xruns seen in succession during write before the data transmission through the FIFO is stopped.

5.197.4.12 maximum_reader_xruns_in_succession_before_stop

`template<class T >`
`unsigned` **`mha_drifter_fifo_t`**`< T >::maximum_reader_xruns_in_succession_before_stop [private]`

A limit to the number of xruns seen in succession during read before the data transmission through the FIFO is stopped.

© 2005-2021 HörTech gGmbH, Oldenburg

5.197.4.13 null_data template<class T >

mha_fifo_t<T>:: **value_type** **mha_drifter_fifo_t**< T >::null_data [private]

The value used in place of missing data.

5.197.4.14 startup_zeros template<class T >

unsigned **mha_drifter_fifo_t**< T >::startup_zeros [private]

When processing starts, that is when both **mha_drifter_fifo_t**<**T**>**::reader_started** (p. 762) and **mha_drifter_fifo_t**<**T**>**::writer_started** (p. 762) are true, then first **mha_drifter_fifo_t**<↩ **T**>**::desired_fill_count** (p. 762) instances of **mha_drifter_fifo_t**<**T**>**::null_data** (p. 763) are delivered to the reader.

These **null_data** (p. 763) instances are not transmitted through the fifo because filling the fifo with enough **null_data** (p. 763) might not be realtime safe and this filling has to be initiated by **starting** (p. 761) or **stop** (p. 761) (this implementation: **starting** (p. 761)) which are be called with realtime constraints.

The documentation for this class was generated from the following file:

- **mha_fifo.h**

5.198 MHA_Error Class Reference

Error reporting exception class.

Inherits exception.

Public Member Functions

- **MHA_Error** (const char ∗file, int line, const char ∗fmt,...) __attribute__((__format__(printf
  *Create an instance of a **MHA_Error** (p. 764).*
- **MHA_Error** (const **MHA_Error** &)
- **MHA_Error** & **operator=** (const **MHA_Error** &)
- ∼**MHA_Error** () throw ()
- const char ∗ **get_msg** () const
  Return the error message without source position.
- const char ∗ **get_longmsg** () const
  Return the error message with source position.
- const char ∗ **what** () const throw ()
  overwrite std::execption::what()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- char * **msg**
- char * **longmsg**

5.198.1 Detailed Description

Error reporting exception class.

This class is used for error handling in the openMHA. It is used by the openMHA kernel and by the openMHA toolbox library. Please note that exceptions should not be used accross ANSI-C interfaces. It is necessary to catch exceptions within the library.

The **MHA_Error** (p. 764) class holds source file name, line number and an error message.

5.198.2 Constructor & Destructor Documentation

5.198.2.1 MHA_Error() [1/2]

```
MHA_Error::MHA_Error (
    const char * s_file,
    int l,
    const char * fmt,
    ... )
```

Create an instance of a **MHA_Error** (p. 764).

Parameters

s_file	source file name (**FILE**)
l	source line (**LINE**)
fmt	format string for error message (as in printf)

5.198.2.2 MHA_Error() [2/2]

```
MHA_Error::MHA_Error (
    const MHA_Error & p )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.198.2.3 ~MHA_Error() `MHA_Error::~MHA_Error ( ) throw ( )`

5.198.3 Member Function Documentation

5.198.3.1 operator=() `MHA_Error & MHA_Error::operator= ( const MHA_Error & p )`

5.198.3.2 get_msg() `const char* MHA_Error::get_msg ( ) const [inline]`

Return the error message without source position.

5.198.3.3 get_longmsg() `const char* MHA_Error::get_longmsg ( ) const [inline]`

Return the error message with source position.

5.198.3.4 what() `const char* MHA_Error::what ( ) const throw ( ) [inline]`

overwrite std::execption::what()

5.198.4 Member Data Documentation

5.198.4.1 msg `char* MHA_Error::msg [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.198.4.2 longmsg `char* MHA_Error::longmsg [private]`

The documentation for this class was generated from the following files:

- **mha_error.hh**
- **mha_error.cpp**

5.199 mha_fifo_lf_t< T > Class Template Reference

A lock-free FIFO class for transferring data from a producer thread to a consumer thread.

Inheritance diagram for mha_fifo_lf_t< T >:

Public Member Functions

- virtual void **write** (const T *data, unsigned count) override
  Write specified ammount of data to the fifo.
- virtual void **read** (T *outbuf, unsigned count) override
  Read data from fifo.
- virtual unsigned **get_fill_count** () const override
  get_fill_count() *(p. 769) must only be called by the reader thread*
- virtual unsigned **get_available_space** () const override
  get_available_space() *(p. 769) must only be called by the writer thread*
- **mha_fifo_lf_t** (unsigned max_fill_count, const T &t=T())
  Create FIFO with fixed buffer size.

Private Attributes

- std::atomic< const T * > **atomic_write_ptr**
  atomic copy of the write_ptr, only modified by the producer thread
- std::atomic< const T * > **atomic_read_ptr**
  atomic copy of the read ptr, only modified by the consumer thread

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.199.1 Detailed Description

template<class T>
class mha_fifo_lf_t< T >

A lock-free FIFO class for transferring data from a producer thread to a consumer thread.

Inherits basic functionality from **mha_fifo_t** (p. 776), adds release-acquire semantics to ensure consumer that the fill count or free space deduced from read and write pointers is consistent with the actual data. Copying, moving, and assignment of FIFO are forbidden by base class.

5.199.2 Constructor & Destructor Documentation

5.199.2.1 mha_fifo_lf_t() template<class T >

```
mha_fifo_lf_t< T >::  mha_fifo_lf_t (
            unsigned max_fill_count,
            const T & t = T() )  [inline], [explicit]
```

Create FIFO with fixed buffer size.

All (initially unused) instances of T are initialized as copies of t

5.199.3 Member Function Documentation

5.199.3.1 write() template<class T >

```
virtual void  mha_fifo_lf_t< T >::write (
            const T * data,
            unsigned count )  [inline], [override], [virtual]
```

Write specified ammount of data to the fifo.

Must only be called by the writer thread.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

data	Pointer to source data.
count	Number of instances to copy.

Exceptions

MHA_Error *(p. 764)*	when there is not enough space available.

Reimplemented from **mha_fifo_t< T >** (p. 779).

5.199.3.2 read()

```
template<class T >
virtual void mha_fifo_lf_t< T >::read (
            T * outbuf,
            unsigned count ) [inline], [override], [virtual]
```

Read data from fifo.

Must only be called by the reader thread.

Parameters

outbuf	Pointer to the target buffer.
count	Number of instances to copy.

Exceptions

MHA_Error *(p. 764)*	when there is not enough data available.

Reimplemented from **mha_fifo_t< T >** (p. 780).

5.199.3.3 get_fill_count()

```
template<class T >
virtual unsigned mha_fifo_lf_t< T >::get_fill_count ( ) const [inline], [override],
[virtual]
```

get_fill_count() (p. 769) must only be called by the reader thread

Reimplemented from **mha_fifo_t< T >** (p. 780).

© 2005-2021 HörTech gGmbH, Oldenburg

5.199.3.4 get_available_space() `template<class T >`

`virtual unsigned` **`mha_fifo_lf_t`**`< T >::get_available_space ( ) const [inline], [override], [virtual]`

get_available_space() (p. 769) must only be called by the writer thread

Reimplemented from **mha_fifo_t< T >** (p. 780).

5.199.4 Member Data Documentation

5.199.4.1 atomic_write_ptr `template<class T >`

`std::atomic<const T *>` **`mha_fifo_lf_t`**`< T >::atomic_write_ptr [private]`

atomic copy of the write_ptr, only modified by the producer thread

5.199.4.2 atomic_read_ptr `template<class T >`

`std::atomic<const T *>` **`mha_fifo_lf_t`**`< T >::atomic_read_ptr [private]`

atomic copy of the read ptr, only modified by the consumer thread

The documentation for this class was generated from the following file:

- **mha_fifo.h**

5.200 mha_fifo_lw_t< T > Class Template Reference

This FIFO uses locks to synchronize access.

Inheritance diagram for mha_fifo_lw_t< T >:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- virtual void **write** (const T *data, unsigned count)
  write specified ammount of data to the fifo.
- virtual void **read** (T * **buf**, unsigned count)
  read data from fifo.
- **mha_fifo_lw_t** (unsigned max_fill_count)
  Create FIFO with fixed buffer size.
- virtual ~**mha_fifo_lw_t** ()
  release synchronization object
- virtual void **set_error** (unsigned index, **MHA_Error** * **error**)
  Process waiting for more data or space should bail out, throwing this error.

Private Attributes

- **mha_fifo_thread_platform_t** * **sync**
  platform specific thread synchronization
- **MHA_Error** * **error** [2]
  If waiting for synchronization should be aborted then exception to be thrown by reader process (index 0) or writer process (index 1) has to be placed here.

Additional Inherited Members

5.200.1 Detailed Description

template<class T>
class mha_fifo_lw_t< T >

This FIFO uses locks to synchronize access.

Reading and writing can block until the operation can be executed.

5.200.2 Constructor & Destructor Documentation

5.200.2.1 mha_fifo_lw_t() `template<class T >`

```
mha_fifo_lw_t< T >::  mha_fifo_lw_t (
            unsigned max_fill_count ) [explicit]
```

Create FIFO with fixed buffer size.

© 2005-2021 HörTech gGmbH, Oldenburg

5.200.2.2 ~mha_fifo_lw_t() template<class T >

mha_fifo_lw_t< T >::~ **mha_fifo_lw_t** [virtual]

release synchronization object

5.200.3 Member Function Documentation

5.200.3.1 write() template<class T >

```
void mha_fifo_lw_t< T >::write (
            const T * data,
            unsigned count ) [virtual]
```

write specified ammount of data to the fifo.

If there is not enough space, then wait for more space.

Parameters

data	Pointer to source data.
count	Number of instances to copy.

Exceptions

MHA_Error *(p. 764)*	when detecting a deadlock situation.

Reimplemented from **mha_fifo_t< T >** (p. 779).

5.200.3.2 read() template<class T >

```
void mha_fifo_lw_t< T >::read (
            T * buf,
            unsigned count ) [virtual]
```

read data from fifo.

If there is not enough data, then wait for more data.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

buf	Pointer to the target buffer.
count	Number of instances to copy.

Exceptions

MHA_Error *(p. 764)*	when detecting a deadlock situation.

Reimplemented from **mha_fifo_t< T >** (p. 780).

5.200.3.3 set_error()

```
template<class T >
void mha_fifo_lw_t< T >::set_error (
            unsigned index,
             MHA_Error * error ) [virtual]
```

Process waiting for more data or space should bail out, throwing this error.

Parameters

index	Use 0 for terminating reader, 1 for terminating writer.
error	**MHA_Error** (p. 764) to be thrown

5.200.4 Member Data Documentation

5.200.4.1 sync

```
template<class T >
mha_fifo_thread_platform_t* mha_fifo_lw_t< T >::sync [private]
```

platform specific thread synchronization

5.200.4.2 error

```
template<class T >
MHA_Error* mha_fifo_lw_t< T >::error[2] [private]
```

If waiting for synchronization should be aborted then exception to be thrown by reader process (index 0) or writer process (index 1) has to be placed here.

The documentation for this class was generated from the following files:

- **mha_fifo.h**
- **mha_fifo.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.201 mha_fifo_posix_threads_t Class Reference

Inheritance diagram for mha_fifo_posix_threads_t:

Public Member Functions

- **mha_fifo_posix_threads_t** ()
- virtual void **aquire_mutex** ()
- virtual void **release_mutex** ()
- virtual void **wait_for_decrease** ()
- virtual void **wait_for_increase** ()
- virtual void **increment** ()
- virtual void **decrement** ()
- virtual ~**mha_fifo_posix_threads_t** ()

Private Attributes

- pthread_mutex_t **mutex**
- pthread_cond_t **decrease_condition**
- pthread_cond_t **increase_condition**

5.201.1 Constructor & Destructor Documentation

5.201.1.1 mha_fifo_posix_threads_t() `mha_fifo_posix_threads_t::mha_fifo_posix_threads_t ( ) [inline]`

5.201.1.2 ~mha_fifo_posix_threads_t() `virtual mha_fifo_posix_threads_t::~mha_fifo_posix_threads_t ( ) [inline], [virtual]`

5.201.2 Member Function Documentation

5.201.2.1 aquire_mutex() `virtual void mha_fifo_posix_threads_t::aquire_mutex ( ) [inline], [virtual]`

Implements **mha_fifo_thread_platform_t** (p. 786).

5.201.2.2 release_mutex() `virtual void mha_fifo_posix_threads_t::release_mutex ( ) [inline], [virtual]`

Implements **mha_fifo_thread_platform_t** (p. 786).

5.201.2.3 wait_for_decrease() `virtual void mha_fifo_posix_threads_t::wait_for_decrease ( ) [inline], [virtual]`

Implements **mha_fifo_thread_platform_t** (p. 786).

5.201.2.4 wait_for_increase() `virtual void mha_fifo_posix_threads_t::wait_for_increase ( ) [inline], [virtual]`

Implements **mha_fifo_thread_platform_t** (p. 787).

5.201.2.5 increment() `virtual void mha_fifo_posix_threads_t::increment ( ) [inline], [virtual]`

Implements **mha_fifo_thread_platform_t** (p. 787).

© 2005-2021 HörTech gGmbH, Oldenburg

5.201.2.6 decrement() `virtual void mha_fifo_posix_threads_t::decrement ( ) [inline], [virtual]`

Implements **mha_fifo_thread_platform_t** (p. 787).

5.201.3 Member Data Documentation

5.201.3.1 mutex `pthread_mutex_t mha_fifo_posix_threads_t::mutex [private]`

5.201.3.2 decrease_condition `pthread_cond_t mha_fifo_posix_threads_t::decrease_condition [private]`

5.201.3.3 increase_condition `pthread_cond_t mha_fifo_posix_threads_t::increase_condition [private]`

The documentation for this class was generated from the following file:

- **mha_fifo.h**

5.202 mha_fifo_t< T > Class Template Reference

A FIFO class.

Inheritance diagram for mha_fifo_t< T >:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Types

- typedef std::vector< T >:: **value_type value_type**
  The data type exchanged by this fifo.

Public Member Functions

- virtual void **write** (const T *data, unsigned count)
  write specified ammount of data to the fifo.
- virtual void **read** (T *outbuf, unsigned count)
  read data from fifo
- virtual unsigned **get_fill_count** () const
  Read-only access to fill_count.
- virtual unsigned **get_available_space** () const
  Read-only access to available_space.
- virtual unsigned **get_max_fill_count** () const
  The capacity of this fifo.
- **mha_fifo_t** (unsigned max_fill_count, const T &t=T())
  Create FIFO with fixed buffer size, where all (initially unused) instances of T are initialized as copies of t.
- virtual ~**mha_fifo_t** ()=default
  Make destructor virtual.
- **mha_fifo_t** (const **mha_fifo_t** &)=delete
  Copy constructor.
- **mha_fifo_t** (**mha_fifo_t** &&)=delete
  Move constructor.
- **mha_fifo_t**< T > & **operator=** (const **mha_fifo_t**< T > &)=delete
  Assignment operator.
- **mha_fifo_t**< T > & **operator=** (**mha_fifo_t**< T > &&)=delete
  Move assignment operator.

Protected Member Functions

- void **clear** ()
  Empty the fifo at once.
- const T * **get_write_ptr** () const
  read-only access to the write pointer for derived classes
- const T * **get_read_ptr** () const
  read-only access to read pointer for derived classes
- unsigned **get_fill_count** (const T *wp, const T *rp) const
  Compute fill count from given write pointer and read pointer.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- std::vector< T > **buf**
  The memory allocated to store the data in the fifo.
- T * **write_ptr**
  points to location where to write next
- const T * **read_ptr**
  points to location where to read next

5.202.1 Detailed Description

template<class T>
class mha_fifo_t< T >

A FIFO class.

Synchronization: None. Use external synchronisation or synchronization in inheriting class. Assignment, copy and move constructors are disabled.

5.202.2 Member Typedef Documentation

5.202.2.1 value_type `template<class T >`

`typedef std::vector<T>::` **`value_type`** **`mha_fifo_t`**`< T >::` **`value_type`**

The data type exchanged by this fifo.

5.202.3 Constructor & Destructor Documentation

5.202.3.1 mha_fifo_t() [1/3] `template<class T >`

```
mha_fifo_t< T >::  mha_fifo_t (
            unsigned max_fill_count,
            const T & t = T() )  [explicit]
```

Create FIFO with fixed buffer size, where all (initially unused) instances of T are initialized as copies of t.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

max_fill_count	The maximum number of instances of T that can be held at the same time inside the fifo.
The	fifo allocates a vector of max_fill_count+1 instances of T for storage, one of which is always unused.

5.202.3.2 ~mha_fifo_t() `template<class T >`

```
virtual mha_fifo_t< T >::~ mha_fifo_t ( ) [virtual], [default]
```

Make destructor virtual.

5.202.3.3 mha_fifo_t() [2/3] `template<class T >`

```
mha_fifo_t< T >:: mha_fifo_t (
            const mha_fifo_t< T > & ) [delete]
```

Copy constructor.

5.202.3.4 mha_fifo_t() [3/3] `template<class T >`

```
mha_fifo_t< T >:: mha_fifo_t (
             mha_fifo_t< T > && ) [delete]
```

Move constructor.

5.202.4 Member Function Documentation

5.202.4.1 write() `template<class T >`

```
void mha_fifo_t< T >::write (
            const T * data,
            unsigned count ) [virtual]
```

write specified ammount of data to the fifo.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

data	Pointer to source data.
count	Number of instances to copy

Exceptions

MHA_Error *(p. 764)*	when there is not enough space available.

Reimplemented in **mha_fifo_lf_t< T >** (p. 768), **mha_fifo_lw_t< T >** (p. 772), **mha_drifter_fifo_t< T >** (p. 759), **mha_fifo_lf_t< mha_real_t >** (p. 768), **mha_fifo_lw_t< mha_real_t >** (p. 772), and **mha_fifo_lf_t< MHA_AC::acspace2matrix_t >** (p. 768).

5.202.4.2 read()

```
template<class T >
void mha_fifo_t< T >::read (
            T * outbuf,
            unsigned count ) [virtual]
```

read data from fifo

Parameters

outbuf	Pointer to the target buffer.
count	Number of instances to copy.

Exceptions

MHA_Error *(p. 764)*	when there is not enough data available.

Reimplemented in **mha_fifo_lf_t< T >** (p. 769), **mha_fifo_lw_t< T >** (p. 772), **mha_drifter_fifo_t< T >** (p. 759), **mha_fifo_lf_t< mha_real_t >** (p. 769), **mha_fifo_lw_t< mha_real_t >** (p. 772), and **mha_fifo_lf_t< MHA_AC::acspace2matrix_t >** (p. 769).

5.202.4.3 get_fill_count() [1/2]

```
template<class T >
virtual unsigned mha_fifo_t< T >::get_fill_count ( ) const [inline], [virtual]
```

Read-only access to fill_count.

Reimplemented in **mha_fifo_lf_t< T >** (p. 769), **mha_fifo_lf_t< mha_real_t >** (p. 769), **mha_fifo_lf_t< MHA_AC::acspace2matrix_t >** (p. 769), and **mha_drifter_fifo_t< T >** (p. 760).

© 2005-2021 HörTech gGmbH, Oldenburg

5.202.4.4 get_available_space() template<class T >

```
unsigned mha_fifo_t< T >::get_available_space [virtual]
```

Read-only access to available_space.

Reimplemented in **mha_fifo_lf_t< T >** (p. 769), **mha_fifo_lf_t< mha_real_t >** (p. 769), **mha_fifo_lf_t< MHA_AC::acspace2matrix_t >** (p. 769), and **mha_drifter_fifo_t< T >** (p. 760).

5.202.4.5 get_max_fill_count() template<class T >

```
virtual unsigned mha_fifo_t< T >::get_max_fill_count ( ) const [inline], [virtual]
```

The capacity of this fifo.

5.202.4.6 operator=() [1/2] template<class T >

```
mha_fifo_t<T>& mha_fifo_t< T >::operator= (
            const mha_fifo_t< T > & ) [delete]
```

Assignment operator.

5.202.4.7 operator=() [2/2] template<class T >

```
mha_fifo_t<T>& mha_fifo_t< T >::operator= (
             mha_fifo_t< T > && ) [delete]
```

Move assignment operator.

5.202.4.8 clear() template<class T >

```
void mha_fifo_t< T >::clear ( ) [inline], [protected]
```

Empty the fifo at once.

Should be called by the reader, or when the reader is inactive.

© 2005-2021 HörTech gGmbH, Oldenburg

5.202.4.9 get_write_ptr() template<class T >

```
const T* mha_fifo_t< T >::get_write_ptr ( ) const [inline], [protected]
```

read-only access to the write pointer for derived classes

5.202.4.10 get_read_ptr() template<class T >

```
const T* mha_fifo_t< T >::get_read_ptr ( ) const [inline], [protected]
```

read-only access to read pointer for derived classes

5.202.4.11 get_fill_count() [2/2] template<class T >

```
unsigned mha_fifo_t< T >::get_fill_count (
            const T * wp,
            const T * rp ) const [inline], [protected]
```

Compute fill count from given write pointer and read pointer.

Parameters

wp	Write pointer.
rp	Read pointer.

Precondition

wp and rp must point to an instance of T inside buf.

Returns

Number of elements that can be read from the fifo when wp and rp have the given values.

5.202.5 Member Data Documentation

5.202.5.1 buf template<class T >

```
std::vector<T> mha_fifo_t< T >::buf [private]
```

The memory allocated to store the data in the fifo.

At least one location in buf is always unused, because we have max_fill_count + 1 possible fillcounts [0:max_fill_count] that we need to distinguish.

© 2005-2021 HörTech gGmbH, Oldenburg

5.202.5.2 write_ptr `template<class T >`

`T* mha_fifo_t< T >::write_ptr [private]`

points to location where to write next

5.202.5.3 read_ptr `template<class T >`

`const T* mha_fifo_t< T >::read_ptr [private]`

points to location where to read next

The documentation for this class was generated from the following file:

- **mha_fifo.h**

5.203 mha_fifo_thread_guard_t Class Reference

Simple Mutex Guard Class.

Public Member Functions

- **mha_fifo_thread_guard_t** (**mha_fifo_thread_platform_t** ∗ **sync**)
- ∼**mha_fifo_thread_guard_t** ()

Private Attributes

- **mha_fifo_thread_platform_t** ∗ **sync**

5.203.1 Detailed Description

Simple Mutex Guard Class.

5.203.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.203.2.1 mha_fifo_thread_guard_t() `mha_fifo_thread_guard_t::mha_fifo_thread_guard_t (`

`mha_fifo_thread_platform_t * sync ) [inline]`

5.203.2.2 ~mha_fifo_thread_guard_t() `mha_fifo_thread_guard_t::~mha_fifo_thread_guard_t ( ) [inline]`

5.203.3 Member Data Documentation

5.203.3.1 sync `mha_fifo_thread_platform_t* mha_fifo_thread_guard_t::sync [private]`

The documentation for this class was generated from the following file:

- **mha_fifo.h**

5.204 mha_fifo_thread_platform_t Class Reference

Abstract base class for synchronizing multithreaded (producer/consumer) fifo operations.

Inheritance diagram for mha_fifo_thread_platform_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- virtual void **aquire_mutex** ()=0
  Calling thread waits until it aquires the lock.
- virtual void **release_mutex** ()=0
  Calling thread releases the lock.
- virtual void **wait_for_decrease** ()=0
  Calling producer thread must own the lock.
- virtual void **wait_for_increase** ()=0
  Calling consumer thread must own the lock.
- virtual void **increment** ()=0
  To be called by producer thread after producing.
- virtual void **decrement** ()=0
  To be called by consumer thread after consuming.
- virtual ~**mha_fifo_thread_platform_t** ()
  Make destructor virtual.
- **mha_fifo_thread_platform_t** ()
  Make default constructor accessible.

Private Member Functions

- **mha_fifo_thread_platform_t** (const **mha_fifo_thread_platform_t** &)
- **mha_fifo_thread_platform_t** & **operator=** (const **mha_fifo_thread_platform_t** &)

5.204.1 Detailed Description

Abstract base class for synchronizing multithreaded (producer/consumer) fifo operations.

Works only with single producer and single consumer.

5.204.2 Constructor & Destructor Documentation

5.204.2.1 ~mha_fifo_thread_platform_t() `virtual mha_fifo_thread_platform_t::~mha↩_fifo_thread_platform_t ( ) [inline], [virtual]`

Make destructor virtual.

© 2005-2021 HörTech gGmbH, Oldenburg

5.204.2.2 mha_fifo_thread_platform_t() [1/2]

```
mha_fifo_thread_platform_t::mha_fifo↩
_thread_platform_t (
            const mha_fifo_thread_platform_t & ) [private]
```

5.204.2.3 mha_fifo_thread_platform_t() [2/2]

```
mha_fifo_thread_platform_t::mha_fifo↩
_thread_platform_t ( ) [inline]
```

Make default constructor accessible.

5.204.3 Member Function Documentation

5.204.3.1 aquire_mutex()

```
virtual void mha_fifo_thread_platform_t::aquire_mutex ( )
[pure virtual]
```

Calling thread waits until it aquires the lock.

Must not be called when the lock is already aquired.

Implemented in **mha_fifo_posix_threads_t** (p. 775).

5.204.3.2 release_mutex()

```
virtual void mha_fifo_thread_platform_t::release_mutex (
) [pure virtual]
```

Calling thread releases the lock.

May only be called when lock is owned.

Implemented in **mha_fifo_posix_threads_t** (p. 775).

© 2005-2021 HörTech gGmbH, Oldenburg

5.204.3.3 wait_for_decrease() `virtual void mha_fifo_thread_platform_t::wait_for_decrease ( ) [pure virtual]`

Calling producer thread must own the lock.

Method releases lock, and waits for consumer thread to call decrease(). Then reaquires lock and returns

Implemented in **mha_fifo_posix_threads_t** (p. 775).

5.204.3.4 wait_for_increase() `virtual void mha_fifo_thread_platform_t::wait_for_increase ( ) [pure virtual]`

Calling consumer thread must own the lock.

Method releases lock, and waits for producer thread to call increase(). Then reaquires lock and returns

Implemented in **mha_fifo_posix_threads_t** (p. 775).

5.204.3.5 increment() `virtual void mha_fifo_thread_platform_t::increment ( ) [pure virtual]`

To be called by producer thread after producing.

Producer thread needs to own the lock to call this method.

Implemented in **mha_fifo_posix_threads_t** (p. 775).

5.204.3.6 decrement() `virtual void mha_fifo_thread_platform_t::decrement ( ) [pure virtual]`

To be called by consumer thread after consuming.

Consumer thread needs to own the lock to call this method.

Implemented in **mha_fifo_posix_threads_t** (p. 775).

© 2005-2021 HörTech gGmbH, Oldenburg

5.204.3.7 operator=()

```
mha_fifo_thread_platform_t& mha_fifo_thread_platform_t↩
::operator= (
            const mha_fifo_thread_platform_t & ) [private]
```

The documentation for this class was generated from the following file:

- **mha_fifo.h**

5.205 mha_real_test_array_t Struct Reference

Public Attributes

- **mha_real_t r** [4]

5.205.1 Member Data Documentation

5.205.1.1 r

```
mha_real_t mha_real_test_array_t::r[4]
```

The documentation for this struct was generated from the following file:

- **mha.hh**

5.206 mha_rt_fifo_element_t< T > Class Template Reference

Object wrapper for **mha_rt_fifo_t** (p. 790).

Public Member Functions

- **mha_rt_fifo_element_t** (T ∗ **data**)
  Constructor.
- ∼**mha_rt_fifo_element_t** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **mha_rt_fifo_element_t**< T > * **next**
  Pointer to next fifo element. NULL for the last (newest) fifo element.
- bool **abandonned**
  Indicates that this element will no longer be used and may be deleted.
- T * **data**
  Pointer to user data.

5.206.1 Detailed Description

template<class T>
class mha_rt_fifo_element_t< T >

Object wrapper for **mha_rt_fifo_t** (p. 790).

5.206.2 Constructor & Destructor Documentation

5.206.2.1 mha_rt_fifo_element_t() `template<class T >`

```
mha_rt_fifo_element_t< T >::  mha_rt_fifo_element_t (
            T * data )  [inline]
```

Constructor.

This element assumes ownership of user data.

Parameters

data	User data. Has to be allocated on the heap with standard operator new, because it will be deleted in this element's destructor.

5.206.2.2 ~mha_rt_fifo_element_t() `template<class T >`

```
mha_rt_fifo_element_t< T >::~ mha_rt_fifo_element_t ( )  [inline]
```

5.206.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.206.3.1 next `template<class T >`

`mha_rt_fifo_element_t<T>* mha_rt_fifo_element_t< T >::next`

Pointer to next fifo element. NULL for the last (newest) fifo element.

5.206.3.2 abandonned `template<class T >`

`bool mha_rt_fifo_element_t< T >::abandonned`

Indicates that this element will no longer be used and may be deleted.

5.206.3.3 data `template<class T >`

`T* mha_rt_fifo_element_t< T >::data`

Pointer to user data.

The documentation for this class was generated from the following file:

- **mha_fifo.h**

5.207 mha_rt_fifo_t< T > Class Template Reference

Template class for thread safe, half real time safe fifo without explixit locks.

Public Member Functions

- **mha_rt_fifo_t** ()
  Construct empty fifo.
- ~**mha_rt_fifo_t** ()
  Destructor will delete all data currently in the fifo.
- T * **poll** ()
  Retrieve the latest element in the Fifo.
- T * **poll_1** ()
  Retrieve the next element in the Fifo, if there is one, and mark the previous element as abandonned.
- void **push** (T *data)
  Add element to the Fifo.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **remove_abandonned** ()
  Deletes abandonned elements.
- void **remove_all** ()
  Deletes all elements.

Private Attributes

- **mha_rt_fifo_element_t**< T > * **root**
  The first element in the fifo. Deleting elements starts here.
- **mha_rt_fifo_element_t**< T > * **current**
  *The element most recently returned by **poll*** (p. 792) *or **poll_1*** (p. 792).

5.207.1 Detailed Description

template<class T>
class mha_rt_fifo_t< T >

Template class for thread safe, half real time safe fifo without explixit locks.

Reading from this fifo is realtime safe, writing to it is not. This fifo is designed for objects that were constructed on the heap. It assumes ownership of these objects and calls delete on them when they are no longer used. Objects remain inside the Fifo while being used by the reader.

A new fifo element is inserted by using **push** (p. 792). The push operation is not real time safe, it allocates and deallocates memory. The latest element is retrieved by calling **poll** (p. 792). This operation will skip fifo elements if more than one **push** (p. 792) has been occured since the last poll. To avoid skipping, call the **poll_1** (p. 792) operation instead.

5.207.2 Constructor & Destructor Documentation

5.207.2.1 mha_rt_fifo_t() `template<class T >`

`mha_rt_fifo_t< T >::` **`mha_rt_fifo_t`** `( ) [inline]`

Construct empty fifo.

© 2005-2021 HörTech gGmbH, Oldenburg

5.207.2.2 ~mha_rt_fifo_t() `template<class T >`

`mha_rt_fifo_t< T >::~ mha_rt_fifo_t ( ) [inline]`

Destructor will delete all data currently in the fifo.

5.207.3 Member Function Documentation

5.207.3.1 poll() `template<class T >`

`T* mha_rt_fifo_t< T >::poll ( ) [inline]`

Retrieve the latest element in the Fifo.

Will skip fifo elements if more than one element has been added since last poll invocation. Will return the same element as on last call if no elements have been added in the mean time. Marks former elements as abandonned.

Returns

The latest element in this Fifo. Returns NULL if the Fifo is empty.

5.207.3.2 poll_1() `template<class T >`

`T* mha_rt_fifo_t< T >::poll_1 ( ) [inline]`

Retrieve the next element in the Fifo, if there is one, and mark the previous element as abandonned.

Else, if there is no newer element, returns the same element as on last **poll()** (p. 792) or **poll_1()** (p. 792) invocation.

Returns

The next element in this Fifo, if there is one, or the same as before. Returns NULL if the Fifo is empty.

5.207.3.3 push() `template<class T >`

```
void mha_rt_fifo_t< T >::push (
            T * data ) [inline]
```

Add element to the Fifo.

Deletes abandonned elements in the fifo.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

data	The new user data to place at the end of the fifo. After this invocation, the fifo is the owner of this object and will delete it when it is no longer used. data must have been allocated on the heap with standard operator new.

5.207.3.4 remove_abandonned() `template<class T >`

`void` **`mha_rt_fifo_t`**`< T >::remove_abandonned ( ) [inline], [private]`

Deletes abandonned elements.

5.207.3.5 remove_all() `template<class T >`

`void` **`mha_rt_fifo_t`**`< T >::remove_all ( ) [inline], [private]`

Deletes all elements.

5.207.4 Member Data Documentation

5.207.4.1 root `template<class T >`

`mha_rt_fifo_element_t``<T>*` **`mha_rt_fifo_t`**`< T >::root [private]`

The first element in the fifo. Deleting elements starts here.

5.207.4.2 current `template<class T >`

`mha_rt_fifo_element_t``<T>*` **`mha_rt_fifo_t`**`< T >::current [private]`

The element most recently returned by **poll** (p. 792) or **poll_1** (p. 792).

Searching for new elements starts here.

The documentation for this class was generated from the following file:

- **mha_fifo.h**

© 2005-2021 HörTech gGmbH, Oldenburg

5.208 mha_spec_t Struct Reference

Inheritance diagram for mha_spec_t:

Public Attributes

- **mha_complex_t** $*$ **buf**
  signal buffer
- unsigned int **num_channels**
  number of channels
- unsigned int **num_frames**
  number of frames in each channel
- **mha_channel_info_t** $*$ **channel_info**
  detailed channel description

5.208.1 Detailed Description

```
\ingroup mhasignal
\brief Spectrum signal structure
```

This structure contains the short time fourier transform output of the windowed input signal. The member `num_frames` describes the number of frequency bins in each channel. For an even FFT length N, this is $N/2+1$. With odd FFT lengths, it is $(N+1)/2$. The imaginary part

© 2005-2021 HörTech gGmbH, Oldenburg

of the first bin is zero. For even FFT lengths, also the imaginary part at the Nyquist frequency is zero.

buf[k].re	*Re(0)*	*Re(1)*	*Re(2)*	*Re(3)*	*Re(4)*	...	*Re(n/2−1)*	*Re(n/2)*
buf[k].im		*Im(1)*	*Im(2)*	*Im(3)*	*Im(4)*	...	*Im(n/2−1)*	
k	*0*	*1*	*2*	*3*	*4*		*n/2−1*	*n/2*

Figure 4 Data order of FFT spectrum.

Only the FFT bins for the positive frequencies, 0, and the Nyquist frequency are stored in this structure. The negative frequencies are not stored, because for a real-valued time signal they are the complex conjugates of the positive frequencies.

The negative frequencies still contribute to the signal's level. Refer to **Central Calibration** (p. 3) for a description of the scaling and how the level would be computed from the spectrum. It is recommended to use the library function **MHASignal::rmslevel** (p. 53) to compute the unweighted level correctly in Pascal, or **MHASignal::colored_intensity** (p. 54) to compute a possibly weighted intensity.

5.208.2 Member Data Documentation

5.208.2.1 buf `mha_complex_t* mha_spec_t::buf`

signal buffer

5.208.2.2 num_channels `unsigned int mha_spec_t::num_channels`

number of channels

5.208.2.3 num_frames `unsigned int mha_spec_t::num_frames`

number of frames in each channel

© 2005-2021 HörTech gGmbH, Oldenburg

5.208.2.4 channel_info `mha_channel_info_t* mha_spec_t::channel_info`

detailed channel description

The documentation for this struct was generated from the following file:

- **mha.hh**

5.209 mha_stash_environment_variable_t Class Reference

This class changes the value of an environment variable when constructed and restores the original state of the environment variable when destroyed.

Public Member Functions

- **mha_stash_environment_variable_t** (const std::string & **variable_name**, const std↩ ::string &new_content)
- ∼**mha_stash_environment_variable_t** ()

Private Attributes

- const bool **existed_before**

  Flag indicates if the environment variable existed before constructor.
- const std::string **variable_name**

  Name of environment variable.
- const std::string **original_content**

  Content of environment variable before constructor executed.

5.209.1 Detailed Description

This class changes the value of an environment variable when constructed and restores the original state of the environment variable when destroyed.

Can be used for testing functionality related to environment variables.

5.209.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.209.2.1 mha_stash_environment_variable_t()

```
mha_stash_environment_variable_t↩
::mha_stash_environment_variable_t (
            const std::string & variable_name,
            const std::string & new_content ) [inline]
```

5.209.2.2 ~mha_stash_environment_variable_t()

```
mha_stash_environment_variable_t↩
::~mha_stash_environment_variable_t ( ) [inline]
```

5.209.3 Member Data Documentation

5.209.3.1 existed_before

```
const bool mha_stash_environment_variable_t::existed_↩
before [private]
```

Flag indicates if the environment variable existed before constructor.

5.209.3.2 variable_name

```
const std::string mha_stash_environment_variable_t::variable↩
_name [private]
```

Name of environment variable.

5.209.3.3 original_content

```
const std::string mha_stash_environment_variable_t↩
::original_content [private]
```

Content of environment variable before constructor executed.

The documentation for this class was generated from the following file:

- **mha_os.h**

© 2005-2021 HörTech gGmbH, Oldenburg

5.210 MHA_TCP::Async_Notify Class Reference

Portable Multiplexable cross-thread notification.

Inheritance diagram for MHA_TCP::Async_Notify:

Public Member Functions

- **Async_Notify** ()
- virtual void **reset** ()
- virtual void **set** ()
- virtual ~**Async_Notify** ()

Private Attributes

- int **pipe** [2]

Additional Inherited Members

5.210.1 Detailed Description

Portable Multiplexable cross-thread notification.

5.210.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.210.2.1 Async_Notify() `Async_Notify::Async_Notify ( )`

5.210.2.2 ∼Async_Notify() `Async_Notify::∼Async_Notify ( ) [virtual]`

5.210.3 Member Function Documentation

5.210.3.1 reset() `void Async_Notify::reset ( ) [virtual]`

Reimplemented from **MHA_TCP::Wakeup_Event** (p. 838).

5.210.3.2 set() `void Async_Notify::set ( ) [virtual]`

5.210.4 Member Data Documentation

5.210.4.1 pipe `int MHA_TCP::Async_Notify::pipe[2] [private]`

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

5.211 mha_tcp::buffered_socket_t Class Reference

An asio TCP socket with an associated streambuf buffer for receiving lines of text, as well as string buffers for sending responses.

Inherits socket, and enable_shared_from_this< buffered_socket_t >.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- asio::streambuf & **get_buffer** ()
  Access to associated streambuf.
- void **queue_write** (const std::string &message)
  Send the given message through this connection to the client asynchronously.

Private Attributes

- asio::streambuf **streambuf**
  associated streambuf object to collect received pieces into lines
- std::string **current_message**
  The message that is currently sent back to the client.
- std::string **next_message**
  A buffer for the next message(s) that must be sent back to the client after the sending of current_message has completed.

5.211.1 Detailed Description

An asio TCP socket with an associated streambuf buffer for receiving lines of text, as well as string buffers for sending responses.

Used for communicating with MHA TCP clients. The life time of the connection objects is managed with shared pointers registered together with callbacks in the asio event loop. This is a common idiom in asio. To support this, we inherit from enable_shared_from_this which is also common in code that uses asio.

5.211.2 Member Function Documentation

5.211.2.1 get_buffer() `asio::streambuf& mha_tcp::buffered_socket_t::get_buffer ( ) [inline]`

Access to associated streambuf.

Needed to invoke async_read.

Returns

associated streambuf object by reference

5.211.2.2 queue_write() `void mha_tcp::buffered_socket_t::queue_write ( const std::string & message )`

Send the given message through this connection to the client asynchronously.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

message	The text to send. Method copies the message before returning.

5.211.3 Member Data Documentation

5.211.3.1 streambuf `asio::streambuf mha_tcp::buffered_socket_t::streambuf [private]`

associated streambuf object to collect received pieces into lines

5.211.3.2 current_message `std::string mha_tcp::buffered_socket_t::current_message [private]`

The message that is currently sent back to the client.

5.211.3.3 next_message `std::string mha_tcp::buffered_socket_t::next_message [private]`

A buffer for the next message(s) that must be sent back to the client after the sending of current_message has completed.

The documentation for this class was generated from the following files:

- **mha_tcp_server.hh**
- **mha_tcp_server.cpp**

5.212 MHA_TCP::Client Class Reference

A portable class for a tcp client connections.

Inheritance diagram for MHA_TCP::Client:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **Client** (const std::string &host, unsigned short port)

  Constructor connects to host, port via TCP.
- **Client** (const std::string &host, unsigned short port, **Timeout_Watcher** &timeout_↩watcher)

  Constructor connects to host, port via TCP, using a timeout.

Additional Inherited Members

5.212.1 Detailed Description

A portable class for a tcp client connections.

5.212.2 Constructor & Destructor Documentation

5.212.2.1 Client() [1/2]

```
Client::Client (
    const std::string & host,
    unsigned short port )
```

Constructor connects to host, port via TCP.

Parameters

host	The hostname of the TCP **Server** (p. 815).
port	The port or the TCP **Server** (p. 815).

5.212.2.2 Client() [2/2]

```
Client::Client (
    const std::string & host,
    unsigned short port,
    Timeout_Watcher & timeout_watcher )
```

Constructor connects to host, port via TCP, using a timeout.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

host	The hostname of the TCP **Server** (p. 815).
port	The port or the TCP **Server** (p. 815).
timeout_watcher	an Event watcher that implements a timeout.

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

5.213 MHA_TCP::Connection Class Reference

Connection (p. 803) handles Communication between client and server, is used on both sides.

Inheritance diagram for MHA_TCP::Connection:

Public Member Functions

- **Sockread_Event** ∗ **get_read_event** ()
- **Sockwrite_Event** ∗ **get_write_event** ()
- std::string **get_peer_address** ()

  Get peer's IP Address.
- unsigned short **get_peer_port** ()

  Get peer's TCP port.
- **SOCKET** **get_fd** () const

  Return the (protected) file descriptor of the connection.
- virtual ∼**Connection** ()

  Destructor closes the underlying file descriptor.
- bool **eof** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Checks if the peer has closed the connection.

- bool **can_read_line** (char delim='\n')

  Checks if a full line of text has arrived by now.
- bool **can_read_bytes** (unsigned howmany)

  Checks if the specified ammount of data can be read.
- std::string **read_line** (char delim='\n')

  Reads a single line of data from the socket.
- std::string **read_bytes** (unsigned howmany)

  Reads the specified ammount of dat from the socket.
- void **try_write** (const std::string &data="")

  Adds data to the internal "outgoing" buffer, and then tries to write as much data from that buffer to the socket as possible without blocking.
- void **write** (const std::string &data="")

  Adds data to the internal "outgoing" buffer, and then writes that that buffer to the socket, regardless of blocking.
- bool **needs_write** ()

  Checks if the internal "outgoing" buffer contains data.
- unsigned **buffered_incoming_bytes** () const

  Returns the number of bytes in the internal "incoming" buffer.
- unsigned **buffered_outgoing_bytes** () const

  Returns the number of bytes in the internal "outgoing" buffer.

Protected Member Functions

- **Connection** (**SOCKET** _fd)

  Create a connection instance from a socket filedescriptor.

Protected Attributes

- **SOCKET fd**

  The file descriptor of the TCP Socket.

Private Member Functions

- void **init_peer_data** ()

  determine peer address and port
- bool **can_sysread** ()

  Determine wether at least 1 byte can be read without blocking.
- bool **can_syswrite** ()

  Determine wether at least 1 byte can be written without blocking.
- std::string **sysread** (unsigned bytes)

  Call the system's read function and try to read bytes.
- std::string **syswrite** (const std::string &data)

  Call the system's write function and try to write all characters in the string data.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- std::string **outbuf**
- std::string **inbuf**
- **Sockread_Event** ∗ **read_event**
- **Sockwrite_Event** ∗ **write_event**
- bool **closed**
- struct sockaddr_in **peer_addr**

5.213.1 Detailed Description

Connection (p. 803) handles Communication between client and server, is used on both sides.

5.213.2 Constructor & Destructor Documentation

5.213.2.1 Connection() `MHA_TCP::Connection::Connection ( SOCKET _fd ) [protected]`

Create a connection instance from a socket filedescriptor.

Parameters

↩ _↩ *fd*	The file descriptor of the TCP Socket. This file descriptor is closed again in the destructor.

Exceptions

MHA_Error *(p. 764)*	If the file descriptor is < 0.

5.213.2.2 ∼Connection() `Connection::~Connection ( ) [virtual]`

Destructor closes the underlying file descriptor.

5.213.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.213.3.1 init_peer_data() `void MHA_TCP::Connection::init_peer_data ( ) [private]`

determine peer address and port

5.213.3.2 can_sysread() `bool Connection::can_sysread ( ) [private]`

Determine wether at least 1 byte can be read without blocking.

5.213.3.3 can_syswrite() `bool Connection::can_syswrite ( ) [private]`

Determine wether at least 1 byte can be written without blocking.

5.213.3.4 sysread() `std::string Connection::sysread ( unsigned bytes ) [private]`

Call the system's read function and try to read bytes.

This will block in a situation where can_sysread returns false.

Parameters

bytes	The desired number of characters.

Returns

The characters read from the socket. The result may have fewer characters than specified by bytes. If the result is an empty string, then the socket has been closed by the peer.

5.213.3.5 syswrite() `std::string Connection::syswrite ( const std::string & data ) [private]`

Call the system's write function and try to write all characters in the string data.

May write fewer characters, but will at least write one character.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

data	A string of characters to write to the socket.

Returns

The rest of the characters that have not yet been written.

5.213.3.6 get_read_event() `Sockread_Event * Connection::get_read_event ( )`

5.213.3.7 get_write_event() `Sockwrite_Event * Connection::get_write_event ( )`

5.213.3.8 get_peer_address() `std::string Connection::get_peer_address ( )`

Get peer's IP Address.

5.213.3.9 get_peer_port() `unsigned short Connection::get_peer_port ( )`

Get peer's TCP port.

5.213.3.10 get_fd() `SOCKET MHA_TCP::Connection::get_fd ( ) const [inline]`

Return the (protected) file descriptor of the connection.

Will be required for SSL.

5.213.3.11 eof() `bool Connection::eof ( )`

Checks if the peer has closed the connection.

As a side effect, this method fills the internal "incoming" buffer if it was empty and the socket is readable and not eof.

5.213.3.12 can_read_line() `bool Connection::can_read_line ( char delim = '\n' )`

Checks if a full line of text has arrived by now.

This method reads data from the socket into the internal "incoming" buffer if it can be done without blocking.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

delim	The line delimiter.

Returns

true if at least one full line of text is present in the internal buffer after this method call, false otherwise.

5.213.3.13 can_read_bytes() `bool Connection::can_read_bytes ( unsigned howmany )`

Checks if the specified ammount of data can be read.

This method reads data from the socket into an internal "incoming" buffer if it can be done without blocking.

Parameters

howmany	The number of bytes that the caller wants to have checked.

Returns

true if at least the specified ammount of data is present in the internal buffer after this method call, false otherwise

5.213.3.14 read_line() `std::string Connection::read_line ( char delim = '\n' )`

Reads a single line of data from the socket.

Blocks if necessary.

Parameters

delim	The line delimiter.

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

The string of characters in this line, including the trailing delimiter. The delimiter may be missing if the last line before EOF does not have a delimiter.

5.213.3.15 read_bytes() `std::string Connection::read_bytes ( unsigned howmany )`

Reads the specified ammount of dat from the socket.

Blocks if necessary.

Parameters

howmany	The number of bytes to read.

Returns

The string of characters read. The string may be shorter if EOF is encountered.

5.213.3.16 try_write() `void Connection::try_write ( const std::string & data = "" )`

Adds data to the internal "outgoing" buffer, and then tries to write as much data from that buffer to the socket as possible without blocking.

Parameters

data	data to send over the socket.

5.213.3.17 write() `void Connection::write ( const std::string & data = "" )`

Adds data to the internal "outgoing" buffer, and then writes that that buffer to the socket, regardless of blocking.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

data	data to send over the socket.

5.213.3.18 needs_write() `bool Connection::needs_write ( )`

Checks if the internal "outgoing" buffer contains data.

5.213.3.19 buffered_incoming_bytes() `unsigned Connection::buffered_incoming_bytes ( ) const`

Returns the number of bytes in the internal "incoming" buffer.

5.213.3.20 buffered_outgoing_bytes() `unsigned Connection::buffered_outgoing_bytes ( ) const`

Returns the number of bytes in the internal "outgoing" buffer.

5.213.4 Member Data Documentation

5.213.4.1 outbuf `std::string MHA_TCP::Connection::outbuf [private]`

5.213.4.2 inbuf `std::string MHA_TCP::Connection::inbuf [private]`

5.213.4.3 read_event `Sockread_Event* MHA_TCP::Connection::read_event [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.213.4.4 write_event `Sockwrite_Event* MHA_TCP::Connection::write_event [private]`

5.213.4.5 closed `bool MHA_TCP::Connection::closed [private]`

5.213.4.6 peer_addr `struct sockaddr_in MHA_TCP::Connection::peer_addr [private]`

5.213.4.7 fd `SOCKET MHA_TCP::Connection::fd [protected]`

The file descriptor of the TCP Socket.

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

5.214 MHA_TCP::Event_Watcher Class Reference

OS-independent event watcher, uses select on Unix and WaitForMultipleObjects on Windows.

Inheritance diagram for MHA_TCP::Event_Watcher:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Types

- typedef **std::set**< **Wakeup_Event** ∗ > **Events**
- typedef **std::set**< **Wakeup_Event** ∗ >:: **iterator iterator**

Public Member Functions

- void **observe** (**Wakeup_Event** ∗event)
  Add an event to this observer.
- void **ignore** (**Wakeup_Event** ∗event)
  Remove an event from this observer.
- **std::set**< **Wakeup_Event** ∗ > **wait** ()
  \ Wait for some event to occur.
- virtual ∼**Event_Watcher** ()

Private Attributes

- **std::set**< **Wakeup_Event** ∗ > **events**
  The list of events to watch.

5.214.1 Detailed Description

OS-independent event watcher, uses select on Unix and WaitForMultipleObjects on Windows.

5.214.2 Member Typedef Documentation

5.214.2.1 Events `typedef std::set< Wakeup_Event*> MHA_TCP::Event_Watcher::↩ Events`

5.214.2.2 iterator `typedef std::set< Wakeup_Event*>:: iterator MHA_TCP::Event_↩ Watcher::iterator`

© 2005-2021 HörTech gGmbH, Oldenburg

5.214.3 Constructor & Destructor Documentation

5.214.3.1 ~Event_Watcher() `Event_Watcher::~Event_Watcher ( ) [virtual]`

5.214.4 Member Function Documentation

5.214.4.1 observe() `void Event_Watcher::observe ( Wakeup_Event * event )`

Add an event to this observer.

5.214.4.2 ignore() `void Event_Watcher::ignore ( Wakeup_Event * event )`

Remove an event from this observer.

5.214.4.3 wait() `std::set< Wakeup_Event * > Event_Watcher::wait ( )`

\ Wait for some event to occur.

Return all events that are ready

5.214.5 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.214.5.1 events `std::set< Wakeup_Event*> MHA_TCP::Event_Watcher::events [private]`

The list of events to watch.

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

5.215 MHA_TCP::OS_EVENT_TYPE Struct Reference

Public Types

- enum { **R** =0, **W** =1, **X** =2, **T** }

Public Attributes

- enum MHA_TCP::OS_EVENT_TYPE:: { ... } **mode**
- union {
  int **fd**
  double **timeout**
  };

5.215.1 Member Enumeration Documentation

5.215.1.1 anonymous enum `anonymous enum`

Enumerator

R	
W	
X	
T	

© 2005-2021 HörTech gGmbH, Oldenburg

5.215.2 Member Data Documentation

5.215.2.1 mode `enum { ... } MHA_TCP::OS_EVENT_TYPE::mode`

5.215.2.2 fd `int MHA_TCP::OS_EVENT_TYPE::fd`

5.215.2.3 timeout `double MHA_TCP::OS_EVENT_TYPE::timeout`

5.215.2.4 "@5 `union { ... }`

The documentation for this struct was generated from the following file:

- **mha_tcp.hh**

5.216 MHA_TCP::Server Class Reference

Public Member Functions

- **Server** (unsigned short **port**=0, const std::string & **iface**="0.0.0.0")
  Create a TCP server socket.
- **Server** (const std::string & **iface**, unsigned short **port**=0)
  Create a TCP server socket.
- ∼**Server** ()
  Close the TCP server socket.
- std::string **get_interface** () const
  Get the name given in the constructor for the network interface.
- unsigned short **get_port** () const
  Get the port that the TCP server socket currently listens to.
- **Sockaccept_Event** ∗ **get_accept_event** ()
  *Produces an event that can be observed by an **Event_Watcher** (p. 811).*
- **Connection** ∗ **accept** ()
  Accept an incoming connection.
- **Connection** ∗ **try_accept** ()
  Accept an incoming connection if it can be done without blocking.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **initialize** (const std::string & **iface**, unsigned short **port**)

Private Attributes

- sockaddr_in **sock_addr**
- **SOCKET serversocket**
- std::string **iface**
- unsigned short **port**
- **Sockaccept_Event** * **accept_event**

5.216.1 Constructor & Destructor Documentation

5.216.1.1 Server() [1/2]

```
Server::Server (
    unsigned short port = 0,
    const std::string & iface = "0.0.0.0" )
```

Create a TCP server socket.

Parameters

port	The TCP port to listen to.
iface	The network interface to bind to.

5.216.1.2 Server() [2/2]

```
Server::Server (
    const std::string & iface,
    unsigned short port = 0 )
```

Create a TCP server socket.

Parameters

port	The TCP port to listen to.
iface	The network interface to bind to.

© 2005-2021 HörTech gGmbH, Oldenburg

5.216.1.3 ~Server() `Server::~Server ( )`

Close the TCP server socket.

5.216.2 Member Function Documentation

5.216.2.1 initialize() `void Server::initialize ( const std::string & iface, unsigned short port ) [private]`

5.216.2.2 get_interface() `std::string Server::get_interface ( ) const`

Get the name given in the constructor for the network interface.

5.216.2.3 get_port() `unsigned short Server::get_port ( ) const`

Get the port that the TCP server socket currently listens to.

5.216.2.4 get_accept_event() `Sockaccept_Event * Server::get_accept_event ( )`

Produces an event that can be observed by an **Event_Watcher** (p. 811).

This event signals incoming connections that can be accepted.

5.216.2.5 accept() `Connection * Server::accept ( )`

Accept an incoming connection.

blocks if necessary.

Returns

The new TCP connection. The connection has to be deleted by the caller.

© 2005-2021 HörTech gGmbH, Oldenburg

5.216.2.6 try_accept() `Connection * Server::try_accept ( )`

Accept an incoming connection if it can be done without blocking.

Returns

The new TCP connection or 0 if there is no immediate connection. The connection has to be deleted by the caller.

5.216.3 Member Data Documentation

5.216.3.1 sock_addr `sockaddr_in MHA_TCP::Server::sock_addr [private]`

5.216.3.2 serversocket `SOCKET MHA_TCP::Server::serversocket [private]`

5.216.3.3 iface `std::string MHA_TCP::Server::iface [private]`

5.216.3.4 port `unsigned short MHA_TCP::Server::port [private]`

5.216.3.5 accept_event `Sockaccept_Event* MHA_TCP::Server::accept_event [private]`

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.217 mha_tcp::server_t Class Reference

Class for accepting TCP connections from clients.

Inheritance diagram for mha_tcp::server_t:

Public Member Functions

- **server_t** (const std::string &interface, uint16_t port)

  Allocates a TCP server.
- uint16_t **get_port** () const
- asio::ip::tcp::endpoint **get_endpoint** () const
- asio::ip::address **get_address** () const
- size_t **get_num_accepted_connections** () const
- void **run** ()

  Accepts connections on the TCP port and serves them.
- virtual bool **on_received_line** (std::shared_ptr< **buffered_socket_t** > c, const std↩ ::string &l)

  This method is invoked when a line of text is received on one of the accepted connections.
- virtual void **shutdown** ()

  *Shuts down the server: Close the acceptor (no new connections), shuts down the receiving direction of all accepted connections (no new commands, but responses can still be finished), registers a timer event that will cause event loop termination and return from the **run()** (p. 822) method 1 second in the future, giving us reasonably enough time for the pending responses to be sent out.*
- virtual ~**server_t** ()=default

  Make destructor virtual.
- asio::io_context & **get_context** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **trigger_accept** ()

  Triggers the acceptance of the next connection.
- void **post_trigger_read_line** (std::shared_ptr< **buffered_socket_t** > c)

  Call trigger_read_line with a single detour through asio's event loop to avoid starving other connections when one connection floods us.
- void **trigger_read_line** (std::shared_ptr< **buffered_socket_t** > c)

  Triggers the reading of the next line from a connection.
- void **add_connection** (std::shared_ptr< **buffered_socket_t** > connection)

  Add new connection to the list of connections, retire stale pointers.

Private Attributes

- asio::io_context **io_context**

  The io context used to run event loops.
- std::shared_ptr< asio::ip::tcp::acceptor > **acceptor**

  The underlying asio object used to accept incoming TCP connections.
- bool **async_accept_has_been_triggered** = false

  Set to true when async_acceptance is triggered in trigger_accept (Only one accept can be in process at any time).
- size_t **num_accepted_connections** = 0U

  Number of accepted connections (not necessarily still existing)
- std::vector< std::weak_ptr< **buffered_socket_t** > > **connections**

  Weak pointers to the existing connections.

5.217.1 Detailed Description

Class for accepting TCP connections from clients.

5.217.2 Constructor & Destructor Documentation

5.217.2.1 server_t()

```
mha_tcp::server_t::server_t (
    const std::string & interface,
    uint16_t port )
```

Allocates a TCP server.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

interface	Host name of the network interface to listen on. Can be "localhost" or "127.0.0.1" for localhost, "0.0.0.0" for any ipv4 interface, ...
port	TCP port to open for incoming connections. If port==0, then the operating system will select a free port.

Exceptions

system_error	if the name given in interface cannot be resolved
system_error	if we cannot bind to the requested interface

5.217.2.2 ~server_t() `virtual mha_tcp::server_t::~server_t ( ) [virtual], [default]`

Make destructor virtual.

5.217.3 Member Function Documentation

5.217.3.1 get_port() `uint16_t mha_tcp::server_t::get_port ( ) const`

Returns

The port number of the TCP port that has been opened. If the port specified in the constructor was 0, this will return the port that the operating system has selected.

5.217.3.2 get_endpoint() `asio::ip::tcp::endpoint mha_tcp::server_t::get_endpoint ( ) const`

Returns

The local endpoint of the acceptor.

© 2005-2021 HörTech gGmbH, Oldenburg

5.217.3.3 get_address() `asio::ip::address mha_tcp::server_t::get_address ( ) const`

Returns

The ip adress that the server is bound to.

5.217.3.4 get_num_accepted_connections() `size_t mha_tcp::server_t::get_num_accepted_connections ( ) const`

Returns

the number of TCP connections that ha ve been accepted

5.217.3.5 run() `void mha_tcp::server_t::run ( )`

Accepts connections on the TCP port and serves them.

Triggers the acceptance of the next connection to start things off.

5.217.3.6 on_received_line() `bool mha_tcp::server_t::on_received_line ( std::shared_ptr< buffered_socket_t > c, const std::string & l ) [virtual]`

This method is invoked when a line of text is received on one of the accepted connections.

Override this method to process the communication with the client.

Parameters

c	the connection that has received this line
l	the line that has been received, without the line ending

Returns

client should return true when client wants to read another line of text, else false.

© 2005-2021 HörTech gGmbH, Oldenburg

Reimplemented in **mhaserver_t::tcp_server_t** (p. 1198).

5.217.3.7 shutdown() `void mha_tcp::server_t::shutdown ( ) [virtual]`

Shuts down the server: Close the acceptor (no new connections), shuts down the receiving direction of all accepted connections (no new commands, but responses can still be finished), registers a timer event that will cause event loop termination and return from the **run()** (p. 822) method 1 second in the future, giving us reasonably enough time for the pending responses to be sent out.

5.217.3.8 get_context() `asio::io_context & mha_tcp::server_t::get_context ( )`

Returns

the asio io context used to run the event loop

5.217.3.9 trigger_accept() `void mha_tcp::server_t::trigger_accept ( ) [private]`

Triggers the acceptance of the next connection.

Called from run to accept the first connection and from the accept handler to accept each next connection. Once a connection from a client is accepted, the accept handler will register it with the event loop for receiving a line of text.

5.217.3.10 post_trigger_read_line() `void mha_tcp::server_t::post_trigger_read_line ( std::shared_ptr< buffered_socket_t > c ) [private]`

Call trigger_read_line with a single detour through asio's event loop to avoid starving other connections when one connection floods us.

5.217.3.11 trigger_read_line() `void mha_tcp::server_t::trigger_read_line ( std::shared_ptr< buffered_socket_t > c ) [private]`

Triggers the reading of the next line from a connection.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

c	The connection where the incoming data is expected from.

5.217.3.12 add_connection()

`void mha_tcp::server_t::add_connection ( std::shared_ptr< buffered_socket_t > connection ) [inline], [private]`

Add new connection to the list of connections, retire stale pointers.

5.217.4 Member Data Documentation

5.217.4.1 io_context

`asio::io_context mha_tcp::server_t::io_context [private]`

The io context used to run event loops.

5.217.4.2 acceptor

`std::shared_ptr<asio::ip::tcp::acceptor> mha_tcp::server_t↩ ::acceptor [private]`

The underlying asio object used to accept incoming TCP connections.

5.217.4.3 async_accept_has_been_triggered

`bool mha_tcp::server_t::async_accept↩ _has_been_triggered = false [private]`

Set to true when async_acceptance is triggered in trigger_accept (Only one accept can be in process at any time).

5.217.4.4 num_accepted_connections

`size_t mha_tcp::server_t::num_accepted_↩ connections = 0U [private]`

Number of accepted connections (not necessarily still existing)

© 2005-2021 HörTech gGmbH, Oldenburg

5.217.4.5 connections `std::vector<std::weak_ptr< buffered_socket_t> > mha_tcp↩ ::server_t::connections [private]`

Weak pointers to the existing connections.

Needed to shutdown the active connections for incoming data when server shuts down.

The documentation for this class was generated from the following files:

- **mha_tcp_server.hh**
- **mha_tcp_server.cpp**

5.218 MHA_TCP::sock_init_t Class Reference

Public Member Functions

- **sock_init_t** ()

5.218.1 Constructor & Destructor Documentation

5.218.1.1 sock_init_t() `MHA_TCP::sock_init_t::sock_init_t ( ) [inline]`

The documentation for this class was generated from the following file:

- **mha_tcp.cpp**

5.219 MHA_TCP::Sockaccept_Event Class Reference

Inheritance diagram for MHA_TCP::Sockaccept_Event:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **Sockaccept_Event** (**SOCKET**)

Additional Inherited Members

5.219.1 Constructor & Destructor Documentation

5.219.1.1 Sockaccept_Event() `MHA_TCP::Sockaccept_Event::Sockaccept_Event ( SOCKET s )`

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

5.220 MHA_TCP::Sockread_Event Class Reference

Watch socket for incoming data.

Inheritance diagram for MHA_TCP::Sockread_Event:

Public Member Functions

- **Sockread_Event** (**SOCKET** s)
  Set socket to watch for.

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.220.1 Detailed Description

Watch socket for incoming data.

5.220.2 Constructor & Destructor Documentation

5.220.2.1 Sockread_Event() `MHA_TCP::Sockread_Event::Sockread_Event ( SOCKET s )`

Set socket to watch for.

Parameters

s	The socket to observe incoming data on.

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

5.221 MHA_TCP::Sockwrite_Event Class Reference

Inheritance diagram for MHA_TCP::Sockwrite_Event:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **Sockwrite_Event** (**SOCKET** s)

Additional Inherited Members

5.221.1 Constructor & Destructor Documentation

5.221.1.1 Sockwrite_Event() `MHA_TCP::Sockwrite_Event::Sockwrite_Event ( SOCKET s )`

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

5.222 MHA_TCP::Thread Class Reference

A very simple class for portable threads.

Public Types

- enum { **PREPARED**, **RUNNING**, **FINISHED** }
  The current state of the thread.
- typedef void ∗(∗ **thr_f**) (void ∗)
  The thread function signature to use with this class.

Public Member Functions

- **Thread** (**thr_f** func, void ∗ **arg**=0)
  Constructor starts a new thread.
- virtual ∼**Thread** ()
  *The destructor should only be called when the **Thread** (p. 828) is finished.*
- virtual void **run** ()
  *The internal method that delegated the new thread to the registered **Thread** (p. 828) function.*

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **Async_Notify thread_finish_event**
  Event will be triggered when the thread exits.
- enum MHA_TCP::Thread:: { ... } **state**
  The current state of the thread.
- **thr_f thread_func**
  The thread function that the client has registered.
- void * **thread_arg**
  The argument that the client wants to be handed through to the thread function.
- **MHA_Error** * **error**
  *The **MHA_Error** (p. 764) that caused the thread to abort, if any.*

Protected Member Functions

- **Thread** ()
  Default constructor may only be used by derived classes that want to start the thread themselves.

Protected Attributes

- void * **arg**
  The argument for the client's thread function.
- void * **return_value**
  The return value from the client's thread function is stored here When that function returns.

Private Attributes

- pthread_t **thread_handle**
  The posix thread handle.
- pthread_attr_t **thread_attr**
  The posix thread attribute structure.

5.222.1 Detailed Description

A very simple class for portable threads.

5.222.2 Member Typedef Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.222.2.1 thr_f `typedef void*(* MHA_TCP::Thread::thr_f) (void *)`

The thread function signature to use with this class.

Derive from this class and call protected standard constructor to start threads differently.

5.222.3 Member Enumeration Documentation

5.222.3.1 anonymous enum `anonymous enum`

The current state of the thread.

Enumerator

PREPARED	
RUNNING	
FINISHED	

5.222.4 Constructor & Destructor Documentation

5.222.4.1 Thread() [1/2] `MHA_TCP::Thread::Thread ( ) [protected]`

Default constructor may only be used by derived classes that want to start the thread themselves.

5.222.4.2 Thread() [2/2]

```
Thread::Thread (
    Thread::thr_f func,
    void * arg = 0 )
```

Constructor starts a new thread.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

func	The function to be executed by the thread.
arg	The argument given to pass to the thread function.

5.222.4.3 ~Thread() `Thread::~Thread ( ) [virtual]`

The destructor should only be called when the **Thread** (p. 828) is finished.

There is preliminary support for forceful thread cancellation in the destructor, but probably not very robust or portable..

5.222.5 Member Function Documentation

5.222.5.1 run() `void Thread::run ( ) [virtual]`

The internal method that delegated the new thread to the registered **Thread** (p. 828) function.

5.222.6 Member Data Documentation

5.222.6.1 thread_handle `pthread_t MHA_TCP::Thread::thread_handle [private]`

The posix thread handle.

5.222.6.2 thread_attr `pthread_attr_t MHA_TCP::Thread::thread_attr [private]`

The posix thread attribute structure.

Required for starting a thread in detached state. Detachment is required to eliminate the need for joining this thread.

© 2005-2021 HörTech gGmbH, Oldenburg

5.222.6.3 arg `void* MHA_TCP::Thread::arg [protected]`

The argument for the client's thread function.

5.222.6.4 return_value `void* MHA_TCP::Thread::return_value [protected]`

The return value from the client's thread function is stored here When that function returns.

5.222.6.5 thread_finish_event `Async_Notify MHA_TCP::Thread::thread_finish_event`

Event will be triggered when the thread exits.

5.222.6.6 state `enum { ... } MHA_TCP::Thread::state`

The current state of the thread.

5.222.6.7 thread_func `thr_f MHA_TCP::Thread::thread_func`

The thread function that the client has registered.

5.222.6.8 thread_arg `void* MHA_TCP::Thread::thread_arg`

The argument that the client wants to be handed through to the thread function.

5.222.6.9 error `MHA_Error* MHA_TCP::Thread::error`

The **MHA_Error** (p. 764) that caused the thread to abort, if any.

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.223 MHA_TCP::Timeout_Event Class Reference

Inheritance diagram for MHA_TCP::Timeout_Event:

Public Member Functions

- **Timeout_Event** (double interval)
- virtual **OS_EVENT_TYPE get_os_event** ()

Private Attributes

- double **end_time**

Additional Inherited Members

5.223.1 Constructor & Destructor Documentation

5.223.1.1 Timeout_Event() `Timeout_Event::Timeout_Event (double interval)`

5.223.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.223.2.1 get_os_event() `OS_EVENT_TYPE Timeout_Event::get_os_event ( ) [virtual]`

Reimplemented from **MHA_TCP::Wakeup_Event** (p. 838).

5.223.3 Member Data Documentation

5.223.3.1 end_time `double MHA_TCP::Timeout_Event::end_time [private]`

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

5.224 MHA_TCP::Timeout_Watcher Class Reference

OS-independent event watcher with internal fixed-end-time timeout.

Inheritance diagram for MHA_TCP::Timeout_Watcher:

Public Member Functions

- **Timeout_Watcher** (double interval)
- virtual ~**Timeout_Watcher** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **Timeout_Event timeout**

Additional Inherited Members

5.224.1 Detailed Description

OS-independent event watcher with internal fixed-end-time timeout.

5.224.2 Constructor & Destructor Documentation

5.224.2.1 Timeout_Watcher() `Timeout_Watcher::Timeout_Watcher ( double interval ) [explicit]`

5.224.2.2 ∼Timeout_Watcher() `Timeout_Watcher::∼Timeout_Watcher ( ) [virtual]`

5.224.3 Member Data Documentation

5.224.3.1 timeout `Timeout_Event MHA_TCP::Timeout_Watcher::timeout [private]`

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.225 MHA_TCP::Wakeup_Event Class Reference

A base class for asynchronous wakeup events.

Inheritance diagram for MHA_TCP::Wakeup_Event:

Public Member Functions

- **Wakeup_Event** ()
  Event Constructor.
- virtual void **observed_by** (**Event_Watcher** ∗observer)
  *Called by the **Event_Watcher** (p. 811) when this event is added to its list of observed events.*
- virtual void **ignored_by** (**Event_Watcher** ∗observer)
  *Called by the **Event_Watcher** (p. 811) when this event is removed from its list of observed events.*
- virtual ∼**Wakeup_Event** ()
  Destructor deregisters from observers.
- virtual **OS_EVENT_TYPE get_os_event** ()
  Get necessary information for the Event Watcher.
- virtual void **reset** ()
  For pure notification events, reset the "signalled" status.
- virtual bool **status** ()
  Query wether the event is in signalled state now.

Protected Attributes

- **OS_EVENT_TYPE os_event**
- bool **os_event_valid**

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **std::set**< class **Event_Watcher** * > **observers**

  *A list of all **Event_Watcher** (p. 811) instances that this **Wakeup_Event** (p. 836) is observed by (stored here for proper deregistering).*

5.225.1 Detailed Description

A base class for asynchronous wakeup events.

5.225.2 Constructor & Destructor Documentation

5.225.2.1 Wakeup_Event() `Wakeup_Event::Wakeup_Event ( )`

Event Constructor.

The new event has invalid state.

5.225.2.2 ~Wakeup_Event() `Wakeup_Event::~Wakeup_Event ( ) [virtual]`

Destructor deregisters from observers.

5.225.3 Member Function Documentation

5.225.3.1 observed_by() `void Wakeup_Event::observed_by ( Event_Watcher * observer ) [virtual]`

Called by the **Event_Watcher** (p. 811) when this event is added to its list of observed events.

© 2005-2021 HörTech gGmbH, Oldenburg

5.225.3.2 ignored_by() `void Wakeup_Event::ignored_by ( Event_Watcher * observer ) [virtual]`

Called by the **Event_Watcher** (p. 811) when this event is removed from its list of observed events.

5.225.3.3 get_os_event() `OS_EVENT_TYPE Wakeup_Event::get_os_event ( ) [virtual]`

Get necessary information for the Event Watcher.

Reimplemented in **MHA_TCP::Timeout_Event** (p. 833).

5.225.3.4 reset() `void Wakeup_Event::reset ( ) [virtual]`

For pure notification events, reset the "signalled" status.

Reimplemented in **MHA_TCP::Async_Notify** (p. 799).

5.225.3.5 status() `bool Wakeup_Event::status ( ) [virtual]`

Query wether the event is in signalled state now.

5.225.4 Member Data Documentation

5.225.4.1 observers `std::set<class Event_Watcher *> MHA_TCP::Wakeup_Event↩ ::observers [private]`

A list of all **Event_Watcher** (p. 811) instances that this **Wakeup_Event** (p. 836) is observed by (stored here for proper deregistering).

© 2005-2021 HörTech gGmbH, Oldenburg

5.225.4.2 os_event `OS_EVENT_TYPE MHA_TCP::Wakeup_Event::os_event [protected]`

5.225.4.3 os_event_valid `bool MHA_TCP::Wakeup_Event::os_event_valid [protected]`

The documentation for this class was generated from the following files:

- **mha_tcp.hh**
- **mha_tcp.cpp**

5.226 mha_tictoc_t Struct Reference

Public Attributes

- struct timeval **tv1**
- struct timeval **tv2**
- struct timezone **tz**
- float **t**

5.226.1 Member Data Documentation

5.226.1.1 tv1 `struct timeval mha_tictoc_t::tv1`

5.226.1.2 tv2 `struct timeval mha_tictoc_t::tv2`

5.226.1.3 tz `struct timezone mha_tictoc_t::tz`

© 2005-2021 HörTech gGmbH, Oldenburg

5.226.1.4 t `float mha_tictoc_t::t`

The documentation for this struct was generated from the following file:

- **mha_profiling.h**

5.227 mha_wave_t Struct Reference

Waveform signal structure.

Inheritance diagram for mha_wave_t:

Public Attributes

- **mha_real_t** $*$ **buf**
  signal buffer
- unsigned int **num_channels**
  number of channels
- unsigned int **num_frames**
  number of frames in each channel
- **mha_channel_info_t** $*$ **channel_info**
  detailed channel description

© 2005-2021 HörTech gGmbH, Oldenburg

5.227.1 Detailed Description

Waveform signal structure.

This structure contains one fragment of a waveform signal. The member num_frames describes the number of audio samples in each audio channel.

In a calibrated openMHA, audio samples are stored in unit Pascal, see **Central Calibration** (p. 3).

The field channel_info must be an array of num_channels entries or NULL.

5.227.2 Member Data Documentation

5.227.2.1 buf `mha_real_t* mha_wave_t::buf`

signal buffer

5.227.2.2 num_channels `unsigned int mha_wave_t::num_channels`

number of channels

5.227.2.3 num_frames `unsigned int mha_wave_t::num_frames`

number of frames in each channel

5.227.2.4 channel_info `mha_channel_info_t* mha_wave_t::channel_info`

detailed channel description

The documentation for this struct was generated from the following file:

- **mha.hh**

© 2005-2021 HörTech gGmbH, Oldenburg

5.228 mhachain::chain_base_t Class Reference

Inheritance diagram for mhachain::chain_base_t:

Public Member Functions

- **chain_base_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **process** (**mha_wave_t** ∗, **mha_wave_t** ∗∗)
- void **process** (**mha_spec_t** ∗, **mha_wave_t** ∗∗)
- void **process** (**mha_wave_t** ∗, **mha_spec_t** ∗∗)
- void **process** (**mha_spec_t** ∗, **mha_spec_t** ∗∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

Protected Attributes

- **MHAParser::bool_t bprofiling**
- **MHAParser::vstring_t algos**

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **update** ()

Private Attributes

- std::vector< std::string > **old_algos**
- **MHAEvents::patchbay_t**< **mhachain::chain_base_t** > **patchbay**
- **mhaconfig_t cfin**
- **mhaconfig_t cfout**
- bool **b_prepared**

Additional Inherited Members

5.228.1 Constructor & Destructor Documentation

5.228.1.1 chain_base_t()

```
mhachain::chain_base_t::chain_base_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.228.2 Member Function Documentation

5.228.2.1 process() [1/4]

```
void mhachain::chain_base_t::process (
    mha_wave_t * sin,
    mha_wave_t ** sout )
```

5.228.2.2 process() [2/4]

```
void mhachain::chain_base_t::process (
    mha_spec_t * sin,
    mha_wave_t ** sout )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.228.2.3 process() [3/4]

```
void mhachain::chain_base_t::process (
    mha_wave_t * sin,
    mha_spec_t ** sout )
```

5.228.2.4 process() [4/4]

```
void mhachain::chain_base_t::process (
    mha_spec_t * sin,
    mha_spec_t ** sout )
```

5.228.2.5 prepare()

```
void mhachain::chain_base_t::prepare (
    mhaconfig_t & cf ) [virtual]
```

Implements **MHAPlugin::plugin_t< mhachain::plugs_t >** (p. 1153).

5.228.2.6 release()

```
void mhachain::chain_base_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< mhachain::plugs_t >** (p. 1154).

5.228.2.7 update()

```
void mhachain::chain_base_t::update ( ) [private]
```

5.228.3 Member Data Documentation

5.228.3.1 bprofiling

```
MHAParser::bool_t mhachain::chain_base_t::bprofiling [protected]
```

5.228.3.2 algos

```
MHAParser::vstring_t mhachain::chain_base_t::algos [protected]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.228.3.3 old_algos

`std::vector<std::string> mhachain::chain_base_t::old_algos [private]`

5.228.3.4 patchbay

`MHAEvents::patchbay_t< mhachain::chain_base_t > mhachain::chain_base_t::patchbay [private]`

5.228.3.5 cfin

`mhaconfig_t mhachain::chain_base_t::cfin [private]`

5.228.3.6 cfout

`mhaconfig_t mhachain::chain_base_t::cfout [private]`

5.228.3.7 b_prepared

`bool mhachain::chain_base_t::b_prepared [private]`

The documentation for this class was generated from the following files:

- **mha_generic_chain.h**
- **mha_generic_chain.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.229 mhachain::mhachain_t Class Reference

Inheritance diagram for mhachain::mhachain_t:

Public Member Functions

- **mhachain_t** (**algo_comm_t** iac, const std::string &configured_name)

Additional Inherited Members

5.229.1 Constructor & Destructor Documentation

5.229.1.1 mhachain_t()

```
mhachain::mhachain_t::mhachain_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

The documentation for this class was generated from the following file:

- **mhachain.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.230 mhachain::plugs_t Class Reference

Public Member Functions

- **plugs_t** (std::vector< std::string > **algos**, **mhaconfig_t** cfin, **mhaconfig_t** cfout, bool do_prepare, **MHAParser::parser_t** &p, **algo_comm_t** iac, bool use_profiling)
- ~**plugs_t** ()
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- void **process** (**mha_wave_t** *, **mha_spec_t** *, **mha_wave_t** **, **mha_spec_t** **)
- bool **prepared** () const

Private Member Functions

- void **alloc_plugs** (std::vector< std::string > **algos**)
- void **cleanup_plugs** ()
- void **update_proc_load** ()

Private Attributes

- bool **b_prepared**
- std::vector< **PluginLoader::mhapluginloader_t** * > **algos**
- **MHAParser::parser_t** & **parser**
- **algo_comm_t** **ac**
- **MHAParser::parser_t** **profiling**
- **MHAParser::vstring_mon_t** **prof_algos**
- **MHAParser::vfloat_mon_t** **prof_init**
- **MHAParser::vfloat_mon_t** **prof_prepare**
- **MHAParser::vfloat_mon_t** **prof_release**
- **MHAParser::vfloat_mon_t** **prof_process**
- **MHAParser::float_mon_t** **prof_process_tt**
- **MHAParser::vfloat_mon_t** **prof_process_load**
- unsigned int **proc_cnt**
- **mhaconfig_t** **prof_cfg**
- **MHAEvents::connector_t**< **mhachain::plugs_t** > **prof_load_con**
- **MHAEvents::connector_t**< **mhachain::plugs_t** > **prof_tt_con**
- bool **b_use_profiling**
- **mha_platform_tictoc_t** **tictoc**

5.230.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.230.1.1 plugs_t()

```
mhachain::plugs_t::plugs_t (
 std::vector< std::string > algos,
  mhaconfig_t cfin,
  mhaconfig_t cfout,
 bool do_prepare,
  MHAParser::parser_t & p,
  algo_comm_t iac,
 bool use_profiling )
```

5.230.1.2 ~plugs_t()

```
mhachain::plugs_t::~plugs_t ( )
```

5.230.2 Member Function Documentation

5.230.2.1 prepare()

```
void mhachain::plugs_t::prepare (
  mhaconfig_t & tf )
```

5.230.2.2 release()

```
void mhachain::plugs_t::release ( )
```

5.230.2.3 process()

```
void mhachain::plugs_t::process (
  mha_wave_t * win,
  mha_spec_t * sin,
  mha_wave_t ** wout,
  mha_spec_t ** sout )
```

5.230.2.4 prepared()

```
bool mhachain::plugs_t::prepared ( ) const [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.230.2.5 alloc_plugs()

`void mhachain::plugs_t::alloc_plugs ( std::vector< std::string > algos ) [private]`

5.230.2.6 cleanup_plugs()

`void mhachain::plugs_t::cleanup_plugs ( ) [private]`

5.230.2.7 update_proc_load()

`void mhachain::plugs_t::update_proc_load ( ) [private]`

5.230.3 Member Data Documentation

5.230.3.1 b_prepared

`bool mhachain::plugs_t::b_prepared [private]`

5.230.3.2 algos

`std::vector< PluginLoader::mhapluginloader_t* > mhachain::plugs_t::algos [private]`

5.230.3.3 parser

`MHAParser::parser_t& mhachain::plugs_t::parser [private]`

5.230.3.4 ac

`algo_comm_t mhachain::plugs_t::ac [private]`

5.230.3.5 profiling

`MHAParser::parser_t mhachain::plugs_t::profiling [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.230.3.6 prof_algos

MHAParser::vstring_mon_t mhachain::plugs_t::prof_algos [private]

5.230.3.7 prof_init

MHAParser::vfloat_mon_t mhachain::plugs_t::prof_init [private]

5.230.3.8 prof_prepare

MHAParser::vfloat_mon_t mhachain::plugs_t::prof_prepare [private]

5.230.3.9 prof_release

MHAParser::vfloat_mon_t mhachain::plugs_t::prof_release [private]

5.230.3.10 prof_process

MHAParser::vfloat_mon_t mhachain::plugs_t::prof_process [private]

5.230.3.11 prof_process_tt

MHAParser::float_mon_t mhachain::plugs_t::prof_process↩_tt [private]

5.230.3.12 prof_process_load

MHAParser::vfloat_mon_t mhachain::plugs_t::prof_↩process_load [private]

5.230.3.13 proc_cnt

unsigned int mhachain::plugs_t::proc_cnt [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.230.3.14 prof_cfg `mhaconfig_t mhachain::plugs_t::prof_cfg [private]`

5.230.3.15 prof_load_con `MHAEvents::connector_t< mhachain::plugs_t> mhachain::plugs_t::prof_load_con [private]`

5.230.3.16 prof_tt_con `MHAEvents::connector_t< mhachain::plugs_t> mhachain::plugs_t::prof_tt_con [private]`

5.230.3.17 b_use_profiling `bool mhachain::plugs_t::b_use_profiling [private]`

5.230.3.18 tictoc `mha_platform_tictoc_t mhachain::plugs_t::tictoc [private]`

The documentation for this class was generated from the following files:

- **mha_generic_chain.h**
- **mha_generic_chain.cpp**

5.231 mhaconfig_t Struct Reference

MHA prepare configuration structure.

Inheritance diagram for mhaconfig_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- unsigned int **channels**
  Number of audio channels.
- unsigned int **domain**
  Signal domain (MHA_WAVEFORM or MHA_SPECTRUM)
- unsigned int **fragsize**
  Fragment size of waveform data.
- unsigned int **wndlen**
  Window length of spectral data.
- unsigned int **fftlen**
  FFT length of spectral data.
- **mha_real_t srate**
  Sampling rate in Hz.

5.231.1 Detailed Description

MHA prepare configuration structure.

This structure contains information about channel number and domain for input and output signals of a openMHA Plugin. Each plugin can change any of these parameters, e.g. by resampling of the signal. The only limitation is that the callback frequency is fixed (except for the plugins `db` and `dbasync`).

5.231.2 Member Data Documentation

5.231.2.1 channels `unsigned int mhaconfig_t::channels`

Number of audio channels.

5.231.2.2 domain `unsigned int mhaconfig_t::domain`

Signal domain (MHA_WAVEFORM or MHA_SPECTRUM)

© 2005-2021 HörTech gGmbH, Oldenburg

5.231.2.3 fragsize `unsigned int mhaconfig_t::fragsize`

Fragment size of waveform data.

5.231.2.4 wndlen `unsigned int mhaconfig_t::wndlen`

Window length of spectral data.

5.231.2.5 fftlen `unsigned int mhaconfig_t::fftlen`

FFT length of spectral data.

5.231.2.6 srate `mha_real_t mhaconfig_t::srate`

Sampling rate in Hz.

The documentation for this struct was generated from the following file:

- **mha.hh**

5.232 MHAEvents::connector_base_t Class Reference

Inheritance diagram for MHAEvents::connector_base_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **connector_base_t** ()
- virtual ~**connector_base_t** ()
- virtual void **emit_event** ()
- virtual void **emit_event** (const std::string &)
- virtual void **emit_event** (const std::string &, unsigned int, unsigned int)
- void **emitter_die** ()

Protected Attributes

- bool **emitter_is_alive**

5.232.1 Constructor & Destructor Documentation

5.232.1.1 connector_base_t() `MHAEvents::connector_base_t::connector_base_t ( )`

5.232.1.2 ~connector_base_t() `MHAEvents::connector_base_t::~connector_base_t ( ) [virtual]`

5.232.2 Member Function Documentation

5.232.2.1 emit_event() `[1/3]` `void MHAEvents::connector_base_t::emit_event ( ) [virtual]`

Reimplemented in **MHAEvents::connector_t< receiver_t >** (p. 857), **MHAEvents::connector_t< MHAOvlFilter::fscale_bw_t >** (p. 857), **MHAEvents::connector_t< MHAParser::mhapluginloader_t >** (p. 857), **MHAEvents::connector_t< mhachain::plugs_t >** (p. 857), and **MHAEvents::connector_t< MHAOvlFilter::fscale_t >** (p. 857).

© 2005-2021 HörTech gGmbH, Oldenburg

5.232.2.2 emit_event() [2/3]

```
void MHAEvents::connector_base_t::emit_event (
    const std::string & ) [virtual]
```

Reimplemented in **MHAEvents::connector_t< receiver_t >** (p. 858), **MHAEvents::connector_t< MHAOvlFilter::fscale_bw_t >** (p. 858), **MHAEvents::connector_t< MHAParser::mhapluginloader_t >** (p. 858), **MHAEvents::connector_t< mhachain::plugs_t >** (p. 858), and **MHAEvents::connector_t< MHAOvlFilter::fscale_t >** (p. 858).

5.232.2.3 emit_event() [3/3]

```
void MHAEvents::connector_base_t::emit_event (
    const std::string & ,
    unsigned int ,
    unsigned int ) [virtual]
```

Reimplemented in **MHAEvents::connector_t< receiver_t >** (p. 858), **MHAEvents::connector_t< MHAOvlFilter::fscale_bw_t >** (p. 858), **MHAEvents::connector_t< MHAParser::mhapluginloader_t >** (p. 858), **MHAEvents::connector_t< mhachain::plugs_t >** (p. 858), and **MHAEvents::connector_t< MHAOvlFilter::fscale_t >** (p. 858).

5.232.2.4 emitter_die()

```
void MHAEvents::connector_base_t::emitter_die ( )
```

5.232.3 Member Data Documentation

5.232.3.1 emitter_is_alive

```
bool MHAEvents::connector_base_t::emitter_is_alive [protected]
```

The documentation for this class was generated from the following files:

- **mha_event_emitter.h**
- **mha_events.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.233 MHAEvents::connector_t< receiver_t > Class Template Reference

Inheritance diagram for MHAEvents::connector_t< receiver_t >:

Public Member Functions

- **connector_t** (**emitter_t** ∗, receiver_t ∗, void(receiver_t::∗)())
- **connector_t** (**emitter_t** ∗, receiver_t ∗, void(receiver_t::∗)(const std::string &))
- **connector_t** (**emitter_t** ∗, receiver_t ∗, void(receiver_t::∗)(const std::string &, unsigned int, unsigned int))
- ∼**connector_t** ()

Private Member Functions

- void **emit_event** ()
- void **emit_event** (const std::string &)
- void **emit_event** (const std::string &, unsigned int, unsigned int)

Private Attributes

- **emitter_t** ∗ **emitter**
- receiver_t ∗ **receiver**
- void(receiver_t::∗ **eventhandler**)()
- void(receiver_t::∗ **eventhandler_s**)(const std::string &)
- void(receiver_t::∗ **eventhandler_suu**)(const std::string &, unsigned int, unsigned int)

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.233.1 Constructor & Destructor Documentation

5.233.1.1 connector_t() [1/3] template<class receiver_t >

```
MHAEvents::connector_t< receiver_t >::  connector_t (
            emitter_t * e,
            receiver_t * r,
            void(receiver_t::*)() rfun )
```

5.233.1.2 connector_t() [2/3] template<class receiver_t >

```
MHAEvents::connector_t< receiver_t >::  connector_t (
            emitter_t * e,
            receiver_t * r,
            void(receiver_t::*)(const std::string &) rfun )
```

5.233.1.3 connector_t() [3/3] template<class receiver_t >

```
MHAEvents::connector_t< receiver_t >::  connector_t (
            emitter_t * e,
            receiver_t * r,
            void(receiver_t::*)(const std::string &, unsigned int, unsigned int)
rfun )
```

5.233.1.4 ~connector_t() template<class receiver_t >

```
MHAEvents::connector_t< receiver_t >::~ connector_t
```

5.233.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.233.2.1 emit_event() [1/3] template<class receiver_t >

```
void MHAEvents::connector_t< receiver_t >::emit_event [private], [virtual]
```

Reimplemented from **MHAEvents::connector_base_t** (p. 854).

5.233.2.2 emit_event() [2/3] template<class receiver_t >

```
void MHAEvents::connector_t< receiver_t >::emit_event (
            const std::string & arg ) [private], [virtual]
```

Reimplemented from **MHAEvents::connector_base_t** (p. 854).

5.233.2.3 emit_event() [3/3] template<class receiver_t >

```
void MHAEvents::connector_t< receiver_t >::emit_event (
            const std::string & arg,
            unsigned int arg2,
            unsigned int arg3 ) [private], [virtual]
```

Reimplemented from **MHAEvents::connector_base_t** (p. 855).

5.233.3 Member Data Documentation

5.233.3.1 emitter template<class receiver_t >

```
emitter_t* MHAEvents::connector_t< receiver_t >::emitter [private]
```

5.233.3.2 receiver template<class receiver_t >

```
receiver_t* MHAEvents::connector_t< receiver_t >::receiver [private]
```

5.233.3.3 eventhandler template<class receiver_t >

```
void(receiver_t::* MHAEvents::connector_t< receiver_t >::eventhandler) () [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.233.3.4 eventhandler_s `template<class receiver_t >`

`void(receiver_t::* MHAEvents::connector_t< receiver_t >::eventhandler_s) (const std::string &) [private]`

5.233.3.5 eventhandler_suu `template<class receiver_t >`

`void(receiver_t::* MHAEvents::connector_t< receiver_t >::eventhandler_suu) (const std::string &, unsigned int, unsigned int) [private]`

The documentation for this class was generated from the following file:

- **mha_events.h**

5.234 MHAEvents::emitter_t Class Reference

Class for emitting openMHA events.

Public Member Functions

- ~**emitter_t** ()
- void **operator()** ()
  Emit an event without parameter.
- void **operator()** (const std::string &)
  Emit an event with string parameter.
- void **operator()** (const std::string &, unsigned int, unsigned int)
  Emit an event with string parameter and two unsigned int parameters.
- void **connect** (**connector_base_t** *)
- void **disconnect** (**connector_base_t** *)

Private Attributes

- std::list< **connector_base_t** * > **connections**

5.234.1 Detailed Description

Class for emitting openMHA events.

Use the template claas **MHAEvents::patchbay_t** (p. 861) for connecting to an emitter.

© 2005-2021 HörTech gGmbH, Oldenburg

5.234.2 Constructor & Destructor Documentation

5.234.2.1 ~emitter_t() `MHAEvents::emitter_t::~emitter_t ( )`

5.234.3 Member Function Documentation

5.234.3.1 operator()() [1/3] `void MHAEvents::emitter_t::operator() ( )`

Emit an event without parameter.

5.234.3.2 operator()() [2/3] `void MHAEvents::emitter_t::operator() ( const std::string & arg )`

Emit an event with string parameter.

5.234.3.3 operator()() [3/3] `void MHAEvents::emitter_t::operator() ( const std::string & arg, unsigned int arg2, unsigned int arg3 )`

Emit an event with string parameter and two unsigned int parameters.

5.234.3.4 connect() `void MHAEvents::emitter_t::connect ( connector_base_t * c )`

5.234.3.5 disconnect() `void MHAEvents::emitter_t::disconnect ( connector_base_t * c )`

© 2005-2021 HörTech gGmbH, Oldenburg

5.234.4 Member Data Documentation

5.234.4.1 connections `std::list< connector_base_t*> MHAEvents::emitter_t::connections [private]`

The documentation for this class was generated from the following files:

- **mha_event_emitter.h**
- **mha_events.cpp**

5.235 MHAEvents::patchbay_t< receiver_t > Class Template Reference

Patchbay which connects any event emitter with any member function of the parameter class.

Public Member Functions

- ~**patchbay_t** ()
- void **connect** (**emitter_t** *, receiver_t *, void(receiver_t::*)())

  Connect a receiver member function void (receiver_t::)() with an event emitter.*
- void **connect** (**emitter_t** *, receiver_t *, void(receiver_t::*)(const std::string &))

  Connect a receiver member function void (receiver_t::)(const std::string&) with an event emitter.*
- void **connect** (**emitter_t** *, receiver_t *, void(receiver_t::*)(const std::string &, unsigned int, unsigned int))

Private Attributes

- std::list< **connector_t**< receiver_t > * > **cons**

5.235.1 Detailed Description

template<class receiver_t>
class MHAEvents::patchbay_t< receiver_t >

Patchbay which connects any event emitter with any member function of the parameter class.

The connections created by the **connect()** (p. 862) function are hold until the destructor is called. To avoid access to invalid function pointers, it is required to destruct the patchbay before the receiver, usually by declaring the patchbay as a member of the receiver.

The receiver can be any claas or structure; the event callback can be either a member function without arguments or with const std::string& argument.

© 2005-2021 HörTech gGmbH, Oldenburg

5.235.2 Constructor & Destructor Documentation

5.235.2.1 ~patchbay_t() template<class receiver_t >

MHAEvents::patchbay_t< receiver_t >::~ **patchbay_t**

5.235.3 Member Function Documentation

5.235.3.1 connect() [1/3] template<class receiver_t >

```
void MHAEvents::patchbay_t< receiver_t >::connect (
            emitter_t * e,
           receiver_t * r,
           void(receiver_t::*)() rfun )
```

Connect a receiver member function void (receiver_t::∗)() with an event emitter.

Create a connection.

The connection is removed when the patchbay is destructed.

Parameters

e	Pointer to an event emitter
r	Pointer to the receiver
rfun	Pointer to a member function of the receiver class

5.235.3.2 connect() [2/3] template<class receiver_t >

```
void MHAEvents::patchbay_t< receiver_t >::connect (
            emitter_t * e,
           receiver_t * r,
           void(receiver_t::*)(const std::string &) rfun )
```

Connect a receiver member function void (receiver_t::∗)(const std::string&) with an event emitter.

Create a connection.

The connection is removed when the patchbay is destructed.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

e	Pointer to an event emitter
r	Pointer to the receiver
rfun	Pointer to a member function of the receiver class

5.235.3.3 connect() [3/3]

```
template<class receiver_t >
void MHAEvents::patchbay_t< receiver_t >::connect (
            emitter_t * e,
           receiver_t * r,
           void(receiver_t::*)(const std::string &, unsigned int, unsigned int)
rfun )
```

5.235.4 Member Data Documentation

5.235.4.1 cons

```
template<class receiver_t >
std::list< connector_t<receiver_t>*> MHAEvents::patchbay_t< receiver_t >::cons
[private]
```

The documentation for this class was generated from the following file:

- **mha_events.h**

5.236 MHAFilter::adapt_filter_param_t Class Reference

Public Member Functions

- **adapt_filter_param_t** (**mha_real_t** imu, bool ierr_in)

Public Attributes

- **mha_real_t mu**
- bool **err_in**

© 2005-2021 HörTech gGmbH, Oldenburg

5.236.1 Constructor & Destructor Documentation

5.236.1.1 adapt_filter_param_t()

```
MHAFilter::adapt_filter_param_t::adapt_filter_↩
param_t (
            mha_real_t imu,
           bool ierr_in )
```

5.236.2 Member Data Documentation

5.236.2.1 mu

```
mha_real_t MHAFilter::adapt_filter_param_t::mu
```

5.236.2.2 err_in

```
bool MHAFilter::adapt_filter_param_t::err_in
```

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.237 MHAFilter::adapt_filter_state_t Class Reference

Public Member Functions

- **adapt_filter_state_t** (int **ntaps**, int **nchannels**)
- void **filter** (**mha_wave_t** y, **mha_wave_t** e, **mha_wave_t** x, **mha_wave_t** d, **mha_↩real_t** mu, bool err_in)

Private Attributes

- int **ntaps**
- int **nchannels**
- **MHASignal::waveform_t W**
- **MHASignal::waveform_t X**
- **MHASignal::waveform_t od**
- **MHASignal::waveform_t oy**

© 2005-2021 HörTech gGmbH, Oldenburg

5.237.1 Constructor & Destructor Documentation

5.237.1.1 adapt_filter_state_t()

```
MHAFilter::adapt_filter_state_t::adapt_filter_↩
state_t (
            int ntaps,
            int nchannels )
```

5.237.2 Member Function Documentation

5.237.2.1 filter()

```
void MHAFilter::adapt_filter_state_t::filter (
            mha_wave_t y,
            mha_wave_t e,
            mha_wave_t x,
            mha_wave_t d,
            mha_real_t mu,
           bool err_in )
```

5.237.3 Member Data Documentation

5.237.3.1 ntaps

```
int MHAFilter::adapt_filter_state_t::ntaps [private]
```

5.237.3.2 nchannels

```
int MHAFilter::adapt_filter_state_t::nchannels [private]
```

5.237.3.3 W

```
MHASignal::waveform_t MHAFilter::adapt_filter_state_t::W [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.237.3.4 X `MHASignal::waveform_t MHAFilter::adapt_filter_state_t::X [private]`

5.237.3.5 od `MHASignal::waveform_t MHAFilter::adapt_filter_state_t::od [private]`

5.237.3.6 oy `MHASignal::waveform_t MHAFilter::adapt_filter_state_t::oy [private]`

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.238 MHAFilter::adapt_filter_t Class Reference

Adaptive filter.

Inheritance diagram for MHAFilter::adapt_filter_t:

Public Member Functions

- **adapt_filter_t** (std::string)
- void **filter** (**mha_wave_t** y, **mha_wave_t** e, **mha_wave_t** x, **mha_wave_t** d)
- void **set_channelcnt** (unsigned int)

Private Member Functions

- void **update_mu** ()
- void **update_ntaps** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::float_t mu**
- **MHAParser::int_t ntaps**
- **MHAParser::bool_t err_in**
- **MHAEvents::patchbay_t< adapt_filter_t > connector**
- unsigned int **nchannels**

Additional Inherited Members

5.238.1 Detailed Description

Adaptive filter.

5.238.2 Constructor & Destructor Documentation

5.238.2.1 adapt_filter_t()

```
MHAFilter::adapt_filter_t::adapt_filter_t (
    std::string help )
```

5.238.3 Member Function Documentation

5.238.3.1 filter()

```
void MHAFilter::adapt_filter_t::filter (
    mha_wave_t y,
    mha_wave_t e,
    mha_wave_t x,
    mha_wave_t d )
```

5.238.3.2 set_channelcnt()

```
void MHAFilter::adapt_filter_t::set_channelcnt (
    unsigned int nch )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.238.3.3 update_mu() `void MHAFilter::adapt_filter_t::update_mu ( ) [private]`

5.238.3.4 update_ntaps() `void MHAFilter::adapt_filter_t::update_ntaps ( ) [private]`

5.238.4 Member Data Documentation

5.238.4.1 mu `MHAParser::float_t MHAFilter::adapt_filter_t::mu [private]`

5.238.4.2 ntaps `MHAParser::int_t MHAFilter::adapt_filter_t::ntaps [private]`

5.238.4.3 err_in `MHAParser::bool_t MHAFilter::adapt_filter_t::err_in [private]`

5.238.4.4 connector `MHAEvents::patchbay_t< adapt_filter_t> MHAFilter::adapt_filter_t::connector [private]`

5.238.4.5 nchannels `unsigned int MHAFilter::adapt_filter_t::nchannels [private]`

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.239 MHAFilter::blockprocessing_polyphase_resampling_t Class Reference

A class that does polyphase resampling and takes into account block processing.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **blockprocessing_polyphase_resampling_t** (float source_srate, unsigned source_↩ fragsize, float target_srate, unsigned target_fragsize, float nyquist_ratio, float irslen, unsigned nchannels, bool add_delay)

  Contructs a polyphase resampling filter that can be used for blockprocessing with the given parameters.
- virtual ∼**blockprocessing_polyphase_resampling_t** ()
- void **write** (**mha_wave_t** &signal)

  Write signal to the ringbuffer.
- void **read** (**mha_wave_t** &signal)

  Read resampled signal.
- bool **can_read** () const

  Checks if the resampling ring buffer can produce another output signal block.

Private Attributes

- **polyphase_resampling_t** ∗ **resampling**
- unsigned **fragsize_in**
- unsigned **fragsize_out**
- unsigned **num_channels**

5.239.1 Detailed Description

A class that does polyphase resampling and takes into account block processing.

5.239.2 Constructor & Destructor Documentation

5.239.2.1 blockprocessing_polyphase_resampling_t()

```
MHAFilter::blockprocessing_↩
polyphase_resampling_t::blockprocessing_polyphase_resampling_t (
            float source_srate,
            unsigned source_fragsize,
            float target_srate,
            unsigned target_fragsize,
            float nyquist_ratio,
            float irslen,
            unsigned nchannels,
            bool add_delay )
```

Contructs a polyphase resampling filter that can be used for blockprocessing with the given parameters.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

source_srate	Source sampling rate / Hz
source_fragsize	Fragment size of incoming audio blocks / frames at source_srate
target_srate	Target sampling rate / Hz
target_fragsize	Fragment size of produced audio blocks / frames at target_srate
nyquist_ratio	Low pass filter cutoff frequency relative to the nyquist frequency of the smaller of the two sampling rates. Example values: 0.8, 0.9
irslen	Impulse response length used for low pass filtering / s
nchannels	Number of audio channels
add_delay	To avoid underruns, a delay is generally necessary for round trip block size adaptations. It is only necessary to add this delay to one of the two resampling chains. Set this parameter to true for the first resampling object of a round trip pair. It will add the necessary delay, and calculate the size of the ring buffer appropriately, When set to false, only the ringbuffer size will be set sufficiently.

5.239.2.2 ~blockprocessing_polyphase_resampling_t() `virtual MHAFilter::blockprocessing_polyphase_resampling_t::~blockprocessing_polyphase_resampling_t ( ) [inline], [virtual]`

5.239.3 Member Function Documentation

5.239.3.1 write() `void MHAFilter::blockprocessing_polyphase_resampling_t::write ( mha_wave_t & signal )`

Write signal to the ringbuffer.

Parameters

signal	input signal in original sampling rate

Exceptions

MHA_Error *(p. 764)*	Raises exception if there is not enough room, if the number of channels does not match, or if the number of frames is not equal to the number specified in the constructor

© 2005-2021 HörTech gGmbH, Oldenburg

5.239.3.2 read() `void MHAFilter::blockprocessing_polyphase_resampling_t::read ( mha_wave_t & signal )`

Read resampled signal.

Will perform the resampling and remove no longer needed samples from the input buffer.

Parameters

signal	buffer to write the resampled signal to.

Exceptions

MHA_Error *(p. 764)*	Raises exception if there is not enough input signal, if the number of channels of frames does not match.

5.239.3.3 can_read() `bool MHAFilter::blockprocessing_polyphase_resampling_t::can_read ( ) const [inline]`

Checks if the resampling ring buffer can produce another output signal block.

5.239.4 Member Data Documentation

5.239.4.1 resampling `polyphase_resampling_t* MHAFilter::blockprocessing_polyphase_resampling_t::resampling [private]`

5.239.4.2 fragsize_in `unsigned MHAFilter::blockprocessing_polyphase_resampling_t::fragsize_in [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.239.4.3 fragsize_out `unsigned MHAFilter::blockprocessing_polyphase_resampling_t↩ ::fragsize_out [private]`

5.239.4.4 num_channels `unsigned MHAFilter::blockprocessing_polyphase_resampling↩ _t::num_channels [private]`

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.240 MHAFilter::complex_bandpass_t Class Reference

Complex bandpass filter.

Public Member Functions

- **complex_bandpass_t** (std::vector< **mha_complex_t** > A, std::vector< **mha_↩ complex_t** > B)

  Constructor with filter coefficients (one per channel)
- void **set_state** (**mha_real_t** val)
- void **set_state** (std::vector< **mha_real_t** > val)
- void **set_state** (**mha_complex_t** val)
- void **set_weights** (std::vector< **mha_complex_t** > new_B)

  Allow to modify the input weights at a later stage.
- std::vector< **mha_complex_t** > **get_weights** () const
- void **filter** (const **mha_wave_t** &X, **mha_spec_t** &Y)

  Filter method for real value input.
- void **filter** (const **mha_wave_t** &X, **mha_wave_t** &Yre, **mha_wave_t** &Yim)

  Filter method for real value input.
- void **filter** (const **mha_spec_t** &X, **mha_spec_t** &Y)

  Filter method for complex value input.
- void **filter** (const **mha_wave_t** &Xre, const **mha_wave_t** &Xim, **mha_wave_t** &Yre, **mha_wave_t** &Yim)

  Filter method for complex value input.
- std::string **inspect** () const

© 2005-2021 HörTech gGmbH, Oldenburg

Static Public Member Functions

- static std::vector< **mha_complex_t** > **creator_A** (std::vector< **mha_real_t** > cf, std↩ ::vector< **mha_real_t** > bw, **mha_real_t** srate, unsigned int order)
- static std::vector< **mha_complex_t** > **creator_B** (std::vector< **mha_complex_t** > A, unsigned int order)

Private Attributes

- std::vector< **mha_complex_t** > **A_**
- std::vector< **mha_complex_t** > **B_**
- std::vector< **mha_complex_t** > **Yn**

5.240.1 Detailed Description

Complex bandpass filter.

5.240.2 Constructor & Destructor Documentation

5.240.2.1 complex_bandpass_t()

```
MHAFilter::complex_bandpass_t::complex_bandpass_t
(
            std::vector< mha_complex_t > A,
            std::vector< mha_complex_t > B )
```

Constructor with filter coefficients (one per channel)

Parameters

A	complex filter coefficients, one per band
B	complex weights

5.240.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.240.3.1 creator_A()

```
std::vector< mha_complex_t > MHAFilter::complex_bandpass_t::creator_A (
            std::vector< mha_real_t > cf,
            std::vector< mha_real_t > bw,
             mha_real_t srate,
            unsigned int order ) [static]
```

5.240.3.2 creator_B()

```
std::vector< mha_complex_t > MHAFilter::complex_bandpass_t::creator_B (
            std::vector< mha_complex_t > A,
            unsigned int order ) [static]
```

5.240.3.3 set_state() [1/3]

```
void MHAFilter::complex_bandpass_t::set_state (
             mha_real_t val )
```

5.240.3.4 set_state() [2/3]

```
void MHAFilter::complex_bandpass_t::set_state (
            std::vector< mha_real_t > val )
```

5.240.3.5 set_state() [3/3]

```
void MHAFilter::complex_bandpass_t::set_state (
             mha_complex_t val )
```

5.240.3.6 set_weights()

```
void MHAFilter::complex_bandpass_t::set_weights (
            std::vector< mha_complex_t > new_B )
```

Allow to modify the input weights at a later stage.

5.240.3.7 get_weights()

```
std::vector< mha_complex_t> MHAFilter::complex_bandpass_t::get_weights ( ) const [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.240.3.8 filter() [1/4]

```
void MHAFilter::complex_bandpass_t::filter (
    const mha_wave_t & X,
    mha_spec_t & Y ) [inline]
```

Filter method for real value input.

5.240.3.9 filter() [2/4]

```
void MHAFilter::complex_bandpass_t::filter (
    const mha_wave_t & X,
    mha_wave_t & Yre,
    mha_wave_t & Yim ) [inline]
```

Filter method for real value input.

5.240.3.10 filter() [3/4]

```
void MHAFilter::complex_bandpass_t::filter (
    const mha_spec_t & X,
    mha_spec_t & Y ) [inline]
```

Filter method for complex value input.

5.240.3.11 filter() [4/4]

```
void MHAFilter::complex_bandpass_t::filter (
    const mha_wave_t & Xre,
    const mha_wave_t & Xim,
    mha_wave_t & Yre,
    mha_wave_t & Yim ) [inline]
```

Filter method for complex value input.

5.240.3.12 inspect()

```
std::string MHAFilter::complex_bandpass_t::inspect ( ) const [inline]
```

5.240.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.240.4.1 A_ `std::vector< mha_complex_t> MHAFilter::complex_bandpass_t::A_ [private]`

5.240.4.2 B_ `std::vector< mha_complex_t> MHAFilter::complex_bandpass_t::B_ [private]`

5.240.4.3 Yn `std::vector< mha_complex_t> MHAFilter::complex_bandpass_t::Yn [private]`

The documentation for this class was generated from the following files:

- **complex_filter.h**
- **complex_filter.cpp**

5.241 MHAFilter::diff_t Class Reference

Differentiator class (non-normalized)

Inheritance diagram for MHAFilter::diff_t:

Public Member Functions

- **diff_t** (unsigned int ch)

Additional Inherited Members

5.241.1 Detailed Description

Differentiator class (non-normalized)

© 2005-2021 HörTech gGmbH, Oldenburg

5.241.2 Constructor & Destructor Documentation

5.241.2.1 diff_t() `MHAFilter::diff_t::diff_t ( unsigned int ch )`

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.242 MHAFilter::fftfilter_t Class Reference

FFT based FIR filter implementation.

Inheritance diagram for MHAFilter::fftfilter_t:

Public Member Functions

- **fftfilter_t** (unsigned int **fragsize**, unsigned int **channels**, unsigned int **fftlen**)
  Constructor.
- ∼**fftfilter_t** ()
- void **update_coeffs** (const **mha_wave_t** ∗pwIRS)
  Update the set of coefficients.
- void **filter** (const **mha_wave_t** ∗pwIn, **mha_wave_t** ∗∗ppwOut, const **mha_wave_t** ∗pwIRS)
  Apply filter with changing coefficients to a waveform fragment.
- void **filter** (const **mha_wave_t** ∗pwIn, **mha_wave_t** ∗∗ppwOut)
  Apply filter to waveform fragment, without changing the coefficients.
- void **filter** (const **mha_wave_t** ∗pwIn, **mha_wave_t** ∗∗ppwOut, const **mha_spec_t** ∗psWeights)
  Apply filter with changing coefficients to a waveform fragment.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- unsigned int **fragsize**
- unsigned int **channels**
- unsigned int **fftlen**
- **MHASignal::waveform_t wInput_fft**
- **mha_wave_t wInput**
- **MHASignal::waveform_t wOutput_fft**
- **mha_wave_t wOutput**
- **MHASignal::spectrum_t sInput**
- **MHASignal::spectrum_t sWeights**
- **MHASignal::waveform_t wIRS_fft**
- **mha_fft_t fft**

5.242.1 Detailed Description

FFT based FIR filter implementation.

The maximal number of coefficients can be FFT length - fragsize + 1.

5.242.2 Constructor & Destructor Documentation

5.242.2.1 fftfilter_t()

```
MHAFilter::fftfilter_t::fftfilter_t (
    unsigned int fragsize,
    unsigned int channels,
    unsigned int fftlen )
```

Constructor.

Parameters

fragsize	Number of frames expected in input signal (each cycle).
channels	Number of channels expected in input signal.
fftlen	FFT length of filter.

5.242.2.2 ~fftfilter_t()

```
MHAFilter::fftfilter_t::~fftfilter_t ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.242.3 Member Function Documentation

5.242.3.1 update_coeffs()

```
void MHAFilter::fftfilter_t::update_coeffs (
    const mha_wave_t * pwIRS )
```

Update the set of coefficients.

Parameters

pwIRS	Coefficients structure

Note

The number of channels in h must match the number of channels given in the constructor. The filter length is limited to fftlen-fragsize+1 (longer IRS will be shortened).

5.242.3.2 filter() [1/3]

```
void MHAFilter::fftfilter_t::filter (
    const mha_wave_t * pwIn,
    mha_wave_t ** ppwOut,
    const mha_wave_t * pwIRS )
```

Apply filter with changing coefficients to a waveform fragment.

Parameters

pw↩ In	Input signal pointer.

Return values

ppwOut	Pointer to output signal pointer, will be set to a valid signal.

Parameters

pwIRS	Pointer to FIR coefficients structure.

© 2005-2021 HörTech gGmbH, Oldenburg

5.242.3.3 filter() [2/3]

```
void MHAFilter::fftfilter_t::filter (
    const mha_wave_t * pwIn,
    mha_wave_t ** ppwOut )
```

Apply filter to waveform fragment, without changing the coefficients.

Parameters

pw↩ In	Input signal pointer.

Return values

ppwOut	Pointer to output signal pointer, will be set to a valid signal

5.242.3.4 filter() [3/3]

```
void MHAFilter::fftfilter_t::filter (
    const mha_wave_t * pwIn,
    mha_wave_t ** ppwOut,
    const mha_spec_t * psWeights )
```

Apply filter with changing coefficients to a waveform fragment.

Parameters

pw↩ In	Input signal pointer.

Return values

ppwOut	Pointer to output signal pointer, will be set to a valid signal.

Parameters

psWeights	Pointer to filter weights structure.

5.242.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.242.4.1 fragsize `unsigned int MHAFilter::fftfilter_t::fragsize [private]`

5.242.4.2 channels `unsigned int MHAFilter::fftfilter_t::channels [private]`

5.242.4.3 fftlen `unsigned int MHAFilter::fftfilter_t::fftlen [private]`

5.242.4.4 wInput_fft `MHASignal::waveform_t MHAFilter::fftfilter_t::wInput_fft [private]`

5.242.4.5 wInput `mha_wave_t MHAFilter::fftfilter_t::wInput [private]`

5.242.4.6 wOutput_fft `MHASignal::waveform_t MHAFilter::fftfilter_t::wOutput_fft [private]`

5.242.4.7 wOutput `mha_wave_t MHAFilter::fftfilter_t::wOutput [private]`

5.242.4.8 sInput `MHASignal::spectrum_t MHAFilter::fftfilter_t::sInput [private]`

5.242.4.9 sWeights `MHASignal::spectrum_t MHAFilter::fftfilter_t::sWeights [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.242.4.10 wIRS_fft `MHASignal::waveform_t MHAFilter::fftfilter_t::wIRS_fft [private]`

5.242.4.11 fft `mha_fft_t MHAFilter::fftfilter_t::fft [private]`

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.243 MHAFilter::fftfilterbank_t Class Reference

FFT based FIR filterbank implementation.

Inheritance diagram for MHAFilter::fftfilterbank_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **fftfilterbank_t** (unsigned int **fragsize**, unsigned int **inputchannels**, unsigned int **firchannels**, unsigned int **fftlen**)
  Constructor.
- ∼**fftfilterbank_t** ()
- void **update_coeffs** (const **mha_wave_t** ∗h)
  Update the set of coefficients.
- void **filter** (const **mha_wave_t** ∗s_in, **mha_wave_t** ∗∗s_out, const **mha_wave_t** ∗h)
  Apply filter with changing coefficients to a waveform fragment.
- void **filter** (const **mha_wave_t** ∗s_in, **mha_wave_t** ∗∗s_out)
  Apply filter to waveform fragment, without changing the coefficients.
- const **mha_wave_t** ∗ **get_irs** () const
  Return the current IRS.

Private Attributes

- unsigned int **fragsize**
- unsigned int **inputchannels**
- unsigned int **firchannels**
- unsigned int **outputchannels**
- unsigned int **fftlen**
- **MHASignal::waveform_t hw**
- **MHASignal::spectrum_t Hs**
- **MHASignal::waveform_t xw**
- **MHASignal::spectrum_t Xs**
- **MHASignal::waveform_t yw**
- **MHASignal::spectrum_t Ys**
- **MHASignal::waveform_t yw_temp**
- **MHASignal::waveform_t tail**
- **mha_fft_t fft**

5.243.1 Detailed Description

FFT based FIR filterbank implementation.

This class convolves n input channels with m filter coefficient sets and returns n∗m output channels.

The maximal number of coefficients can be FFT length - fragsize + 1.

5.243.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.243.2.1 fftfilterbank_t()

```
MHAFilter::fftfilterbank_t::fftfilterbank_t (
    unsigned int fragsize,
    unsigned int inputchannels,
    unsigned int firchannels,
    unsigned int fftlen )
```

Constructor.

Parameters

fragsize	Number of frames expected in input signal (each cycle).
inputchannels	Number of channels expected in input signal.
firchannels	Number of channels expected in FIR filter coefficients (= number of bands).
fftlen	FFT length of filter.

The number of output channels is inputchannels∗firchannels.

5.243.2.2 ∼fftfilterbank_t()

```
MHAFilter::fftfilterbank_t::~fftfilterbank_t ( )
```

5.243.3 Member Function Documentation

5.243.3.1 update_coeffs()

```
void MHAFilter::fftfilterbank_t::update_coeffs (
    const mha_wave_t * h )
```

Update the set of coefficients.

Parameters

h	Coefficients structure

Note

The number of channels in h must match the number of channels given in the constructor, and the number of frames can not be more than fftlen-fragsize+1.

© 2005-2021 HörTech gGmbH, Oldenburg

5.243.3.2 filter() [1/2]

```
void MHAFilter::fftfilterbank_t::filter (
    const mha_wave_t * s_in,
    mha_wave_t ** s_out,
    const mha_wave_t * h )
```

Apply filter with changing coefficients to a waveform fragment.

Parameters

s_in	Input signal pointer.

Return values

s_out	Pointer to output signal pointer, will be set to a valid signal

Parameters

h	FIR coefficients

5.243.3.3 filter() [2/2]

```
void MHAFilter::fftfilterbank_t::filter (
    const mha_wave_t * s_in,
    mha_wave_t ** s_out )
```

Apply filter to waveform fragment, without changing the coefficients.

Parameters

s_in	Input signal pointer.

Return values

s_out	Pointer to output signal pointer, will be set to a valid signal

5.243.3.4 get_irs()

```
const mha_wave_t* MHAFilter::fftfilterbank_t::get_irs ( ) const [inline]
```

Return the current IRS.

© 2005-2021 HörTech gGmbH, Oldenburg

5.243.4 Member Data Documentation

5.243.4.1 fragsize `unsigned int MHAFilter::fftfilterbank_t::fragsize [private]`

5.243.4.2 inputchannels `unsigned int MHAFilter::fftfilterbank_t::inputchannels [private]`

5.243.4.3 firchannels `unsigned int MHAFilter::fftfilterbank_t::firchannels [private]`

5.243.4.4 outputchannels `unsigned int MHAFilter::fftfilterbank_t::outputchannels [private]`

5.243.4.5 fftlen `unsigned int MHAFilter::fftfilterbank_t::fftlen [private]`

5.243.4.6 hw `MHASignal::waveform_t MHAFilter::fftfilterbank_t::hw [private]`

5.243.4.7 Hs `MHASignal::spectrum_t MHAFilter::fftfilterbank_t::Hs [private]`

5.243.4.8 xw `MHASignal::waveform_t MHAFilter::fftfilterbank_t::xw [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.243.4.9 Xs

`MHASignal::spectrum_t MHAFilter::fftfilterbank_t::Xs [private]`

5.243.4.10 yw

`MHASignal::waveform_t MHAFilter::fftfilterbank_t::yw [private]`

5.243.4.11 Ys

`MHASignal::spectrum_t MHAFilter::fftfilterbank_t::Ys [private]`

5.243.4.12 yw_temp

`MHASignal::waveform_t MHAFilter::fftfilterbank_t::yw_temp [private]`

5.243.4.13 tail

`MHASignal::waveform_t MHAFilter::fftfilterbank_t::tail [private]`

5.243.4.14 fft

`mha_fft_t MHAFilter::fftfilterbank_t::fft [private]`

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.244 MHAFilter::filter_t Class Reference

Generic IIR filter class.

Inheritance diagram for MHAFilter::filter_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **filter_t** (unsigned int ch, unsigned int lena, unsigned int lenb)

  Constructor.
- **filter_t** (unsigned int ch, const std::vector< **mha_real_t** > &vA, const std::vector< **mha_real_t** > &vB)

  Constructor with initialization of coefficients.
- **filter_t** (const **MHAFilter::filter_t** &src)

  Copy constructor.
- **filter_t** & **operator=** (const **MHAFilter::filter_t** &)=delete

  Assignment operator is not implemented.
- ∼**filter_t** ()
- void **filter** (**mha_wave_t** ∗out, const **mha_wave_t** ∗in)

  Filter all channels in a waveform structure.
- void **filter** (**mha_real_t** ∗dest, const **mha_real_t** ∗src, unsigned int dframes, unsigned int frame_dist, unsigned int channel_dist, unsigned int channel_begin, unsigned int channel_end)

  Filter parts of a waveform structure.
- **mha_real_t** **filter** (**mha_real_t** x, unsigned int ch)

  Filter one sample.
- unsigned int **get_len_A** () const

  Return length of recursive coefficients.
- unsigned int **get_len_B** () const

  Return length of non-recursive coefficients.

Public Attributes

- double ∗ **A**

  Pointer to recursive coefficients.
- double ∗ **B**

  Pointer to non-recursive coefficients.

Private Attributes

- unsigned int **len_A**
- unsigned int **len_B**
- unsigned int **len**
- unsigned int **channels**
- double ∗ **state**

© 2005-2021 HörTech gGmbH, Oldenburg

5.244.1 Detailed Description

Generic IIR filter class.

This class implements a generic multichannel IIR filter. It is realized as direct form II. It can work on any float array or on **mha_wave_t** (p. 840) structs. The filter coefficients can be directly accessed.

5.244.2 Constructor & Destructor Documentation

5.244.2.1 filter_t() [1/3]

```
MHAFilter::filter_t::filter_t (
    unsigned int ch,
    unsigned int lena,
    unsigned int lenb )
```

Constructor.

Parameters

ch	Number of channels
lena	Number of recursive coefficients
lenb	Number of non-recursive coefficients

5.244.2.2 filter_t() [2/3]

```
MHAFilter::filter_t::filter_t (
    unsigned int ch,
    const std::vector< mha_real_t > & vA,
    const std::vector< mha_real_t > & vB )
```

Constructor with initialization of coefficients.

Parameters

ch	Number of channels.
vA	Recursive coefficients.
vB	Non-recursive coefficients.

© 2005-2021 HörTech gGmbH, Oldenburg

5.244.2.3 filter_t() [3/3]

```
MHAFilter::filter_t::filter_t (
    const MHAFilter::filter_t & src )
```

Copy constructor.

5.244.2.4 ~filter_t()

```
MHAFilter::filter_t::~filter_t ( )
```

5.244.3 Member Function Documentation

5.244.3.1 operator=()

```
filter_t& MHAFilter::filter_t::operator= (
    const MHAFilter::filter_t & ) [delete]
```

Assignment operator is not implemented.

Use copy constructor instead.

5.244.3.2 filter() [1/3]

```
void MHAFilter::filter_t::filter (
    mha_wave_t * out,
    const mha_wave_t * in )
```

Filter all channels in a waveform structure.

Parameters

out	Output signal
in	Input signal

5.244.3.3 filter() [2/3]

```
void MHAFilter::filter_t::filter (
    mha_real_t * dest,
    const mha_real_t * src,
    unsigned int dframes,
    unsigned int frame_dist,
    unsigned int channel_dist,
    unsigned int channel_begin,
    unsigned int channel_end )
```

© 2005-2021 HörTech gGmbH, Oldenburg

Filter parts of a waveform structure.

Parameters

dest	Output signal.
src	Input signal.
dframes	Number of frames to be filtered.
frame_dist	Index distance between frames of one channel
channel_dist	Index distance between audio channels
channel_begin	Number of first channel to be processed
channel_end	Number of last channel to be processed

5.244.3.4 filter() [3/3]

```
mha_real_t MHAFilter::filter_t::filter (
    mha_real_t x,
    unsigned int ch )
```

Filter one sample.

Parameters

x	Input value
ch	Channel number to use in filter state

5.244.3.5 get_len_A()

```
unsigned int MHAFilter::filter_t::get_len_A ( ) const [inline]
```

Return length of recursive coefficients.

5.244.3.6 get_len_B()

```
unsigned int MHAFilter::filter_t::get_len_B ( ) const [inline]
```

Return length of non-recursive coefficients.

5.244.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.244.4.1 A `double* MHAFilter::filter_t::A`

Pointer to recursive coefficients.

5.244.4.2 B `double* MHAFilter::filter_t::B`

Pointer to non-recursive coefficients.

5.244.4.3 len_A `unsigned int MHAFilter::filter_t::len_A [private]`

5.244.4.4 len_B `unsigned int MHAFilter::filter_t::len_B [private]`

5.244.4.5 len `unsigned int MHAFilter::filter_t::len [private]`

5.244.4.6 channels `unsigned int MHAFilter::filter_t::channels [private]`

5.244.4.7 state `double* MHAFilter::filter_t::state [private]`

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.245 MHAFilter::gamma_flt_t Class Reference

Class for gammatone filter.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **gamma_flt_t** (std::vector< **mha_real_t** > cf, std::vector< **mha_real_t** > bw, **mha_real_t** srate, unsigned int order)
  Constructor.
- ~**gamma_flt_t** ()
- void **operator()** (**mha_wave_t** &X, **mha_spec_t** &Y)
  Filter method.
- void **operator()** (**mha_wave_t** &X, **mha_wave_t** &Yre, **mha_wave_t** &Yim)
  Filter method.
- void **operator()** (**mha_wave_t** &Yre, **mha_wave_t** &Yim, unsigned int stage)
  Filter method for specific stage.
- void **phase_correction** (unsigned int desired_delay, unsigned int inchannels)
- void **set_weights** (std::vector< **mha_complex_t** > new_B)
- void **set_weights** (unsigned int stage, std::vector< **mha_complex_t** > new_B)
- std::vector< **mha_complex_t** > **get_weights** () const
- std::vector< **mha_complex_t** > **get_weights** (unsigned int stage) const
- std::vector< **mha_real_t** > **get_resynthesis_gain** () const
- void **reset_state** ()
- const std::vector< **mha_complex_t** > & **get_A** ()
- std::string **inspect** () const

Private Attributes

- std::vector< **mha_complex_t** > **A**
- std::vector< **complex_bandpass_t** > **GF**
- **MHASignal::delay_t** * **delay**
- std::vector< int > **envelope_delay**
- std::vector< **mha_real_t** > **resynthesis_gain**
- std::vector< **mha_real_t** > **cf_**
- std::vector< **mha_real_t** > **bw_**
- **mha_real_t** **srate_**

5.245.1 Detailed Description

Class for gammatone filter.

5.245.2 Constructor & Destructor Documentation

5.245.2.1 gamma_flt_t()

```
MHAFilter::gamma_flt_t::gamma_flt_t (
    std::vector< mha_real_t > cf,
    std::vector< mha_real_t > bw,
    mha_real_t srate,
    unsigned int order )
```

Constructor.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

cf	Center frequency in Hz.
bw	Bandwidth in Hz (same number of entries as in cf).
srate	Sampling frequency in Hz.
order	Filter order.

5.245.2.2 ~gamma_flt_t() `MHAFilter::gamma_flt_t::~gamma_flt_t ( )`

5.245.3 Member Function Documentation

5.245.3.1 operator()() [1/3]

```
void MHAFilter::gamma_flt_t::operator() (
    mha_wave_t & X,
    mha_spec_t & Y ) [inline]
```

Filter method.

5.245.3.2 operator()() [2/3]

```
void MHAFilter::gamma_flt_t::operator() (
    mha_wave_t & X,
    mha_wave_t & Yre,
    mha_wave_t & Yim ) [inline]
```

Filter method.

5.245.3.3 operator()() [3/3]

```
void MHAFilter::gamma_flt_t::operator() (
    mha_wave_t & Yre,
    mha_wave_t & Yim,
    unsigned int stage ) [inline]
```

Filter method for specific stage.

© 2005-2021 HörTech gGmbH, Oldenburg

5.245.3.4 phase_correction()

```
void MHAFilter::gamma_flt_t::phase_correction (
    unsigned int desired_delay,
    unsigned int inchannels )
```

5.245.3.5 set_weights() [1/2]

```
void MHAFilter::gamma_flt_t::set_weights (
    std::vector< mha_complex_t > new_B )
```

5.245.3.6 set_weights() [2/2]

```
void MHAFilter::gamma_flt_t::set_weights (
    unsigned int stage,
    std::vector< mha_complex_t > new_B )
```

5.245.3.7 get_weights() [1/2]

```
std::vector< mha_complex_t> MHAFilter::gamma_flt_t↩
::get_weights ( ) const [inline]
```

5.245.3.8 get_weights() [2/2]

```
std::vector< mha_complex_t> MHAFilter::gamma_flt_t↩
::get_weights (
    unsigned int stage ) const [inline]
```

5.245.3.9 get_resynthesis_gain()

```
std::vector< mha_real_t> MHAFilter::gamma_flt_t↩
::get_resynthesis_gain ( ) const [inline]
```

5.245.3.10 reset_state()

```
void MHAFilter::gamma_flt_t::reset_state ( )
```

5.245.3.11 get_A()

```
const std::vector< mha_complex_t>& MHAFilter::gamma_flt_t↩
::get_A ( ) [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.245.3.12 inspect() `std::string MHAFilter::gamma_flt_t::inspect ( ) const [inline]`

5.245.4 Member Data Documentation

5.245.4.1 A `std::vector< mha_complex_t> MHAFilter::gamma_flt_t::A [private]`

5.245.4.2 GF `std::vector< complex_bandpass_t> MHAFilter::gamma_flt_t::GF [private]`

5.245.4.3 delay `MHASignal::delay_t* MHAFilter::gamma_flt_t::delay [private]`

5.245.4.4 envelope_delay `std::vector<int> MHAFilter::gamma_flt_t::envelope_delay [private]`

5.245.4.5 resynthesis_gain `std::vector< mha_real_t> MHAFilter::gamma_flt_t::resynthesis↩_gain [private]`

5.245.4.6 cf_ `std::vector< mha_real_t> MHAFilter::gamma_flt_t::cf_ [private]`

5.245.4.7 bw_ `std::vector< mha_real_t> MHAFilter::gamma_flt_t::bw_ [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.245.4.8 srate_ `mha_real_t MHAFilter::gamma_flt_t::srate_ [private]`

The documentation for this class was generated from the following files:

- **complex_filter.h**
- **complex_filter.cpp**

5.246 MHAFilter::iir_filter_state_t Class Reference

Inheritance diagram for MHAFilter::iir_filter_state_t:

Public Member Functions

- **iir_filter_state_t** (unsigned int **channels**, std::vector< float > cf_A, std::vector< float > cf_B)

Additional Inherited Members

5.246.1 Constructor & Destructor Documentation

5.246.1.1 iir_filter_state_t()

```
MHAFilter::iir_filter_state_t::iir_filter_state_t (
    unsigned int channels,
    std::vector< float > cf_A,
    std::vector< float > cf_B )
```

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.247 MHAFilter::iir_filter_t Class Reference

IIR filter class wrapper for integration into parser structure.

Inheritance diagram for MHAFilter::iir_filter_t:

Public Member Functions

- **iir_filter_t** (std::string **help**="IIR **filter** structure", std::string def_A="[1]", std::string def↩_B="[1]", unsigned int **channels**=1)

  Constructor of the IIR filter.
- void **filter** (**mha_wave_t** ∗y, const **mha_wave_t** ∗x)

  The filter processes the audio signal.
- **mha_real_t** **filter** (**mha_real_t** x, unsigned int ch)

  Filter a single audio sample.
- void **resize** (unsigned int **channels**)

  Change the number of channels after object creation.

Private Member Functions

- void **update_filter** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::vfloat_t A**
- **MHAParser::vfloat_t B**
- **MHAEvents::patchbay_t< iir_filter_t > connector**
- unsigned int **nchannels**

Additional Inherited Members

5.247.1 Detailed Description

IIR filter class wrapper for integration into parser structure.

This class implements an infinite impulse response filter. Since it inherits from **MHAParser↩::parser_t** (p. 1102), it can easily be integrated in the openMHA configuration tree. It provides the configuration language variables "A" (vector of recursive filter coefficients) and "B" (vector of non-recursive filter coefficients).

The filter instance reacts to changes in filter coefficients through the openMHA configuration language, and uses the updated coefficients in the next invocation of the filter method.

Update of the coefficients is thread-safe and non-blocking. Simply add this subparser to your parser items and use the "filter" member function. Filter states are reset to all 0 on update.

5.247.2 Constructor & Destructor Documentation

5.247.2.1 iir_filter_t()

```
MHAFilter::iir_filter_t::iir_filter_t (
    std::string help = "IIR filter structure",
    std::string def_A = "[1]",
    std::string def_B = "[1]",
    unsigned int channels = 1 )
```

Constructor of the IIR filter.

Initialises the sub-parser structure and the memory for holding the filter's state.

Parameters

help	The help string for the parser that groups the configuration variables of this filter. Could be used to describe the purpose of this IIR filter.
def_A	The initial value of the vector of the recursive filter coefficients, represented as string.
def_B	The initial value of the vector of the non-recursive filter coefficients, represented as string.
channels	The number of independent audio channels to process with this filter. Needed to

© 2005-2021 HörTech gGmbH, Oldenburg

5.247.3 Member Function Documentation

5.247.3.1 filter() [1/2]

```
void MHAFilter::iir_filter_t::filter (
    mha_wave_t * y,
    const mha_wave_t * x )
```

The filter processes the audio signal.

All channels in the audio signal are processed using the same filter coefficients. Indipendent state is stored between calls for each audio channel.

Parameters

y	Pointer to output signal holder. The output signal is stored here. Has to have the same signal dimensions as the input signal x. In-place processing (y and x pointing to the same signal holder) is possible.
x	Pointer to input signal holder. Number of channels has to be the same as given to the constructor, or to the **resize** (p. 900) method.

5.247.3.2 filter() [2/2]

```
mha_real_t MHAFilter::iir_filter_t::filter (
    mha_real_t x,
    unsigned int ch )
```

Filter a single audio sample.

Parameters

x	The single audio sample
ch	Zero-based channel index. Use and change the state of channel ch. ch has to be less than the number of channels given to the constructor or the **resize** (p. 900) method.

Returns

the filtered result sample.

© 2005-2021 HörTech gGmbH, Oldenburg

5.247.3.3 resize()

```
void MHAFilter::iir_filter_t::resize (
    unsigned int channels )
```

Change the number of channels after object creation.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

channels	The new number of channels. Old filter states are lost.

5.247.3.4 update_filter() `void MHAFilter::iir_filter_t::update_filter ( ) [private]`

5.247.4 Member Data Documentation

5.247.4.1 A `MHAParser::vfloat_t MHAFilter::iir_filter_t::A [private]`

5.247.4.2 B `MHAParser::vfloat_t MHAFilter::iir_filter_t::B [private]`

5.247.4.3 connector `MHAEvents::patchbay_t< iir_filter_t> MHAFilter::iir_filter↩_t::connector [private]`

5.247.4.4 nchannels `unsigned int MHAFilter::iir_filter_t::nchannels [private]`

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.248 MHAFilter::iir_ord1_real_t Class Reference

First order recursive filter.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **iir_ord1_real_t** (std::vector< **mha_real_t** > A, std::vector< **mha_real_t** > B)
  Constructor with filter coefficients (one per channel)
- **iir_ord1_real_t** (std::vector< **mha_real_t** > tau, **mha_real_t** srate)
  Constructor for low pass filter (one time constant per channel)
- void **set_state** (**mha_real_t** val)
- void **set_state** (std::vector< **mha_real_t** > val)
- void **set_state** (**mha_complex_t** val)
- **mha_real_t operator()** (unsigned int ch, **mha_real_t** x)
  Filter method for real value input, one element.
- **mha_complex_t operator()** (unsigned int ch, **mha_complex_t** x)
  Filter method for complex input, one element.
- void **operator()** (const **mha_wave_t** &X, **mha_wave_t** &Y)
  Filter method for real value input.
- void **operator()** (const **mha_spec_t** &X, **mha_spec_t** &Y)
  Filter method for complex value input.
- void **operator()** (const **mha_wave_t** &Xre, const **mha_wave_t** &Xim, **mha_wave_t** &Yre, **mha_wave_t** &Yim)
  Filter method for complex value input.

Private Attributes

- std::vector< **mha_real_t** > **A_**
- std::vector< **mha_real_t** > **B_**
- std::vector< **mha_complex_t** > **Yn**

5.248.1 Detailed Description

First order recursive filter.

5.248.2 Constructor & Destructor Documentation

5.248.2.1 iir_ord1_real_t() [1/2]

```
MHAFilter::iir_ord1_real_t::iir_ord1_real_t (
    std::vector< mha_real_t > A,
    std::vector< mha_real_t > B )
```

Constructor with filter coefficients (one per channel)

© 2005-2021 HörTech gGmbH, Oldenburg

5.248.2.2 iir_ord1_real_t() [2/2]

```
MHAFilter::iir_ord1_real_t::iir_ord1_real_t (
    std::vector< mha_real_t > tau,
    mha_real_t srate )
```

Constructor for low pass filter (one time constant per channel)

5.248.3 Member Function Documentation

5.248.3.1 set_state() [1/3]

```
void MHAFilter::iir_ord1_real_t::set_state (
    mha_real_t val )
```

5.248.3.2 set_state() [2/3]

```
void MHAFilter::iir_ord1_real_t::set_state (
    std::vector< mha_real_t > val )
```

5.248.3.3 set_state() [3/3]

```
void MHAFilter::iir_ord1_real_t::set_state (
    mha_complex_t val )
```

5.248.3.4 operator()() [1/5]

```
mha_real_t MHAFilter::iir_ord1_real_t::operator() (
    unsigned int ch,
    mha_real_t x ) [inline]
```

Filter method for real value input, one element.

5.248.3.5 operator()() [2/5]

```
mha_complex_t MHAFilter::iir_ord1_real_t::operator() (
    unsigned int ch,
    mha_complex_t x ) [inline]
```

Filter method for complex input, one element.

© 2005-2021 HörTech gGmbH, Oldenburg

5.248.3.6 operator()() [3/5]

```
void MHAFilter::iir_ord1_real_t::operator() (
    const mha_wave_t & X,
    mha_wave_t & Y ) [inline]
```

Filter method for real value input.

5.248.3.7 operator()() [4/5]

```
void MHAFilter::iir_ord1_real_t::operator() (
    const mha_spec_t & X,
    mha_spec_t & Y ) [inline]
```

Filter method for complex value input.

5.248.3.8 operator()() [5/5]

```
void MHAFilter::iir_ord1_real_t::operator() (
    const mha_wave_t & Xre,
    const mha_wave_t & Xim,
    mha_wave_t & Yre,
    mha_wave_t & Yim ) [inline]
```

Filter method for complex value input.

5.248.4 Member Data Documentation

5.248.4.1 A_

```
std::vector< mha_real_t> MHAFilter::iir_ord1_real_t::A_ [private]
```

5.248.4.2 B_

```
std::vector< mha_real_t> MHAFilter::iir_ord1_real_t::B_ [private]
```

5.248.4.3 Yn

```
std::vector< mha_complex_t> MHAFilter::iir_ord1_real_t::Yn [private]
```

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.249 MHAFilter::o1_ar_filter_t Class Reference

First order attack-release lowpass filter.

Inheritance diagram for MHAFilter::o1_ar_filter_t:

Public Member Functions

- **o1_ar_filter_t** (unsigned int **channels**, **mha_real_t** **fs**=1.0f, std::vector< **mha_real_t** > tau_a=std::vector< float >(1, 0.0f), std::vector< **mha_real_t** > tau_r=std::vector< float >(1, 0.0f))

  Constructor, setting all taus to zero.
- void **set_tau_attack** (unsigned int ch, **mha_real_t** tau)

  Set the attack time constant.
- void **set_tau_release** (unsigned int ch, **mha_real_t** tau)

  Set the release time constant.
- **mha_real_t** **operator()** (unsigned int ch, **mha_real_t** x)

  Apply filter to value x, using state channel ch.
- void **operator()** (const **mha_wave_t** &in, **mha_wave_t** &out)

  *Apply filter to a **mha_wave_t** (p. 840) data.*

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Attributes

- **MHASignal::waveform_t c1_a**
- **MHASignal::waveform_t c2_a**
- **MHASignal::waveform_t c1_r**
- **MHASignal::waveform_t c2_r**
- **mha_real_t fs**

Additional Inherited Members

5.249.1 Detailed Description

First order attack-release lowpass filter.

This filter is the base of first order lowpass filter, maximum tracker and minimum tracker.

5.249.2 Constructor & Destructor Documentation

5.249.2.1 o1_ar_filter_t()

```
MHAFilter::o1_ar_filter_t::o1_ar_filter_t (
    unsigned int channels,
    mha_real_t fs = 1.0f,
    std::vector< mha_real_t > tau_a = std::vector<float>(1,0.0f),
    std::vector< mha_real_t > tau_r = std::vector<float>(1,0.0f) )
```

Constructor, setting all taus to zero.

The filter state can be accessed through the member functions of **MHASignal::waveform_t** (p. 1263).

Parameters

channels	Number of independent filters
fs	Sampling rate (optional, default = 1)
tau_a	Attack time constants (optional, default = 0)
tau_r	Release time constants (optional, default = 0)

© 2005-2021 HörTech gGmbH, Oldenburg

5.249.3 Member Function Documentation

5.249.3.1 set_tau_attack()

```
void MHAFilter::o1_ar_filter_t::set_tau_attack (
    unsigned int ch,
    mha_real_t tau )
```

Set the attack time constant.

Parameters

ch	Channel number
tau	Time constant

5.249.3.2 set_tau_release()

```
void MHAFilter::o1_ar_filter_t::set_tau_release (
    unsigned int ch,
    mha_real_t tau )
```

Set the release time constant.

Parameters

ch	Channel number
tau	Time constant

5.249.3.3 operator()() [1/2]

```
mha_real_t MHAFilter::o1_ar_filter_t::operator() (
    unsigned int ch,
    mha_real_t x ) [inline]
```

Apply filter to value x, using state channel ch.

Parameters

ch	Cannel number
x	Input value

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

Output value

5.249.3.4 operator()() [2/2]

```
void MHAFilter::o1_ar_filter_t::operator() (
    const mha_wave_t & in,
    mha_wave_t & out ) [inline]
```

Apply filter to a **mha_wave_t** (p. 840) data.

Parameters

in	Input signal
out	Output signal

The number of channels must match the number of filter bands.

5.249.4 Member Data Documentation

5.249.4.1 c1_a

```
MHASignal::waveform_t MHAFilter::o1_ar_filter_t::c1_a [protected]
```

5.249.4.2 c2_a

```
MHASignal::waveform_t MHAFilter::o1_ar_filter_t::c2_a [protected]
```

5.249.4.3 c1_r

```
MHASignal::waveform_t MHAFilter::o1_ar_filter_t::c1_r [protected]
```

5.249.4.4 c2_r

```
MHASignal::waveform_t MHAFilter::o1_ar_filter_t::c2_r [protected]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.249.4.5 fs `mha_real_t MHAFilter::o1_ar_filter_t::fs [protected]`

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.250 MHAFilter::o1flt_lowpass_t Class Reference

First order low pass filter.

Inheritance diagram for MHAFilter::o1flt_lowpass_t:

Public Member Functions

- **o1flt_lowpass_t** (const std::vector< **mha_real_t** > &, **mha_real_t**, **mha_real_t**=0)
  Constructor of low pass filter, sets sampling rate and time constants.
- **o1flt_lowpass_t** (const std::vector< **mha_real_t** > &tau, **mha_real_t** **fs**, const std↩::vector< **mha_real_t** > &startval)
  Constructor of low pass filter, sets sampling rate and time constants.
- void **set_tau** (unsigned int ch, **mha_real_t** tau)
  change the time constant in one channel
- void **set_tau** (**mha_real_t** tau)
  set time constant in all channels to tau
- **mha_real_t** **get_c1** (unsigned int ch) const
- **mha_real_t** **get_last_output** (unsigned int ch) const

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.250.1 Detailed Description

First order low pass filter.

5.250.2 Constructor & Destructor Documentation

5.250.2.1 o1flt_lowpass_t() [1/2]

```
MHAFilter::o1flt_lowpass_t::o1flt_lowpass_t (
    const std::vector< mha_real_t > & tau,
    mha_real_t fs,
    mha_real_t startval = 0 )
```

Constructor of low pass filter, sets sampling rate and time constants.

Parameters

tau	Vector of time constants
fs	Sampling rate
startval	Initial internal state value

5.250.2.2 o1flt_lowpass_t() [2/2]

```
MHAFilter::o1flt_lowpass_t::o1flt_lowpass_t (
    const std::vector< mha_real_t > & tau,
    mha_real_t fs,
    const std::vector< mha_real_t > & startval )
```

Constructor of low pass filter, sets sampling rate and time constants.

Parameters

tau	Vector of time constants
fs	Sampling rate
startval	Initial internal state value

© 2005-2021 HörTech gGmbH, Oldenburg

5.250.3 Member Function Documentation

5.250.3.1 set_tau() [1/2]

```
void MHAFilter::o1flt_lowpass_t::set_tau (
    unsigned int ch,
    mha_real_t tau )
```

change the time constant in one channel

5.250.3.2 set_tau() [2/2]

```
void MHAFilter::o1flt_lowpass_t::set_tau (
    mha_real_t tau )
```

set time constant in all channels to tau

5.250.3.3 get_c1()

```
mha_real_t MHAFilter::o1flt_lowpass_t::get_c1 (
    unsigned int ch ) const [inline]
```

5.250.3.4 get_last_output()

```
mha_real_t MHAFilter::o1flt_lowpass_t::get_last_output
(
    unsigned int ch ) const [inline]
```

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.251 MHAFilter::o1flt_maxtrack_t Class Reference

First order maximum tracker.

Inheritance diagram for MHAFilter::o1flt_maxtrack_t:

Public Member Functions

- **o1flt_maxtrack_t** (const std::vector< **mha_real_t** > &, **mha_real_t**, **mha_real_t**=0)
  Constructor of low pass filter, sets sampling rate and time constants.
- **o1flt_maxtrack_t** (const std::vector< **mha_real_t** > &tau, **mha_real_t** **fs**, const std↩ ::vector< **mha_real_t** > &startval)
- void **set_tau** (unsigned int ch, **mha_real_t** tau)
  change the time constant in one channel
- void **set_tau** (**mha_real_t** tau)
  set time constant in all channels to tau

Additional Inherited Members

5.251.1 Detailed Description

First order maximum tracker.

© 2005-2021 HörTech gGmbH, Oldenburg

5.251.2 Constructor & Destructor Documentation

5.251.2.1 o1flt_maxtrack_t() [1/2]

```
MHAFilter::o1flt_maxtrack_t::o1flt_maxtrack_t (
    const std::vector< mha_real_t > & tau,
    mha_real_t fs,
    mha_real_t startval = 0 )
```

Constructor of low pass filter, sets sampling rate and time constants.

Parameters

tau	Vector of time constants
fs	Sampling rate
startval	Initial internal state value

5.251.2.2 o1flt_maxtrack_t() [2/2]

```
MHAFilter::o1flt_maxtrack_t::o1flt_maxtrack_t (
    const std::vector< mha_real_t > & tau,
    mha_real_t fs,
    const std::vector< mha_real_t > & startval )
```

5.251.3 Member Function Documentation

5.251.3.1 set_tau() [1/2]

```
void MHAFilter::o1flt_maxtrack_t::set_tau (
    unsigned int ch,
    mha_real_t tau )
```

change the time constant in one channel

5.251.3.2 set_tau() [2/2]

```
void MHAFilter::o1flt_maxtrack_t::set_tau (
    mha_real_t tau )
```

set time constant in all channels to tau

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.252 MHAFilter::o1flt_mintrack_t Class Reference

First order minimum tracker.

Inheritance diagram for MHAFilter::o1flt_mintrack_t:

Public Member Functions

- **o1flt_mintrack_t** (const std::vector< **mha_real_t** > &, **mha_real_t**, **mha_real_t**=0)
- **o1flt_mintrack_t** (const std::vector< **mha_real_t** > &, **mha_real_t**, const std::vector< **mha_real_t** > &)
- void **set_tau** (unsigned int ch, **mha_real_t** tau)

  change the time constant in one channel
- void **set_tau** (**mha_real_t** tau)

  set time constant in all channels to tau

Additional Inherited Members

5.252.1 Detailed Description

First order minimum tracker.

© 2005-2021 HörTech gGmbH, Oldenburg

5.252.2 Constructor & Destructor Documentation

5.252.2.1 o1flt_mintrack_t() [1/2]

```
MHAFilter::o1flt_mintrack_t::o1flt_mintrack_t (
    const std::vector< mha_real_t > & tau,
    mha_real_t fs_,
    mha_real_t startval = 0 )
```

5.252.2.2 o1flt_mintrack_t() [2/2]

```
MHAFilter::o1flt_mintrack_t::o1flt_mintrack_t (
    const std::vector< mha_real_t > & tau,
    mha_real_t fs,
    const std::vector< mha_real_t > & startval )
```

5.252.3 Member Function Documentation

5.252.3.1 set_tau() [1/2]

```
void MHAFilter::o1flt_mintrack_t::set_tau (
    unsigned int ch,
    mha_real_t tau )
```

change the time constant in one channel

5.252.3.2 set_tau() [2/2]

```
void MHAFilter::o1flt_mintrack_t::set_tau (
    mha_real_t tau )
```

set time constant in all channels to tau

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.253 MHAFilter::partitioned_convolution_t Class Reference

A filter class for partitioned convolution.

Classes

- struct **index_t**
  Bookkeeping class.

Public Member Functions

- **partitioned_convolution_t** (unsigned int **fragsize**, unsigned int **nchannels_in**, unsigned int **nchannels_out**, const **transfer_matrix_t** &transfer)
  Create a new partitioned convolver.
- ∼**partitioned_convolution_t** ()
  Free fftw resource allocated in constructor.
- **mha_wave_t** ∗ **process** (const **mha_wave_t** ∗s_in)
  processing

Public Attributes

- unsigned int **fragsize**
  Audio fragment size, always equal to partition size.
- unsigned int **nchannels_in**
  Number of audio input channels.
- unsigned int **nchannels_out**
  Number of audio output channels.
- unsigned int **output_partitions**
  The maximum number of partitions in any of the impulse responses.
- unsigned int **filter_partitions**
  The total number of non-zero impulse response partitions.
- **MHASignal::waveform_t input_signal_wave**
  Buffer for input signal.
- unsigned int **current_input_signal_buffer_half_index**
  A counter modulo 2.
- **MHASignal::spectrum_t input_signal_spec**
  Buffer for FFT transformed input signal.
- **MHASignal::spectrum_t frequency_response**
  Buffers for frequency response spectra of impulse response partitions.
- std::vector< **index_t** > **bookkeeping**
  Keeps track of input channels, output channels, impulse response partition, and delay.
- std::vector< **MHASignal::spectrum_t** > **output_signal_spec**

© 2005-2021 HörTech gGmbH, Oldenburg

Buffers for FFT transformed output signal.

- unsigned int **current_output_partition_index**

  A counter modulo output_partitions, indexing the "current" output partition.

- **MHASignal::waveform_t output_signal_wave**

  Buffer for the wave output signal.

- **mha_fft_t fft**

  The FFT transformer.

5.253.1 Detailed Description

A filter class for partitioned convolution.

Impulse responses are partitioned into sections of fragment size. Audio signal is convolved with every partition and delayed as needed. Convolution is done according to overlap-save. FFT length used is 2 times fragment size.

5.253.2 Constructor & Destructor Documentation

5.253.2.1 partitioned_convolution_t()

```
MHAFilter::partitioned_convolution_t::partitioned↩
_convolution_t (
            unsigned int fragsize,
            unsigned int nchannels_in,
            unsigned int nchannels_out,
            const  transfer_matrix_t & transfer )
```

Create a new partitioned convolver.

Parameters

fragsize	Audio fragment size, equal to partition size.
nchannels_in	Number of input audio channels.
nchannels_out	Number of output audio channels.
transfer	A sparse matrix of impulse responses.

5.253.2.2 ∼partitioned_convolution_t()

```
MHAFilter::partitioned_convolution_t::∼partitioned↩
_convolution_t ( )
```

Free fftw resource allocated in constructor.

© 2005-2021 HörTech gGmbH, Oldenburg

5.253.3 Member Function Documentation

5.253.3.1 process() `mha_wave_t * MHAFilter::partitioned_convolution_t::process ( const mha_wave_t * s_in )`

processing

5.253.4 Member Data Documentation

5.253.4.1 fragsize `unsigned int MHAFilter::partitioned_convolution_t::fragsize`

Audio fragment size, always equal to partition size.

5.253.4.2 nchannels_in `unsigned int MHAFilter::partitioned_convolution_t::nchannels↩_in`

Number of audio input channels.

5.253.4.3 nchannels_out `unsigned int MHAFilter::partitioned_convolution_t::nchannels↩_out`

Number of audio output channels.

5.253.4.4 output_partitions `unsigned int MHAFilter::partitioned_convolution_t↩::output_partitions`

The maximum number of partitions in any of the impulse responses.

Determines the size if the delay line.

© 2005-2021 HörTech gGmbH, Oldenburg

5.253.4.5 filter_partitions `unsigned int MHAFilter::partitioned_convolution_t::filter↩_partitions`

The total number of non-zero impulse response partitions.

5.253.4.6 input_signal_wave `MHASignal::waveform_t MHAFilter::partitioned_convolution↩_t::input_signal_wave`

Buffer for input signal.

Has nchannels_in channels and fragsize∗2 frames

5.253.4.7 current_input_signal_buffer_half_index `unsigned int MHAFilter::partitioned↩_convolution_t::current_input_signal_buffer_half_index`

A counter modulo 2.

Indicates the buffer half in input signal wave into which to copy the current input signal.

5.253.4.8 input_signal_spec `MHASignal::spectrum_t MHAFilter::partitioned_convolution↩_t::input_signal_spec`

Buffer for FFT transformed input signal.

Has nchannels_in channels and fragsize+1 frames (fft bins).

5.253.4.9 frequency_response `MHASignal::spectrum_t MHAFilter::partitioned_↩convolution_t::frequency_response`

Buffers for frequency response spectra of impulse response partitions.

Each "channel" contains another partition of some impulse response. The bookkeeping array is used to keep track what to do with these frequency responses. This container has filter_↩partitions channels and fragsize+1 frames (fft bins).

5.253.4.10 bookkeeping `std::vector< index_t> MHAFilter::partitioned_convolution↩_t::bookkeeping`

Keeps track of input channels, output channels, impulse response partition, and delay.

The index into this array is the same as the "channel" index into the frequency_response array. Array has filter_partitions entries.

© 2005-2021 HörTech gGmbH, Oldenburg

5.253.4.11 output_signal_spec `std::vector< MHASignal::spectrum_t> MHAFilter↩ ::partitioned_convolution_t::output_signal_spec`

Buffers for FFT transformed output signal.

For each array member, Number of channels is equal to nchannels_out, number of frames (fft bins) is equal to fragsize+1. Array size is equal to output_partitions.

5.253.4.12 current_output_partition_index `unsigned int MHAFilter::partitioned_↩ convolution_t::current_output_partition_index`

A counter modulo output_partitions, indexing the "current" output partition.

5.253.4.13 output_signal_wave `MHASignal::waveform_t MHAFilter::partitioned_↩ convolution_t::output_signal_wave`

Buffer for the wave output signal.

Number of channels is equal to nchannels_out, number of frames is equal to fragsize

5.253.4.14 fft `mha_fft_t MHAFilter::partitioned_convolution_t::fft`

The FFT transformer.

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.254 MHAFilter::partitioned_convolution_t::index_t Struct Reference

Bookkeeping class.

Public Member Functions

- **index_t** (unsigned int src, unsigned int tgt, unsigned int dly)
  Data constructor.
- **index_t** ()
  Default constructor for STL compatibility.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- unsigned int **source_channel_index**
  The input channel index to apply the current partition to.
- unsigned int **target_channel_index**
  The index of the output channel to which the filter result should go.
- unsigned int **delay**
  The delay (in blocks) of this partition.

5.254.1 Detailed Description

Bookkeeping class.

For each impulse response partition, keeps track of which input to filter, which output channel to filter to, and the delay in blocks. Objects of class Index should be kept in an array with the same indices as the corresponding inpulse response partitions.

5.254.2 Constructor & Destructor Documentation

5.254.2.1 index_t() [1/2]

```
MHAFilter::partitioned_convolution_t::index_t::index_t (
    unsigned int src,
    unsigned int tgt,
    unsigned int dly ) [inline]
```

Data constructor.

Parameters

src	The input channel index to apply the current partition to.
tgt	The index of the output channel to which the filter result should go.
dly	The delay (in blocks) of this partition

5.254.2.2 index_t() [2/2]

```
MHAFilter::partitioned_convolution_t::index_t::index_t ( ) [inline]
```

Default constructor for STL compatibility.

© 2005-2021 HörTech gGmbH, Oldenburg

5.254.3 Member Data Documentation

5.254.3.1 source_channel_index `unsigned int MHAFilter::partitioned_convolution_↩ t::index_t::source_channel_index`

The input channel index to apply the current partition to.

5.254.3.2 target_channel_index `unsigned int MHAFilter::partitioned_convolution_t↩ ::index_t::target_channel_index`

The index of the output channel to which the filter result should go.

5.254.3.3 delay `unsigned int MHAFilter::partitioned_convolution_t::index_t::delay`

The delay (in blocks) of this partition.

The documentation for this struct was generated from the following file:

- **mha_filter.hh**

5.255 MHAFilter::polyphase_resampling_t Class Reference

A class that performs polyphase resampling.

Public Member Functions

- **polyphase_resampling_t** (unsigned n_up, unsigned n_down, **mha_real_t** nyquist_↩ ratio, unsigned n_irs, unsigned n_ringbuffer, unsigned n_channels, unsigned n_prefill)
  Construct a polyphase resampler instance.
- void **write** (**mha_wave_t** &signal)
  Write signal to the ringbuffer.
- void **read** (**mha_wave_t** &signal)
  Read resampled signal.
- unsigned **readable_frames** () const
  Number of frames at target sampling rate that can be produced.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- unsigned **upsampling_factor**
  Integer upsampling factor.
- unsigned **downsampling_factor**
  Integer downsampling factor.
- unsigned **now_index**
  Index of "now" in the interpolated sampling rate.
- bool **underflow**
  Set to true when an underflow has occurred.
- **MHAWindow::hanning_t impulse_response**
  Contains the impulse response of the lowpass filter needed for anti-aliasing.
- **MHASignal::ringbuffer_t ringbuffer**
  Storage of input signal.

5.255.1 Detailed Description

A class that performs polyphase resampling.

Background information: When resampling from one sampling rate to another, it helps when one sampling rate is a multiple of the other sampling rate: In the case of upsampling, the samples at the original rate are copied to the upsampled signal spread out with a constant number of zero samples between the originally adjacent samples. The signal is then low-pass filtered to avoid frequency aliasing and to fill the zero-samples with interpolated values. In the case of down-sampling, the signal is first low-pass filtered for anti-aliasing, and only every n^{th} sample of the filtered output is used for the signal at the new sample rate. Of course, for finite-impulse-response (FIR) filters this means that only every n^{th} sample needs to be computed.

When resampling from one sampling rate to another where neither is a multiple of the other, the signal first needs to be upsampled to a sampling rate that is a multiple of both (source and target) sampling rates, and then downsampled again to the target sampling rate. Instead of applying two separate lowpass filters directly after each other (one filter for upsampling and another for downsampling), it is sufficient to apply only one low-pass filter, when producing the output at the final target rate, with a cut-off frequency equal to the lower cut-off-frequency of the replaced two low-pass filters. Not filtering to produce a filtered signal already at the common multiple sampling rate has the side effect that this intermediate signal at the common multiple sampling rate keeps its filler zero samples unaltered. These zero samples can be taken advantage of when filtering to produce the output at the target rate: The zeros do not need to be multiplied with their corresponding filter coefficients, because the result is known to be zero again, and this zero product has no effect on the summation operation to compute a target sample at the target rate. To summarize, the following optimization techniques are available:

- The signal does not need to be stored in memory at the interpolation rate. It is sufficient to have the signal available at the source rate and to know where the zeros would be.
- The signal needs to be low-pass-filtered only once.

© 2005-2021 HörTech gGmbH, Oldenburg

- The FIR low-pass filtering can take advantage of
  - computing only filter outputs for the required samples at the target rate,
  - skipping over zero-samples at the interpolation rate.

The procedure that takes advantage of these optimization possibilites is known as polyphase resampling.

This class implements polyphase resampling in this way for a source sampling rate and a target sampling rate that have common multiple, the interpolation sampling rate. Non-rational and drifting sample rates are outside the scope of this resampler.

5.255.2 Constructor & Destructor Documentation

5.255.2.1 polyphase_resampling_t()

```
MHAFilter::polyphase_resampling_t::polyphase_↩
resampling_t (
            unsigned n_up,
            unsigned n_down,
             mha_real_t nyquist_ratio,
            unsigned n_irs,
            unsigned n_ringbuffer,
            unsigned n_channels,
            unsigned n_prefill )
```

Construct a polyphase resampler instance.

Allocates a ringbuffer with the given capacity *n_ringbuffer*. Client that triggers the constructor must ensure that the capacity *n_ringbuffer* and the delay *n_prefill* are sufficient, i.e. enough old and new samples are always available to compute sufficient samples in using an impulse response of length *n_irs*. Audio block sizes at both sides of the resampler have to be taken into account. Class `MHASignal::blockprocessing_polyphase_resampling_t` takes care of this, and it is recommended to use this class for block-based processing.

Based on *n_up*, *n_down*, *n_irs* and *nyquist_ratio*, a suitable sinc impulse response is computed and windowed with a hanning window to limit its extent.

The actual source sampling rate, target sampling rate, and interpolation sampling rate are not parameters to this constructors, because only their ratios matter.

Parameters

n_up	upsampling factor, ratio between interpolation rate and source rate.
n_down	downsampling factor, ratio between interpolation rate and target rate.
nyquist_ratio	low pass filter cutoff frequency relative to the nyquist frequency of the smaller of the two sampling rates. Example values: E.g. 0.8, 0.9
n_irs	length of impulse response (in samples at interpolation rate)

© 2005-2021 HörTech gGmbH, Oldenburg

5.255.3 Member Function Documentation

5.255.3.1 write() `void MHAFilter::polyphase_resampling_t::write ( mha_wave_t & signal )`

Write signal to the ringbuffer.

Signal contained in signal is appended to the audio frames already present in the ringbuffer.

Parameters

signal	input signal in original sampling rate

Exceptions

MHA_Error *(p. 764)*	Raises exception if there is not enough room or if the number of channels does not match.

5.255.3.2 read() `void MHAFilter::polyphase_resampling_t::read ( mha_wave_t & signal )`

Read resampled signal.

Will perform the resampling and remove no longer needed samples from the input buffer.

Parameters

signal	buffer to write the resampled signal to.

Exceptions

MHA_Error *(p. 764)*	Raises exception if there is not enough input signal or if the number of channels is too high.

5.255.3.3 readable_frames() `unsigned MHAFilter::polyphase_resampling_t::readable_frames ( ) const [inline]`

© 2005-2021 HörTech gGmbH, Oldenburg

Number of frames at target sampling rate that can be produced.

This method only checks for enough future samples present, therefore, this number can be positive and a read operation can still fail if there are not enough past samples present to perform the filtering for the first output sample. This could only happen if the constructor parameters *n_ringbuffer* or *n_prefill* have been chosen too small, because otherwise the method **read** (p. 926) ensures that enough past samples are present to compute the next target sample.

5.255.4 Member Data Documentation

5.255.4.1 upsampling_factor `unsigned MHAFilter::polyphase_resampling_t::upsampling↩_factor [private]`

Integer upsampling factor.

Interpolation rate divided by source rate.

5.255.4.2 downsampling_factor `unsigned MHAFilter::polyphase_resampling_t::downsampling↩_factor [private]`

Integer downsampling factor.

Interpolation rate divided by target rate.

5.255.4.3 now_index `unsigned MHAFilter::polyphase_resampling_t::now_index [private]`

Index of "now" in the interpolated sampling rate.

5.255.4.4 underflow `bool MHAFilter::polyphase_resampling_t::underflow [private]`

Set to true when an underflow has occurred.

When this is true, then the object can no longer be used. Underflows have to be avoided by clients, e.g. by checking that enough **readable_frames** (p. 926) are present before calling **read** (p. 926)

© 2005-2021 HörTech gGmbH, Oldenburg

5.255.4.5 impulse_response `MHAWindow::hanning_t MHAFilter::polyphase_resampling↩_t::impulse_response [private]`

Contains the impulse response of the lowpass filter needed for anti-aliasing.

The impulse response is stored at the interpolation sampling rate. We use an instance of **MHAWindow::hanning_t** (p. 1297) here because we are limiting the sinc impulse response with a Hanning window (otherwise the impulse response would extend indefinitely into past and future). And the samples inside an **MHAWindow::hanning_t** (p. 1297) can be altered with $*=$, which our constructor does.

5.255.4.6 ringbuffer `MHASignal::ringbuffer_t MHAFilter::polyphase_resampling_t↩::ringbuffer [private]`

Storage of input signal.

Part of the polyphase resampling optimization is that apart from the FIR impulse response, nothing is stored at the interpolation rate, saving memory and computation cycles.

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.256 MHAFilter::resampling_filter_t Class Reference

Hann shaped low pass filter for resampling.

Inheritance diagram for MHAFilter::resampling_filter_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **resampling_filter_t** (unsigned int **fftlen**, unsigned int irslen, unsigned int **channels**, unsigned int Nup, unsigned int Ndown, double fCutOff)

  Constructor.

Static Public Member Functions

- static unsigned int **fragsize_validator** (unsigned int **fftlen**, unsigned int irslen)

Private Attributes

- unsigned int **fragsize**

5.256.1 Detailed Description

Hann shaped low pass filter for resampling.

This class uses FFT filter at upsampled rate.

5.256.2 Constructor & Destructor Documentation

5.256.2.1 resampling_filter_t() `MHAFilter::resampling_filter_t::resampling_filter_t`

```
(
            unsigned int fftlen,
            unsigned int irslen,
            unsigned int channels,
            unsigned int Nup,
            unsigned int Ndown,
            double fCutOff )
```

Constructor.

Parameters

fftlen	FFT length.
irslen	Length of filter.
channels	Number of channels to be filtered.
Nup	Upsampling ratio.
Ndown	Downsampling ratio.
fCutOff	Cut off frequency (relative to lower Nyquist Frequency)

© 2005-2021 HörTech gGmbH, Oldenburg

5.256.3 Member Function Documentation

5.256.3.1 fragsize_validator() `unsigned int MHAFilter::resampling_filter_t::fragsize↩_validator (`

```
unsigned int fftlen,
unsigned int irslen ) [static]
```

5.256.4 Member Data Documentation

5.256.4.1 fragsize `unsigned int MHAFilter::resampling_filter_t::fragsize [private]`

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.257 MHAFilter::smoothspec_t Class Reference

Smooth spectral gains, create a windowed impulse response.

Public Member Functions

- **smoothspec_t** (unsigned int **fftlen**, unsigned int **nchannels**, const **MHAWindow↩::base_t** & **window**, bool **minphase**, bool linphase_asym=false)

  Constructor.
- void **smoothspec** (const **mha_spec_t** &s_in, **mha_spec_t** &s_out)

  Create a smoothed spectrum.
- void **smoothspec** (**mha_spec_t** &spec)

  Create a smoothed spectrum (in place)
- void **spec2fir** (const **mha_spec_t** &spec, **mha_wave_t** &fir)

  Return FIR coefficients.
- ∼**smoothspec_t** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **internal_fir** (const **mha_spec_t** &)

Private Attributes

- unsigned int **fftlen**
- unsigned int **nchannels**
- **MHAWindow::base_t window**
- **MHASignal::waveform_t tmp_wave**
- **MHASignal::spectrum_t tmp_spec**
- **MHASignal::minphase_t** * **minphase**
- bool **_linphase_asym**
- **mha_fft_t fft**

5.257.1 Detailed Description

Smooth spectral gains, create a windowed impulse response.

Spectral gains are smoothed by multiplying the impulse response with a window function.

If a minimal phase is used, then the original phase is discarded and replaced by the minimal phase function. In this case, the window is applied to the beginning of the inverse Fourier transform of the input spectrum, and the remaining signal set to zero. If the original phase is kept, the window is applied symmetrically arround zero, i.e. to the first and last samples of the inverse Fourier transform of the input spectrum. The **spec2fir()** (p. 933) function creates a causal impulse response by circularly shifting the impulse response by half of the window length.

The signal dimensions of the arguments of **smoothspec()** (p. 932) must correspond to the FFT length and number of channels provided in the constructor. The function **spec2fir()** (p. 933) can fill signal structures with more than window length frames.

5.257.2 Constructor & Destructor Documentation

5.257.2.1 smoothspec_t()

```
MHAFilter::smoothspec_t::smoothspec_t (
    unsigned int fftlen,
    unsigned int nchannels,
    const MHAWindow::base_t & window,
    bool minphase,
    bool linphase_asym = false )
```

Constructor.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

fftlen	FFT length of input spectrum (fftlen/2+1 bins)
nchannels	Number of channels in input spectrum
window	Window used for smoothing
minphase	Use minimal phase (true) or original phase (false)
linphase_asym	Keep phase, but apply full window at beginning of IRS

5.257.2.2 ~smoothspec_t() `MHAFilter::smoothspec_t::~smoothspec_t ( )`

5.257.3 Member Function Documentation

5.257.3.1 smoothspec() [1/2] `void MHAFilter::smoothspec_t::smoothspec ( const mha_spec_t & s_in, mha_spec_t & s_out )`

Create a smoothed spectrum.

Parameters

s_in	Input spectrum

Return values

s_out	Output spectrum

5.257.3.2 smoothspec() [2/2] `void MHAFilter::smoothspec_t::smoothspec ( mha_spec_t & spec ) [inline]`

Create a smoothed spectrum (in place)

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

spec	Spectrum to be smoothed.

5.257.3.3 spec2fir()

```
void MHAFilter::smoothspec_t::spec2fir (
    const mha_spec_t & spec,
    mha_wave_t & fir )
```

Return FIR coefficients.

Parameters

spec	Input spectrum

Return values

fir	FIR coefficients, minimum length is window length

5.257.3.4 internal_fir()

```
void MHAFilter::smoothspec_t::internal_fir (
    const mha_spec_t & s_in ) [private]
```

5.257.4 Member Data Documentation

5.257.4.1 fftlen

```
unsigned int MHAFilter::smoothspec_t::fftlen [private]
```

5.257.4.2 nchannels

```
unsigned int MHAFilter::smoothspec_t::nchannels [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.257.4.3 window `MHAWindow::base_t MHAFilter::smoothspec_t::window [private]`

5.257.4.4 tmp_wave `MHASignal::waveform_t MHAFilter::smoothspec_t::tmp_wave [private]`

5.257.4.5 tmp_spec `MHASignal::spectrum_t MHAFilter::smoothspec_t::tmp_spec [private]`

5.257.4.6 minphase `MHASignal::minphase_t* MHAFilter::smoothspec_t::minphase [private]`

5.257.4.7 _linphase_asym `bool MHAFilter::smoothspec_t::_linphase_asym [private]`

5.257.4.8 fft `mha_fft_t MHAFilter::smoothspec_t::fft [private]`

The documentation for this class was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.258 MHAFilter::thirdoctave_analyzer_t Class Reference

Public Member Functions

- **thirdoctave_analyzer_t** (**mhaconfig_t** cfg)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- unsigned int **nbands** ()
- unsigned int **nchannels** ()
- std::vector< **mha_real_t** > **get_cf_hz** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Static Public Member Functions

- static std::vector< **mha_real_t** > **cf_generator** (**mhaconfig_t** cfg)
- static std::vector< **mha_real_t** > **bw_generator** (**mhaconfig_t** cfg)
- static std::vector< **mha_real_t** > **dup** (std::vector< **mha_real_t** >, **mhaconfig_t** cfg)

Private Attributes

- **mhaconfig_t cfg_**
- std::vector< **mha_real_t** > **cf**
- **MHAFilter::gamma_flt_t fb**
- **MHASignal::waveform_t out_chunk**
- **MHASignal::waveform_t out_chunk_im**

5.258.1 Constructor & Destructor Documentation

5.258.1.1 thirdoctave_analyzer_t()

```
MHAFilter::thirdoctave_analyzer_t::thirdoctave_↩
analyzer_t (
            mhaconfig_t cfg )
```

5.258.2 Member Function Documentation

5.258.2.1 process()

```
mha_wave_t * MHAFilter::thirdoctave_analyzer_t::process (
            mha_wave_t * sIn )
```

5.258.2.2 nbands()

```
unsigned int MHAFilter::thirdoctave_analyzer_t::nbands ( )
```

5.258.2.3 nchannels()

```
unsigned int MHAFilter::thirdoctave_analyzer_t::nchannels (
)
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.258.2.4 get_cf_hz()

`std::vector< mha_real_t > MHAFilter::thirdoctave_analyzer_t::get_cf_hz ( )`

5.258.2.5 cf_generator()

```
std::vector< mha_real_t > MHAFilter::thirdoctave_analyzer_t::cf_generator (
            mhaconfig_t cfg ) [static]
```

5.258.2.6 bw_generator()

```
std::vector< mha_real_t > MHAFilter::thirdoctave_analyzer_t::bw_generator (
            mhaconfig_t cfg ) [static]
```

5.258.2.7 dup()

```
std::vector< mha_real_t > MHAFilter::thirdoctave_analyzer_t::dup
(
           std::vector< mha_real_t > vec,
            mhaconfig_t cfg ) [static]
```

5.258.3 Member Data Documentation

5.258.3.1 cfg_

`mhaconfig_t MHAFilter::thirdoctave_analyzer_t::cfg_ [private]`

5.258.3.2 cf

`std::vector< mha_real_t> MHAFilter::thirdoctave_analyzer_t::cf [private]`

5.258.3.3 fb

`MHAFilter::gamma_flt_t MHAFilter::thirdoctave_analyzer_t::fb [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.258.3.4 out_chunk `MHASignal::waveform_t MHAFilter::thirdoctave_analyzer_t↩ ::out_chunk [private]`

5.258.3.5 out_chunk_im `MHASignal::waveform_t MHAFilter::thirdoctave_analyzer_t↩ ::out_chunk_im [private]`

The documentation for this class was generated from the following files:

- **complex_filter.h**
- **complex_filter.cpp**

5.259 MHAFilter::transfer_function_t Struct Reference

a structure containing a source channel number, a target channel number, and an impulse response.

Public Member Functions

- **transfer_function_t** ()
  Default constructor for STL conformity.
- **transfer_function_t** (unsigned int **source_channel_index**, unsigned int **target_↩ channel_index**, const std::vector< float > & **impulse_response**)
  Data constructor.
- unsigned int **partitions** (unsigned int fragsize) const
  for the given partition size, return the number of partitions of the impulse response.
- unsigned int **non_empty_partitions** (unsigned int fragsize) const
  for the given partition size, return the number of non-empty partitions of the impulse response.
- bool **isempty** (unsigned int fragsize, unsigned int index) const
  checks if the partition contains only zeros

Public Attributes

- unsigned int **source_channel_index**
  Source audio channel index for this transfer function.
- unsigned int **target_channel_index**
  Target audio channel index for this transfer function.
- std::vector< float > **impulse_response**
  Impulse response of transfer from source to target channel.

© 2005-2021 HörTech gGmbH, Oldenburg

5.259.1 Detailed Description

a structure containing a source channel number, a target channel number, and an impulse response.

5.259.2 Constructor & Destructor Documentation

5.259.2.1 transfer_function_t() [1/2] `MHAFilter::transfer_function_t::transfer_function_t ( ) [inline]`

Default constructor for STL conformity.

Not used.

5.259.2.2 transfer_function_t() [2/2]

```
MHAFilter::transfer_function_t::transfer_function_t (
            unsigned int source_channel_index,
            unsigned int target_channel_index,
            const std::vector< float > & impulse_response )
```

Data constructor.

Parameters

source_channel_index	Source audio channel index for this transfer function
target_channel_index	Target audio channel index for this transfer function
impulse_response	Impulse response of transfer from source to target channel

5.259.3 Member Function Documentation

5.259.3.1 partitions()

```
unsigned int MHAFilter::transfer_function_t::partitions (
            unsigned int fragsize ) const [inline]
```

for the given partition size, return the number of partitions of the impulse response.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

fragsize	partition size

Returns

number of partitions occupied by the impulse response

5.259.3.2 non_empty_partitions()

```
unsigned int MHAFilter::transfer_function_t↩
::non_empty_partitions (
            unsigned int fragsize ) const [inline]
```

for the given partition size, return the number of non-empty partitions of the impulse response.

Parameters

fragsize	partition size

Returns

the number of non-empty partitions of the impulse response, i.e. partitions containing only zeros are not counted.

5.259.3.3 isempty()

```
bool MHAFilter::transfer_function_t::isempty (
            unsigned int fragsize,
            unsigned int index ) const [inline]
```

checks if the partition contains only zeros

Parameters

fragsize	partition size
index	partition index

Returns

true when this partition of the impulse response contains only zeros.

© 2005-2021 HörTech gGmbH, Oldenburg

5.259.4 Member Data Documentation

5.259.4.1 source_channel_index `unsigned int MHAFilter::transfer_function_t::source↩_channel_index`

Source audio channel index for this transfer function.

5.259.4.2 target_channel_index `unsigned int MHAFilter::transfer_function_t::target↩_channel_index`

Target audio channel index for this transfer function.

5.259.4.3 impulse_response `std::vector<float> MHAFilter::transfer_function_t↩::impulse_response`

Impulse response of transfer from source to target channel.

The documentation for this struct was generated from the following files:

- **mha_filter.hh**
- **mha_filter.cpp**

5.260 MHAFilter::transfer_matrix_t Struct Reference

A sparse matrix of transfer function partitionss.

Inherits vector< transfer_function_t >.

Public Member Functions

- std::valarray< unsigned int > **partitions** (unsigned fragsize) const

  *Returns an array of the results of calling the **partitions()** (p. 941) method on every matrix member.*
- std::valarray< unsigned int > **non_empty_partitions** (unsigned int fragsize) const

  *Returns an array of the results of calling the **non_empty_partitions()** (p. 941) method on every matrix member.*

© 2005-2021 HörTech gGmbH, Oldenburg

5.260.1 Detailed Description

A sparse matrix of transfer function partitionss.

Each matrix element knows its position in the matrix, so they can be stored as a vector.

5.260.2 Member Function Documentation

5.260.2.1 partitions()

```
std::valarray<unsigned int> MHAFilter::transfer_matrix_t↩
::partitions (
            unsigned fragsize ) const [inline]
```

Returns an array of the results of calling the **partitions()** (p. 941) method on every matrix member.

5.260.2.2 non_empty_partitions()

```
std::valarray<unsigned int> MHAFilter::transfer↩
_matrix_t::non_empty_partitions (
            unsigned int fragsize ) const [inline]
```

Returns an array of the results of calling the **non_empty_partitions()** (p. 941) method on every matrix member.

The documentation for this struct was generated from the following file:

- **mha_filter.hh**

5.261 MHAIOJack::io_jack_t Class Reference

Main class for JACK IO.

Inheritance diagram for MHAIOJack::io_jack_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **io_jack_t** (unsigned int **fragsize**, float **samplerate**, **IOProcessEvent_t proc_event**, void * **proc_handle**, **IOStartedEvent_t start_event**, void * **start_handle**, **IOStoppedEvent_t stop_event**, void * **stop_handle**)
- void **prepare** (int, int)

  Allocate buffers, activate JACK client and install internal ports.
- void **release** ()

Private Member Functions

- void **reconnect_inports** ()

  Connect the input ports when connection variable is accessed.
- void **reconnect_outports** ()

  Connect the output ports when connection variable is accessed.
- void **get_physical_input_ports** ()
- void **get_physical_output_ports** ()
- void **get_all_input_ports** ()
- void **get_all_output_ports** ()
- void **get_delays_in** ()
- void **get_delays_out** ()
- void **read_get_cpu_load** ()
- void **read_get_xruns** ()
- void **read_get_scheduler** ()

Private Attributes

- unsigned int **fw_fragsize**
- float **fw_samplerate**
- **MHAParser::string_t servername**
- **MHAParser::string_t clientname**
- **MHAParser::vstring_t connections_in**
- **MHAParser::vint_mon_t delays_in**
- **MHAParser::vstring_t connections_out**
- **MHAParser::vint_mon_t delays_out**
- **MHAParser::vstring_t portnames_in**
- **MHAParser::vstring_t portnames_out**
- **MHAParser::vstring_mon_t ports_in_physical**
- **MHAParser::vstring_mon_t ports_out_physical**
- **MHAParser::vstring_mon_t ports_in_all**
- **MHAParser::vstring_mon_t ports_out_all**
- **MHAParser::parser_t ports_parser**
- **MHAParser::float_mon_t state_cpuload**
- **MHAParser::int_mon_t state_xruns**
- **MHAParser::int_mon_t state_priority**
- **MHAParser::string_mon_t state_scheduler**
- **MHAParser::parser_t state_parser**
- **MHAEvents::patchbay_t< io_jack_t > patchbay**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.261.1 Detailed Description

Main class for JACK IO.

This class registers a JACK client. JACK and framework states are managed by this class.

5.261.2 Constructor & Destructor Documentation

5.261.2.1 io_jack_t()

```
io_jack_t::io_jack_t (
    unsigned int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle )
```

5.261.3 Member Function Documentation

5.261.3.1 prepare()

```
void io_jack_t::prepare (
    int nch_in,
    int nch_out )
```

Allocate buffers, activate JACK client and install internal ports.

5.261.3.2 release()

```
void io_jack_t::release ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.261.3.3 reconnect_inports() `void io_jack_t::reconnect_inports ( ) [private]`

Connect the input ports when connection variable is accessed.

5.261.3.4 reconnect_outports() `void io_jack_t::reconnect_outports ( ) [private]`

Connect the output ports when connection variable is accessed.

5.261.3.5 get_physical_input_ports() `void io_jack_t::get_physical_input_ports ( ) [private]`

5.261.3.6 get_physical_output_ports() `void io_jack_t::get_physical_output_ports ( ) [private]`

5.261.3.7 get_all_input_ports() `void io_jack_t::get_all_input_ports ( ) [private]`

5.261.3.8 get_all_output_ports() `void io_jack_t::get_all_output_ports ( ) [private]`

5.261.3.9 get_delays_in() `void io_jack_t::get_delays_in ( ) [private]`

5.261.3.10 get_delays_out() `void io_jack_t::get_delays_out ( ) [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.261.3.11 read_get_cpu_load() `void io_jack_t::read_get_cpu_load ( ) [private]`

5.261.3.12 read_get_xruns() `void io_jack_t::read_get_xruns ( ) [private]`

5.261.3.13 read_get_scheduler() `void io_jack_t::read_get_scheduler ( ) [private]`

5.261.4 Member Data Documentation

5.261.4.1 fw_fragsize `unsigned int MHAIOJack::io_jack_t::fw_fragsize [private]`

5.261.4.2 fw_samplerate `float MHAIOJack::io_jack_t::fw_samplerate [private]`

5.261.4.3 servername `MHAParser::string_t MHAIOJack::io_jack_t::servername [private]`

5.261.4.4 clientname `MHAParser::string_t MHAIOJack::io_jack_t::clientname [private]`

5.261.4.5 connections_in `MHAParser::vstring_t MHAIOJack::io_jack_t::connections↩_in [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.261.4.6 delays_in

MHAParser::vint_mon_t MHAIOJack::io_jack_t::delays_in [private]

5.261.4.7 connections_out

MHAParser::vstring_t MHAIOJack::io_jack_t::connections↩_out [private]

5.261.4.8 delays_out

MHAParser::vint_mon_t MHAIOJack::io_jack_t::delays_out [private]

5.261.4.9 portnames_in

MHAParser::vstring_t MHAIOJack::io_jack_t::portnames_in [private]

5.261.4.10 portnames_out

MHAParser::vstring_t MHAIOJack::io_jack_t::portnames_↩out [private]

5.261.4.11 ports_in_physical

MHAParser::vstring_mon_t MHAIOJack::io_jack_t::ports↩_in_physical [private]

5.261.4.12 ports_out_physical

MHAParser::vstring_mon_t MHAIOJack::io_jack_t↩::ports_out_physical [private]

5.261.4.13 ports_in_all

MHAParser::vstring_mon_t MHAIOJack::io_jack_t::ports_in_↩all [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.261.4.14 ports_out_all

MHAParser::vstring_mon_t MHAIOJack::io_jack_t::ports_↩ out_all [private]

5.261.4.15 ports_parser

MHAParser::parser_t MHAIOJack::io_jack_t::ports_parser [private]

5.261.4.16 state_cpuload

MHAParser::float_mon_t MHAIOJack::io_jack_t::state_↩ cpuload [private]

5.261.4.17 state_xruns

MHAParser::int_mon_t MHAIOJack::io_jack_t::state_xruns [private]

5.261.4.18 state_priority

MHAParser::int_mon_t MHAIOJack::io_jack_t::state_priority [private]

5.261.4.19 state_scheduler

MHAParser::string_mon_t MHAIOJack::io_jack_t::state_↩ scheduler [private]

5.261.4.20 state_parser

MHAParser::parser_t MHAIOJack::io_jack_t::state_parser [private]

5.261.4.21 patchbay

MHAEvents::patchbay_t< io_jack_t> MHAIOJack::io_jack_t↩ ::patchbay [private]

The documentation for this class was generated from the following file:

- **MHAIOJack.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.262 MHAIOJackdb::io_jack_t Class Reference

Main class for JACK IO.

Inheritance diagram for MHAIOJackdb::io_jack_t:

Public Member Functions

- **io_jack_t** (unsigned int **fragsize**, float **samplerate**, **IOProcessEvent_t proc_event**, void * **proc_handle**, **IOStartedEvent_t start_event**, void * **start_handle**, **IOStoppedEvent_t stop_event**, void * **stop_handle**)
- void **prepare** (int, int)

  Allocate buffers, activate JACK client and install internal ports.
- void **release** ()
- bool **fail_on_async_jackerror** () const

Private Member Functions

- int **IOProcessEvent_inner** (**mha_wave_t** *sIn, **mha_wave_t** **sOut)
- void **reconnect_inports** ()

  Connect the input ports when connection variable is accessed.
- void **reconnect_outports** ()

  Connect the output ports when connection variable is accessed.
- void **get_physical_input_ports** ()
- void **get_physical_output_ports** ()
- void **get_all_input_ports** ()
- void **get_all_output_ports** ()
- void **read_get_cpu_load** ()
- void **read_get_xruns** ()
- void **read_get_scheduler** ()
- void **set_use_jack_transport** ()
- void **set_locate** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Static Private Member Functions

- static int **IOProcessEvent_inner** (void *handle, **mha_wave_t** *sIn, **mha_wave_t** **s↩ Out)

Private Attributes

- **IOProcessEvent_t proc_event**
- void * **proc_handle**
- unsigned int **mha_fragsize**
- float **mha_samplerate**
- unsigned int **fragsize_ratio**
- **MHAParser::string_t servername**
- **MHAParser::string_t clientname**
- **MHAParser::vstring_t connections_in**
- **MHAParser::vstring_t connections_out**
- **MHAParser::vstring_t portnames_in**
- **MHAParser::vstring_t portnames_out**
- **MHAParser::bool_t fail_on_async_jackerr**
- **MHAParser::bool_t use_jack_transport**
- **MHAParser::float_t locate**
- **MHAParser::float_mon_t server_srate**
- **MHAParser::int_mon_t server_fragsize**
- **MHAParser::vstring_mon_t ports_in_physical**
- **MHAParser::vstring_mon_t ports_out_physical**
- **MHAParser::vstring_mon_t ports_in_all**
- **MHAParser::vstring_mon_t ports_out_all**
- **MHAParser::parser_t ports_parser**
- **MHAParser::float_mon_t state_cpuload**
- **MHAParser::int_mon_t state_xruns**
- **MHAParser::int_mon_t state_priority**
- **MHAParser::string_mon_t state_scheduler**
- **MHAParser::parser_t state_parser**
- **MHASignal::waveform_t** * **pwinner_out**
- **MHAEvents::patchbay_t**< **io_jack_t** > **patchbay**

Additional Inherited Members

5.262.1 Detailed Description

Main class for JACK IO.

This class registers a JACK client. JACK and framework states are managed by this class.

© 2005-2021 HörTech gGmbH, Oldenburg

5.262.2 Constructor & Destructor Documentation

5.262.2.1 io_jack_t()

```
io_jack_t::io_jack_t (
    unsigned int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle )
```

5.262.3 Member Function Documentation

5.262.3.1 prepare()

```
void io_jack_t::prepare (
    int nch_in,
    int nch_out )
```

Allocate buffers, activate JACK client and install internal ports.

5.262.3.2 release()

```
void io_jack_t::release ( )
```

5.262.3.3 fail_on_async_jackerror()

```
bool MHAIOJackdb::io_jack_t::fail_on_async_↩
jackerror ( ) const [inline]
```

5.262.3.4 IOProcessEvent_inner() [1/2]

```
int io_jack_t::IOProcessEvent_inner (
    void * handle,
    mha_wave_t * sIn,
    mha_wave_t ** sOut ) [static], [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.262.3.5 IOProcessEvent_inner() [2/2]

```
int io_jack_t::IOProcessEvent_inner (
    mha_wave_t * sIn,
    mha_wave_t ** sOut ) [private]
```

5.262.3.6 reconnect_inports()

```
void io_jack_t::reconnect_inports ( ) [private]
```

Connect the input ports when connection variable is accessed.

5.262.3.7 reconnect_outports()

```
void io_jack_t::reconnect_outports ( ) [private]
```

Connect the output ports when connection variable is accessed.

5.262.3.8 get_physical_input_ports()

```
void io_jack_t::get_physical_input_ports ( ) [private]
```

5.262.3.9 get_physical_output_ports()

```
void io_jack_t::get_physical_output_ports ( ) [private]
```

5.262.3.10 get_all_input_ports()

```
void io_jack_t::get_all_input_ports ( ) [private]
```

5.262.3.11 get_all_output_ports()

```
void io_jack_t::get_all_output_ports ( ) [private]
```

5.262.3.12 read_get_cpu_load()

```
void io_jack_t::read_get_cpu_load ( ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.262.3.13 read_get_xruns() `void io_jack_t::read_get_xruns ( ) [private]`

5.262.3.14 read_get_scheduler() `void io_jack_t::read_get_scheduler ( ) [private]`

5.262.3.15 set_use_jack_transport() `void io_jack_t::set_use_jack_transport ( ) [private]`

5.262.3.16 set_locate() `void io_jack_t::set_locate ( ) [private]`

5.262.4 Member Data Documentation

5.262.4.1 proc_event `IOProcessEvent_t MHAIOJackdb::io_jack_t::proc_event [private]`

5.262.4.2 proc_handle `void* MHAIOJackdb::io_jack_t::proc_handle [private]`

5.262.4.3 mha_fragsize `unsigned int MHAIOJackdb::io_jack_t::mha_fragsize [private]`

5.262.4.4 mha_samplerate `float MHAIOJackdb::io_jack_t::mha_samplerate [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.262.4.5 fragsize_ratio

`unsigned int MHAIOJackdb::io_jack_t::fragsize_ratio [private]`

5.262.4.6 servername

`MHAParser::string_t MHAIOJackdb::io_jack_t::servername [private]`

5.262.4.7 clientname

`MHAParser::string_t MHAIOJackdb::io_jack_t::clientname [private]`

5.262.4.8 connections_in

`MHAParser::vstring_t MHAIOJackdb::io_jack_t::connections↩_in [private]`

5.262.4.9 connections_out

`MHAParser::vstring_t MHAIOJackdb::io_jack_t::connections↩_out [private]`

5.262.4.10 portnames_in

`MHAParser::vstring_t MHAIOJackdb::io_jack_t::portnames↩_in [private]`

5.262.4.11 portnames_out

`MHAParser::vstring_t MHAIOJackdb::io_jack_t::portnames↩_out [private]`

5.262.4.12 fail_on_async_jackerr

`MHAParser::bool_t MHAIOJackdb::io_jack_t::fail_↩on_async_jackerr [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.262.4.13 use_jack_transport

MHAParser::bool_t MHAIOJackdb::io_jack_t::use_↩ jack_transport [private]

5.262.4.14 locate

MHAParser::float_t MHAIOJackdb::io_jack_t::locate [private]

5.262.4.15 server_srate

MHAParser::float_mon_t MHAIOJackdb::io_jack_t::server_↩ srate [private]

5.262.4.16 server_fragsize

MHAParser::int_mon_t MHAIOJackdb::io_jack_t::server_↩ fragsize [private]

5.262.4.17 ports_in_physical

MHAParser::vstring_mon_t MHAIOJackdb::io_jack_t↩ ::ports_in_physical [private]

5.262.4.18 ports_out_physical

MHAParser::vstring_mon_t MHAIOJackdb::io_jack_t↩ ::ports_out_physical [private]

5.262.4.19 ports_in_all

MHAParser::vstring_mon_t MHAIOJackdb::io_jack_t::ports_↩ in_all [private]

5.262.4.20 ports_out_all

MHAParser::vstring_mon_t MHAIOJackdb::io_jack_t::ports_↩ out_all [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.262.4.21 ports_parser

MHAParser::parser_t MHAIOJackdb::io_jack_t::ports_parser [private]

5.262.4.22 state_cpuload

MHAParser::float_mon_t MHAIOJackdb::io_jack_t::state_↩cpuload [private]

5.262.4.23 state_xruns

MHAParser::int_mon_t MHAIOJackdb::io_jack_t::state_xruns [private]

5.262.4.24 state_priority

MHAParser::int_mon_t MHAIOJackdb::io_jack_t::state_↩priority [private]

5.262.4.25 state_scheduler

MHAParser::string_mon_t MHAIOJackdb::io_jack_t::state↩_scheduler [private]

5.262.4.26 state_parser

MHAParser::parser_t MHAIOJackdb::io_jack_t::state_parser [private]

5.262.4.27 pwinner_out

MHASignal::waveform_t* MHAIOJackdb::io_jack_t::pwinner_out [private]

5.262.4.28 patchbay

MHAEvents::patchbay_t< io_jack_t> MHAIOJackdb::io_jack_t↩::patchbay [private]

The documentation for this class was generated from the following file:

- **MHAIOJackdb.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.263 MHAIOPortAudio::device_info_t Class Reference

Inheritance diagram for MHAIOPortAudio::device_info_t:

Public Member Functions

- **device_info_t** ()
- void **fill_info** ()

Public Attributes

- **MHAParser::int_mon_t numDevices**
- **MHAParser::vint_mon_t structVersion**
- **MHAParser::vstring_mon_t name**
- **MHAParser::vint_mon_t hostApi**
- **MHAParser::vint_mon_t maxInputChannels**
- **MHAParser::vint_mon_t maxOutputChannels**
- **MHAParser::vfloat_mon_t defaultLowInputLatency**
- **MHAParser::vfloat_mon_t defaultLowOutputLatency**
- **MHAParser::vfloat_mon_t defaultHighInputLatency**
- **MHAParser::vfloat_mon_t defaultHighOutputLatency**
- **MHAParser::vfloat_mon_t defaultSampleRate**

Additional Inherited Members

5.263.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.263.1.1 device_info_t() `MHAIOPortAudio::device_info_t::device_info_t ( ) [inline]`

5.263.2 Member Function Documentation

5.263.2.1 fill_info() `void MHAIOPortAudio::device_info_t::fill_info ( ) [inline]`

5.263.3 Member Data Documentation

5.263.3.1 numDevices `MHAParser::int_mon_t MHAIOPortAudio::device_info_t::num↩ Devices`

5.263.3.2 structVersion `MHAParser::vint_mon_t MHAIOPortAudio::device_info_t::struct↩ Version`

5.263.3.3 name `MHAParser::vstring_mon_t MHAIOPortAudio::device_info_t::name`

5.263.3.4 hostApi `MHAParser::vint_mon_t MHAIOPortAudio::device_info_t::hostApi`

5.263.3.5 maxInputChannels `MHAParser::vint_mon_t MHAIOPortAudio::device_info_t↩ ::maxInputChannels`

© 2005-2021 HörTech gGmbH, Oldenburg

5.263.3.6 maxOutputChannels

MHAParser::vint_mon_t MHAIOPortAudio::device_info↩ _t::maxOutputChannels

5.263.3.7 defaultLowInputLatency

MHAParser::vfloat_mon_t MHAIOPortAudio::device↩ _info_t::defaultLowInputLatency

5.263.3.8 defaultLowOutputLatency

MHAParser::vfloat_mon_t MHAIOPortAudio::device↩ _info_t::defaultLowOutputLatency

5.263.3.9 defaultHighInputLatency

MHAParser::vfloat_mon_t MHAIOPortAudio::device↩ _info_t::defaultHighInputLatency

5.263.3.10 defaultHighOutputLatency

MHAParser::vfloat_mon_t MHAIOPortAudio↩ ::device_info_t::defaultHighOutputLatency

5.263.3.11 defaultSampleRate

MHAParser::vfloat_mon_t MHAIOPortAudio::device_↩ info_t::defaultSampleRate

The documentation for this class was generated from the following file:

- **MHAIOPortAudio.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.264 MHAIOPortAudio::io_portaudio_t Class Reference

Main class for Portaudio sound IO.

Inheritance diagram for MHAIOPortAudio::io_portaudio_t:

Public Member Functions

- **io_portaudio_t** (unsigned int **fragsize**, float **samplerate**, **IOProcessEvent_t proc↩ _event**, void ∗ **proc_handle**, **IOStartedEvent_t start_event**, void ∗ **start_handle**, **I↩ OStoppedEvent_t stop_event**, void ∗ **stop_handle**)
- void **device_name_in_updated** ()
- void **device_name_out_updated** ()
- void **device_index_in_updated** ()
- void **device_index_out_updated** ()
- ∼**io_portaudio_t** ()
- void **cmd_prepare** (int, int)
- void **cmd_start** ()
- void **cmd_stop** ()
- void **cmd_release** ()
- int **portaudio_callback** (const void ∗input, void ∗output, unsigned long frame_count, const PaStreamCallbackTimeInfo ∗time_info, PaStreamCallbackFlags status_flags)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **device_info_t device_info**
- **stream_info_t stream_info**
- **MHASignal::waveform_t** $*$ **s_in**
- **mha_wave_t** $*$ **s_out**
- float **samplerate**
- unsigned int **nchannels_out**
- unsigned int **nchannels_in**
- unsigned int **fragsize**
- **IOProcessEvent_t proc_event**
- void $*$ **proc_handle**
- **IOStartedEvent_t start_event**
- void $*$ **start_handle**
- **IOStoppedEvent_t stop_event**
- void $*$ **stop_handle**
- PaStream $*$ **portaudio_stream**
- **MHAParser::string_t device_name_in**
- **MHAParser::int_t device_index_in**
- **MHAParser::string_t device_name_out**
- **MHAParser::int_t device_index_out**
- **MHAParser::float_t suggestedInputLatency**
- **MHAParser::float_t suggestedOutputLatency**
- **MHAEvents::patchbay_t**$<$ **io_portaudio_t** $>$ **patchbay**

Additional Inherited Members

5.264.1 Detailed Description

Main class for Portaudio sound IO.

5.264.2 Constructor & Destructor Documentation

5.264.2.1 io_portaudio_t()

```
MHAIOPortAudio::io_portaudio_t::io_portaudio_t (
    unsigned int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle ) [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.264.2.2 ~io_portaudio_t() `MHAIOPortAudio::io_portaudio_t::~io_portaudio_t ( ) [inline]`

5.264.3 Member Function Documentation

5.264.3.1 device_name_in_updated() `void MHAIOPortAudio::io_portaudio_t::device_name_in_updated ( ) [inline]`

5.264.3.2 device_name_out_updated() `void MHAIOPortAudio::io_portaudio_t::device_name_out_updated ( ) [inline]`

5.264.3.3 device_index_in_updated() `void MHAIOPortAudio::io_portaudio_t::device_index_in_updated ( ) [inline]`

5.264.3.4 device_index_out_updated() `void MHAIOPortAudio::io_portaudio_t::device_index_out_updated ( ) [inline]`

5.264.3.5 cmd_prepare()

```
void MHAIOPortAudio::io_portaudio_t::cmd_prepare (
    int nchannels_in,
    int nchannels_out )
```

5.264.3.6 cmd_start() `void MHAIOPortAudio::io_portaudio_t::cmd_start ( )`

© 2005-2021 HörTech gGmbH, Oldenburg

5.264.3.7 cmd_stop() `void MHAIOPortAudio::io_portaudio_t::cmd_stop ( )`

5.264.3.8 cmd_release() `void MHAIOPortAudio::io_portaudio_t::cmd_release ( )`

5.264.3.9 portaudio_callback() `int MHAIOPortAudio::io_portaudio_t::portaudio_callback (`

```
const void * input,
void * output,
unsigned long frame_count,
const PaStreamCallbackTimeInfo * time_info,
PaStreamCallbackFlags status_flags )
```

5.264.4 Member Data Documentation

5.264.4.1 device_info `device_info_t MHAIOPortAudio::io_portaudio_t::device_info [private]`

5.264.4.2 stream_info `stream_info_t MHAIOPortAudio::io_portaudio_t::stream_info [private]`

5.264.4.3 s_in `MHASignal::waveform_t* MHAIOPortAudio::io_portaudio_t::s_in [private]`

5.264.4.4 s_out `mha_wave_t* MHAIOPortAudio::io_portaudio_t::s_out [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.264.4.5 samplerate

`float MHAIOPortAudio::io_portaudio_t::samplerate [private]`

5.264.4.6 nchannels_out

`unsigned int MHAIOPortAudio::io_portaudio_t::nchannels_out [private]`

5.264.4.7 nchannels_in

`unsigned int MHAIOPortAudio::io_portaudio_t::nchannels_in [private]`

5.264.4.8 fragsize

`unsigned int MHAIOPortAudio::io_portaudio_t::fragsize [private]`

5.264.4.9 proc_event

`IOProcessEvent_t MHAIOPortAudio::io_portaudio_t::proc_event [private]`

5.264.4.10 proc_handle

`void* MHAIOPortAudio::io_portaudio_t::proc_handle [private]`

5.264.4.11 start_event

`IOStartedEvent_t MHAIOPortAudio::io_portaudio_t::start_event [private]`

5.264.4.12 start_handle

`void* MHAIOPortAudio::io_portaudio_t::start_handle [private]`

5.264.4.13 stop_event

`IOStoppedEvent_t MHAIOPortAudio::io_portaudio_t::stop_event [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.264.4.14 stop_handle `void* MHAIOPortAudio::io_portaudio_t::stop_handle [private]`

5.264.4.15 portaudio_stream `PaStream* MHAIOPortAudio::io_portaudio_t::portaudio_stream [private]`

5.264.4.16 device_name_in `MHAParser::string_t MHAIOPortAudio::io_portaudio_t::device_name_in [private]`

5.264.4.17 device_index_in `MHAParser::int_t MHAIOPortAudio::io_portaudio_t::device_index_in [private]`

5.264.4.18 device_name_out `MHAParser::string_t MHAIOPortAudio::io_portaudio_t::device_name_out [private]`

5.264.4.19 device_index_out `MHAParser::int_t MHAIOPortAudio::io_portaudio_t::device_index_out [private]`

5.264.4.20 suggestedInputLatency `MHAParser::float_t MHAIOPortAudio::io_portaudio_t::suggestedInputLatency [private]`

5.264.4.21 suggestedOutputLatency `MHAParser::float_t MHAIOPortAudio::io_portaudio_t::suggestedOutputLatency [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.264.4.22 patchbay

`MHAEvents::patchbay_t< io_portaudio_t> MHAIOPortAudio::io_portaudio_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **MHAIOPortAudio.cpp**

5.265 MHAIOPortAudio::stream_info_t Class Reference

Inheritance diagram for MHAIOPortAudio::stream_info_t:

Public Member Functions

- **stream_info_t** ()
- void **fill_info** (PaStream *stream_)

Public Attributes

- **MHAParser::float_mon_t paInputLatency**
- **MHAParser::float_mon_t paOutputLatency**
- **MHAParser::float_mon_t paSampleRate**

Additional Inherited Members

5.265.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.265.1.1 stream_info_t() `MHAIOPortAudio::stream_info_t::stream_info_t ( ) [inline]`

5.265.2 Member Function Documentation

5.265.2.1 fill_info() `void MHAIOPortAudio::stream_info_t::fill_info ( PaStream * stream_ ) [inline]`

5.265.3 Member Data Documentation

5.265.3.1 paInputLatency `MHAParser::float_mon_t MHAIOPortAudio::stream_info_t↩ ::paInputLatency`

5.265.3.2 paOutputLatency `MHAParser::float_mon_t MHAIOPortAudio::stream_info_t↩ ::paOutputLatency`

5.265.3.3 paSampleRate `MHAParser::float_mon_t MHAIOPortAudio::stream_info_t↩ ::paSampleRate`

The documentation for this class was generated from the following file:

- **MHAIOPortAudio.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.266 MHAJack::client_avg_t Class Reference

Generic JACK client for averaging a system response across time.

Inheritance diagram for MHAJack::client_avg_t:

Public Member Functions

- **client_avg_t** (const std::string & **name**, const unsigned int &nrep_)

  Constructor for averaging client.
- void **io** (**mha_wave_t** ∗ **s_out**, **mha_wave_t** ∗ **s_in**, const std::vector< std::string > &p_out, const std::vector< std::string > &p_in, float ∗srate=NULL, unsigned int ∗ **fragsize**=NULL)

  Recording function.

Private Member Functions

- void **proc** (**mha_wave_t** ∗sIn, **mha_wave_t** ∗∗sOut)
- void **IOStoppedEvent** ()

Static Private Member Functions

- static int **proc** (void ∗handle, **mha_wave_t** ∗sIn, **mha_wave_t** ∗∗sOut)
- static void **IOStoppedEvent** (void ∗handle, int proc_err, int io_err)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- bool **b_stopped**
- unsigned int **pos**
- **mha_wave_t** ∗ **sn_in**
- **mha_wave_t** ∗ **sn_out**
- std::string **name**
- **MHASignal::waveform_t** ∗ **frag_out**
- const unsigned int **nrep**
- unsigned int **n**
- bool **b_ready**

Additional Inherited Members

5.266.1 Detailed Description

Generic JACK client for averaging a system response across time.

5.266.2 Constructor & Destructor Documentation

5.266.2.1 client_avg_t()

```
MHAJack::client_avg_t::client_avg_t (
    const std::string & name_,
    const unsigned int & nrep_ )
```

Constructor for averaging client.

Parameters

*name_*	Name of JACK client
*nrep_*	Number of repetitions

5.266.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.266.3.1 io()

```
void MHAJack::client_avg_t::io (
    mha_wave_t * is_out,
    mha_wave_t * is_in,
    const std::vector< std::string > & p_out,
    const std::vector< std::string > & p_in,
    float * srate = NULL,
    unsigned int * fragsize = NULL )
```

Recording function.

long-description

Parameters

is_out	Input (test) signal, which will be repeated
is_in	System response (averaged, same length as input required)
p_out	Ports to play back the test signal
p_in	Ports to record from the system response
srate	Pointer to sampling rate variable, will be filled with server sampling rate
fragsize	Pointer to fragment size variable, will be filled with server fragment size

5.266.3.2 proc() [1/2]

```
int MHAJack::client_avg_t::proc (
    void * handle,
    mha_wave_t * sIn,
    mha_wave_t ** sOut ) [static], [private]
```

5.266.3.3 IOStoppedEvent() [1/2]

```
void MHAJack::client_avg_t::IOStoppedEvent (
    void * handle,
    int proc_err,
    int io_err ) [static], [private]
```

5.266.3.4 proc() [2/2]

```
void MHAJack::client_avg_t::proc (
    mha_wave_t * sIn,
    mha_wave_t ** sOut ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.266.3.5 IOStoppedEvent() [2/2] void MHAJack::client_avg_t::IOStoppedEvent ( ) [private]

5.266.4 Member Data Documentation

5.266.4.1 b_stopped bool MHAJack::client_avg_t::b_stopped [private]

5.266.4.2 pos unsigned int MHAJack::client_avg_t::pos [private]

5.266.4.3 sn_in mha_wave_t* MHAJack::client_avg_t::sn_in [private]

5.266.4.4 sn_out mha_wave_t* MHAJack::client_avg_t::sn_out [private]

5.266.4.5 name std::string MHAJack::client_avg_t::name [private]

5.266.4.6 frag_out MHASignal::waveform_t* MHAJack::client_avg_t::frag_out [private]

5.266.4.7 nrep const unsigned int MHAJack::client_avg_t::nrep [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.266.4.8 n `unsigned int MHAJack::client_avg_t::n [private]`

5.266.4.9 b_ready `bool MHAJack::client_avg_t::b_ready [private]`

The documentation for this class was generated from the following files:

- **mhajack.h**
- **mhajack.cpp**

5.267 MHAJack::client_noncont_t Class Reference

Generic client for synchronous playback and recording of waveform fragments.

Inheritance diagram for MHAJack::client_noncont_t:

Public Member Functions

- **client_noncont_t** (const std::string & **name**, bool **use_jack_transport**=false)
- void **io** (**mha_wave_t** ∗ **s_out**, **mha_wave_t** ∗ **s_in**, const std::vector< std::string > &p_out, const std::vector< std::string > &p_in, float ∗srate=NULL, unsigned int ∗ **fragsize**=NULL)

Private Member Functions

- void **proc** (**mha_wave_t** ∗sIn, **mha_wave_t** ∗∗sOut)
- void **IOStoppedEvent** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Static Private Member Functions

- static int **proc** (void ∗handle, **mha_wave_t** ∗sIn, **mha_wave_t** ∗∗sOut)
- static void **IOStoppedEvent** (void ∗handle, int proc_err, int io_err)

Private Attributes

- bool **b_stopped**
- unsigned int **pos**
- **mha_wave_t** ∗ **sn_in**
- **mha_wave_t** ∗ **sn_out**
- std::string **name**
- **MHASignal::waveform_t** ∗ **frag_out**

Additional Inherited Members

5.267.1 Detailed Description

Generic client for synchronous playback and recording of waveform fragments.

5.267.2 Constructor & Destructor Documentation

5.267.2.1 client_noncont_t()

```
MHAJack::client_noncont_t::client_noncont_t (
    const std::string & name,
    bool use_jack_transport = false )
```

5.267.3 Member Function Documentation

5.267.3.1 io()

```
void MHAJack::client_noncont_t::io (
    mha_wave_t * s_out,
    mha_wave_t * s_in,
    const std::vector< std::string > & p_out,
    const std::vector< std::string > & p_in,
    float * srate = NULL,
    unsigned int * fragsize = NULL )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.267.3.2 proc() [1/2]

```
int MHAJack::client_noncont_t::proc (
    void * handle,
    mha_wave_t * sIn,
    mha_wave_t ** sOut ) [static], [private]
```

5.267.3.3 IOStoppedEvent() [1/2]

```
void MHAJack::client_noncont_t::IOStoppedEvent (
    void * handle,
    int proc_err,
    int io_err ) [static], [private]
```

5.267.3.4 proc() [2/2]

```
void MHAJack::client_noncont_t::proc (
    mha_wave_t * sIn,
    mha_wave_t ** sOut ) [private]
```

5.267.3.5 IOStoppedEvent() [2/2]

```
void MHAJack::client_noncont_t::IOStoppedEvent (
) [private]
```

5.267.4 Member Data Documentation

5.267.4.1 b_stopped

```
bool MHAJack::client_noncont_t::b_stopped [private]
```

5.267.4.2 pos

```
unsigned int MHAJack::client_noncont_t::pos [private]
```

5.267.4.3 sn_in

```
mha_wave_t* MHAJack::client_noncont_t::sn_in [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.267.4.4 sn_out `mha_wave_t* MHAJack::client_noncont_t::sn_out [private]`

5.267.4.5 name `std::string MHAJack::client_noncont_t::name [private]`

5.267.4.6 frag_out `MHASignal::waveform_t* MHAJack::client_noncont_t::frag_out [private]`

The documentation for this class was generated from the following files:

- **mhajack.h**
- **mhajack.cpp**

5.268 MHAJack::client_t Class Reference

Generic asynchronous JACK client.

Inheritance diagram for MHAJack::client_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **client_t** (**IOProcessEvent_t** **proc_event**, void ∗ **proc_handle**=NULL, **IOStarted↩ Event_t** **start_event**=NULL, void ∗ **start_handle**=NULL, **IOStoppedEvent_t** **stop_↩ event**=NULL, void ∗ **stop_handle**=NULL, bool **use_jack_transport**=false)
- void **prepare** (const std::string &client_name, const unsigned int & **nchannels_in**, const unsigned int & **nchannels_out**)

  Allocate buffers, activate JACK client and install internal ports.
- void **prepare** (const std::string &server_name, const std::string &client_name, const unsigned int & **nchannels_in**, const unsigned int & **nchannels_out**)

  Allocate buffers, ports, and activates JACK client.
- void **release** ()

  Remove JACK client and deallocate internal ports and buffers.
- void **start** (bool fail_on_async_jack_error=true)
- void **stop** ()
- void **connect_input** (const std::vector< std::string > &)

  Connect the input ports when connection variable is accessed.
- void **connect_output** (const std::vector< std::string > &)

  Connect the output ports when connection variable is accessed.
- unsigned int **get_fragsize** () const
- float **get_srate** () const
- unsigned long **get_xruns** ()
- unsigned long **get_xruns_reset** ()
- std::string **str_error** (int err)
- void **get_ports** (std::vector< std::string > &, unsigned long jack_flags)

  Get a list of Jack ports.
- std::vector< std::string > **get_my_input_ports** ()
- std::vector< std::string > **get_my_output_ports** ()
- void **set_input_portnames** (const std::vector< std::string > &)
- void **set_output_portnames** (const std::vector< std::string > &)
- float **get_cpu_load** ()
- void **set_use_jack_transport** (bool ut)
- bool **is_prepared** () const

Protected Attributes

- jack_client_t ∗ **jc**

Private Member Functions

- void **prepare_impl** (const char ∗server_name, const char ∗client_name, const unsigned int & **nchannels_in**, const unsigned int & **nchannels_out**)

  Allocate buffers, activate JACK client and allocates jack ports Registers the jack client with the given server and activates it.
- void **internal_start** ()
- void **internal_stop** ()
- void **stopped** (int, int)
- int **jack_proc_cb** (jack_nframes_t)

  This is the main processing callback.
- int **jack_xrun_cb** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Static Private Member Functions

- static int **jack_proc_cb** (jack_nframes_t, void ∗)
- static int **jack_xrun_cb** (void ∗)

Private Attributes

- unsigned long **num_xruns**
- unsigned int **fragsize**
- float **samplerate**
- unsigned int **nchannels_in**
- unsigned int **nchannels_out**
- **IOProcessEvent_t proc_event**
- void ∗ **proc_handle**
- **IOStartedEvent_t start_event**
- void ∗ **start_handle**
- **IOStoppedEvent_t stop_event**
- void ∗ **stop_handle**
- **MHASignal::waveform_t** ∗ **s_in**
- **mha_wave_t** ∗ **s_out**
- **MHAJack::port_t** ∗∗ **inch**
- **MHAJack::port_t** ∗∗ **outch**
- unsigned int **flags**
- bool **b_prepared**
- bool **use_jack_transport**
- jack_transport_state_t **jstate_prev**
- std::vector< std::string > **input_portnames**
- std::vector< std::string > **output_portnames**
- bool **fail_on_async_jackerror**

5.268.1 Detailed Description

Generic asynchronous JACK client.

5.268.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.268.2.1 client_t()

```
MHAJack::client_t::client_t (
    IOProcessEvent_t proc_event,
    void * proc_handle = NULL,
    IOStartedEvent_t start_event = NULL,
    void * start_handle = NULL,
    IOStoppedEvent_t stop_event = NULL,
    void * stop_handle = NULL,
    bool use_jack_transport = false )
```

5.268.3 Member Function Documentation

5.268.3.1 prepare() [1/2]

```
void MHAJack::client_t::prepare (
    const std::string & client_name,
    const unsigned int & nch_in,
    const unsigned int & nch_out )
```

Allocate buffers, activate JACK client and install internal ports.

Registers the jack client with the default jack server and activates it.

Parameters

client_name	Name of this jack client
nch_in	Input ports to register
nch_out	Output ports to register

5.268.3.2 prepare() [2/2]

```
void MHAJack::client_t::prepare (
    const std::string & server_name,
    const std::string & client_name,
    const unsigned int & nch_in,
    const unsigned int & nch_out )
```

Allocate buffers, ports, and activates JACK client.

Registers the jack client with specified jack server and activates it.

Parameters

server_name	Name of the jack server to register with
client_name	Name of this jack client
nch_in	Input ports to register
nch_out	Output ports to register

© 2005-2021 HörTech gGmbH, Oldenburg

5.268.3.3 release() `void MHAJack::client_t::release ( )`

Remove JACK client and deallocate internal ports and buffers.

5.268.3.4 start() `void MHAJack::client_t::start ( bool fail_on_async_jack_error = true )`

5.268.3.5 stop() `void MHAJack::client_t::stop ( )`

5.268.3.6 connect_input() `void MHAJack::client_t::connect_input ( const std::vector< std::string > & con )`

Connect the input ports when connection variable is accessed.

5.268.3.7 connect_output() `void MHAJack::client_t::connect_output ( const std::vector< std::string > & con )`

Connect the output ports when connection variable is accessed.

5.268.3.8 get_fragsize() `unsigned int MHAJack::client_t::get_fragsize ( ) const [inline]`

5.268.3.9 get_srate() `float MHAJack::client_t::get_srate ( ) const [inline]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.268.3.10 get_xruns() `unsigned long MHAJack::client_t::get_xruns ( ) [inline]`

5.268.3.11 get_xruns_reset() `unsigned long MHAJack::client_t::get_xruns_reset ( )`

5.268.3.12 str_error() `std::string MHAJack::client_t::str_error ( int err )`

5.268.3.13 get_ports() `void MHAJack::client_t::get_ports ( std::vector< std::string > & res, unsigned long jack_flags )`

Get a list of Jack ports.

Parameters

res	Result string vector
jack_flags	Jack port flags (JackPortInput etc.)

5.268.3.14 get_my_input_ports() `std::vector< std::string > MHAJack::client_t↩ ::get_my_input_ports ( )`

5.268.3.15 get_my_output_ports() `std::vector< std::string > MHAJack::client_t↩ ::get_my_output_ports ( )`

5.268.3.16 set_input_portnames() `void MHAJack::client_t::set_input_portnames ( const std::vector< std::string > & names )`

© 2005-2021 HörTech gGmbH, Oldenburg

5.268.3.17 set_output_portnames()

```
void MHAJack::client_t::set_output_portnames (
const std::vector< std::string > & names )
```

5.268.3.18 get_cpu_load()

```
float MHAJack::client_t::get_cpu_load ( )
```

5.268.3.19 set_use_jack_transport()

```
void MHAJack::client_t::set_use_jack_transport
(
bool ut ) [inline]
```

5.268.3.20 is_prepared()

```
bool MHAJack::client_t::is_prepared ( ) const [inline]
```

5.268.3.21 prepare_impl()

```
void MHAJack::client_t::prepare_impl (
const char * server_name,
const char * client_name,
const unsigned int & nch_in,
const unsigned int & nch_out ) [private]
```

Allocate buffers, activate JACK client and allocates jack ports Registers the jack client with the given server and activates it.

Parameters

server_name	Name of the jack server to register with
client_name	Name of this jack client
nch_in	Input ports to register
nch_out	Output ports to register

5.268.3.22 internal_start()

```
void MHAJack::client_t::internal_start ( ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.268.3.23 internal_stop()

```
void MHAJack::client_t::internal_stop ( ) [private]
```

5.268.3.24 stopped()

```
void MHAJack::client_t::stopped (
int proc_err,
int io_err ) [private]
```

5.268.3.25 jack_proc_cb() [1/2]

```
int MHAJack::client_t::jack_proc_cb (
jack_nframes_t n,
void * h ) [static], [private]
```

5.268.3.26 jack_proc_cb() [2/2]

```
int MHAJack::client_t::jack_proc_cb (
jack_nframes_t n ) [private]
```

This is the main processing callback.

Here happens double buffering and downsampling.

5.268.3.27 jack_xrun_cb() [1/2]

```
int MHAJack::client_t::jack_xrun_cb (
void * h ) [static], [private]
```

5.268.3.28 jack_xrun_cb() [2/2]

```
int MHAJack::client_t::jack_xrun_cb ( ) [inline],
[private]
```

5.268.4 Member Data Documentation

5.268.4.1 num_xruns

```
unsigned long MHAJack::client_t::num_xruns [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.268.4.2 fragsize `unsigned int MHAJack::client_t::fragsize [private]`

5.268.4.3 samplerate `float MHAJack::client_t::samplerate [private]`

5.268.4.4 nchannels_in `unsigned int MHAJack::client_t::nchannels_in [private]`

5.268.4.5 nchannels_out `unsigned int MHAJack::client_t::nchannels_out [private]`

5.268.4.6 proc_event `IOProcessEvent_t MHAJack::client_t::proc_event [private]`

5.268.4.7 proc_handle `void* MHAJack::client_t::proc_handle [private]`

5.268.4.8 start_event `IOStartedEvent_t MHAJack::client_t::start_event [private]`

5.268.4.9 start_handle `void* MHAJack::client_t::start_handle [private]`

5.268.4.10 stop_event `IOStoppedEvent_t MHAJack::client_t::stop_event [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.268.4.11 stop_handle `void* MHAJack::client_t::stop_handle [private]`

5.268.4.12 s_in `MHASignal::waveform_t* MHAJack::client_t::s_in [private]`

5.268.4.13 s_out `mha_wave_t* MHAJack::client_t::s_out [private]`

5.268.4.14 inch `MHAJack::port_t** MHAJack::client_t::inch [private]`

5.268.4.15 outch `MHAJack::port_t** MHAJack::client_t::outch [private]`

5.268.4.16 jc `jack_client_t* MHAJack::client_t::jc [protected]`

5.268.4.17 flags `unsigned int MHAJack::client_t::flags [private]`

5.268.4.18 b_prepared `bool MHAJack::client_t::b_prepared [private]`

5.268.4.19 use_jack_transport `bool MHAJack::client_t::use_jack_transport [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.268.4.20 jstate_prev `jack_transport_state_t MHAJack::client_t::jstate_prev [private]`

5.268.4.21 input_portnames `std::vector<std::string> MHAJack::client_t::input_portnames [private]`

5.268.4.22 output_portnames `std::vector<std::string> MHAJack::client_t::output_portnames [private]`

5.268.4.23 fail_on_async_jackerror `bool MHAJack::client_t::fail_on_async_jackerror [private]`

The documentation for this class was generated from the following files:

- **mhajack.h**
- **mhajack.cpp**

5.269 MHAJack::port_t Class Reference

Class for one channel/port.

Public Types

- enum **dir_t** { **input**, **output** }

Public Member Functions

- **port_t** (jack_client_t ∗ **jc**, **dir_t** dir, int id)
- **port_t** (jack_client_t ∗ **jc**, **dir_t** dir, const std::string &id)

  Constructor to create port with specific name.
- ∼**port_t** ()
- void **read** (**mha_wave_t** ∗s, unsigned int ch)
- void **write** (**mha_wave_t** ∗s, unsigned int ch)
- void **mute** (unsigned int n)
- void **connect_to** (const char ∗pn)
- const char ∗ **get_short_name** ()

  Return the port name.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **dir_t dir_type**
- jack_port_t * **port**
- jack_default_audio_sample_t * **iob**
- jack_client_t * **jc**

5.269.1 Detailed Description

Class for one channel/port.

This class represents one JACK port. Double buffering for asynchronous process callbacks is managed by this class.

5.269.2 Member Enumeration Documentation

5.269.2.1 dir_t `enum` **MHAJack::port_t::dir_t**

Enumerator

input	
output	

5.269.3 Constructor & Destructor Documentation

5.269.3.1 port_t() [1/2] `MHAJack::port_t::port_t (`

```
jack_client_t * jc,
 dir_t dir,
int id )
```

Parameters

jc	JACK client.
dir	Direction (input/output).
id	Number in port name (starting with 1).

© 2005-2021 HörTech gGmbH, Oldenburg

5.269.3.2 port_t() [2/2]

```
MHAJack::port_t::port_t (
    jack_client_t * jc,
    dir_t dir,
    const std::string & id )
```

Constructor to create port with specific name.

Parameters

jc	JACK client.
dir	Direction (input/output).
id	Port name.

5.269.3.3 ~port_t()

```
MHAJack::port_t::~port_t ( )
```

5.269.4 Member Function Documentation

5.269.4.1 read()

```
void MHAJack::port_t::read (
    mha_wave_t * s,
    unsigned int ch )
```

Parameters

s	Signal structure to store the audio data.
ch	Channel number in audio data structure to be used.

5.269.4.2 write()

```
void MHAJack::port_t::write (
    mha_wave_t * s,
    unsigned int ch )
```

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s	Signal structure from which the audio data is read.
ch	Channel number in audio data structure to be used.

5.269.4.3 mute()

```
void MHAJack::port_t::mute (
    unsigned int n )
```

Parameters

n	Number of samples to be muted (must be the same as reported by Jack processing callback).

5.269.4.4 connect_to()

```
void MHAJack::port_t::connect_to (
    const char * pn )
```

Parameters

pn	Port name to connect to

5.269.4.5 get_short_name()

```
const char * MHAJack::port_t::get_short_name ( )
```

Return the port name.

5.269.5 Member Data Documentation

5.269.5.1 dir_type

```
dir_t MHAJack::port_t::dir_type [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.269.5.2 port `jack_port_t* MHAJack::port_t::port [private]`

5.269.5.3 iob `jack_default_audio_sample_t* MHAJack::port_t::iob [private]`

5.269.5.4 jc `jack_client_t* MHAJack::port_t::jc [private]`

The documentation for this class was generated from the following files:

- **mhajack.h**
- **mhajack.cpp**

5.270 MHAKernel::algo_comm_class_t Class Reference

Inheritance diagram for MHAKernel::algo_comm_class_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **algo_comm_class_t** ()
- virtual ~**algo_comm_class_t** ()
- **algo_comm_t get_c_handle** ()
- virtual void **local_insert_var** (const char *, **comm_var_t**)
- virtual void **local_remove_var** (const char *)
- virtual void **local_remove_ref** (void *)
- virtual bool **local_is_var** (const char *)
- virtual void **local_get_var** (const char *, **comm_var_t** *)
- virtual std::string **local_get_entries** ()
- virtual comm_var_map_t::size_type **size** () const

Static Public Member Functions

- static int **insert_var** (void *, const char *, **comm_var_t**)
- static int **insert_var_int** (void *, const char *, int *)
- static int **insert_var_vfloat** (void *handle, const char *name, std::vector< float > &ivar)
- static int **insert_var_float** (void *, const char *, float *)
- static int **insert_var_double** (void *, const char *, double *)
- static int **remove_var** (void *, const char *)
- static int **remove_ref** (void *, void *)
- static int **is_var** (void *, const char *)
- static int **get_var** (void *, const char *, **comm_var_t** *)
- static int **get_var_int** (void *, const char *, int *)
- static int **get_var_float** (void *, const char *, float *)
- static int **get_var_double** (void *, const char *, double *)
- static int **get_entries** (void *, char *, unsigned int)
- static const char * **get_error** (int)

Public Attributes

- char * **algo_comm_id_string**

Private Attributes

- **algo_comm_t ac**
- int **algo_comm_id_string_len**
- **comm_var_map_t vars**

5.270.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.270.1.1 algo_comm_class_t() `MHAKernel::algo_comm_class_t::algo_comm_class_t ( )`

5.270.1.2 ∼algo_comm_class_t() `MHAKernel::algo_comm_class_t::∼algo_comm_class_t ( ) [virtual]`

5.270.2 Member Function Documentation

5.270.2.1 get_c_handle() `algo_comm_t MHAKernel::algo_comm_class_t::get_c_handle ( )`

5.270.2.2 insert_var()

```
int MHAKernel::algo_comm_class_t::insert_var (
    void * handle,
    const char * name,
    comm_var_t var ) [static]
```

5.270.2.3 insert_var_int()

```
int MHAKernel::algo_comm_class_t::insert_var_int (
    void * handle,
    const char * name,
    int * ivar ) [static]
```

5.270.2.4 insert_var_vfloat()

```
int MHAKernel::algo_comm_class_t::insert_var_vfloat (
    void * handle,
    const char * name,
    std::vector< float > & ivar ) [static]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.270.2.5 insert_var_float()

```
int MHAKernel::algo_comm_class_t::insert_var_float (
    void * handle,
    const char * name,
    float * ivar ) [static]
```

5.270.2.6 insert_var_double()

```
int MHAKernel::algo_comm_class_t::insert_var_double (
    void * handle,
    const char * name,
    double * ivar ) [static]
```

5.270.2.7 remove_var()

```
int MHAKernel::algo_comm_class_t::remove_var (
    void * handle,
    const char * name ) [static]
```

5.270.2.8 remove_ref()

```
int MHAKernel::algo_comm_class_t::remove_ref (
    void * handle,
    void * ref ) [static]
```

5.270.2.9 is_var()

```
int MHAKernel::algo_comm_class_t::is_var (
    void * handle,
    const char * name ) [static]
```

5.270.2.10 get_var()

```
int MHAKernel::algo_comm_class_t::get_var (
    void * handle,
    const char * name,
    comm_var_t * var ) [static]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.270.2.11 get_var_int()

```
int MHAKernel::algo_comm_class_t::get_var_int (
void * handle,
const char * name,
int * ivar ) [static]
```

5.270.2.12 get_var_float()

```
int MHAKernel::algo_comm_class_t::get_var_float (
void * handle,
const char * name,
float * ivar ) [static]
```

5.270.2.13 get_var_double()

```
int MHAKernel::algo_comm_class_t::get_var_double (
void * handle,
const char * name,
double * ivar ) [static]
```

5.270.2.14 get_entries()

```
int MHAKernel::algo_comm_class_t::get_entries (
void * handle,
char * ret,
unsigned int len ) [static]
```

5.270.2.15 get_error()

```
const char * MHAKernel::algo_comm_class_t::get_error (
int e ) [static]
```

5.270.2.16 local_insert_var()

```
void MHAKernel::algo_comm_class_t::local_insert_var (
const char * name,
 comm_var_t var ) [virtual]
```

5.270.2.17 local_remove_var()

```
void MHAKernel::algo_comm_class_t::local_remove_var
(
const char * name ) [virtual]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.270.2.18 local_remove_ref()

`void MHAKernel::algo_comm_class_t::local_remove_ref ( void * addr ) [virtual]`

5.270.2.19 local_is_var()

`bool MHAKernel::algo_comm_class_t::local_is_var ( const char * name ) [virtual]`

5.270.2.20 local_get_var()

`void MHAKernel::algo_comm_class_t::local_get_var ( const char * name, comm_var_t * var ) [virtual]`

5.270.2.21 local_get_entries()

`std::string MHAKernel::algo_comm_class_t::local_get_entries ( ) [virtual]`

5.270.2.22 size()

`MHAKernel::comm_var_map_t::size_type MHAKernel::algo_comm_class_t::size ( ) const [virtual]`

5.270.3 Member Data Documentation

5.270.3.1 algo_comm_id_string

`char* MHAKernel::algo_comm_class_t::algo_comm_id_string`

5.270.3.2 ac

`algo_comm_t MHAKernel::algo_comm_class_t::ac [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.270.3.3 algo_comm_id_string_len `int MHAKernel::algo_comm_class_t::algo_comm_↩ id_string_len [private]`

5.270.3.4 vars `comm_var_map_t MHAKernel::algo_comm_class_t::vars [private]`

The documentation for this class was generated from the following files:

- **mha_algo_comm.hh**
- **mha_algo_comm.cpp**

5.271 MHAKernel::comm_var_map_t Class Reference

Inherits map< std::string, comm_var_t >.

Public Member Functions

- bool **has_key** (const std::string &name)

5.271.1 Member Function Documentation

5.271.1.1 has_key() `bool MHAKernel::comm_var_map_t::has_key ( const std::string & name ) [inline]`

The documentation for this class was generated from the following file:

- **mha_algo_comm.hh**

© 2005-2021 HörTech gGmbH, Oldenburg

5.272 MHAMultiSrc::base_t Class Reference

Base class for source selection.

Inheritance diagram for MHAMultiSrc::base_t:

Public Member Functions

- **base_t** (**algo_comm_t** iac)
- void **select_source** (const std::vector< std::string > &src, int in_channels)

  Change the selection of input sources.

Protected Attributes

- **algo_comm_t ac**

Additional Inherited Members

5.272.1 Detailed Description

Base class for source selection.

See also

MHAMultiSrc::channel_t (p. 997)
MHAMultiSrc::channels_t (p. 997)

© 2005-2021 HörTech gGmbH, Oldenburg

5.272.2 Constructor & Destructor Documentation

5.272.2.1 base_t()

```
MHAMultiSrc::base_t::base_t (
    algo_comm_t iac )
```

5.272.3 Member Function Documentation

5.272.3.1 select_source()

```
void MHAMultiSrc::base_t::select_source (
    const std::vector< std::string > & src,
    int in_channels )
```

Change the selection of input sources.

This function is real-time and thread safe.

Parameters

src	List of input sources
in_channels	Number of input channels in direct input (the processed signal)

5.272.4 Member Data Documentation

5.272.4.1 ac

```
algo_comm_t MHAMultiSrc::base_t::ac [protected]
```

The documentation for this class was generated from the following files:

- **mha_multisrc.h**
- **mha_multisrc.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.273 MHAMultiSrc::channel_t Class Reference

Public Attributes

- std::string **name**
- int **channel**

5.273.1 Member Data Documentation

5.273.1.1 name `std::string MHAMultiSrc::channel_t::name`

5.273.1.2 channel `int MHAMultiSrc::channel_t::channel`

The documentation for this class was generated from the following file:

- **mha_multisrc.h**

5.274 MHAMultiSrc::channels_t Class Reference

Inherits vector< MHAMultiSrc::channel_t >.

Public Member Functions

- **channels_t** (const std::vector< std::string > &src, int in_channels)
  Separate a list of input sources into a parsable channel list.

5.274.1 Constructor & Destructor Documentation

5.274.1.1 channels_t()

```
MHAMultiSrc::channels_t::channels_t (
    const std::vector< std::string > & route,
    int in_channels )
```

Separate a list of input sources into a parsable channel list.

The number of input channels if verified, a list of **MHAMultiSrc::channel_t** (p. 997) is filled.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

route	vector of source channel ids
in_channels	number of channels in the processed input signal

The documentation for this class was generated from the following files:

- **mha_multisrc.h**
- **mha_multisrc.cpp**

5.275 MHAMultiSrc::spectrum_t Class Reference

Inheritance diagram for MHAMultiSrc::spectrum_t:

Public Member Functions

- **spectrum_t** (**algo_comm_t** iac, std::string **name**, unsigned int frames, unsigned int **channels**)
- **mha_spec_t** ∗ **update** (**mha_spec_t** ∗s)

  Update data of spectrum to hold actual input data.

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.275.1 Constructor & Destructor Documentation

5.275.1.1 spectrum_t()

```
MHAMultiSrc::spectrum_t::spectrum_t (
    algo_comm_t iac,
    std::string name,
    unsigned int frames,
    unsigned int channels )
```

5.275.2 Member Function Documentation

5.275.2.1 update()

```
mha_spec_t * MHAMultiSrc::spectrum_t::update (
    mha_spec_t * s )
```

Update data of spectrum to hold actual input data.

Parameters

s	Input signal chunk

Returns

Return pointer to spectrum structure

The documentation for this class was generated from the following files:

- **mha_multisrc.h**
- **mha_multisrc.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.276 MHAMultiSrc::waveform_t Class Reference

Inheritance diagram for MHAMultiSrc::waveform_t:

Public Member Functions

- **waveform_t** (**algo_comm_t** iac, std::string **name**, unsigned int frames, unsigned int **channels**)
- **mha_wave_t** ∗ **update** (**mha_wave_t** ∗s)

  Update data of waveform to hold actual input data.

Additional Inherited Members

5.276.1 Constructor & Destructor Documentation

5.276.1.1 waveform_t()

```
MHAMultiSrc::waveform_t::waveform_t (
    algo_comm_t iac,
    std::string name,
    unsigned int frames,
    unsigned int channels )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.276.2 Member Function Documentation

5.276.2.1 update() `mha_wave_t * MHAMultiSrc::waveform_t::update ( mha_wave_t * s )`

Update data of waveform to hold actual input data.

Parameters

s	Input signal chunk

Returns

Return pointer to waveform structure

The documentation for this class was generated from the following files:

- **mha_multisrc.h**
- **mha_multisrc.cpp**

5.277 MHAOvlFilter::band_descriptor_t Class Reference

Public Attributes

- **mha_real_t cf_l**
- **mha_real_t ef_l**
- **mha_real_t cf**
- **mha_real_t ef_h**
- **mha_real_t cf_h**
- bool **low_side_flat**
- bool **high_side_flat**

5.277.1 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.277.1.1 cf_l `mha_real_t MHAOvlFilter::band_descriptor_t::cf_l`

5.277.1.2 ef_l `mha_real_t MHAOvlFilter::band_descriptor_t::ef_l`

5.277.1.3 cf `mha_real_t MHAOvlFilter::band_descriptor_t::cf`

5.277.1.4 ef_h `mha_real_t MHAOvlFilter::band_descriptor_t::ef_h`

5.277.1.5 cf_h `mha_real_t MHAOvlFilter::band_descriptor_t::cf_h`

5.277.1.6 low_side_flat `bool MHAOvlFilter::band_descriptor_t::low_side_flat`

5.277.1.7 high_side_flat `bool MHAOvlFilter::band_descriptor_t::high_side_flat`

The documentation for this class was generated from the following file:

- **mha_fftfb.hh**

© 2005-2021 HörTech gGmbH, Oldenburg

5.278 MHAOvlFilter::barkscale::bark2hz_t Class Reference

Inheritance diagram for MHAOvlFilter::barkscale::bark2hz_t:

Public Member Functions

- **bark2hz_t** ()
- ∼**bark2hz_t** ()

Additional Inherited Members

5.278.1 Constructor & Destructor Documentation

5.278.1.1 bark2hz_t() `MHAOvlFilter::barkscale::bark2hz_t::bark2hz_t ( )`

5.278.1.2 ∼bark2hz_t() `MHAOvlFilter::barkscale::bark2hz_t::~bark2hz_t ( )`

The documentation for this class was generated from the following file:

- **mha_fftfb.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.279 MHAOvlFilter::barkscale::hz2bark_t Class Reference

Inheritance diagram for MHAOvlFilter::barkscale::hz2bark_t:

Public Member Functions

- **hz2bark_t** ()
- ∼**hz2bark_t** ()

Additional Inherited Members

5.279.1 Constructor & Destructor Documentation

5.279.1.1 hz2bark_t() `MHAOvlFilter::barkscale::hz2bark_t::hz2bark_t ( )`

5.279.1.2 ∼hz2bark_t() `MHAOvlFilter::barkscale::hz2bark_t::∼hz2bark_t ( )`

The documentation for this class was generated from the following file:

- **mha_fftfb.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.280 MHAOvlFilter::fftfb_ac_info_t Class Reference

Public Member Functions

- **fftfb_ac_info_t** (const **MHAOvlFilter::fftfb_t** &fb, **algo_comm_t** ac, const std::string &prefix)
- void **insert** ()

Private Attributes

- **MHA_AC::waveform_t cfv**
  vector of nominal center frequencies / Hz
- **MHA_AC::waveform_t efv**
  vector of edge frequencies / Hz
- **MHA_AC::waveform_t bwv**
  vector of band-weigths (sum of squared fft-bin-weigths)/num_frames
- **MHA_AC::waveform_t cLTASS**
  vector of LTASS correction

5.280.1 Constructor & Destructor Documentation

5.280.1.1 fftfb_ac_info_t()

```
MHAOvlFilter::fftfb_ac_info_t::fftfb_ac_info_t (
    const MHAOvlFilter::fftfb_t & fb,
    algo_comm_t ac,
    const std::string & prefix )
```

5.280.2 Member Function Documentation

5.280.2.1 insert()

```
void MHAOvlFilter::fftfb_ac_info_t::insert ( )
```

5.280.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.280.3.1 cfv `MHA_AC::waveform_t MHAOvlFilter::fftfb_ac_info_t::cfv [private]`

vector of nominal center frequencies / Hz

5.280.3.2 efv `MHA_AC::waveform_t MHAOvlFilter::fftfb_ac_info_t::efv [private]`

vector of edge frequencies / Hz

5.280.3.3 bwv `MHA_AC::waveform_t MHAOvlFilter::fftfb_ac_info_t::bwv [private]`

vector of band-weigths (sum of squared fft-bin-weigths)/num_frames

5.280.3.4 cLTASS `MHA_AC::waveform_t MHAOvlFilter::fftfb_ac_info_t::cLTASS [private]`

vector of LTASS correction

The documentation for this class was generated from the following files:

- **mha_fftfb.hh**
- **mha_fftfb.cpp**

5.281 MHAOvlFilter::fftfb_t Class Reference

FFT based overlapping filter bank.

Inheritance diagram for MHAOvlFilter::fftfb_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **fftfb_t** (**MHAOvlFilter::fftfb_vars_t** &par, unsigned int nfft, **mha_real_t** fs)
  Constructor for a FFT-based overlapping filter bank.
- ∼**fftfb_t** ()
- void **apply_gains** (**mha_spec_t** ∗s_out, const **mha_spec_t** ∗s_in, const **mha_wave↩_t** ∗gains)
- void **get_fbpower** (**mha_wave_t** ∗fbpow, const **mha_spec_t** ∗s_in)
- void **get_fbpower_db** (**mha_wave_t** ∗fbpow, const **mha_spec_t** ∗s_in)
- std::vector< **mha_real_t** > **get_ltass_gain_db** () const
- unsigned int **bin1** (unsigned int band) const
  Return index of first non-zero filter shape window.
- unsigned int **bin2** (unsigned int band) const
  Return index of first zero filter shape window above center frequency.
- unsigned int **get_fftlen** () const
  Return fft length.
- **mha_real_t** **w** (unsigned int k, unsigned int b) const
  Return filter shape window at index k in band b.

Private Attributes

- unsigned int ∗ **vbin1**
- unsigned int ∗ **vbin2**
- **mha_real_t**(∗ **shape**)(**mha_real_t**)
- unsigned int **fftlen**
- **mha_real_t** **samplingrate**

Additional Inherited Members

5.281.1 Detailed Description

FFT based overlapping filter bank.

5.281.2 Constructor & Destructor Documentation

5.281.2.1 fftfb_t()

```
MHAOvlFilter::fftfb_t::fftfb_t (
    MHAOvlFilter::fftfb_vars_t & par,
    unsigned int nfft,
    mha_real_t fs )
```

Constructor for a FFT-based overlapping filter bank.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

par	Parameters for the FFT filterbank that can not be deduced from the signal dimensions are taken from this set of configuration variables.
nfft	FFT length
fs	Sampling rate / Hz

5.281.2.2 ~fftfb_t() `MHAOvlFilter::fftfb_t::~fftfb_t ( )`

5.281.3 Member Function Documentation

5.281.3.1 apply_gains()

```
void MHAOvlFilter::fftfb_t::apply_gains (
    mha_spec_t * s_out,
    const mha_spec_t * s_in,
    const mha_wave_t * gains )
```

5.281.3.2 get_fbpower()

```
void MHAOvlFilter::fftfb_t::get_fbpower (
    mha_wave_t * fbpow,
    const mha_spec_t * s_in )
```

5.281.3.3 get_fbpower_db()

```
void MHAOvlFilter::fftfb_t::get_fbpower_db (
    mha_wave_t * fbpow,
    const mha_spec_t * s_in )
```

5.281.3.4 get_ltass_gain_db()

```
std::vector< float > MHAOvlFilter::fftfb_t::get_↩
ltass_gain_db ( ) const
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.281.3.5 bin1()

```
unsigned int MHAOvlFilter::fftfb_t::bin1 (
    unsigned int band ) const [inline]
```

Return index of first non-zero filter shape window.

5.281.3.6 bin2()

```
unsigned int MHAOvlFilter::fftfb_t::bin2 (
    unsigned int band ) const [inline]
```

Return index of first zero filter shape window above center frequency.

5.281.3.7 get_fftlen()

```
unsigned int MHAOvlFilter::fftfb_t::get_fftlen ( ) const [inline]
```

Return fft length.

5.281.3.8 w()

```
mha_real_t MHAOvlFilter::fftfb_t::w (
    unsigned int k,
    unsigned int b ) const [inline]
```

Return filter shape window at index k in band b.

Parameters

k	Frequency index
b	Band index

5.281.4 Member Data Documentation

5.281.4.1 vbin1

```
unsigned int* MHAOvlFilter::fftfb_t::vbin1 [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.281.4.2 vbin2 `unsigned int* MHAOvlFilter::fftfb_t::vbin2 [private]`

5.281.4.3 shape `mha_real_t(* MHAOvlFilter::fftfb_t::shape) ( mha_real_t) [private]`

5.281.4.4 fftlen `unsigned int MHAOvlFilter::fftfb_t::fftlen [private]`

5.281.4.5 samplingrate `mha_real_t MHAOvlFilter::fftfb_t::samplingrate [private]`

The documentation for this class was generated from the following files:

- **mha_fftfb.hh**
- **mha_fftfb.cpp**

5.282 MHAOvlFilter::fftfb_vars_t Class Reference

Set of configuration variables for FFT-based overlapping filters.

Inheritance diagram for MHAOvlFilter::fftfb_vars_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **fftfb_vars_t** (**MHAParser::parser_t** &p)

  construct a set of openMHA configuration language variables suitable for configuring the FFT-based overlapping filterbank.

Public Attributes

- **scale_var_t fscale**

  Frequency scale type (lin/bark/log/erb).
- **scale_var_t ovltype**

  Filter shape (rect/lin/hann).
- **MHAParser::float_t plateau**

  relative plateau width.
- **MHAParser::kw_t ftype**

  Flag to decide wether edge or center frequencies are used.
- **fscale_t f**

  Frequency.
- **MHAParser::bool_t normalize**

  Normalize sum of channels.
- **MHAParser::bool_t fail_on_nonmonotonic**

  Fail if frequency entries are non-monotonic (otherwise sort)
- **MHAParser::bool_t fail_on_unique_bins**

  Fail if center frequencies share the same FFT bin.
- **MHAParser::bool_t flag_allow_empty_bands**

  Allow that frequency bands contain only zeros.
- **MHAParser::vfloat_mon_t cf**

  Final center frequencies in Hz.
- **MHAParser::vfloat_mon_t ef**

  Final edge frequencies in Hz.
- **MHAParser::vfloat_mon_t cLTASS**

  Bandwidth correction for LTASS noise (level of 0 dB RMS LTASS noise)
- **MHAParser::mfloat_mon_t shapes**

5.282.1 Detailed Description

Set of configuration variables for FFT-based overlapping filters.

This class enables easy configuration of the FFT-based overlapping filterbank. An instance of **fftfb_vars_t** (p. 1010) creates openMHA configuration language variables needed for configuring the filterbank, and inserts these variables in the openMHA configuration tree.

This way, the variables are visible to the user and can be configured using the openMHA configuration language.

© 2005-2021 HörTech gGmbH, Oldenburg

5.282.2 Constructor & Destructor Documentation

5.282.2.1 fftfb_vars_t() `MHAOvlFilter::fftfb_vars_t::fftfb_vars_t ( MHAParser::parser_t & p )`

construct a set of openMHA configuration language variables suitable for configuring the FFT-based overlapping filterbank.

Parameters

p	The node of the configuration tree where the variables created by this instance are inserted.

5.282.3 Member Data Documentation

5.282.3.1 fscale `scale_var_t MHAOvlFilter::fftfb_vars_t::fscale`

Frequency scale type (lin/bark/log/erb).

5.282.3.2 ovltype `scale_var_t MHAOvlFilter::fftfb_vars_t::ovltype`

Filter shape (rect/lin/hann).

5.282.3.3 plateau `MHAParser::float_t MHAOvlFilter::fftfb_vars_t::plateau`

relative plateau width.

© 2005-2021 HörTech gGmbH, Oldenburg

5.282.3.4 ftype `MHAParser::kw_t MHAOvlFilter::fftfb_vars_t::ftype`

Flag to decide wether edge or center frequencies are used.

5.282.3.5 f `fscale_t MHAOvlFilter::fftfb_vars_t::f`

Frequency.

5.282.3.6 normalize `MHAParser::bool_t MHAOvlFilter::fftfb_vars_t::normalize`

Normalize sum of channels.

5.282.3.7 fail_on_nonmonotonic `MHAParser::bool_t MHAOvlFilter::fftfb_vars_t↩ ::fail_on_nonmonotonic`

Fail if frequency entries are non-monotonic (otherwise sort)

5.282.3.8 fail_on_unique_bins `MHAParser::bool_t MHAOvlFilter::fftfb_vars_t::fail↩ _on_unique_bins`

Fail if center frequencies share the same FFT bin.

5.282.3.9 flag_allow_empty_bands `MHAParser::bool_t MHAOvlFilter::fftfb_vars_t↩ ::flag_allow_empty_bands`

Allow that frequency bands contain only zeros.

© 2005-2021 HörTech gGmbH, Oldenburg

5.282.3.10 cf `MHAParser::vfloat_mon_t` `MHAOvlFilter::fftfb_vars_t::cf`

Final center frequencies in Hz.

5.282.3.11 ef `MHAParser::vfloat_mon_t` `MHAOvlFilter::fftfb_vars_t::ef`

Final edge frequencies in Hz.

5.282.3.12 cLTASS `MHAParser::vfloat_mon_t` `MHAOvlFilter::fftfb_vars_t::cLTASS`

Bandwidth correction for LTASS noise (level of 0 dB RMS LTASS noise)

5.282.3.13 shapes `MHAParser::mfloat_mon_t` `MHAOvlFilter::fftfb_vars_t::shapes`

The documentation for this class was generated from the following files:

- **mha_fftfb.hh**
- **mha_fftfb.cpp**

5.283 MHAOvlFilter::fscale_bw_t Class Reference

Inheritance diagram for MHAOvlFilter::fscale_bw_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **fscale_bw_t** (**MHAParser::parser_t** &parent)
- std::vector< **mha_real_t** > **get_bw_hz** () const

Protected Attributes

- **MHAParser::vfloat_t bw**
- **MHAParser::vfloat_mon_t bw_hz**

Private Member Functions

- void **update_hz** ()

Private Attributes

- **MHAEvents::connector_t**< **fscale_bw_t** > **updater**

Additional Inherited Members

5.283.1 Constructor & Destructor Documentation

5.283.1.1 fscale_bw_t() `MHAOvlFilter::fscale_bw_t::fscale_bw_t ( MHAParser::parser_t & parent )`

5.283.2 Member Function Documentation

5.283.2.1 get_bw_hz() `std::vector< mha_real_t > MHAOvlFilter::fscale_bw_t::get_bw_hz ( ) const`

5.283.2.2 update_hz() `void MHAOvlFilter::fscale_bw_t::update_hz ( ) [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.283.3 Member Data Documentation

5.283.3.1 bw `MHAParser::vfloat_t MHAOvlFilter::fscale_bw_t::bw [protected]`

5.283.3.2 bw_hz `MHAParser::vfloat_mon_t MHAOvlFilter::fscale_bw_t::bw_hz [protected]`

5.283.3.3 updater `MHAEvents::connector_t< fscale_bw_t> MHAOvlFilter::fscale_bw_t::updater [private]`

The documentation for this class was generated from the following files:

- **mha_fftfb.hh**
- **mha_fftfb.cpp**

5.284 MHAOvlFilter::fscale_t Class Reference

Inheritance diagram for MHAOvlFilter::fscale_t:

Public Member Functions

- **fscale_t** (**MHAParser::parser_t** &parent)
- std::vector< **mha_real_t** > **get_f_hz** () const

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **scale_var_t unit**
- **MHAParser::vfloat_t f**
- **MHAParser::vfloat_mon_t f_hz**

Private Member Functions

- void **update_hz** ()

Private Attributes

- **MHAEvents::connector_t< fscale_t > updater**

5.284.1 Constructor & Destructor Documentation

5.284.1.1 fscale_t() `MHAOvlFilter::fscale_t::fscale_t ( MHAParser::parser_t & parent )`

5.284.2 Member Function Documentation

5.284.2.1 get_f_hz() `std::vector< mha_real_t > MHAOvlFilter::fscale_t::get_f_hz ( ) const`

5.284.2.2 update_hz() `void MHAOvlFilter::fscale_t::update_hz ( ) [private]`

5.284.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.284.3.1 unit `scale_var_t MHAOvlFilter::fscale_t::unit`

5.284.3.2 f `MHAParser::vfloat_t MHAOvlFilter::fscale_t::f`

5.284.3.3 f_hz `MHAParser::vfloat_mon_t MHAOvlFilter::fscale_t::f_hz`

5.284.3.4 updater `MHAEvents::connector_t< fscale_t> MHAOvlFilter::fscale_t↩ ::updater [private]`

The documentation for this class was generated from the following files:

- **mha_fftfb.hh**
- **mha_fftfb.cpp**

5.285 MHAOvlFilter::fspacing_t Class Reference

Class for frequency spacing, used by filterbank shape generator class.

Inheritance diagram for MHAOvlFilter::fspacing_t:

Public Member Functions

- **fspacing_t** (const **MHAOvlFilter::fftfb_vars_t** &par, unsigned int nfft, **mha_real_t** fs)
- std::vector< unsigned int > **get_cf_fftbin** () const
- std::vector< **mha_real_t** > **get_cf_hz** () const
- std::vector< **mha_real_t** > **get_ef_hz** () const
- unsigned int **nbands** () const

  Return number of bands in filter bank.

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Member Functions

- void **fail_on_nonmonotonic_cf** ()
- void **fail_on_unique_fftbins** ()

Protected Attributes

- std::vector< **MHAOvlFilter::band_descriptor_t** > **bands**
- **mha_real_t**(* **symmetry_scale**)(**mha_real_t**)

Private Member Functions

- void **ef2bands** (std::vector< **mha_real_t** > vef)
- void **cf2bands** (std::vector< **mha_real_t** > vcf)
- void **equidist2bands** (std::vector< **mha_real_t** > vcf)

Private Attributes

- unsigned int **nfft_**
- **mha_real_t fs_**

5.285.1 Detailed Description

Class for frequency spacing, used by filterbank shape generator class.

5.285.2 Constructor & Destructor Documentation

5.285.2.1 fspacing_t()

```
MHAOvlFilter::fspacing_t::fspacing_t (
    const MHAOvlFilter::fftfb_vars_t & par,
    unsigned int nfft,
    mha_real_t fs )
```

5.285.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.285.3.1 get_cf_fftbin() `std::vector< unsigned int > MHAOvlFilter::fspacing_t↩ ::get_cf_fftbin ( ) const`

5.285.3.2 get_cf_hz() `std::vector< mha_real_t > MHAOvlFilter::fspacing_t::get_cf↩ _hz ( ) const`

5.285.3.3 get_ef_hz() `std::vector< mha_real_t > MHAOvlFilter::fspacing_t::get_ef↩ _hz ( ) const`

5.285.3.4 nbands() `unsigned int MHAOvlFilter::fspacing_t::nbands ( ) const [inline]`

Return number of bands in filter bank.

5.285.3.5 fail_on_nonmonotonic_cf() `void MHAOvlFilter::fspacing_t::fail_on_↩ nonmonotonic_cf ( ) [protected]`

5.285.3.6 fail_on_unique_fftbins() `void MHAOvlFilter::fspacing_t::fail_on_unique_↩ fftbins ( ) [protected]`

5.285.3.7 ef2bands() `void MHAOvlFilter::fspacing_t::ef2bands ( std::vector< mha_real_t > vef ) [private]`

5.285.3.8 cf2bands() `void MHAOvlFilter::fspacing_t::cf2bands ( std::vector< mha_real_t > vcf ) [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.285.3.9 equidist2bands()

`void MHAOvlFilter::fspacing_t::equidist2bands ( std::vector< mha_real_t > vcf ) [private]`

5.285.4 Member Data Documentation

5.285.4.1 bands

`std::vector< MHAOvlFilter::band_descriptor_t> MHAOvlFilter::fspacing↩_t::bands [protected]`

5.285.4.2 symmetry_scale

`mha_real_t(* MHAOvlFilter::fspacing_t::symmetry_scale) ( mha_real_t) [protected]`

5.285.4.3 nfft_

`unsigned int MHAOvlFilter::fspacing_t::nfft_ [private]`

5.285.4.4 fs_

`mha_real_t MHAOvlFilter::fspacing_t::fs_ [private]`

The documentation for this class was generated from the following files:

- **mha_fftfb.hh**
- **mha_fftfb.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.286 MHAOvlFilter::overlap_save_filterbank_analytic_t Class Reference

Inheritance diagram for MHAOvlFilter::overlap_save_filterbank_analytic_t:

Public Member Functions

- **overlap_save_filterbank_analytic_t** (**MHAOvlFilter::overlap_save_filterbank_t**↩ **::vars_t** &fbpar, **mhaconfig_t** channelconfig_in)
- void **filter_analytic** (const **mha_wave_t** ∗sIn, **mha_wave_t** ∗∗fltRe, **mha_wave_t** ∗∗fltIm)

Private Attributes

- **MHAFilter::fftfilterbank_t imagfb**

Additional Inherited Members

5.286.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.286.1.1 overlap_save_filterbank_analytic_t()

```
MHAOvlFilter::overlap_save_filterbank↩
_analytic_t::overlap_save_filterbank_analytic_t (
            MHAOvlFilter::overlap_save_filterbank_t::vars_t & fbpar,
            mhaconfig_t channelconfig_in )
```

5.286.2 Member Function Documentation

5.286.2.1 filter_analytic()

```
void MHAOvlFilter::overlap_save_filterbank_analytic_t↩
::filter_analytic (
            const mha_wave_t * sIn,
            mha_wave_t ** fltRe,
            mha_wave_t ** fltIm )
```

5.286.3 Member Data Documentation

5.286.3.1 imagfb

```
MHAFilter::fftfilterbank_t MHAOvlFilter::overlap_save_filterbank↩
_analytic_t::imagfb [private]
```

The documentation for this class was generated from the following files:

- **mha_fftfb.hh**
- **mha_fftfb.cpp**

5.287 MHAOvlFilter::overlap_save_filterbank_t Class Reference

A time-domain minimal phase filter bank with frequency shapes from **MHAOvlFilter::fftfb_t** (p. 1006).

© 2005-2021 HörTech gGmbH, Oldenburg

Inheritance diagram for MHAOvlFilter::overlap_save_filterbank_t:

Classes

- class **vars_t**

Public Member Functions

- **overlap_save_filterbank_t** (**MHAOvlFilter::overlap_save_filterbank_t::vars_t** &fb-par, **mhaconfig_t** channelconfig_in)
- **mhaconfig_t** **get_channelconfig** () const

Private Attributes

- **mhaconfig_t** **channelconfig_out_**

Additional Inherited Members

5.287.1 Detailed Description

A time-domain minimal phase filter bank with frequency shapes from **MHAOvlFilter::fftfb_t** (p. 1006).

© 2005-2021 HörTech gGmbH, Oldenburg

5.287.2 Constructor & Destructor Documentation

5.287.2.1 overlap_save_filterbank_t()

```
MHAOvlFilter::overlap_save_filterbank_t↩
::overlap_save_filterbank_t (
            MHAOvlFilter::overlap_save_filterbank_t::vars_t & fbpar,
            mhaconfig_t channelconfig_in )
```

5.287.3 Member Function Documentation

5.287.3.1 get_channelconfig()

```
mhaconfig_t MHAOvlFilter::overlap_save_filterbank_↩
t::get_channelconfig ( ) const [inline]
```

5.287.4 Member Data Documentation

5.287.4.1 channelconfig_out_

```
mhaconfig_t MHAOvlFilter::overlap_save_filterbank_↩
t::channelconfig_out_ [private]
```

The documentation for this class was generated from the following files:

- **mha_fftfb.hh**
- **mha_fftfb.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.288 MHAOvlFilter::overlap_save_filterbank_t::vars_t Class Reference

Inheritance diagram for MHAOvlFilter::overlap_save_filterbank_t::vars_t:

Public Member Functions

- **vars_t** (**MHAParser::parser_t** &p)

Public Attributes

- **MHAParser::int_t fftlen**
- **MHAParser::kw_t phasemodel**
- **MHAParser::window_t irswnd**

5.288.1 Constructor & Destructor Documentation

5.288.1.1 vars_t()

`MHAOvlFilter::overlap_save_filterbank_t::vars_t::vars_t ( MHAParser::parser_t & p )`

© 2005-2021 HörTech gGmbH, Oldenburg

5.288.2 Member Data Documentation

5.288.2.1 fftlen

`MHAParser::int_t MHAOvlFilter::overlap_save_filterbank_t::vars_t↩ ::fftlen`

5.288.2.2 phasemodel

`MHAParser::kw_t MHAOvlFilter::overlap_save_filterbank_t↩ ::vars_t::phasemodel`

5.288.2.3 irswnd

`MHAParser::window_t MHAOvlFilter::overlap_save_filterbank_t↩ ::vars_t::irswnd`

The documentation for this class was generated from the following files:

- **mha_fftfb.hh**
- **mha_fftfb.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.289 MHAOvlFilter::scale_var_t Class Reference

Inheritance diagram for MHAOvlFilter::scale_var_t:

Public Member Functions

- **scale_var_t** (const std::string & **help**)
- void **add_fun** (const std::string &name, **scale_fun_t** ∗fun)
- std::string **get_name** () const
- **scale_fun_t** ∗ **get_fun** () const
- **mha_real_t** **hz2unit** (**mha_real_t** x) const
- **mha_real_t** **unit2hz** (**mha_real_t** x) const

Private Attributes

- std::vector< std::string > **names**
- std::vector< **scale_fun_t** ∗ > **funs**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.289.1 Constructor & Destructor Documentation

5.289.1.1 scale_var_t()

```
MHAOvlFilter::scale_var_t::scale_var_t (
    const std::string & help )
```

5.289.2 Member Function Documentation

5.289.2.1 add_fun()

```
void MHAOvlFilter::scale_var_t::add_fun (
    const std::string & name,
    scale_fun_t * fun )
```

5.289.2.2 get_name()

```
std::string MHAOvlFilter::scale_var_t::get_name ( ) const [inline]
```

5.289.2.3 get_fun()

```
scale_fun_t* MHAOvlFilter::scale_var_t::get_fun ( ) const [inline]
```

5.289.2.4 hz2unit()

```
mha_real_t MHAOvlFilter::scale_var_t::hz2unit (
    mha_real_t x ) const
```

5.289.2.5 unit2hz()

```
mha_real_t MHAOvlFilter::scale_var_t::unit2hz (
    mha_real_t x ) const
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.289.3 Member Data Documentation

5.289.3.1 names `std::vector<std::string> MHAOvlFilter::scale_var_t::names [private]`

5.289.3.2 funs `std::vector< scale_fun_t*> MHAOvlFilter::scale_var_t::funs [private]`

The documentation for this class was generated from the following files:

- **mha_fftfb.hh**
- **mha_fftfb.cpp**

5.290 MHAParser::base_t Class Reference

Base class for all parser items.

© 2005-2021 HörTech gGmbH, Oldenburg

Inheritance diagram for MHAParser::base_t:

Classes

- class **replace_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **base_t** (const std::string &)

  Constructor for base class of all parser nodes.
- **base_t** (const **base_t** &)

  *Copy constructor for **base_t** (p. 1030).*
- **base_t** & **operator=** (const **base_t** &)=default
- **base_t** (**base_t** &&)=delete
- **base_t** & **operator=** (**base_t** &&)=delete
- virtual ~**base_t** ()
- virtual std::string **parse** (const std::string &)

  Causes this node to process a command in the openMHA configuration language.
- virtual void **parse** (const char *, char *, unsigned int)

  This function parses a command and writes the parsing result into a C character array.
- virtual void **parse** (const std::vector< std::string > &, std::vector< std::string > &)
- virtual std::string **op_subparse** (**expression_t** &)
- virtual std::string **op_setval** (**expression_t** &)
- virtual std::string **op_query** (**expression_t** &)
- virtual std::string **query_dump** (const std::string &)
- virtual std::string **query_entries** (const std::string &)
- virtual std::string **query_perm** (const std::string &)
- virtual std::string **query_range** (const std::string &)
- virtual std::string **query_type** (const std::string &)
- virtual std::string **query_val** (const std::string &)
- virtual std::string **query_readfile** (const std::string &)
- virtual std::string **query_savefile** (const std::string &)
- virtual std::string **query_savefile_compact** (const std::string &)
- virtual std::string **query_savemons** (const std::string &)
- virtual std::string **query_listids** (const std::string &)
- std::string **query_version** (const std::string &)
- std::string **query_id** (const std::string &)
- std::string **query_subst** (const std::string &)
- std::string **query_addsubst** (const std::string &)
- std::string **query_help** (const std::string &)
- std::string **query_cmds** (const std::string &)
- void **set_node_id** (const std::string &)

  Set the identification string of this parser node.
- void **set_help** (const std::string &)

  Set the help comment of a variable or parser.
- void **add_parent_on_insert** (**parser_t** *, std::string)
- void **rm_parent_on_remove** (**parser_t** *)
- const std::string & **fullname** () const

  Return the full dot-separated path name of this parser node in the openMHA configuration tree.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **MHAEvents::emitter_t writeaccess**
  Event emitted on write access.
- **MHAEvents::emitter_t valuechanged**
  Event emitted if the value has changed.
- **MHAEvents::emitter_t readaccess**
  Event emitted on read access.
- **MHAEvents::emitter_t prereadaccess**
  Event emitted on read access, before the data field is accessed.

Protected Member Functions

- void **activate_query** (const std::string &, **query_t**)
- void **notify** ()

Protected Attributes

- **query_map_t queries**
- bool **data_is_initialized**

Private Types

- typedef std::vector< **replace_t** > **repl_list_t**

Private Member Functions

- void **add_replace_pair** (const std::string &, const std::string &)
- std::string **oplist** ()

Private Attributes

- std::string **help**
- std::string **id_str**
- **opact_map_t operators**
- **repl_list_t repl_list**
- bool **nested_lock**
- **parser_t** * **parent**
- std::string **thefullname**

© 2005-2021 HörTech gGmbH, Oldenburg

5.290.1 Detailed Description

Base class for all parser items.

The key method of the parser base class is the std::string **parse(const std::string&)** (p. 1035) method. Parser proxy derivatives which overwrite any of the other **parse()** (p. 1035) methods to be the key method must make sure that the original **parse()** (p. 1035) method utilizes the new key method.

5.290.2 Member Typedef Documentation

5.290.2.1 repl_list_t `typedef std::vector< replace_t> MHAParser::base_t::repl_list_t [private]`

5.290.3 Constructor & Destructor Documentation

5.290.3.1 base_t() [1/3] `MHAParser::base_t::base_t ( const std::string & h )`

Constructor for base class of all parser nodes.

Parameters

h	Help text describing this parser node. This help text is accessible to the configuration language through the "?help" query command.

5.290.3.2 base_t() [2/3] `MHAParser::base_t::base_t ( const base_t & src )`

Copy constructor for **base_t** (p. 1030).

Copies help text and id string, but does not insert the new node into the parser tree structure.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

src	Source parser

Deprecated Copying parser nodes makes little sense, avoid wherever possible

5.290.3.3 base_t() [3/3]

`MHAParser::base_t::base_t ( base_t && ) [delete]`

5.290.3.4 ~base_t()

`MHAParser::base_t::~base_t ( ) [virtual]`

5.290.4 Member Function Documentation

5.290.4.1 operator=() [1/2]

`base_t& MHAParser::base_t::operator= ( const base_t & ) [default]`

5.290.4.2 operator=() [2/2]

`base_t& MHAParser::base_t::operator= ( base_t && ) [delete]`

5.290.4.3 parse() [1/3]

`std::string MHAParser::base_t::parse ( const std::string & cs ) [virtual]`

Causes this node to process a command in the openMHA configuration language.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

cs	The command to parse

Returns

The response to the command, if successful

Exceptions

MHA_Error *(p. 764)*	If the command cannot be executed successfully. The reason for failure is given in the message string of the exception.

Reimplemented in **plug_wrapperl** (p. 1353), **PluginLoader::mhapluginloader_t** (p. 1371), **plug_wrapper** (p. 1351), **io_wrapper** (p. 644), and **altplugs_t** (p. 302).

5.290.4.4 parse() [2/3]

```
void MHAParser::base_t::parse (
    const char * cmd,
    char * retv,
    unsigned int len ) [virtual]
```

This function parses a command and writes the parsing result into a C character array.

This base class implementation delegates to **parse(const std::string &)** (p. 1035).

Parameters

cmd	Command to be parsed
retv	Buffer for the result
len	Length of buffer

Reimplemented in **altplugs_t** (p. 302).

5.290.4.5 parse() [3/3]

```
void MHAParser::base_t::parse (
    const std::vector< std::string > & cs,
    std::vector< std::string > & retv ) [virtual]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.290.4.6 op_subparse()

```
std::string MHAParser::base_t::op_subparse (
    expression_t & ) [virtual]
```

Reimplemented in **MHAParser::c_ifc_parser_t** (p. 1051), and **MHAParser::parser_t** (p. 1105).

5.290.4.7 op_setval()

```
std::string MHAParser::base_t::op_setval (
    expression_t & ) [virtual]
```

Reimplemented in **MHAParser::c_ifc_parser_t** (p. 1051), **MHAParser::mcomplex_t** (p. 1082), **MHAParser::mfloat_t** (p. 1087), **MHAParser::mint_t** (p. 1099), **MHAParser::vcomplex_t** (p. 1123), **MHAParser::vfloat_t** (p. 1129), **MHAParser::vint_t** (p. 1133), **MHAParser::complex_t** (p. 1058), **MHAParser::float_t** (p. 1065), **MHAParser::int_t** (p. 1070), **MHAParser::bool_t** (p. 1049), **MHAParser::vstring_t** (p. 1137), **MHAParser::string_t** (p. 1117), **MHAParser::kw_t** (p. 1077), **MHAParser::variable_t** (p. 1119), and **MHAParser::parser_t** (p. 1106).

5.290.4.8 op_query()

```
std::string MHAParser::base_t::op_query (
    expression_t & ) [virtual]
```

Reimplemented in **MHAParser::c_ifc_parser_t** (p. 1051), **MHAParser::monitor_t** (p. 1101), and **MHAParser::parser_t** (p. 1106).

5.290.4.9 query_dump()

```
std::string MHAParser::base_t::query_dump (
    const std::string & ) [virtual]
```

Reimplemented in **MHAParser::monitor_t** (p. 1101), and **MHAParser::parser_t** (p. 1106).

5.290.4.10 query_entries()

```
std::string MHAParser::base_t::query_entries (
    const std::string & ) [virtual]
```

Reimplemented in **MHAParser::parser_t** (p. 1106).

© 2005-2021 HörTech gGmbH, Oldenburg

5.290.4.11 query_perm() `std::string MHAParser::base_t::query_perm ( const std::string & ) [virtual]`

Reimplemented in **MHAParser::variable_t** (p. 1119), and **MHAParser::monitor_t** (p. 1101).

5.290.4.12 query_range() `std::string MHAParser::base_t::query_range ( const std::string & ) [virtual]`

Reimplemented in **MHAParser::kw_t** (p. 1078), and **MHAParser::range_var_t** (p. 1110).

5.290.4.13 query_type() `std::string MHAParser::base_t::query_type ( const std::string & ) [virtual]`

Reimplemented in **MHAParser::mcomplex_mon_t** (p. 1080), **MHAParser::vcomplex_mon_t** (p. 1121), **MHAParser::complex_mon_t** (p. 1056), **MHAParser::float_mon_t** (p. 1063), **MHAParser::mfloat_mon_t** (p. 1084), **MHAParser::vfloat_mon_t** (p. 1126), **MHAParser::mint_mon_t** (p. 1096), **MHAParser::vint_mon_t** (p. 1131), **MHAParser::vstring_mon_t** (p. 1136), **MHAParser::string_mon_t** (p. 1114), **MHAParser::bool_mon_t** (p. 1046), **MHAParser::int_mon_t** (p. 1068), **MHAParser::mcomplex_t** (p. 1082), **MHAParser::mfloat_t** (p. 1087), **MHAParser::mint_t** (p. 1099), **MHAParser::vcomplex_t** (p. 1124), **MHAParser::vfloat_t** (p. 1129), **MHAParser::vint_t** (p. 1133), **MHAParser::complex_t** (p. 1058), **MHAParser::float_t** (p. 1065), **MHAParser::int_t** (p. 1071), **MHAParser::bool_t** (p. 1049), **MHAParser::vstring_t** (p. 1137), **MHAParser::string_t** (p. 1117), **MHAParser::kw_t** (p. 1078), and **MHAParser::parser_t** (p. 1106).

5.290.4.14 query_val() `std::string MHAParser::base_t::query_val ( const std::string & ) [virtual]`

Reimplemented in **MHAParser::mcomplex_mon_t** (p. 1080), **MHAParser::vcomplex_mon_t** (p. 1121), **MHAParser::complex_mon_t** (p. 1056), **MHAParser::float_mon_t** (p. 1062), **MHAParser::mfloat_mon_t** (p. 1084), **MHAParser::vfloat_mon_t** (p. 1126), **MHAParser::mint_mon_t** (p. 1096), **MHAParser::vint_mon_t** (p. 1131), **MHAParser::vstring_mon_t** (p. 1135), **MHAParser::string_mon_t** (p. 1114), **MHAParser::bool_mon_t** (p. 1046), **MHAParser::int_mon_t** (p. 1068), **MHAParser::mcomplex_t** (p. 1082), **MHAParser::mfloat_t** (p. 1087), **MHAParser::mint_t** (p. 1099), **MHAParser::vcomplex_t** (p. 1124), **MHAParser::vfloat_t** (p. 1129), **MHAParser::vint_t** (p. 1133), **MHAParser::complex_t** (p. 1058), **MHAParser::float_t** (p. 1066), **MHAParser::int_t** (p. 1071), **MHAParser::bool_t** (p. 1049), **MHAParser::vstring_t** (p. 1138), **MHAParser::string_t** (p. 1117), **MHAParser::kw_t** (p. 1078), and **MHAParser::parser_t** (p. 1107).

© 2005-2021 HörTech gGmbH, Oldenburg

5.290.4.15 query_readfile()

```
std::string MHAParser::base_t::query_readfile (
    const std::string & ) [virtual]
```

Reimplemented in **MHAParser::parser_t** (p. 1106).

5.290.4.16 query_savefile()

```
std::string MHAParser::base_t::query_savefile (
    const std::string & ) [virtual]
```

Reimplemented in **MHAParser::parser_t** (p. 1107).

5.290.4.17 query_savefile_compact()

```
std::string MHAParser::base_t::query_savefile↩
_compact (
    const std::string & ) [virtual]
```

Reimplemented in **MHAParser::parser_t** (p. 1107).

5.290.4.18 query_savemons()

```
std::string MHAParser::base_t::query_savemons (
    const std::string & ) [virtual]
```

Reimplemented in **MHAParser::parser_t** (p. 1107).

5.290.4.19 query_listids()

```
std::string MHAParser::base_t::query_listids (
    const std::string & ) [virtual]
```

Reimplemented in **MHAParser::parser_t** (p. 1107).

5.290.4.20 query_version()

```
std::string MHAParser::base_t::query_version (
    const std::string & )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.290.4.21 query_id() `std::string MHAParser::base_t::query_id ( const std::string & )`

5.290.4.22 query_subst() `std::string MHAParser::base_t::query_subst ( const std::string & )`

5.290.4.23 query_addsubst() `std::string MHAParser::base_t::query_addsubst ( const std::string & s )`

5.290.4.24 query_help() `std::string MHAParser::base_t::query_help ( const std::string & )`

5.290.4.25 query_cmds() `std::string MHAParser::base_t::query_cmds ( const std::string & )`

5.290.4.26 set_node_id() `void MHAParser::base_t::set_node_id ( const std::string & s )`

Set the identification string of this parser node.

The id can be queried from the configuration language using the ?id query command. Nodes can be found by id using the ?listid query command on a containing parser node.

Parameters

s	The new identification string.

5.290.4.27 set_help() `void MHAParser::base_t::set_help (`

© 2005-2021 HörTech gGmbH, Oldenburg

```
const std::string & s )
```

Set the help comment of a variable or parser.

Parameters

s	New help comment.

5.290.4.28 add_parent_on_insert()

```
void MHAParser::base_t::add_parent_on_insert (
 parser_t * p,
std::string n )
```

5.290.4.29 rm_parent_on_remove()

```
void MHAParser::base_t::rm_parent_on_remove (
 parser_t *  )
```

5.290.4.30 fullname()

```
const std::string & MHAParser::base_t::fullname ( ) const
```

Return the full dot-separated path name of this parser node in the openMHA configuration tree.

5.290.4.31 activate_query()

```
void MHAParser::base_t::activate_query (
const std::string & n,
 query_t a )  [protected]
```

5.290.4.32 notify()

```
void MHAParser::base_t::notify ( )  [protected]
```

5.290.4.33 add_replace_pair()

```
void MHAParser::base_t::add_replace_pair (
const std::string & a,
const std::string & b )  [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.290.4.34 oplist() `std::string MHAParser::base_t::oplist ( ) [private]`

5.290.5 Member Data Documentation

5.290.5.1 writeaccess `MHAEvents::emitter_t MHAParser::base_t::writeaccess`

Event emitted on write access.

To connect a callback that is invoked on write access to this parser variable, use MHAEvents↩ ::patchbay_t<receiver_t> method connect(&writeaccess,&receiver_t::callback) where callback is a method that expects no parameters and returns void.

5.290.5.2 valuechanged `MHAEvents::emitter_t MHAParser::base_t::valuechanged`

Event emitted if the value has changed.

To connect a callback that is invoked when write access to this parser variable actually changes its value, use MHAEvents::patchbay_t<receiver_t> method connect(&valuechanged,&receiver↩ _t::callback) where callback is a method that expects no parameters and returns void.

5.290.5.3 readaccess `MHAEvents::emitter_t MHAParser::base_t::readaccess`

Event emitted on read access.

To connect a callback that is invoked after the value of this variable has been read through the configuration interface, use MHAEvents::patchbay_t<receiver_t> method connect(&readaccess,&receiver_t::callback) where callback is a method that expects no parameters and returns void.

5.290.5.4 prereadaccess `MHAEvents::emitter_t MHAParser::base_t::prereadaccess`

Event emitted on read access, before the data field is accessed.

To connect a callback that is invoked when the value of this variable is about to be read through the configuration interface, so that the callback can influence the value that is reported, use MHAEvents::patchbay_t<receiver_t> method connect(&prereadaccess,&receiver_t::callback) where callback is a method that expects no parameters and returns void.

© 2005-2021 HörTech gGmbH, Oldenburg

5.290.5.5 queries `query_map_t MHAParser::base_t::queries [protected]`

5.290.5.6 data_is_initialized `bool MHAParser::base_t::data_is_initialized [protected]`

5.290.5.7 help `std::string MHAParser::base_t::help [private]`

5.290.5.8 id_str `std::string MHAParser::base_t::id_str [private]`

5.290.5.9 operators `opact_map_t MHAParser::base_t::operators [private]`

5.290.5.10 repl_list `repl_list_t MHAParser::base_t::repl_list [private]`

5.290.5.11 nested_lock `bool MHAParser::base_t::nested_lock [private]`

5.290.5.12 parent `parser_t* MHAParser::base_t::parent [private]`

5.290.5.13 thefullname `std::string MHAParser::base_t::thefullname [private]`

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.291 MHAParser::base_t::replace_t Class Reference

Public Member Functions

- **replace_t** (const std::string &, const std::string &)
- void **replace** (std::string &)
- const std::string & **get_a** () const
- const std::string & **get_b** () const

Private Attributes

- std::string **a**
- std::string **b**

5.291.1 Constructor & Destructor Documentation

5.291.1.1 replace_t()

```
MHAParser::base_t::replace_t::replace_t (
    const std::string & ia,
    const std::string & ib )
```

5.291.2 Member Function Documentation

5.291.2.1 replace()

```
void MHAParser::base_t::replace_t::replace (
    std::string & s )
```

5.291.2.2 get_a()

```
const std::string& MHAParser::base_t::replace_t::get_a ( ) const
[inline]
```

5.291.2.3 get_b()

```
const std::string& MHAParser::base_t::replace_t::get_b ( ) const
[inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.291.3 Member Data Documentation

5.291.3.1 a `std::string MHAParser::base_t::replace_t::a [private]`

5.291.3.2 b `std::string MHAParser::base_t::replace_t::b [private]`

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.292 MHAParser::bool_mon_t Class Reference

Monitor with string value.

Inheritance diagram for MHAParser::bool_mon_t:

Public Member Functions

- **bool_mon_t** (const std::string &hlp)
  Create a monitor variable for string values.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- bool **data**
  Data field.

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.292.1 Detailed Description

Monitor with string value.

5.292.2 Constructor & Destructor Documentation

5.292.2.1 bool_mon_t() `MHAParser::bool_mon_t::bool_mon_t ( const std::string & hlp )`

Create a monitor variable for string values.

Parameters

hlp	A help text describing this monitor variable.

5.292.3 Member Function Documentation

5.292.3.1 query_val() `std::string MHAParser::bool_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

© 2005-2021 HörTech gGmbH, Oldenburg

5.292.3.2 query_type() `std::string MHAParser::bool_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.292.4 Member Data Documentation

5.292.4.1 data `bool MHAParser::bool_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.293 MHAParser::bool_t Class Reference

Variable with a boolean value ("yes"/"no")

Inheritance diagram for MHAParser::bool_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **bool_t** (const std::string &help_text, const std::string &initial_value)
  Constructor for a configuration language variable for boolean values.

Public Attributes

- bool **data**
  Data field.

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

Additional Inherited Members

5.293.1 Detailed Description

Variable with a boolean value ("yes"/"no")

5.293.2 Constructor & Destructor Documentation

5.293.2.1 bool_t()

```
MHAParser::bool_t::bool_t (
    const std::string & help_text,
    const std::string & initial_value )
```

Constructor for a configuration language variable for boolean values.

Parameters

help_text	A human-readable text describing the purpose of this configuration variable.
initial_value	The initial value for this variable as a string. The string representation of 'true' is either "yes" or "1". The string representation of 'false' is either "no" or "0".

© 2005-2021 HörTech gGmbH, Oldenburg

5.293.3 Member Function Documentation

5.293.3.1 op_setval() `std::string MHAParser::bool_t::op_setval ( expression_t & x ) [protected], [virtual]`

Reimplemented from **MHAParser::variable_t** (p. 1119).

5.293.3.2 query_type() `std::string MHAParser::bool_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.293.3.3 query_val() `std::string MHAParser::bool_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.293.4 Member Data Documentation

5.293.4.1 data `bool MHAParser::bool_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.294 MHAParser::c_ifc_parser_t Class Reference

Inheritance diagram for MHAParser::c_ifc_parser_t:

Public Member Functions

- **c_ifc_parser_t** (const std::string &modulename_)
- ~**c_ifc_parser_t** ()
- void **set_parse_cb** (**c_parse_cmd_t**, **c_parse_err_t**, void *)

Protected Member Functions

- std::string **op_subparse** (**MHAParser::expression_t** &)
- std::string **op_setval** (**MHAParser::expression_t** &)
- std::string **op_query** (**MHAParser::expression_t** &)

Private Member Functions

- void **test_error** ()

Private Attributes

- std::string **modulename**
- **c_parse_cmd_t c_parse_cmd**
- **c_parse_err_t c_parse_err**
- int **liberr**
- void * **libdata**
- unsigned int **ret_size**
- char * **retv**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.294.1 Constructor & Destructor Documentation

5.294.1.1 c_ifc_parser_t()

```
MHAParser::c_ifc_parser_t::c_ifc_parser_t (
    const std::string & modulename_ )
```

5.294.1.2 ~c_ifc_parser_t()

```
MHAParser::c_ifc_parser_t::~c_ifc_parser_t ( )
```

5.294.2 Member Function Documentation

5.294.2.1 set_parse_cb()

```
void MHAParser::c_ifc_parser_t::set_parse_cb (
    c_parse_cmd_t ,
    c_parse_err_t ,
    void * )
```

5.294.2.2 op_subparse()

```
std::string MHAParser::c_ifc_parser_t::op_subparse (
    MHAParser::expression_t & x ) [protected], [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1036).

5.294.2.3 op_setval()

```
std::string MHAParser::c_ifc_parser_t::op_setval (
    MHAParser::expression_t & x ) [protected], [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1037).

© 2005-2021 HörTech gGmbH, Oldenburg

5.294.2.4 op_query()

`std::string MHAParser::c_ifc_parser_t::op_query ( MHAParser::expression_t & x ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1037).

5.294.2.5 test_error()

`void MHAParser::c_ifc_parser_t::test_error ( ) [private]`

5.294.3 Member Data Documentation

5.294.3.1 modulename

`std::string MHAParser::c_ifc_parser_t::modulename [private]`

5.294.3.2 c_parse_cmd

`c_parse_cmd_t MHAParser::c_ifc_parser_t::c_parse_cmd [private]`

5.294.3.3 c_parse_err

`c_parse_err_t MHAParser::c_ifc_parser_t::c_parse_err [private]`

5.294.3.4 liberr

`int MHAParser::c_ifc_parser_t::liberr [private]`

5.294.3.5 libdata

`void* MHAParser::c_ifc_parser_t::libdata [private]`

5.294.3.6 ret_size

`unsigned int MHAParser::c_ifc_parser_t::ret_size [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.294.3.7 retv `char* MHAParser::c_ifc_parser_t::retv [private]`

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.295 MHAParser::commit_t< receiver_t > Class Template Reference

Parser variable with event-emission functionality.

Inheritance diagram for MHAParser::commit_t< receiver_t >:

Public Member Functions

- **commit_t** (receiver_t *, void(receiver_t::*)(), const std::string & **help**="Variable changes action")

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAEvents::connector_t**< receiver_t > **extern_connector**

Additional Inherited Members

5.295.1 Detailed Description

template<class receiver_t>
class MHAParser::commit_t< receiver_t >

Parser variable with event-emission functionality.

The **commit_t** (p. 1053) variable can register an event receiver in its constructor, which is called whenever the variable is set to "commit".

5.295.2 Constructor & Destructor Documentation

5.295.2.1 commit_t() template<class receiver_t >

```
MHAParser::commit_t< receiver_t >::  commit_t (
            receiver_t * r,
            void(receiver_t::*)() rfun,
            const std::string & help = "Variable changes action" )
```

5.295.3 Member Data Documentation

5.295.3.1 extern_connector template<class receiver_t >

```
MHAEvents::connector_t<receiver_t>  MHAParser::commit_t< receiver_t >::extern_↩
connector  [private]
```

The documentation for this class was generated from the following file:

- **mha_parser.hh**

© 2005-2021 HörTech gGmbH, Oldenburg

5.296 MHAParser::complex_mon_t Class Reference

Monitor with complex value.

Inheritance diagram for MHAParser::complex_mon_t:

Public Member Functions

- **complex_mon_t** (const std::string &hlp)
  Create a complex monitor variable.

Public Attributes

- **mha_complex_t data**
  Data field.

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.296.1 Detailed Description

Monitor with complex value.

© 2005-2021 HörTech gGmbH, Oldenburg

5.296.2 Constructor & Destructor Documentation

5.296.2.1 complex_mon_t() `MHAParser::complex_mon_t::complex_mon_t ( const std::string & hlp )`

Create a complex monitor variable.

Parameters

hlp	A help text describing this monitor variable.

5.296.3 Member Function Documentation

5.296.3.1 query_val() `std::string MHAParser::complex_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.296.3.2 query_type() `std::string MHAParser::complex_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.296.4 Member Data Documentation

5.296.4.1 data `mha_complex_t MHAParser::complex_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.297 MHAParser::complex_t Class Reference

Variable with complex value.

Inheritance diagram for MHAParser::complex_t:

Public Member Functions

- **complex_t** (const std::string &, const std::string &, const std::string &="")

Public Attributes

- **mha_complex_t data**
  Data field.

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.297.1 Detailed Description

Variable with complex value.

5.297.2 Constructor & Destructor Documentation

5.297.2.1 complex_t()

```
MHAParser::complex_t::complex_t (
    const std::string & h,
    const std::string & v,
    const std::string & rg = "" )
```

5.297.3 Member Function Documentation

5.297.3.1 op_setval()

```
std::string MHAParser::complex_t::op_setval (
    expression_t & x ) [protected], [virtual]
```

Reimplemented from **MHAParser::variable_t** (p. 1119).

5.297.3.2 query_type()

```
std::string MHAParser::complex_t::query_type (
    const std::string & ) [protected], [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1038).

5.297.3.3 query_val()

```
std::string MHAParser::complex_t::query_val (
    const std::string & s ) [protected], [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1038).

© 2005-2021 HörTech gGmbH, Oldenburg

5.297.4 Member Data Documentation

5.297.4.1 data `mha_complex_t MHAParser::complex_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.298 MHAParser::entry_t Class Reference

Public Member Functions

- **entry_t** (const std::string &, **base_t** ∗)

Public Attributes

- std::string **name**
- **base_t** ∗ **entry**

5.298.1 Constructor & Destructor Documentation

5.298.1.1 entry_t()

```
MHAParser::entry_t::entry_t (
    const std::string & n,
    base_t * e )
```

5.298.2 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.298.2.1 name `std::string MHAParser::entry_t::name`

5.298.2.2 entry `base_t* MHAParser::entry_t::entry`

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.299 MHAParser::expression_t Class Reference

Public Member Functions

- **expression_t** (const std::string &, const std::string &)
  Constructor.
- **expression_t** ()

Public Attributes

- std::string **lval**
- std::string **rval**
- std::string **op**

5.299.1 Constructor & Destructor Documentation

5.299.1.1 expression_t() [1/2]

```
expression_t::expression_t (
    const std::string & s,
    const std::string & o )
```

Constructor.

Parameters

s	String to be splitted
o	List of valid operators (single character only)

© 2005-2021 HörTech gGmbH, Oldenburg

5.299.1.2 expression_t() [2/2] `expression_t::expression_t ( )`

5.299.2 Member Data Documentation

5.299.2.1 lval `std::string MHAParser::expression_t::lval`

5.299.2.2 rval `std::string MHAParser::expression_t::rval`

5.299.2.3 op `std::string MHAParser::expression_t::op`

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.300 MHAParser::float_mon_t Class Reference

Monitor with float value.

Inheritance diagram for MHAParser::float_mon_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **float_mon_t** (const std::string &hlp)
  Initialize a floating point (32 bits) monitor variable.

Public Attributes

- float **data**
  Data field.

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.300.1 Detailed Description

Monitor with float value.

5.300.2 Constructor & Destructor Documentation

5.300.2.1 float_mon_t() `MHAParser::float_mon_t::float_mon_t ( const std::string & hlp )`

Initialize a floating point (32 bits) monitor variable.

Parameters

hlp	A help text describing this monitor variable.

5.300.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.300.3.1 query_val() `std::string MHAParser::float_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.300.3.2 query_type() `std::string MHAParser::float_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.300.4 Member Data Documentation

5.300.4.1 data `float MHAParser::float_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.301 MHAParser::float_t Class Reference

Variable with float value.

© 2005-2021 HörTech gGmbH, Oldenburg

Inheritance diagram for MHAParser::float_t:

Public Member Functions

- **float_t** (const std::string &help_text, const std::string &initial_value, const std::string & **range**="")

  Constructor for a configuration language variable for 32bit ieee floating-point values.

Public Attributes

- float **data**

  Data field.

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.301.1 Detailed Description

Variable with float value.

5.301.2 Constructor & Destructor Documentation

5.301.2.1 float_t()

```
MHAParser::float_t::float_t (
    const std::string & help_text,
    const std::string & initial_value,
    const std::string & range = "" )
```

Constructor for a configuration language variable for 32bit ieee floating-point values.

Parameters

help_text	A human-readable text describing the purpose of this configuration variable.
initial_value	The initial value for this variable as a string (decimal representation of the floating-point variable). If a range is given in the third parameter, then the initial value has to be within the range. A human-readable text describing the purpose of this configuration variable.
range	The range of values that this variable can hold can be restricted. A range is a string of the form "[a,b]", where a and b are decimal representations of the inclusive boundaries of the range. a<=b. In a range of the form "]a,b[", both boundaries are excluded. Mixed forms are permitted. a or b can also be omitted if there is no lower or upper limit. The range of values is always restricted by the representable range of the underlying C data type.

5.301.3 Member Function Documentation

5.301.3.1 op_setval()

```
std::string MHAParser::float_t::op_setval (
    expression_t & x ) [protected], [virtual]
```

Reimplemented from **MHAParser::variable_t** (p. 1119).

© 2005-2021 HörTech gGmbH, Oldenburg

5.301.3.2 query_type() `std::string MHAParser::float_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.301.3.3 query_val() `std::string MHAParser::float_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.301.4 Member Data Documentation

5.301.4.1 data `float MHAParser::float_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.302 MHAParser::int_mon_t Class Reference

Monitor variable with int value.

Inheritance diagram for MHAParser::int_mon_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **int_mon_t** (const std::string &hlp)
  Create a monitor variable for integral values.

Public Attributes

- int **data**
  Data field.

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.302.1 Detailed Description

Monitor variable with int value.

Monitor variables can be of many types. These variables can be queried through the parser. The public data element contains the monitored state. Write access is only possible from the C++ code by direct access to the data field.

5.302.2 Constructor & Destructor Documentation

5.302.2.1 int_mon_t() `MHAParser::int_mon_t::int_mon_t ( const std::string & hlp )`

Create a monitor variable for integral values.

Parameters

hlp	A help text describing this monitor variable.

© 2005-2021 HörTech gGmbH, Oldenburg

5.302.3 Member Function Documentation

5.302.3.1 query_val() `std::string MHAParser::int_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.302.3.2 query_type() `std::string MHAParser::int_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.302.4 Member Data Documentation

5.302.4.1 data `int MHAParser::int_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.303 MHAParser::int_t Class Reference

Variable with integer value.

Inheritance diagram for MHAParser::int_t:

Public Member Functions

- **int_t** (const std::string &help_text, const std::string &initial_value, const std::string & **range**="")

  Constructor for a configuration language variable for integral values.

Public Attributes

- int **data**

  Data field.

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

Additional Inherited Members

5.303.1 Detailed Description

Variable with integer value.

5.303.2 Constructor & Destructor Documentation

5.303.2.1 int_t()

```
MHAParser::int_t::int_t (
    const std::string & help_text,
    const std::string & initial_value,
    const std::string & range = "" )
```

Constructor for a configuration language variable for integral values.

Parameters

help_text	A human-readable text describing the purpose of this configuration variable.
initial_value	The initial value for this variable as a string (decimal representation of the integer variable). If a range is given in the third parameter, then the initial value has to be within the range.
range	The range of values that this variable can hold can be restricted. A range is a string of the form "[a,b]", where a and b are decimal representations of the integral inclusive boundaries of the range. a<=b. In a range of the form "]a,b[", both boundaries are excluded. Mixed forms are permitted. a or b can also be omitted if there is no lower or upper limit. The range of values is always restricted by the representable range of the underlying C data type (usually 32 bits, [-2147483648,2147483647]).

5.303.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.303.3.1 op_setval() `std::string MHAParser::int_t::op_setval ( expression_t & x ) [protected], [virtual]`

Reimplemented from **MHAParser::variable_t** (p. 1119).

5.303.3.2 query_type() `std::string MHAParser::int_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.303.3.3 query_val() `std::string MHAParser::int_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.303.4 Member Data Documentation

5.303.4.1 data `int MHAParser::int_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.304 MHAParser::keyword_list_t Class Reference

Keyword list class.

Public Types

- typedef std::vector< std::string >::size_type **size_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- void **set_value** (const std::string &)
  Select a value from keyword list.
- void **set_entries** (const std::string &)
  Set keyword list entries.
- const std::string & **get_value** () const
  Return selected value.
- const std::vector< std::string > & **get_entries** () const
  Return keyword list.
- const **size_t** & **get_index** () const
  Return index of selected value.
- void **set_index** (unsigned int)
- void **validate** () const
  Check if index of selected value is valid.
- void **add_entry** (const std::string &en)
- **keyword_list_t** ()
  Constructor.

Private Attributes

- **size_t index**
  Index into list.
- std::vector< std::string > **entries**
  List of valid entries.
- std::string **empty_string**

5.304.1 Detailed Description

Keyword list class.

The stucture **keyword_list_t** (p. 1071) defines a keyword list (vector of strings) with an index into the list. Used as **MHAParser::kw_t** (p. 1075), it can be used to access a set of valid keywords through the parser (i.e. one of "pear apple banana").

5.304.2 Member Typedef Documentation

5.304.2.1 size_t `typedef std::vector<std::string>::size_type` **MHAParser::keyword_list_t::size_t**

5.304.3 Constructor & Destructor Documentation

5.304.3.1 keyword_list_t() `MHAParser::keyword_list_t::keyword_list_t ( )`

Constructor.

5.304.4 Member Function Documentation

5.304.4.1 set_value() `void MHAParser::keyword_list_t::set_value ( const std::string & s )`

Select a value from keyword list.

This function selects a value from the keyword list. The index is set to the last matching entry.

Parameters

s	Value to be selected.

5.304.4.2 set_entries() `void MHAParser::keyword_list_t::set_entries ( const std::string & s )`

Set keyword list entries.

With this function, the keyword list can be set from a space separated string list.

Parameters

s	Space separated entry list.

5.304.4.3 get_value() `const std::string & MHAParser::keyword_list_t::get_value ( )`

© 2005-2021 HörTech gGmbH, Oldenburg

`const`

Return selected value.

5.304.4.4 get_entries() `const std::vector< std::string > & MHAParser::keyword_list_t::get_entries ( ) const`

Return keyword list.

5.304.4.5 get_index() `const` **`MHAParser::keyword_list_t::size_t`** `& MHAParser::keyword_list_t::get_index ( ) const`

Return index of selected value.

5.304.4.6 set_index() `void MHAParser::keyword_list_t::set_index ( unsigned int` *`idx`* `)`

5.304.4.7 validate() `void MHAParser::keyword_list_t::validate ( ) const`

Check if index of selected value is valid.

5.304.4.8 add_entry() `void MHAParser::keyword_list_t::add_entry ( const std::string &` *`en`* `) [inline]`

5.304.5 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.304.5.1 index `size_t MHAParser::keyword_list_t::index [private]`

Index into list.

5.304.5.2 entries `std::vector<std::string> MHAParser::keyword_list_t::entries [private]`

List of valid entries.

5.304.5.3 empty_string `std::string MHAParser::keyword_list_t::empty_string [private]`

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.305 MHAParser::kw_t Class Reference

Variable with keyword list value.

Inheritance diagram for MHAParser::kw_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **kw_t** (const std::string &, const std::string &, const std::string &)
  Constructor of a keyword list openMHA configuration variable.
- **kw_t** (const **kw_t** &)
  Copy constructor.
- void **set_range** (const std::string &)
  Set/change the list of valid entries.
- bool **isval** (const std::string &) const
  Test if the given value is selected.

Public Attributes

- **keyword_list_t data**
  Variable data in its native type.

Protected Member Functions

- void **validate** (const **keyword_list_t** &)
- std::string **op_setval** (**expression_t** &)
- std::string **query_range** (const std::string &)
- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.305.1 Detailed Description

Variable with keyword list value.

5.305.2 Constructor & Destructor Documentation

5.305.2.1 kw_t() [1/2]

```
MHAParser::kw_t::kw_t (
    const std::string & h,
    const std::string & v,
    const std::string & rg )
```

Constructor of a keyword list openMHA configuration variable.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

h	A help string describing the purpose of this variable.
v	The initial value, has to be a value from the list of possible values given in the last parameter.
rg	A string containing the list of valid entries. The entries have to be separated by spaces. The list of entries has to be delimited by brackets "[", "]".

5.305.2.2 kw_t() [2/2]

```
MHAParser::kw_t::kw_t (
    const kw_t & src )
```

Copy constructor.

5.305.3 Member Function Documentation

5.305.3.1 set_range()

```
void MHAParser::kw_t::set_range (
    const std::string & r )
```

Set/change the list of valid entries.

Parameters

r	A string containing the list of valid entries. The entries have to be separated by spaces. The list of entries has to be delimited by brackets "[", "]".

5.305.3.2 isval()

```
bool MHAParser::kw_t::isval (
    const std::string & testval ) const
```

Test if the given value is selected.

5.305.3.3 validate()

```
void MHAParser::kw_t::validate (
    const keyword_list_t & s ) [protected]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.305.3.4 op_setval() `std::string MHAParser::kw_t::op_setval ( expression_t & x ) [protected], [virtual]`

Reimplemented from **MHAParser::variable_t** (p. 1119).

5.305.3.5 query_range() `std::string MHAParser::kw_t::query_range ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.305.3.6 query_val() `std::string MHAParser::kw_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.305.3.7 query_type() `std::string MHAParser::kw_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.305.4 Member Data Documentation

5.305.4.1 data `keyword_list_t MHAParser::kw_t::data`

Variable data in its native type.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.306 MHAParser::mcomplex_mon_t Class Reference

Matrix of complex numbers monitor.

Inheritance diagram for MHAParser::mcomplex_mon_t:

Public Member Functions

- **mcomplex_mon_t** (const std::string &hlp)
  Create a matrix of complex floating point monitor values.

Public Attributes

- std::vector< std::vector< **mha_complex_t** > > **data**
  Data field.

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.306.1 Detailed Description

Matrix of complex numbers monitor.

© 2005-2021 HörTech gGmbH, Oldenburg

5.306.2 Constructor & Destructor Documentation

5.306.2.1 mcomplex_mon_t() `MHAParser::mcomplex_mon_t::mcomplex_mon_t ( const std::string & hlp )`

Create a matrix of complex floating point monitor values.

Parameters

hlp	A help text describing this monitor variable.

5.306.3 Member Function Documentation

5.306.3.1 query_val() `std::string MHAParser::mcomplex_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.306.3.2 query_type() `std::string MHAParser::mcomplex_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.306.4 Member Data Documentation

5.306.4.1 data `std::vector< std::vector< mha_complex_t> > MHAParser::mcomplex_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.307 MHAParser::mcomplex_t Class Reference

Matrix variable with complex value.

Inheritance diagram for MHAParser::mcomplex_t:

Public Member Functions

- **mcomplex_t** (const std::string &, const std::string &, const std::string &="")

Public Attributes

- std::vector< std::vector< **mha_complex_t** > > **data**
  Data field.

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.307.1 Detailed Description

Matrix variable with complex value.

5.307.2 Constructor & Destructor Documentation

5.307.2.1 mcomplex_t()

```
MHAParser::mcomplex_t::mcomplex_t (
    const std::string & h,
    const std::string & v,
    const std::string & rg = "" )
```

5.307.3 Member Function Documentation

5.307.3.1 op_setval()

```
std::string MHAParser::mcomplex_t::op_setval (
    expression_t & x ) [protected], [virtual]
```

Reimplemented from **MHAParser::variable_t** (p. 1119).

5.307.3.2 query_type()

```
std::string MHAParser::mcomplex_t::query_type (
    const std::string & ) [protected], [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1038).

5.307.3.3 query_val()

```
std::string MHAParser::mcomplex_t::query_val (
    const std::string & s ) [protected], [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1038).

© 2005-2021 HörTech gGmbH, Oldenburg

5.307.4 Member Data Documentation

5.307.4.1 data `std::vector<std::vector< mha_complex_t> > MHAParser::mcomplex_t↩::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.308 MHAParser::mfloat_mon_t Class Reference

Matrix of floats monitor.

Inheritance diagram for MHAParser::mfloat_mon_t:

Public Member Functions

- **mfloat_mon_t** (const std::string &hlp)
  Create a matrix of floating point monitor values.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- std::vector< std::vector< float > > **data**
  Data field.

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.308.1 Detailed Description

Matrix of floats monitor.

5.308.2 Constructor & Destructor Documentation

5.308.2.1 mfloat_mon_t() `MHAParser::mfloat_mon_t::mfloat_mon_t ( const std::string & hlp )`

Create a matrix of floating point monitor values.

Parameters

hlp	A help text describing this monitor variable.

5.308.3 Member Function Documentation

5.308.3.1 query_val() `std::string MHAParser::mfloat_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

© 2005-2021 HörTech gGmbH, Oldenburg

5.308.3.2 query_type() `std::string MHAParser::mfloat_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.308.4 Member Data Documentation

5.308.4.1 data `std::vector< std::vector<float> > MHAParser::mfloat_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.309 MHAParser::mfloat_t Class Reference

Matrix variable with float value.

Inheritance diagram for MHAParser::mfloat_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **mfloat_t** (const std::string &, const std::string &, const std::string &="")
  Create a float matrix parser variable.

Public Attributes

- std::vector< std::vector< float > > **data**
  Data field.

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

Additional Inherited Members

5.309.1 Detailed Description

Matrix variable with float value.

5.309.2 Constructor & Destructor Documentation

5.309.2.1 mfloat_t()

```
MHAParser::mfloat_t::mfloat_t (
    const std::string & h,
    const std::string & v,
    const std::string & rg = "" )
```

Create a float matrix parser variable.

Parameters

h	A human-readable text describing the purpose of this configuration variable.
v	The initial value of the variable, as a string, in openMHA configuration language: (e.g. "[[0 1]; [2 3]]" for a matrix), described in the "Multidimensional Variables" s2.1.3 section of the openMHA User Manual.
rg	The numeric range to enforce on all members of the matrix.

© 2005-2021 HörTech gGmbH, Oldenburg

5.309.3 Member Function Documentation

5.309.3.1 op_setval() `std::string MHAParser::mfloat_t::op_setval ( expression_t & x ) [protected], [virtual]`

Reimplemented from **MHAParser::variable_t** (p. 1119).

5.309.3.2 query_type() `std::string MHAParser::mfloat_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.309.3.3 query_val() `std::string MHAParser::mfloat_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.309.4 Member Data Documentation

5.309.4.1 data `std::vector<std::vector<float> > MHAParser::mfloat_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.310 MHAParser::mhaconfig_mon_t Class Reference

Inheritance diagram for MHAParser::mhaconfig_mon_t:

Public Member Functions

- **mhaconfig_mon_t** (const std::string & **help**="")
- void **update** (const **mhaconfig_t** &cf)

Private Attributes

- **MHAParser::int_mon_t channels**

  Number of audio channels.
- **MHAParser::string_mon_t domain**

  Signal domain (MHA_WAVEFORM or MHA_SPECTRUM)
- **MHAParser::int_mon_t fragsize**

  Fragment size of waveform data.
- **MHAParser::int_mon_t wndlen**

  Window length of spectral data.
- **MHAParser::int_mon_t fftlen**

  FFT length of spectral data.
- **MHAParser::float_mon_t srate**

  Sampling rate in Hz.

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.310.1 Constructor & Destructor Documentation

5.310.1.1 mhaconfig_mon_t()

```
MHAParser::mhaconfig_mon_t::mhaconfig_mon_t (
    const std::string & help = "" )
```

5.310.2 Member Function Documentation

5.310.2.1 update()

```
void MHAParser::mhaconfig_mon_t::update (
    const mhaconfig_t & cf )
```

5.310.3 Member Data Documentation

5.310.3.1 channels

```
MHAParser::int_mon_t MHAParser::mhaconfig_mon_t::channels [private]
```

Number of audio channels.

5.310.3.2 domain

```
MHAParser::string_mon_t MHAParser::mhaconfig_mon_t::domain [private]
```

Signal domain (MHA_WAVEFORM or MHA_SPECTRUM)

5.310.3.3 fragsize

```
MHAParser::int_mon_t MHAParser::mhaconfig_mon_t::fragsize [private]
```

Fragment size of waveform data.

© 2005-2021 HörTech gGmbH, Oldenburg

5.310.3.4 wndlen `MHAParser::int_mon_t MHAParser::mhaconfig_mon_t::wndlen [private]`

Window length of spectral data.

5.310.3.5 fftlen `MHAParser::int_mon_t MHAParser::mhaconfig_mon_t::fftlen [private]`

FFT length of spectral data.

5.310.3.6 srate `MHAParser::float_mon_t MHAParser::mhaconfig_mon_t::srate [private]`

Sampling rate in Hz.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.311 MHAParser::mhapluginloader_t Class Reference

Class to create a plugin loader in a parser, including the load logic.

Inheritance diagram for MHAParser::mhapluginloader_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **mhapluginloader_t** (**MHAParser::parser_t** &parent, const **algo_comm_t** &ac, const std::string &plugname_name="plugin_name", const std::string &prefix="")
- ∼**mhapluginloader_t** ()
- void **prepare** (**mhaconfig_t** &cf)
- void **release** ()
- void **process** (**mha_wave_t** ∗sIn, **mha_wave_t** ∗∗sOut)
- void **process** (**mha_spec_t** ∗sIn, **mha_spec_t** ∗∗sOut)
- void **process** (**mha_wave_t** ∗sIn, **mha_spec_t** ∗∗sOut)
- void **process** (**mha_spec_t** ∗sIn, **mha_wave_t** ∗∗sOut)
- **mhaconfig_t** **get_cfin** () const
- **mhaconfig_t** **get_cfout** () const
- const std::string & **get_last_name** () const

Protected Attributes

- **PluginLoader::mhapluginloader_t** ∗ **plug**

Private Member Functions

- void **load_plug** ()

Private Attributes

- **MHAParser::parser_t** & **parent_**
- **MHAParser::string_t** **plugname**
- std::string **prefix_**
- **MHAEvents::connector_t**< **mhapluginloader_t** > **connector**
- **algo_comm_t** **ac_**
- std::string **last_name**
- std::string **plugname_name_**
- **mhaconfig_t** **cf_in_**
- **mhaconfig_t** **cf_out_**

Static Private Attributes

- static double **bookkeeping**

5.311.1 Detailed Description

Class to create a plugin loader in a parser, including the load logic.

© 2005-2021 HörTech gGmbH, Oldenburg

5.311.2 Constructor & Destructor Documentation

5.311.2.1 mhapluginloader_t()

```
MHAParser::mhapluginloader_t::mhapluginloader_t (
    MHAParser::parser_t & parent,
    const algo_comm_t & ac,
    const std::string & plugname_name = "plugin_name",
    const std::string & prefix = "" )
```

5.311.2.2 ∼mhapluginloader_t()

```
MHAParser::mhapluginloader_t::∼mhapluginloader_t ( )
```

5.311.3 Member Function Documentation

5.311.3.1 prepare()

```
void MHAParser::mhapluginloader_t::prepare (
    mhaconfig_t & cf )
```

5.311.3.2 release()

```
void MHAParser::mhapluginloader_t::release ( )
```

5.311.3.3 process() [1/4]

```
void MHAParser::mhapluginloader_t::process (
    mha_wave_t * sIn,
    mha_wave_t ** sOut ) [inline]
```

5.311.3.4 process() [2/4]

```
void MHAParser::mhapluginloader_t::process (
    mha_spec_t * sIn,
    mha_spec_t ** sOut ) [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.311.3.5 process() [3/4]

```
void MHAParser::mhapluginloader_t::process (
    mha_wave_t * sIn,
    mha_spec_t ** sOut ) [inline]
```

5.311.3.6 process() [4/4]

```
void MHAParser::mhapluginloader_t::process (
    mha_spec_t * sIn,
    mha_wave_t ** sOut ) [inline]
```

5.311.3.7 get_cfin()

```
mhaconfig_t MHAParser::mhapluginloader_t::get_cfin ( ) const [inline]
```

5.311.3.8 get_cfout()

```
mhaconfig_t MHAParser::mhapluginloader_t::get_cfout ( ) const [inline]
```

5.311.3.9 get_last_name()

```
const std::string& MHAParser::mhapluginloader_t::get_↩
last_name ( ) const [inline]
```

5.311.3.10 load_plug()

```
void MHAParser::mhapluginloader_t::load_plug ( ) [private]
```

5.311.4 Member Data Documentation

5.311.4.1 plug

```
PluginLoader::mhapluginloader_t* MHAParser::mhapluginloader_t::plug [protected]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.311.4.2 parent_ **MHAParser::parser_t**& MHAParser::mhapluginloader_t::parent_↩ [private]

5.311.4.3 plugname **MHAParser::string_t** MHAParser::mhapluginloader_t::plugname [private]

5.311.4.4 prefix_ std::string MHAParser::mhapluginloader_t::prefix_ [private]

5.311.4.5 connector **MHAEvents::connector_t**< **mhapluginloader_t**> MHAParser::mhapluginloader↩ _t::connector [private]

5.311.4.6 ac_ **algo_comm_t** MHAParser::mhapluginloader_t::ac_ [private]

5.311.4.7 last_name std::string MHAParser::mhapluginloader_t::last_name [private]

5.311.4.8 plugname_name_ std::string MHAParser::mhapluginloader_t::plugname_↩ name_ [private]

5.311.4.9 cf_in_ **mhaconfig_t** MHAParser::mhapluginloader_t::cf_in_ [private]

5.311.4.10 cf_out_ **mhaconfig_t** MHAParser::mhapluginloader_t::cf_out_ [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.311.4.11 bookkeeping `double MHAParser::mhapluginloader_t::bookkeeping [static], [private]`

The documentation for this class was generated from the following files:

- **mhapluginloader.h**
- **mhapluginloader.cpp**

5.312 MHAParser::mint_mon_t Class Reference

Matrix of ints monitor.

Inheritance diagram for MHAParser::mint_mon_t:

Public Member Functions

- **mint_mon_t** (const std::string &hlp)
  Create a matrix of integer monitor values.

Public Attributes

- std::vector< std::vector< int > > **data**
  Data field.

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.312.1 Detailed Description

Matrix of ints monitor.

5.312.2 Constructor & Destructor Documentation

5.312.2.1 mint_mon_t() `MHAParser::mint_mon_t::mint_mon_t ( const std::string & hlp )`

Create a matrix of integer monitor values.

Parameters

hlp	A help text describing this monitor variable.

5.312.3 Member Function Documentation

5.312.3.1 query_val() `std::string MHAParser::mint_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.312.3.2 query_type() `std::string MHAParser::mint_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

© 2005-2021 HörTech gGmbH, Oldenburg

5.312.4 Member Data Documentation

5.312.4.1 data `std::vector< std::vector<int> > MHAParser::mint_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.313 MHAParser::mint_t Class Reference

Matrix variable with int value.

Inheritance diagram for MHAParser::mint_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **mint_t** (const std::string &, const std::string &, const std::string &="")
  Create a int matrix parser variable.

Public Attributes

- std::vector< std::vector< int > > **data**
  Data field.

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

Additional Inherited Members

5.313.1 Detailed Description

Matrix variable with int value.

5.313.2 Constructor & Destructor Documentation

5.313.2.1 mint_t()

```
MHAParser::mint_t::mint_t (
    const std::string & h,
    const std::string & v,
    const std::string & rg = "" )
```

Create a int matrix parser variable.

Parameters

h	A human-readable text describing the purpose of this configuration variable.
v	The initial value of the variable, as a string, in openMHA configuration language: (e.g. "[[0 1]; [2 3]]" for a matrix), described in the "Multidimensional Variables" s2.1.3 section of the openMHA User Manual.
rg	The numeric range to enforce on all members of the matrix.

© 2005-2021 HörTech gGmbH, Oldenburg

5.313.3 Member Function Documentation

5.313.3.1 op_setval() `std::string MHAParser::mint_t::op_setval ( expression_t & x ) [protected], [virtual]`

Reimplemented from **MHAParser::variable_t** (p. 1119).

5.313.3.2 query_type() `std::string MHAParser::mint_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.313.3.3 query_val() `std::string MHAParser::mint_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.313.4 Member Data Documentation

5.313.4.1 data `std::vector<std::vector<int> > MHAParser::mint_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.314 MHAParser::monitor_t Class Reference

Base class for monitors and variable nodes.

Inheritance diagram for MHAParser::monitor_t:

Public Member Functions

- **monitor_t** (const std::string &)
- **monitor_t** (const **monitor_t** &)
- **monitor_t** & **operator=** (const **monitor_t** &)=default
- std::string **op_query** (**expression_t** &)
- std::string **query_dump** (const std::string &)
- std::string **query_perm** (const std::string &)

Additional Inherited Members

5.314.1 Detailed Description

Base class for monitors and variable nodes.

5.314.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.314.2.1 monitor_t() [1/2]

```
MHAParser::monitor_t::monitor_t (
    const std::string & h )
```

5.314.2.2 monitor_t() [2/2]

```
MHAParser::monitor_t::monitor_t (
    const monitor_t & src )
```

5.314.3 Member Function Documentation

5.314.3.1 operator=()

```
monitor_t& MHAParser::monitor_t::operator= (
    const monitor_t & ) [default]
```

5.314.3.2 op_query()

```
std::string MHAParser::monitor_t::op_query (
    expression_t & x ) [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1037).

5.314.3.3 query_dump()

```
std::string MHAParser::monitor_t::query_dump (
    const std::string & ) [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1037).

5.314.3.4 query_perm()

```
std::string MHAParser::monitor_t::query_perm (
    const std::string & ) [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1037).

Reimplemented in **MHAParser::variable_t** (p. 1119).

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.315 MHAParser::parser_t Class Reference

Parser node class.

Inherits **MHAParser::base_t**.

Inherited by **alsa_dev_par_parser_t**, **AuditoryProfile::parser_t**, **AuditoryProfile::parser_t::ear_t**, **AuditoryProfile::parser_t::fmap_t**, **DynComp::dc_afterburn_vars_t**, **fw_t**, **io_alsa_t**, **io_asterisk_parser_t**, **io_file_t**, **io_parser_t**, **io_tcp_parser_t**, **MHAFilter::adapt_filter_t**, **MHAFilter::iir_filter_t**, **MHAIOJack::io_jack_t**, **MHAIOJackdb::io_jack_t**, **MHAIOPortAudio::device_info_t**, **MHAIOPortAudio::io_portaudio_t**, **MHAIOPortAudio::stream_info_t**, **MHAParser::mhaconfig_mon_t**, **MHAParser::window_t**, **MHAPlugin::plugin_t< runtime_cfg_t >**, **MHAPlugin_Split::split_t**, **MHAPlugin::plugin_t< ac2wave_t >**, **MHAPlugin::plugin_t< acConcat_wave_config >**, **MHAPlugin::plugin_t< acPooling_wave_config >**, **MHAPlugin::plugin_t< acSteer_config >**, **MHAPlugin::plugin_t< acTransform_wave_config >**, **MHAPlugin::plugin_t< acwriter_t >**, **MHAPlugin::plugin_t< adaptive_feedback_canceller_config >**, **MHAPlugin::plugin_t< adm_rtconfig_t >**, **MHAPlugin::plugin_t< analysepath_t >**, **MHAPlugin::plugin_t< cfg_t >**, **MHAPlugin::plugin_t< char >**, **MHAPlugin::plugin_t< cohflt_t >**, **MHAPlugin::plugin_t< combc_t >**, **MHAPlugin::plugin_t< cpuload_cfg_t >**, **MHAPlugin::plugin_t< db_t >**, **MHAPlugin::plugin_t< dbasync_t >**, **MHAPlugin::plugin_t< dc_t >**, **MHAPlugin::plugin_t< delaysum_t >**, **MHAPlugin::plugin_t< delaysum_wave_t >**, **MHAPlugin::plugin_t< doasvm_classification_config >**, **MHAPlugin::plugin_t< doasvm_feature_extraction_config >**, **MHAPlugin::plugin_t< example5_t >**, **MHAPlugin::plugin_t< fftfb_plug_t >**, **MHAPlugin::plugin_t< fftfbpow_t >**, **MHAPlugin::plugin_t< fftfilter_t >**, **MHAPlugin::plugin_t< fshift_config_t >**, **MHAPlugin::plugin_t< gsc_adaptive_stage >**, **MHAPlugin::plugin_t< gtfb_analyzer_cfg_t >**, **MHAPlugin::plugin_t< gtfb_simd_cfg_t >**, **MHAPlugin::plugin_t< gtfb_simple_rt_t >**, **MHAPlugin::plugin_t< hilbert_shifter_t >**, **MHAPlugin::plugin_t< int >**, **MHAPlugin::plugin_t< level_matching_config_t >**, **MHAPlugin::plugin_t< lpc_bl_predictor_config >**, **MHAPlugin::plugin_t< lpc_burglattice_config >**, **MHAPlugin::plugin_t< lpc_config >**, **MHAPlugin::plugin_t< matlab_wrapper_rt_cfg_t >**, **MHAPlugin::plugin_t< MHA_AC::spectrum_t >**, **MHAPlugin::plugin_t< MHA_AC::waveform_t >**, **MHAPlugin::plugin_t< mhachain::plugs_t >**, **MHAPlugin::plugin_t< MHAFilter::partitioned_convolution_t >**, **MHAPlugin::plugin_t< MHASignal::async_rmslevel_t >**, **MHAPlugin::plugin_t< MHASignal::delay_t >**, **MHAPlugin::plugin_t< MHASignal::waveform_t >**, **MHAPlugin::plugin_t< MHAWindow::fun_t >**, **MHAPlugin::plugin_t< noise_psd_estimator_t >**, **MHAPlugin::plugin_t< overlapadd_t >**, **MHAPlugin::plugin_t< plingploing_t >**, **MHAPlugin::plugin_t< resampling_t >**, **MHAPlugin::plugin_t< rmslevel_t >**, **MHAPlugin::plugin_t< rohConfig >**, **MHAPlugin::plugin_t< route::process_t >**, **MHAPlugin::plugin_t< rt_nlms_t >**, **MHAPlugin::plugin_t< scaler_t >**, **MHAPlugin::plugin_t< sine_cfg_t >**, **MHAPlugin::plugin_t< smooth_cepstrum_t >**, **MHAPlugin::plugin_t< smoothspec_wrap_t >**, **MHAPlugin::plugin_t< spec2wave_t >**, **MHAPlugin::plugin_t< spec_fader_t >**, **MHAPlugin::plugin_t< steerbf_config >**, **MHAPlugin::plugin_t< wave2spec_t >**, **MHAPlugin::plugin_t< wavwriter_t >**, **softclipper_variables_t**, **testplugin::ac_parser_t**, **testplugin::config_parser_t**, and **testplugin::signal_parser_t**.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **parser_t** (const std::string &help_text="")
  Construct detached node to be used in the configuration tree.
- ~**parser_t** ()
- void **insert_item** (const std::string &, **base_t** *)
  Register a parser item into this sub-parser.
- void **remove_item** (const std::string &)
  Remove an item by name.
- void **force_remove_item** (const std::string &)
  Remove an item by name.
- void **remove_item** (const **base_t** *)
  Remove an item by address.

Protected Member Functions

- std::string **op_subparse** (**expression_t** &)
- std::string **op_setval** (**expression_t** &)
- std::string **op_query** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_dump** (const std::string &)
- std::string **query_entries** (const std::string &)
- std::string **query_readfile** (const std::string &)
- std::string **query_savefile** (const std::string &)
- std::string **query_savefile_compact** (const std::string &)
- std::string **query_savemons** (const std::string &)
- std::string **query_val** (const std::string &)
- std::string **query_listids** (const std::string &)
- void **set_id_string** (const std::string &)
- bool **has_entry** (const std::string &)

Private Attributes

- **entry_map_t entries**
- std::string **id_string**
  identification string
- std::string **last_errormsg**

Additional Inherited Members

5.315.1 Detailed Description

Parser node class.

A **parser_t** (p. 1102) instance is a node in the configuration tree. A parser node can contain any number of other **parser_t** (p. 1102) instances or configuration language variables. These items are inserted into a parser node using the **parser_t::insert_item** (p. 1104) method.

© 2005-2021 HörTech gGmbH, Oldenburg

5.315.2 Constructor & Destructor Documentation

5.315.2.1 parser_t() `MHAParser::parser_t::parser_t ( const std::string & help_text = "" )`

Construct detached node to be used in the configuration tree.

Parameters

help_text	A text describing this node. E.g. if this node lives at the root of some openMHA plugin, then the help text should describe the functionality of the plugin.

5.315.2.2 ~parser_t() `MHAParser::parser_t::~parser_t ( )`

5.315.3 Member Function Documentation

5.315.3.1 insert_item() `void MHAParser::parser_t::insert_item ( const std::string & n, MHAParser::base_t * e )`

Register a parser item into this sub-parser.

This function registers an item under a given name into this sub-parser and makes it accessible to the parser interface.

Parameters

n	Name of the item in the configuration tree
e	C++ pointer to the item instance. e can either point to a variable, to a monitor, or to another sub-parser.

© 2005-2021 HörTech gGmbH, Oldenburg

5.315.3.2 remove_item() [1/2]

```
void MHAParser::parser_t::remove_item (
    const std::string & n )
```

Remove an item by name.

If the item does not exist, an error is being reported.

Parameters

n	Name of parser item to be removed from list.

5.315.3.3 force_remove_item()

```
void MHAParser::parser_t::force_remove_item (
    const std::string & n )
```

Remove an item by name.

Non-existing items are ignored.

Parameters

n	Name of parser item to be removed from list.

5.315.3.4 remove_item() [2/2]

```
void MHAParser::parser_t::remove_item (
    const base_t * addr )
```

Remove an item by address.

The item belonging to an address is being removed from the list of items.

Parameters

addr	Address of parser item to be removed.

5.315.3.5 op_subparse()

```
std::string MHAParser::parser_t::op_subparse (
    expression_t & x ) [protected], [virtual]
```

© 2005-2021 HörTech gGmbH, Oldenburg

Reimplemented from **MHAParser::base_t** (p. 1036).

5.315.3.6 op_setval()

`std::string MHAParser::parser_t::op_setval ( expression_t & x ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1037).

5.315.3.7 op_query()

`std::string MHAParser::parser_t::op_query ( expression_t & x ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1037).

5.315.3.8 query_type()

`std::string MHAParser::parser_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.315.3.9 query_dump()

`std::string MHAParser::parser_t::query_dump ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1037).

5.315.3.10 query_entries()

`std::string MHAParser::parser_t::query_entries ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1037).

© 2005-2021 HörTech gGmbH, Oldenburg

5.315.3.11 query_readfile()

`std::string MHAParser::parser_t::query_readfile ( const std::string & fname ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.315.3.12 query_savefile()

`std::string MHAParser::parser_t::query_savefile ( const std::string & fname ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1039).

5.315.3.13 query_savefile_compact()

`std::string MHAParser::parser_t::query_savefile_compact ( const std::string & fname ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1039).

5.315.3.14 query_savemons()

`std::string MHAParser::parser_t::query_savemons ( const std::string & fname ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1039).

5.315.3.15 query_val()

`std::string MHAParser::parser_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.315.3.16 query_listids()

`std::string MHAParser::parser_t::query_listids ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1039).

© 2005-2021 HörTech gGmbH, Oldenburg

5.315.3.17 set_id_string() `void MHAParser::parser_t::set_id_string ( const std::string & s ) [protected]`

5.315.3.18 has_entry() `bool MHAParser::parser_t::has_entry ( const std::string & s ) [protected]`

5.315.4 Member Data Documentation

5.315.4.1 entries `entry_map_t MHAParser::parser_t::entries [private]`

5.315.4.2 id_string `std::string MHAParser::parser_t::id_string [private]`

identification string

5.315.4.3 last_errormsg `std::string MHAParser::parser_t::last_errormsg [private]`

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.316 MHAParser::range_var_t Class Reference

Base class for all variables with a numeric value range.

Inheritance diagram for MHAParser::range_var_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **range_var_t** (const std::string &, const std::string &="")
- **range_var_t** (const **range_var_t** &)
- std::string **query_range** (const std::string &)
- void **set_range** (const std::string &r)

  Change the valid range of a variable.
- void **validate** (const int &)
- void **validate** (const float &)
- void **validate** (const **mha_complex_t** &)
- void **validate** (const std::vector< int > &)
- void **validate** (const std::vector< float > &)
- void **validate** (const std::vector< **mha_complex_t** > &)
- void **validate** (const std::vector< std::vector< int > > &)
- void **validate** (const std::vector< std::vector< float > > &)
- void **validate** (const std::vector< std::vector< **mha_complex_t** > > &)

Protected Attributes

- float **low_limit**

  Lower limit of range.

- float **up_limit**

  Upper limit of range.

- bool **low_incl**

  Lower limit is included (or excluded) in range.
- bool **up_incl**

  Upper limit is included (or excluded) in range.
- bool **check_low**

  Check lower limit.
- bool **check_up**

  Check upper limit.
- bool **check_range**

  Range checking is active.

Additional Inherited Members

5.316.1 Detailed Description

Base class for all variables with a numeric value range.

© 2005-2021 HörTech gGmbH, Oldenburg

5.316.2 Constructor & Destructor Documentation

5.316.2.1 range_var_t() [1/2]

```
MHAParser::range_var_t::range_var_t (
    const std::string & h,
    const std::string & r = "" )
```

5.316.2.2 range_var_t() [2/2]

```
MHAParser::range_var_t::range_var_t (
    const range_var_t & src )
```

5.316.3 Member Function Documentation

5.316.3.1 query_range()

```
std::string MHAParser::range_var_t::query_range (
    const std::string & ) [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1038).

5.316.3.2 set_range()

```
void MHAParser::range_var_t::set_range (
    const std::string & r )
```

Change the valid range of a variable.

Parameters

r	New range of the variable (string representation)

5.316.3.3 validate() [1/9]

```
void MHAParser::range_var_t::validate (
    const int & v )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.316.3.4 validate() [2/9]

```
void MHAParser::range_var_t::validate (
    const float & v )
```

5.316.3.5 validate() [3/9]

```
void MHAParser::range_var_t::validate (
    const mha_complex_t & v )
```

5.316.3.6 validate() [4/9]

```
void MHAParser::range_var_t::validate (
    const std::vector< int > & v )
```

5.316.3.7 validate() [5/9]

```
void MHAParser::range_var_t::validate (
    const std::vector< float > & v )
```

5.316.3.8 validate() [6/9]

```
void MHAParser::range_var_t::validate (
    const std::vector< mha_complex_t > & v )
```

5.316.3.9 validate() [7/9]

```
void MHAParser::range_var_t::validate (
    const std::vector< std::vector< int > > & v )
```

5.316.3.10 validate() [8/9]

```
void MHAParser::range_var_t::validate (
    const std::vector< std::vector< float > > & v )
```

5.316.3.11 validate() [9/9]

```
void MHAParser::range_var_t::validate (
    const std::vector< std::vector< mha_complex_t > > & v )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.316.4 Member Data Documentation

5.316.4.1 low_limit `float MHAParser::range_var_t::low_limit [protected]`

Lower limit of range.

5.316.4.2 up_limit `float MHAParser::range_var_t::up_limit [protected]`

Upper limit of range.

5.316.4.3 low_incl `bool MHAParser::range_var_t::low_incl [protected]`

Lower limit is included (or excluded) in range.

5.316.4.4 up_incl `bool MHAParser::range_var_t::up_incl [protected]`

Upper limit is included (or excluded) in range.

5.316.4.5 check_low `bool MHAParser::range_var_t::check_low [protected]`

Check lower limit.

5.316.4.6 check_up `bool MHAParser::range_var_t::check_up [protected]`

Check upper limit.

© 2005-2021 HörTech gGmbH, Oldenburg

5.316.4.7 check_range `bool MHAParser::range_var_t::check_range [protected]`

Range checking is active.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.317 MHAParser::string_mon_t Class Reference

Monitor with string value.

Inheritance diagram for MHAParser::string_mon_t:

Public Member Functions

- **string_mon_t** (const std::string &hlp)
  Create a monitor variable for string values.

Public Attributes

- std::string **data**
  Data field.

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.317.1 Detailed Description

Monitor with string value.

5.317.2 Constructor & Destructor Documentation

5.317.2.1 string_mon_t() `MHAParser::string_mon_t::string_mon_t ( const std::string & hlp )`

Create a monitor variable for string values.

Parameters

hlp	A help text describing this monitor variable.

5.317.3 Member Function Documentation

5.317.3.1 query_val() `std::string MHAParser::string_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.317.3.2 query_type() `std::string MHAParser::string_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

© 2005-2021 HörTech gGmbH, Oldenburg

5.317.4 Member Data Documentation

5.317.4.1 data `std::string MHAParser::string_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.318 MHAParser::string_t Class Reference

Variable with a string value.

Inheritance diagram for MHAParser::string_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **string_t** (const std::string &, const std::string &)
  Constructor of a openMHA configuration variable for string values.

Public Attributes

- std::string **data**
  Data field.

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

Additional Inherited Members

5.318.1 Detailed Description

Variable with a string value.

5.318.2 Constructor & Destructor Documentation

5.318.2.1 string_t()

```
MHAParser::string_t::string_t (
    const std::string & h,
    const std::string & v )
```

Constructor of a openMHA configuration variable for string values.

Parameters

h	A help string describing the purpose of this variable.
v	The initial string value

© 2005-2021 HörTech gGmbH, Oldenburg

5.318.3 Member Function Documentation

5.318.3.1 op_setval() `std::string MHAParser::string_t::op_setval ( expression_t & x ) [protected], [virtual]`

Reimplemented from **MHAParser::variable_t** (p. 1119).

5.318.3.2 query_type() `std::string MHAParser::string_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.318.3.3 query_val() `std::string MHAParser::string_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.318.4 Member Data Documentation

5.318.4.1 data `std::string MHAParser::string_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.319 MHAParser::variable_t Class Reference

Base class for variable nodes.

Inheritance diagram for MHAParser::variable_t:

Public Member Functions

- **variable_t** (const std::string &)
- std::string **op_setval** (**expression_t** &)
- std::string **query_perm** (const std::string &)
- void **setlock** (const bool &)

  Lock a variable against write access.

Private Attributes

- bool **locked**

Additional Inherited Members

5.319.1 Detailed Description

Base class for variable nodes.

© 2005-2021 HörTech gGmbH, Oldenburg

5.319.2 Constructor & Destructor Documentation

5.319.2.1 variable_t() `MHAParser::variable_t::variable_t ( const std::string & h )`

5.319.3 Member Function Documentation

5.319.3.1 op_setval() `std::string MHAParser::variable_t::op_setval ( expression_t & x ) [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1037).

Reimplemented in **MHAParser::mcomplex_t** (p. 1082), **MHAParser::mfloat_t** (p. 1087), **MHAParser::mint_t** (p. 1099), **MHAParser::vcomplex_t** (p. 1123), **MHAParser::vfloat_t** (p. 1129), **MHAParser::vint_t** (p. 1133), **MHAParser::complex_t** (p. 1058), **MHAParser::float_t** (p. 1065), **MHAParser::int_t** (p. 1070), **MHAParser::bool_t** (p. 1049), **MHAParser::vstring_t** (p. 1137), **MHAParser::string_t** (p. 1117), and **MHAParser::kw_t** (p. 1077).

5.319.3.2 query_perm() `std::string MHAParser::variable_t::query_perm ( const std::string & ) [virtual]`

Reimplemented from **MHAParser::monitor_t** (p. 1101).

5.319.3.3 setlock() `void MHAParser::variable_t::setlock ( const bool & b )`

Lock a variable against write access.

Parameters

b	Lock state

© 2005-2021 HörTech gGmbH, Oldenburg

5.319.4 Member Data Documentation

5.319.4.1 locked `bool MHAParser::variable_t::locked [private]`

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.320 MHAParser::vcomplex_mon_t Class Reference

Monitor with vector of complex values.

Inheritance diagram for MHAParser::vcomplex_mon_t:

Public Member Functions

- **vcomplex_mon_t** (const std::string &hlp)
  Create a vector of complex monitor values.

Public Attributes

- std::vector< **mha_complex_t** > **data**
  Data field.

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.320.1 Detailed Description

Monitor with vector of complex values.

5.320.2 Constructor & Destructor Documentation

5.320.2.1 vcomplex_mon_t()

`MHAParser::vcomplex_mon_t::vcomplex_mon_t ( const std::string & hlp )`

Create a vector of complex monitor values.

Parameters

hlp	A help text describing this monitor variable.

5.320.3 Member Function Documentation

5.320.3.1 query_val()

`std::string MHAParser::vcomplex_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.320.3.2 query_type()

`std::string MHAParser::vcomplex_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

© 2005-2021 HörTech gGmbH, Oldenburg

5.320.4 Member Data Documentation

5.320.4.1 data `std::vector< mha_complex_t> MHAParser::vcomplex_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.321 MHAParser::vcomplex_t Class Reference

Vector variable with complex value.

Inheritance diagram for MHAParser::vcomplex_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **vcomplex_t** (const std::string &, const std::string &, const std::string &="")

Public Attributes

- std::vector< **mha_complex_t** > **data**
  Data field.

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

Additional Inherited Members

5.321.1 Detailed Description

Vector variable with complex value.

5.321.2 Constructor & Destructor Documentation

5.321.2.1 vcomplex_t()

```
MHAParser::vcomplex_t::vcomplex_t (
    const std::string & h,
    const std::string & v,
    const std::string & rg = "" )
```

5.321.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.321.3.1 op_setval()

`std::string MHAParser::vcomplex_t::op_setval ( expression_t & x ) [protected], [virtual]`

Reimplemented from **MHAParser::variable_t** (p. 1119).

5.321.3.2 query_type()

`std::string MHAParser::vcomplex_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.321.3.3 query_val()

`std::string MHAParser::vcomplex_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.321.4 Member Data Documentation

5.321.4.1 data

`std::vector< mha_complex_t> MHAParser::vcomplex_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.322 MHAParser::vfloat_mon_t Class Reference

Vector of floats monitor.

Inheritance diagram for MHAParser::vfloat_mon_t:

Public Member Functions

- **vfloat_mon_t** (const std::string &hlp)
  Create a vector of floating point monitor values.

Public Attributes

- std::vector< float > **data**
  Data field.

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.322.1 Detailed Description

Vector of floats monitor.

5.322.2 Constructor & Destructor Documentation

5.322.2.1 vfloat_mon_t() `MHAParser::vfloat_mon_t::vfloat_mon_t ( const std::string & hlp )`

Create a vector of floating point monitor values.

Parameters

hlp	A help text describing this monitor variable.

5.322.3 Member Function Documentation

5.322.3.1 query_val() `std::string MHAParser::vfloat_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.322.3.2 query_type() `std::string MHAParser::vfloat_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.322.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.322.4.1 data `std::vector<float> MHAParser::vfloat_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.323 MHAParser::vfloat_t Class Reference

Vector variable with float value.

Inheritance diagram for MHAParser::vfloat_t:

Public Member Functions

- **vfloat_t** (const std::string &, const std::string &, const std::string &="")
  Create a float vector parser variable.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- std::vector< float > **data**
  Data field.

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

Additional Inherited Members

5.323.1 Detailed Description

Vector variable with float value.

5.323.2 Constructor & Destructor Documentation

5.323.2.1 vfloat_t()

```
MHAParser::vfloat_t::vfloat_t (
    const std::string & h,
    const std::string & v,
    const std::string & rg = "" )
```

Create a float vector parser variable.

Parameters

h	A human-readable text describing the purpose of this configuration variable.
v	The initial value of the variable, as a string, in openMHA configuration language: (e.g. "[0 1 2.1 3]" for a vector), described in the "Multidimensional Variables" s2.1.3 section of the openMHA User Manual.
rg	The numeric range to enforce on all members of the vector.

-

© 2005-2021 HörTech gGmbH, Oldenburg

5.323.3 Member Function Documentation

5.323.3.1 op_setval() `std::string MHAParser::vfloat_t::op_setval ( expression_t & x ) [protected], [virtual]`

Reimplemented from **MHAParser::variable_t** (p. 1119).

5.323.3.2 query_type() `std::string MHAParser::vfloat_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.323.3.3 query_val() `std::string MHAParser::vfloat_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.323.4 Member Data Documentation

5.323.4.1 data `std::vector<float> MHAParser::vfloat_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.324 MHAParser::vint_mon_t Class Reference

Vector of ints monitor.

Inheritance diagram for MHAParser::vint_mon_t:

Public Member Functions

- **vint_mon_t** (const std::string &hlp)
  Create a vector of integer monitor values.

Public Attributes

- std::vector< int > **data**
  Data field.

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.324.1 Detailed Description

Vector of ints monitor.

© 2005-2021 HörTech gGmbH, Oldenburg

5.324.2 Constructor & Destructor Documentation

5.324.2.1 vint_mon_t() `MHAParser::vint_mon_t::vint_mon_t ( const std::string & hlp )`

Create a vector of integer monitor values.

Parameters

hlp	A help text describing this monitor variable.

5.324.3 Member Function Documentation

5.324.3.1 query_val() `std::string MHAParser::vint_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.324.3.2 query_type() `std::string MHAParser::vint_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.324.4 Member Data Documentation

5.324.4.1 data `std::vector<int> MHAParser::vint_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.325 MHAParser::vint_t Class Reference

Variable with vector<int> value.

Inheritance diagram for MHAParser::vint_t:

Public Member Functions

- **vint_t** (const std::string &, const std::string &, const std::string &="")
  Constructor.

Public Attributes

- std::vector< int > **data**
  Data field.

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.325.1 Detailed Description

Variable with vector<int> value.

5.325.2 Constructor & Destructor Documentation

5.325.2.1 vint_t()

```
MHAParser::vint_t::vint_t (
    const std::string & h,
    const std::string & v,
    const std::string & rg = "" )
```

Constructor.

Parameters

h	help string
v	initial value
rg	optional: range constraint for all elements

5.325.3 Member Function Documentation

5.325.3.1 op_setval()

```
std::string MHAParser::vint_t::op_setval (
    expression_t & x ) [protected], [virtual]
```

Reimplemented from **MHAParser::variable_t** (p. 1119).

5.325.3.2 query_type()

```
std::string MHAParser::vint_t::query_type (
    const std::string & ) [protected], [virtual]
```

Reimplemented from **MHAParser::base_t** (p. 1038).

© 2005-2021 HörTech gGmbH, Oldenburg

5.325.3.3 query_val() `std::string MHAParser::vint_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.325.4 Member Data Documentation

5.325.4.1 data `std::vector<int> MHAParser::vint_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.326 MHAParser::vstring_mon_t Class Reference

Vector of monitors with string value.

Inheritance diagram for MHAParser::vstring_mon_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **vstring_mon_t** (const std::string &hlp)
  Create a vector of string monitor values.

Public Attributes

- std::vector< std::string > **data**
  Data field.

Protected Member Functions

- std::string **query_val** (const std::string &)
- std::string **query_type** (const std::string &)

Additional Inherited Members

5.326.1 Detailed Description

Vector of monitors with string value.

5.326.2 Constructor & Destructor Documentation

5.326.2.1 vstring_mon_t() `MHAParser::vstring_mon_t::vstring_mon_t ( const std::string & hlp )`

Create a vector of string monitor values.

Parameters

hlp	A help text describing this monitor variable.

5.326.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.326.3.1 query_val() `std::string MHAParser::vstring_mon_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.326.3.2 query_type() `std::string MHAParser::vstring_mon_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.326.4 Member Data Documentation

5.326.4.1 data `std::vector<std::string> MHAParser::vstring_mon_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.327 MHAParser::vstring_t Class Reference

Vector variable with string values.

Inheritance diagram for MHAParser::vstring_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **vstring_t** (const std::string &, const std::string &)

Public Attributes

- std::vector< std::string > **data**
  Data field.

Protected Member Functions

- std::string **op_setval** (**expression_t** &)
- std::string **query_type** (const std::string &)
- std::string **query_val** (const std::string &)

Additional Inherited Members

5.327.1 Detailed Description

Vector variable with string values.

5.327.2 Constructor & Destructor Documentation

5.327.2.1 vstring_t()

```
MHAParser::vstring_t::vstring_t (
    const std::string & h,
    const std::string & v )
```

5.327.3 Member Function Documentation

5.327.3.1 op_setval()

```
std::string MHAParser::vstring_t::op_setval (
    expression_t & x ) [protected], [virtual]
```

Reimplemented from **MHAParser::variable_t** (p. 1119).

© 2005-2021 HörTech gGmbH, Oldenburg

5.327.3.2 query_type() `std::string MHAParser::vstring_t::query_type ( const std::string & ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.327.3.3 query_val() `std::string MHAParser::vstring_t::query_val ( const std::string & s ) [protected], [virtual]`

Reimplemented from **MHAParser::base_t** (p. 1038).

5.327.4 Member Data Documentation

5.327.4.1 data `std::vector<std::string> MHAParser::vstring_t::data`

Data field.

The documentation for this class was generated from the following files:

- **mha_parser.hh**
- **mha_parser.cpp**

5.328 MHAParser::window_t Class Reference

MHA configuration interface for a window function generator.

Inheritance diagram for MHAParser::window_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Types

- enum **wtype_t** {
  wnd_rect =0, **wnd_hann** =1, **wnd_hamming** =2, **wnd_blackman** =3,
  wnd_bartlett =4, **wnd_user** =5 }

Public Member Functions

- **window_t** (const std::string & **help**="Window type configuration.")
  Constructor to create parser class.
- **MHAWindow::base_t get_window** (unsigned int len) const
  Create a window instance, use default parameters.
- **MHAWindow::base_t get_window** (unsigned int len, float xmin) const
  Create a window instance.
- **MHAWindow::base_t get_window** (unsigned int len, float xmin, float xmax) const
  Create a window instance.
- **MHAWindow::base_t get_window** (unsigned int len, float xmin, float xmax, bool minincluded) const
  Create a window instance.
- **MHAWindow::base_t get_window** (unsigned int len, float xmin, float xmax, bool minincluded, bool maxincluded) const
  Create a window instance.
- **MHAParser::window_t::wtype_t get_type** () const
  Return currently selected window type.
- void **setlock** (bool b)

Private Attributes

- **MHAParser::kw_t wtype**
- **MHAParser::vfloat_t user**

Additional Inherited Members

5.328.1 Detailed Description

MHA configuration interface for a window function generator.

This class implements a configuration interface (sub-parser) for window type selection and user-defined window type. It provides member functions to generate an instance of **MHA↩ Window::base_t** (p. 1291) based on the values provided by the configuration interface.

The configuration interface is derived from **MHAParser::parser_t** (p. 1102) and can thus be inserted into the configuration tree using the **insert_item()** (p. 1104) method of the parent parser.

If one of the pre-defined window types is used, then the window is generated using the **MHA↩ Window::fun_t** (p. 1295) class constructor; for the user-defined type the values from the "user" variable are copied.

© 2005-2021 HörTech gGmbH, Oldenburg

5.328.2 Member Enumeration Documentation

5.328.2.1 wtype_t `enum` **`MHAParser::window_t::wtype_t`**

Enumerator

wnd_rect	
wnd_hann	
wnd_hamming	
wnd_blackman	
wnd_bartlett	
wnd_user	

5.328.3 Constructor & Destructor Documentation

5.328.3.1 window_t() `MHAParser::window_t::window_t (`
`const std::string & help = "Window type configuration." )`

Constructor to create parser class.

5.328.4 Member Function Documentation

5.328.4.1 get_window() `[1/5]` **`MHAWindow::base_t`** `MHAParser::window_t::get_window (`
`unsigned int len ) const`

Create a window instance, use default parameters.

© 2005-2021 HörTech gGmbH, Oldenburg

5.328.4.2 get_window() [2/5]

```
MHAWindow::base_t MHAParser::window_t::get_window (
    unsigned int len,
    float xmin ) const
```

Create a window instance.

5.328.4.3 get_window() [3/5]

```
MHAWindow::base_t MHAParser::window_t::get_window (
    unsigned int len,
    float xmin,
    float xmax ) const
```

Create a window instance.

5.328.4.4 get_window() [4/5]

```
MHAWindow::base_t MHAParser::window_t::get_window (
    unsigned int len,
    float xmin,
    float xmax,
    bool minincluded ) const
```

Create a window instance.

5.328.4.5 get_window() [5/5]

```
MHAWindow::base_t MHAParser::window_t::get_window (
    unsigned int len,
    float xmin,
    float xmax,
    bool minincluded,
    bool maxincluded ) const
```

Create a window instance.

5.328.4.6 get_type()

```
MHAParser::window_t::wtype_t MHAParser::window_t::get_type (
) const
```

Return currently selected window type.

© 2005-2021 HörTech gGmbH, Oldenburg

5.328.4.7 setlock() `void MHAParser::window_t::setlock ( bool b ) [inline]`

5.328.5 Member Data Documentation

5.328.5.1 wtype `MHAParser::kw_t MHAParser::window_t::wtype [private]`

5.328.5.2 user `MHAParser::vfloat_t MHAParser::window_t::user [private]`

The documentation for this class was generated from the following files:

- **mha_windowparser.h**
- **mha_windowparser.cpp**

5.329 mhaplug_cfg_t Class Reference

Inheritance diagram for mhaplug_cfg_t:

Public Member Functions

- **mhaplug_cfg_t** (**algo_comm_t** iac, const std::string & **libname**, bool use_own_ac)
- ~**mhaplug_cfg_t** () throw ()

Additional Inherited Members

5.329.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.329.1.1 mhaplug_cfg_t()

```
mhaplug_cfg_t::mhaplug_cfg_t (
    algo_comm_t iac,
    const std::string & libname,
    bool use_own_ac )
```

5.329.1.2 ~mhaplug_cfg_t()

```
mhaplug_cfg_t::~mhaplug_cfg_t ( ) throw ( )   [inline]
```

The documentation for this class was generated from the following file:

- **altplugs.cpp**

5.330 MHAPlugin::cfg_node_t< runtime_cfg_t > Class Template Reference

A node class for storing MHA plugin runtime configurations as a singly linked list, where the nodes store besides the usual "next" and "data" pointers also a flag that indicates whether this node can be deleted.

Public Member Functions

- **cfg_node_t** (runtime_cfg_t *runtime_cfg)

  Constructor for a singly linked list node.
- ~**cfg_node_t** ()

  Destructor of the singly linked list node.

Public Attributes

- std::atomic< **cfg_node_t**< runtime_cfg_t > * > **next**

  A pointer to the next node in the singly linked list.
- std::atomic< bool > **not_in_use**

  Initially this data member is set to false by the constructor.
- runtime_cfg_t * **data**

  A native pointer to the runtime configuration managed by this node.

© 2005-2021 HörTech gGmbH, Oldenburg

5.330.1 Detailed Description

template<class runtime_cfg_t>
class MHAPlugin::cfg_node_t< runtime_cfg_t >

A node class for storing MHA plugin runtime configurations as a singly linked list, where the nodes store besides the usual "next" and "data" pointers also a flag that indicates whether this node can be deleted.

The singly linked list is designed for a single producer thread and a single consumer thread, where the producer is also responsible for destroying objects when they are no longer needed because the consumer is the signal processing thread that cannot afford memory allocation or deallocation operations.

5.330.2 Constructor & Destructor Documentation

5.330.2.1 cfg_node_t() template<class runtime_cfg_t >

```
MHAPlugin::cfg_node_t< runtime_cfg_t >::  cfg_node_t (
            runtime_cfg_t * runtime_cfg )  [explicit]
```

Constructor for a singly linked list node.

Parameters

runtime_cfg	Pointer to a runtime configuration object that was just created on the heap. The newly constructed **cfg_node_t** (p. 1143) object takes over object ownership of the pointed-to runtime configuration and will call delete on it in its destructor.

5.330.2.2 ~cfg_node_t() template<class runtime_cfg_t >

```
MHAPlugin::cfg_node_t< runtime_cfg_t >::~ cfg_node_t
```

Destructor of the singly linked list node.

Will also delete the pointed-to data object (the runtime configuration)

5.330.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.330.3.1 next

`template<class runtime_cfg_t >`
`std::atomic< cfg_node_t<runtime_cfg_t>*> MHAPlugin::cfg_node_t< runtime_cfg_t >::next`

A pointer to the next node in the singly linked list.

On construction, this will be a NULL pointer. New objects can be appended to the singly linked list by writing the address to the next node into this data member. Since this pointer is std::atomic, writing to it is a release operation which means all threads seeing the new value of the next pointer will also see all other writes to memory that the thread doing this write has performed, which includes anything the constructor of the new runtime config has written and the assignment of the address of the new runtime config to the data pointer of the next node. This prevents making half-constructed runtime configuration objects visible to the consumer thread.

5.330.3.2 not_in_use

`template<class runtime_cfg_t >`
`std::atomic<bool> MHAPlugin::cfg_node_t< runtime_cfg_t >::not_in_use`

Initially this data member is set to false by the constructor.

It is set to true by the consumer thread when it no longer needs the run time configuration pointed to by this node's data member. This bool is atomic because this node and the runtime object it points to can be deleted by the configuration thread as soon as value true is stored in this bool, which happens right after the processing thread acquires a pointer to the next node in the singly linked list. The atomic ensures all threads agree on this "right after" ordering.

5.330.3.3 data

`template<class runtime_cfg_t >`
`runtime_cfg_t* MHAPlugin::cfg_node_t< runtime_cfg_t >::data`

A native pointer to the runtime configuration managed by this node.

The runtime configuration lives on the heap and is owned by this node. It is deleted in this node's destructor. The runtime configuration object must be created by client code with operator new before ownership is transferred to this node by passing a pointer to it as the constructor's parameter. This pointer does not need to be atomic, memory access ordering is ensured by the atomic next pointer and placing accesses to "data" in the correct places relative to accesses to "next".

The documentation for this class was generated from the following file:

- **mha_plugin.hh**

© 2005-2021 HörTech gGmbH, Oldenburg

5.331 MHAPlugin::config_t< runtime_cfg_t > Class Template Reference

Template class for thread safe configuration.

Inheritance diagram for MHAPlugin::config_t< runtime_cfg_t >:

Public Member Functions

- **config_t** ()
- ~**config_t** ()

Protected Member Functions

- runtime_cfg_t * **poll_config** ()

  Receive the latest run time configuration.
- runtime_cfg_t * **peek_config** () const

  Receive the latest run time configuration without changing the configuration pointer.
- void **push_config** (runtime_cfg_t *ncfg)

  Push a new run time configuration into the configuration fifo.
- void **cleanup_unused_cfg** ()

  *To be called by the **push_config()*** (p. 1150) *for housekeeping.*
- void **remove_all_cfg** ()

  To be called on Plugin destruction, will delete all runtime configuration list nodes and objects regardless of their in_use flag.

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Attributes

- runtime_cfg_t * **cfg**
  Pointer to the runtime configuration currently used by the signal processing thread.

Private Attributes

- std::atomic< **MHAPlugin::cfg_node_t**< runtime_cfg_t > * > **cfg_root**
  Start of a singly linked list of runtime configuration objects.
- **MHAPlugin::cfg_node_t**< runtime_cfg_t > * **cfg_node_current**
  Pointer to the currently used plugin runtime configurations.

5.331.1 Detailed Description

template<class runtime_cfg_t>
class MHAPlugin::config_t< runtime_cfg_t >

Template class for thread safe configuration.

This template class provides a mechanism for the handling of thread safe configuration which is required for run time configuration changes of the openMHA plugins.

The template parameter runtime_cfg_t is the run time configuration class of the openMHA plugin. The constructor of that class should transform the **MHAParser** (p. 122) variables into derived runtime configuration. The constructor should fail if the configuration is invalid by any reason.

A new runtime configuration is provided by the function **push_config()** (p. 1150). In the processing thread, the actual configuration can be received by a call of **poll_config()** (p. 1149).

Figure 5 Schematic drawing of runtime configuration update: configuration updated, but not used yet.

© 2005-2021 HörTech gGmbH, Oldenburg

Figure 6 Schematic drawing of runtime configuration update: configuration in use.

To ensure lock-free thread safety, we use C++ atomics and rely on the C++ memory model. We only use store-release and load-acquire operations by using C++ atomics with the default memory ordering. The semantics of these are:

The store-release operation atomically writes to an atomic variable, while the load-acquire operation atomically reads from an atomic variable.

The C++ memory model guarantees that all previous writes to memory performed by the thread doing the store-release are visible to other threads when they see the new value in the shared atomic variable when that value is read by the other thread with a load-acquire operation.

An important precondition of this synchronization scheme is that there is only ever one audio thread and one configuration thread per plugin, i.e. there is only one thread doing the push_↩config and one thread doing the poll_config for each instance of **config_t** (p. 1146).

For more details on atomics, refer to the C++11 or later documentation, or to these conference talks by Sutter:

- Atomic Weapons, 2012
- Lock-Free Programming, 2014

5.331.2 Constructor & Destructor Documentation

5.331.2.1 config_t() `template<class runtime_cfg_t >`

MHAPlugin::config_t`< runtime_cfg_t >::` **config_t**

5.331.2.2 ~config_t() template<class runtime_cfg_t >

MHAPlugin::config_t< runtime_cfg_t >::~ **config_t**

5.331.3 Member Function Documentation

5.331.3.1 poll_config() template<class runtime_cfg_t >

runtime_cfg_t * **MHAPlugin::config_t**< runtime_cfg_t >::poll_config [protected]

Receive the latest run time configuration.

This function stores the latest run time configuration into the protected class member variable 'cfg'. If no configuration exists, then an exception will be thrown. If no changes occured, then the value of 'cfg' will be untouched.

This function should be only called from the *processing* thread.

Should be called at the start of each process() callback to get the latest runtime configuration.

When this function finds newer run time configurations, it returns the newest and ensures the older run time configurations have their not_in_use flag set to true.

Returns

Pointer to the latest runtime configuration object (same pointer as stored by this function in data member 'cfg').

Exceptions

MHA_Error *(p. 764)*	if the resulting runtime configuration is NULL. This usually means that no push_config has occured.

5.331.3.2 peek_config() template<class runtime_cfg_t >

runtime_cfg_t * **MHAPlugin::config_t**< runtime_cfg_t >::peek_config [protected]

Receive the latest run time configuration without changing the configuration pointer.

This function retrieves the latest run time configuration. Returns a pointer to the latest runtime configuration without updating the data member cfg. For use in the configuration thread when

© 2005-2021 HörTech gGmbH, Oldenburg

creation of a new runtime configuration object needs access to the previously created runtime configuration object. Should normally not be used because it introduces synchronization requirements between configuration thread and signal processing thread.

5.331.3.3 push_config()

```
template<class runtime_cfg_t >
void MHAPlugin::config_t< runtime_cfg_t >::push_config (
        runtime_cfg_t * ncfg ) [protected]
```

Push a new run time configuration into the configuration fifo.

Should be called only by the configuration thread when a new runtime configuration object has been constructed in response to configuration changes, or during execution of the prepare() method to ensure that there is a valid runtime configuration for the signal processing which can start after prepare() returns.

For housekeeping, this method will also delete any runtime configuration objects that have previously been passed to **push_config()** (p. 1150) if they are no longer needed.

Parameters

ncfg	A pointer to the new runtime configuration object. This object must have been created on the heap by the configuration thread with operator new. By passing the pointer to this method, client code gives up ownership. The object will be deleted in a future invocation of push_config, or on destruction of this **config_t** (p. 1146) instance.

5.331.3.4 cleanup_unused_cfg()

```
template<class runtime_cfg_t >
void MHAPlugin::config_t< runtime_cfg_t >::cleanup_unused_cfg [protected]
```

To be called by the **push_config()** (p. 1150) for housekeeping.

Will delete any no longer used runtime configuration objects.

5.331.3.5 remove_all_cfg()

```
template<class runtime_cfg_t >
void MHAPlugin::config_t< runtime_cfg_t >::remove_all_cfg [protected]
```

To be called on Plugin destruction, will delete all runtime configuration list nodes and objects regardless of their in_use flag.

5.331.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.331.4.1 cfg `template<class runtime_cfg_t >`

`runtime_cfg_t* MHAPlugin::config_t< runtime_cfg_t >::cfg [protected]`

Pointer to the runtime configuration currently used by the signal processing thread.

Should be used to access the current runtime configuration during signal processing. This pointer is updated as a side effect of calling **poll_config()** (p. 1149) on this object.

5.331.4.2 cfg_root `template<class runtime_cfg_t >`

`std::atomic< MHAPlugin::cfg_node_t<runtime_cfg_t> *> MHAPlugin::config_t< runtime_cfg_t >::cfg_root [private]`

Start of a singly linked list of runtime configuration objects.

cfg_root points to the oldest still existing node of that list. After object creation this pointer is updated by the configuration thread and then read by the signal processing thread. To ensure proper order of memory accesses for this transfer between threads, it needs to be atomic, this ensures that the start of the singly linked list of runtime configurations will be properly visible to the signal processing on startup.

5.331.4.3 cfg_node_current `template<class runtime_cfg_t >`

`MHAPlugin::cfg_node_t<runtime_cfg_t>* MHAPlugin::config_t< runtime_cfg_t >::cfg_node_current [private]`

Pointer to the currently used plugin runtime configurations.

Used as a hint for poll_config where to start looking for the newest node. This optimization allows poll_config to scale better with the number of nodes not yet cleaned up by push_config. Does not need to be atomic because it is only used within the signal processing thread.

The documentation for this class was generated from the following file:

- **mha_plugin.hh**

5.332 MHAPlugin::plugin_t< runtime_cfg_t > Class Template Reference

The template class for C++ openMHA plugins.

Inheritance diagram for MHAPlugin::plugin_t< runtime_cfg_t >:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **plugin_t** (const std::string &, const **algo_comm_t** &)
  Constructor of plugin template.
- virtual ~**plugin_t** ()
- virtual void **prepare** (**mhaconfig_t** &)=0
- virtual void **release** ()
- void **prepare_** (**mhaconfig_t** &)
- void **release_** ()
- bool **is_prepared** () const
  Flag, if the prepare method is successfully called (or currently evaluated)
- **mhaconfig_t input_cfg** () const
  Current input channel configuration.
- **mhaconfig_t output_cfg** () const
  Current output channel configuration.

Protected Attributes

- **mhaconfig_t tftype**
  Member for storage of plugin interface configuration.
- **algo_comm_t ac**
  AC handle of the chain.

Private Attributes

- bool **is_prepared_**
- **mhaconfig_t input_cfg_**
- **mhaconfig_t output_cfg_**
- **MHAParser::mhaconfig_mon_t mhaconfig_in**
- **MHAParser::mhaconfig_mon_t mhaconfig_out**

Additional Inherited Members

5.332.1 Detailed Description

template<class runtime_cfg_t>
class MHAPlugin::plugin_t< runtime_cfg_t >

The template class for C++ openMHA plugins.

© 2005-2021 HörTech gGmbH, Oldenburg

Template Parameters

runtime_cfg_t	run-time configuration.

This template class provides thread safe configuration handling and standard methods to be compatible to the C++ openMHA plugin wrapper macro **MHAPLUGIN_CALLBACKS** (p. 1625).

The template parameter runtime_cfg_t should be the runtime configuration of the plugin.

See **MHAPlugin::config_t** (p. 1146) for details on the thread safe communication update mechanism.

5.332.2 Constructor & Destructor Documentation

5.332.2.1 plugin_t() `template<class runtime_cfg_t >`

```
MHAPlugin::plugin_t< runtime_cfg_t >::  plugin_t (
            const std::string & help,
            const  algo_comm_t & iac )
```

Constructor of plugin template.

Parameters

help	Help comment to provide some general information about the plugin.
iac	AC space handle (will be stored into the member variable ac).

5.332.2.2 ~plugin_t() `template<class runtime_cfg_t >`

```
MHAPlugin::plugin_t< runtime_cfg_t >::~ plugin_t  [virtual]
```

5.332.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.332.3.1 prepare() template<class runtime_cfg_t >

```
virtual void MHAPlugin::plugin_t< runtime_cfg_t >::prepare (
            mhaconfig_t & ) [pure virtual]
```

Implemented in **calibrator_t** (p. 339), **dc_simple::dc_if_t** (p. 393), **dc::dc_if_t** (p. 381), **fftfbpow::fftfbpow_interface_t** (p. 495), **save_spec_t** (p. 1423), **save_wave_t** (p. 1425), **wave2spec_if_t** (p. 1494), **windnoise::if_t** (p. 1515), **attenuate20_t** (p. 314), **matlab_wrapper::matlab_wrapper_t** (p. 700), **example3_t** (p. 475), **example4_t** (p. 479), **droptect_t** (p. 441), **example1_t** (p. 468), **example2_t** (p. 471), **mconv::MConv** (p. 719), **gtfb_simple_t** (p. 572), **plingploing::if_t** (p. 1342), **plugins::hoertech::acrec::acrec_t** (p. 1380), **wavrec_t** (p. 1504), **altconfig_t** (p. 297), **bbcalib_interface_t** (p. 334), **gtfb_simd_t** (p. 563), **addsndfile::addsndfile_if_t** (p. 251), **adm_if_t** (p. 272), **testplugin::if_t** (p. 1487), **analysispath_if_t** (p. 311), **osc2ac_t** (p. 1322), **dbasync_native::db_if_t** (p. 374), **ac2lsl::ac2lsl_t** (p. 163), **audiometerbackend::audiometer_if_t** (p. 316), **noise_psd_estimator::noise_psd_estimator_if_t** (p. 1313), **lsl2ac::lsl2ac_t** (p. 683), **fftfilter::interface_t** (p. 502), **smooth_cepstrum::smooth_cepstrum_if_t** (p. 1440), **rohBeam::rohBeam** (p. 1401), **multibandcompressor::interface_t** (p. 1305), **combc_if_t** (p. 356), **gtfb_analyzer::gtfb_analyzer_t** (p. 555), **rmslevel::rmslevel_if_t** (p. 1393), **coherence::cohflt_if_t** (p. 348), **smoothgains_bridge::overlapadd_if_t** (p. 1454), **cpuload::cpuload_if_t** (p. 367), **noise_t** (p. 1319), **MHAPlugin_Resampling::resampling_if_t** (p. 1158), **shadowfilter_end::shadowfilter_end_t** (p. 1432), **ac2wave_if_t** (p. 187), **adaptive_feedback_canceller** (p. 242), **nlms_t** (p. 1310), **fshift_hilbert::frequency_translator_t** (p. 517), **spec2wave_if_t** (p. 1467), **level_matching::level_matching_t** (p. 656), **fader_wave::fader_wave_if_t** (p. 490), **overlapadd::overlapadd_if_t** (p. 1332), **acsave::acsave_t** (p. 220), **plugin_interface_t** (p. 1355), **mhachain::chain_base_t** (p. 844), **complex_scale_channel_t** (p. 362), **example6_t** (p. 483), **doasvm_feature_extraction** (p. 428), **shadowfilter_begin::shadowfilter_begin_t** (p. 1428), **fshift::fshift_t** (p. 513), **lpc_bl_predictor** (p. 666), **lpc_burglattice** (p. 672), **acPooling_wave** (p. 212), **delaysum::delaysum_wave_if_t** (p. 414), **sine_t** (p. 1436), **steerbf** (p. 1476), **db_if_t** (p. 370), **fader_if_t** (p. 488), **lpc** (p. 662), **acConcat_wave** (p. 200), **gain::gain_if_t** (p. 535), **acSteer** (p. 229), **acTransform_wave** (p. 235), **doasvm_classification** (p. 423), **fftfilterbank::fftfb_interface_t** (p. 505), **route::interface_t** (p. 1413), **matrixmixer::matmix_t** (p. 715), **equalize::freqgains_t** (p. 465), **altplugs_t** (p. 301), **delaysum_spec::delaysum_spec_if_t** (p. 419), **softclip_t** (p. 1459), **ac2osc_t** (p. 182), **ac_proc::interface_t** (p. 197), **gsc_adaptive_stage::gsc_adaptive_stage_if** (p. 547), **acmon::acmon_t** (p. 208), **example7_t** (p. 486), **dropgen_t** (p. 438), **identity_t** (p. 577), **levelmeter_t** (p. 659), **delay::interface_t** (p. 411), **ds_t** (p. 445), and **us_t** (p. 1491).

5.332.3.2 release() template<class runtime_cfg_t >

```
void MHAPlugin::plugin_t< runtime_cfg_t >::release [virtual]
```

Reimplemented in **windnoise::if_t** (p. 1516), **attenuate20_t** (p. 314), **smooth_cepstrum::smooth_cepstrum_if_t** (p. 1440), **rohBeam::rohBeam** (p. 1401), **adaptive_feedback_canceller** (p. 242), **level_matching::level_matching_t** (p. 656), **doasvm_feature_extraction** (p. 429), **fshift::fshift_t** (p. 514), **example3_t** (p. 475), **example4_t** (p. 480), **lpc_bl_predictor** (p. 666), **lpc_burglattice** (p. 672), **acPooling_wave** (p. 213), **steerbf** (p. 1477), **droptect_t** (p. 442), **lpc** (p. 663), **acConcat_wave** (p. 201), **acSteer** (p. 230), **acTransform_wave** (p. 236), **example2_t** (p. 472), **doasvm_classification** (p. 423),

© 2005-2021 HörTech gGmbH, Oldenburg

example1_t (p. 468), **gsc_adaptive_stage::gsc_adaptive_stage_if** (p. 547), **example7_t** (p. 486), **bbcalib_interface_t** (p. 335), **calibrator_t** (p. 339), **gtfb_simple_t** (p. 572), **addsndfile::addsndfile_if_t** (p. 252), **adm_if_t** (p. 273), **analysispath_if_t** (p. 311), **ac2lsl::ac2lsl_t** (p. 164), **osc2ac_t** (p. 1322), **dbasync_native::db_if_t** (p. 375), **matlab_wrapper::matlab_wrapper_t** (p. 701), **dc_simple::dc_if_t** (p. 394), **lsl2ac::lsl2ac_t** (p. 683), **multibandcompressor::interface_t** (p. 1305), **wave2spec_if_t** (p. 1494), **gtfb_analyzer::gtfb_analyzer_t** (p. 555), **plugins::hoertech::acrec::acrec_t** (p. 1380), **coherence::cohflt_if_t** (p. 348), **smoothgains_bridge::overlapadd_if_t** (p. 1454), **MHAPlugin_Resampling::resampling_if_t** (p. 1159), **ac2wave_if_t** (p. 187), **nlms_t** (p. 1310), **fshift_hilbert::frequency_translator_t** (p. 517), **spec2wave_if_t** (p. 1467), **fader_wave::fader_wave_if_t** (p. 490), **overlapadd::overlapadd_if_t** (p. 1332), **acsave::acsave_t** (p. 220), **mhachain::chain_base_t** (p. 844), **delaysum::delaysum_wave_if_t** (p. 414), **wavrec_t** (p. 1504), **altconfig_t** (p. 297), **db_if_t** (p. 370), **gain::gain_if_t** (p. 535), **fftfilterbank::fftfb_interface_t** (p. 505), **route::interface_t** (p. 1414), **ac2osc_t** (p. 182), **altplugs_t** (p. 301), **mconv::MConv** (p. 719), **ac_proc::interface_t** (p. 197), **acmon::acmon_t** (p. 208), **dropgen_t** (p. 438), **identity_t** (p. 577), **ds_t** (p. 445), and **us_t** (p. 1491).

5.332.3.3 prepare_()

```
template<class runtime_cfg_t >
void MHAPlugin::plugin_t< runtime_cfg_t >::prepare_ (
            mhaconfig_t & cf )
```

5.332.3.4 release_()

```
template<class runtime_cfg_t >
void MHAPlugin::plugin_t< runtime_cfg_t >::release_
```

5.332.3.5 is_prepared()

```
template<class runtime_cfg_t >
bool MHAPlugin::plugin_t< runtime_cfg_t >::is_prepared ( ) const [inline]
```

Flag, if the prepare method is successfully called (or currently evaluated)

5.332.3.6 input_cfg()

```
template<class runtime_cfg_t >
mhaconfig_t MHAPlugin::plugin_t< runtime_cfg_t >::input_cfg ( ) const [inline]
```

Current input channel configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

5.332.3.7 output_cfg() template<class runtime_cfg_t >

mhaconfig_t MHAPlugin::plugin_t< runtime_cfg_t >::output_cfg ( ) const [inline]

Current output channel configuration.

5.332.4 Member Data Documentation

5.332.4.1 tftype template<class runtime_cfg_t >

mhaconfig_t MHAPlugin::plugin_t< runtime_cfg_t >::tftype [protected]

Member for storage of plugin interface configuration.

This member is defined for convenience of the developer. Typically, the actual contents of **mhaconfig_t** (p. 851) are stored in this member in the **prepare()** (p. 1153) method.

Note

This member is likely to be removed in later versions, use **input_cfg()** (p. 1155) and **output_cfg()** (p. 1155) instead.

5.332.4.2 ac template<class runtime_cfg_t >

algo_comm_t MHAPlugin::plugin_t< runtime_cfg_t >::ac [protected]

AC handle of the chain.

This variable is initialized in the constructor and can be used by derived plugins to access the AC space. Its contents should not be modified.

5.332.4.3 is_prepared_ template<class runtime_cfg_t >

bool **MHAPlugin::plugin_t**< runtime_cfg_t >::is_prepared_ [private]

5.332.4.4 input_cfg_ template<class runtime_cfg_t >

mhaconfig_t MHAPlugin::plugin_t< runtime_cfg_t >::input_cfg_ [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.332.4.5 output_cfg_ template<class runtime_cfg_t >

mhaconfig_t MHAPlugin::plugin_t< runtime_cfg_t >::output_cfg_ [private]

5.332.4.6 mhaconfig_in template<class runtime_cfg_t >

MHAParser::mhaconfig_mon_t MHAPlugin::plugin_t< runtime_cfg_t >::mhaconfig_in [private]

5.332.4.7 mhaconfig_out template<class runtime_cfg_t >

MHAParser::mhaconfig_mon_t MHAPlugin::plugin_t< runtime_cfg_t >::mhaconfig_out [private]

The documentation for this class was generated from the following file:

- **mha_plugin.hh**

5.333 MHAPlugin_Resampling::resampling_if_t Class Reference

Inheritance diagram for MHAPlugin_Resampling::resampling_if_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **resampling_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

Private Attributes

- **MHAParser::float_t srate**
- **MHAParser::int_t fragsize**
- **MHAParser::float_t nyquist_ratio**
- **MHAParser::float_t irslen_outer2inner**
- **MHAParser::float_t irslen_inner2outer**
- **MHAParser::mhapluginloader_t plugloader**
- std::string **algo**

Additional Inherited Members

5.333.1 Constructor & Destructor Documentation

5.333.1.1 resampling_if_t()

```
MHAPlugin_Resampling::resampling_if_t::resampling_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.333.2 Member Function Documentation

5.333.2.1 process()

```
mha_wave_t * MHAPlugin_Resampling::resampling_if_t::process (
    mha_wave_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.333.2.2 prepare()

`void MHAPlugin_Resampling::resampling_if_t::prepare ( mhaconfig_t & conf ) [virtual]`

Implements **MHAPlugin::plugin_t< resampling_t >** (p. 1153).

5.333.2.3 release()

`void MHAPlugin_Resampling::resampling_if_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t< resampling_t >** (p. 1154).

5.333.3 Member Data Documentation

5.333.3.1 srate

`MHAParser::float_t MHAPlugin_Resampling::resampling_if_t::srate [private]`

5.333.3.2 fragsize

`MHAParser::int_t MHAPlugin_Resampling::resampling_if_t::fragsize [private]`

5.333.3.3 nyquist_ratio

`MHAParser::float_t MHAPlugin_Resampling::resampling_if_t↩ ::nyquist_ratio [private]`

5.333.3.4 irslen_outer2inner

`MHAParser::float_t MHAPlugin_Resampling::resampling↩ _if_t::irslen_outer2inner [private]`

5.333.3.5 irslen_inner2outer

`MHAParser::float_t MHAPlugin_Resampling::resampling↩ _if_t::irslen_inner2outer [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.333.3.6 plugloader `MHAParser::mhapluginloader_t MHAPlugin_Resampling::resampling_if_t::plugloader [private]`

5.333.3.7 algo `std::string MHAPlugin_Resampling::resampling_if_t::algo [private]`

The documentation for this class was generated from the following file:

- **resampling.cpp**

5.334 MHAPlugin_Resampling::resampling_t Class Reference

Public Member Functions

- **resampling_t** (unsigned int **outer_fragsize**, float **outer_srate**, unsigned int **inner_fragsize**, float inner_scrate, unsigned int nch_in, float filter_length_in, unsigned int nch_out, float filter_length_out, float nyquist_ratio, **MHAParser::mhapluginloader_t** &plug)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Private Attributes

- unsigned **outer_fragsize**
- unsigned **inner_fragsize**
- float **outer_srate**
- float **inner_srate**
- unsigned **nchannels_in**
- unsigned **nchannels_out**
- **MHAFilter::blockprocessing_polyphase_resampling_t outer2inner_resampling**
- **MHAFilter::blockprocessing_polyphase_resampling_t inner2outer_resampling**
- **MHAParser::mhapluginloader_t** & **plugloader**
- **MHASignal::waveform_t inner_signal**
- **MHASignal::waveform_t output_signal**

5.334.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.334.1.1 resampling_t()

```
MHAPlugin_Resampling::resampling_t::resampling_t (
    unsigned int outer_fragsize,
    float outer_srate,
    unsigned int inner_fragsize,
    float inner_scrate,
    unsigned int nch_in,
    float filter_length_in,
    unsigned int nch_out,
    float filter_length_out,
    float nyquist_ratio,
    MHAParser::mhapluginloader_t & plug )
```

5.334.2 Member Function Documentation

5.334.2.1 process()

```
mha_wave_t * MHAPlugin_Resampling::resampling_t::process (
    mha_wave_t * s )
```

5.334.3 Member Data Documentation

5.334.3.1 outer_fragsize

```
unsigned MHAPlugin_Resampling::resampling_t::outer_fragsize [private]
```

5.334.3.2 inner_fragsize

```
unsigned MHAPlugin_Resampling::resampling_t::inner_fragsize [private]
```

5.334.3.3 outer_srate

```
float MHAPlugin_Resampling::resampling_t::outer_srate [private]
```

5.334.3.4 inner_srate

```
float MHAPlugin_Resampling::resampling_t::inner_srate [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.334.3.5 nchannels_in `unsigned MHAPlugin_Resampling::resampling_t::nchannels_in [private]`

5.334.3.6 nchannels_out `unsigned MHAPlugin_Resampling::resampling_t::nchannels_↩ out [private]`

5.334.3.7 outer2inner_resampling **`MHAFilter::blockprocessing_polyphase_resampling↩ _t`** `MHAPlugin_Resampling::resampling_t::outer2inner_resampling [private]`

5.334.3.8 inner2outer_resampling **`MHAFilter::blockprocessing_polyphase_resampling↩ _t`** `MHAPlugin_Resampling::resampling_t::inner2outer_resampling [private]`

5.334.3.9 plugloader **`MHAParser::mhapluginloader_t`**`& MHAPlugin_Resampling::resampling↩ _t::plugloader [private]`

5.334.3.10 inner_signal **`MHASignal::waveform_t`** `MHAPlugin_Resampling::resampling_t↩ ::inner_signal [private]`

5.334.3.11 output_signal **`MHASignal::waveform_t`** `MHAPlugin_Resampling::resampling↩ _t::output_signal [private]`

The documentation for this class was generated from the following file:

- **resampling.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.335 MHAPlugin_Split::domain_handler_t Class Reference

Handles domain-specific partial input and output signal.

Inheritance diagram for MHAPlugin_Split::domain_handler_t:

Public Member Functions

- void **set_input_domain** (const **mhaconfig_t** &settings_in)

  Set parameters of input signal.
- void **set_output_domain** (const **mhaconfig_t** &settings_out)

  Set output signal parameters.
- void **deallocate_domains** ()

  Deallocate domain indicators and signal holders.
- **domain_handler_t** (const **mhaconfig_t** &settings_in, const **mhaconfig_t** &settings↩ _out, **PluginLoader::fourway_processor_t** ∗ **processor**)

  Construct a new domain handler once the domains and dimensions of input and output signal of one of the child plugins of split are known.
- virtual ∼**domain_handler_t** ()

  Deallocation of signal holders.
- unsigned **put_signal** (**mha_wave_t** ∗s_in, unsigned start_channel)

  Store the relevant channels from the input signal for processing.
- unsigned **put_signal** (**mha_spec_t** ∗s_in, unsigned start_channel)

  Store the relevant channels from the input signal for processing.
- unsigned **get_signal** (**MHASignal::waveform_t** ∗s_out, unsigned start_channel)

  Store all partial signal output channels in the combined waveform signal with the given channel offset.
- unsigned **get_signal** (**MHASignal::spectrum_t** ∗s_out, unsigned start_channel)

  Store all partial signal output channels in the combined spectrum signal with the given channel offset.
- void **process** ()

  Call the processing method of the processor with configured input/output signal domains.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **MHASignal::waveform_t** $*$ **wave_in**
  Partial wave input signal.
- **mha_wave_t** $**$ **wave_out**
  Partial wave output signal.
- **MHASignal::spectrum_t** $*$ **spec_in**
  Partial spec input signal.
- **mha_spec_t** $**$ **spec_out**
  Partial spec input signal.
- **PluginLoader::fourway_processor_t** $*$ **processor**
  The domain-specific signal processing methods are implemented here.

Private Member Functions

- **domain_handler_t** (const **domain_handler_t** &)
  Disallow copy constructor.
- **domain_handler_t** & **operator=** (const **domain_handler_t** &)
  Disallow assignment operator.

5.335.1 Detailed Description

Handles domain-specific partial input and output signal.

5.335.2 Constructor & Destructor Documentation

5.335.2.1 domain_handler_t() [1/2]

```
MHAPlugin_Split::domain_handler_t::domain_↩
handler_t (
            const domain_handler_t & ) [private]
```

Disallow copy constructor.

© 2005-2021 HörTech gGmbH, Oldenburg

5.335.2.2 domain_handler_t() [2/2]

```
MHAPlugin_Split::domain_handler_t::domain_↩
handler_t (
            const mhaconfig_t & settings_in,
            const mhaconfig_t & settings_out,
             PluginLoader::fourway_processor_t * processor ) [inline]
```

Construct a new domain handler once the domains and dimensions of input and output signal of one of the child plugins of split are known.

5.335.2.3 ~domain_handler_t()

```
virtual MHAPlugin_Split::domain_handler_t::~domain↩
_handler_t ( ) [inline], [virtual]
```

Deallocation of signal holders.

5.335.3 Member Function Documentation

5.335.3.1 operator=()

```
domain_handler_t& MHAPlugin_Split::domain_handler_t::operator=
(
            const domain_handler_t & ) [private]
```

Disallow assignment operator.

5.335.3.2 set_input_domain()

```
void MHAPlugin_Split::domain_handler_t::set_input_↩
domain (
            const mhaconfig_t & settings_in ) [inline]
```

Set parameters of input signal.

Parameters

settings↩ _in	domain and dimensions of partial input signal

© 2005-2021 HörTech gGmbH, Oldenburg

5.335.3.3 set_output_domain() void MHAPlugin_Split::domain_handler_t::set_output↩_domain (

```
const mhaconfig_t & settings_out ) [inline]
```

Set output signal parameters.

Parameters

settings_out	domain and dimensions of partial output signal

5.335.3.4 deallocate_domains() void MHAPlugin_Split::domain_handler_t::deallocate↩_domains ( ) [inline]

Deallocate domain indicators and signal holders.

5.335.3.5 put_signal() [1/2] unsigned MHAPlugin_Split::domain_handler_t::put_signal (

```
mha_wave_t * s_in,
unsigned start_channel ) [inline]
```

Store the relevant channels from the input signal for processing.

The number of channels to store is taken from the dimensions of the partial input signal holder **wave_in** (p. 1168).

Parameters

s_in	The combined waveform input signal.
start_channel	The index (0-based) of the first channel in s_in to be copied to the partial input signal.

Returns

The number of channels that were copied from the input signal

© 2005-2021 HörTech gGmbH, Oldenburg

5.335.3.6 put_signal() [2/2]

```
unsigned MHAPlugin_Split::domain_handler_t::put_signal
(
            mha_spec_t * s_in,
            unsigned start_channel ) [inline]
```

Store the relevant channels from the input signal for processing.

The number of channels to store is taken from the dimensions of the partial input signal holder **spec_in** (p. 1169).

Parameters

s_in	The combined spectrum input signal.
start_channel	The index (0-based) of the first channel in s_in to be copied to the partial input signal.

Returns

The number of channels that were copied from the input signal

5.335.3.7 get_signal() [1/2]

```
unsigned MHAPlugin_Split::domain_handler_t::get_signal
(
            MHASignal::waveform_t * s_out,
            unsigned start_channel ) [inline]
```

Store all partial signal output channels in the combined waveform signal with the given channel offset.

All channels present in **wave_out** (p. 1168) will be copied. Caller may use (∗wave_out)->num↩_channels to check the number of channels in advance.

Parameters

s_out	The combined waveform output signal.
start_channel	The channel offset (0-based) in s_out.

Returns

The number of channels that were copied to the output signal

© 2005-2021 HörTech gGmbH, Oldenburg

5.335.3.8 get_signal() [2/2] unsigned MHAPlugin_Split::domain_handler_t::get_signal (

```
MHASignal::spectrum_t * s_out,
unsigned start_channel ) [inline]
```

Store all partial signal output channels in the combined spectrum signal with the given channel offset.

All channels present in **spec_out** (p. 1169) will be copied. Caller may use (*spec_out)->num↩_channels to check the number of channels in advance.

Parameters

s_out	The combined spectrum output signal.
start_channel	The channel offset (0-based) in s_out.

Returns

The number of channels that were copied to the output signal

5.335.3.9 process() void MHAPlugin_Split::domain_handler_t::process ( ) [inline], [virtual]

Call the processing method of the processor with configured input/output signal domains.

The input signal has to be stored using **put_signal** (p. 1166) before this method may be called.

Implements **MHAPlugin_Split::uni_processor_t** (p. 1192).

5.335.4 Member Data Documentation

5.335.4.1 wave_in MHASignal::waveform_t* MHAPlugin_Split::domain_handler_t::wave↩_in

Partial wave input signal.

© 2005-2021 HörTech gGmbH, Oldenburg

5.335.4.2 wave_out `mha_wave_t** MHAPlugin_Split::domain_handler_t::wave_out`

Partial wave output signal.

5.335.4.3 spec_in `MHASignal::spectrum_t* MHAPlugin_Split::domain_handler_t::spec↩_in`

Partial spec input signal.

5.335.4.4 spec_out `mha_spec_t** MHAPlugin_Split::domain_handler_t::spec_out`

Partial spec input signal.

5.335.4.5 processor `PluginLoader::fourway_processor_t* MHAPlugin_Split::domain_↩handler_t::processor`

The domain-specific signal processing methods are implemented here.

The documentation for this class was generated from the following file:

- **split.cpp**

5.336 MHAPlugin_Split::dummy_threads_t Class Reference

Dummy specification of a thread platform: This class implements everything in a single thread.

Inheritance diagram for MHAPlugin_Split::dummy_threads_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- void **kick_thread** ()

  perform signal processing immediately (no multiple threads in this dummy class)
- void **catch_thread** ()

  No implementation needed: Processing has been completed during ummy_threads_t::kick_thread.
- **dummy_threads_t** (**uni_processor_t** *proc, const std::string &thread_scheduler, int thread_priority)

  Constructor.

Additional Inherited Members

5.336.1 Detailed Description

Dummy specification of a thread platform: This class implements everything in a single thread.

5.336.2 Constructor & Destructor Documentation

5.336.2.1 dummy_threads_t()

```
MHAPlugin_Split::dummy_threads_t::dummy_threads_t (
    uni_processor_t * proc,
    const std::string & thread_scheduler,
    int thread_priority ) [inline]
```

Constructor.

Parameters

proc	Pointer to the associated plugin loader
thread_scheduler	Unused in dummy thread platform
thread_priority	Unused in dummy thread platform

5.336.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.336.3.1 kick_thread() `void MHAPlugin_Split::dummy_threads_t::kick_thread ( ) [inline], [virtual]`

perform signal processing immediately (no multiple threads in this dummy class)

Implements **MHAPlugin_Split::thread_platform_t** (p. 1190).

5.336.3.2 catch_thread() `void MHAPlugin_Split::dummy_threads_t::catch_thread ( ) [inline], [virtual]`

No implementation needed: Processing has been completed during ummy_threads_t::kick_↩ thread.

Implements **MHAPlugin_Split::thread_platform_t** (p. 1190).

The documentation for this class was generated from the following file:

- **split.cpp**

5.337 MHAPlugin_Split::posix_threads_t Class Reference

Posix threads specification of thread platform.

Inheritance diagram for MHAPlugin_Split::posix_threads_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- void **kick_thread** ()

  Start signal processing in separate thread.
- void **catch_thread** ()

  Wait for signal processing to finish.
- **posix_threads_t** (**uni_processor_t** ∗proc, const std::string &thread_scheduler, int thread_priority)

  Constructor.
- ∼**posix_threads_t** ()

  Terminate thread.
- void **main** ()

  Thread main loop. Wait for process/termination trigger, then act.

Static Public Member Functions

- static void ∗ **thread_start** (void ∗thr)

  Thread start function.
- static std::string **current_thread_scheduler** ()
- static int **current_thread_priority** ()

Private Attributes

- pthread_mutex_t **mutex**

  The mutex.
- pthread_cond_t **kick_condition**

  The condition for signalling the kicking and termination.
- pthread_cond_t **catch_condition**

  The condition for signalling the processing is finished.
- pthread_attr_t **attr**

  Thread attributes.
- struct sched_param **priority**

  Thread scheduling priority.
- int **scheduler**
- pthread_t **thread**

  The thread object.
- bool **kicked**

  A flag that is set to true by kick_thread and to false by the thread after it has woken up from the kicking.
- bool **processing_done**

  A flag that is set to true by the thread when it returns from processing and to false by catch_↩ thread after it has waited for that return.
- bool **termination_request**

  Set to true by the destructor.

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.337.1 Detailed Description

Posix threads specification of thread platform.

5.337.2 Constructor & Destructor Documentation

5.337.2.1 posix_threads_t()

```
MHAPlugin_Split::posix_threads_t::posix_threads_t (
    uni_processor_t * proc,
    const std::string & thread_scheduler,
    int thread_priority ) [inline]
```

Constructor.

Parameters

proc	Pointer to the associated signal processor instance
thread_scheduler	A string describing the posix thread scheduler. Possible values: "SCHED_OTHER", "SCHED_RR", "SCHED_FIFO".
thread_priority	The scheduling priority of the new thread.

5.337.2.2 ~posix_threads_t()

```
MHAPlugin_Split::posix_threads_t::~posix_threads_t ( ) [inline]
```

Terminate thread.

5.337.3 Member Function Documentation

5.337.3.1 kick_thread()

```
void MHAPlugin_Split::posix_threads_t::kick_thread ( ) [inline], [virtual]
```

Start signal processing in separate thread.

Implements **MHAPlugin_Split::thread_platform_t** (p. 1190).

© 2005-2021 HörTech gGmbH, Oldenburg

5.337.3.2 catch_thread() `void MHAPlugin_Split::posix_threads_t::catch_thread ( ) [inline], [virtual]`

Wait for signal processing to finish.

Implements **MHAPlugin_Split::thread_platform_t** (p. 1190).

5.337.3.3 thread_start() `static void* MHAPlugin_Split::posix_threads_t::thread_start ( void * thr ) [inline], [static]`

Thread start function.

5.337.3.4 main() `void MHAPlugin_Split::posix_threads_t::main ( ) [inline]`

Thread main loop. Wait for process/termination trigger, then act.

5.337.3.5 current_thread_scheduler() `static std::string MHAPlugin_Split::posix_threads_t::current_thread_scheduler ( ) [inline], [static]`

5.337.3.6 current_thread_priority() `static int MHAPlugin_Split::posix_threads_t::current_thread_priority ( ) [inline], [static]`

5.337.4 Member Data Documentation

5.337.4.1 mutex `pthread_mutex_t MHAPlugin_Split::posix_threads_t::mutex [private]`

The mutex.

© 2005-2021 HörTech gGmbH, Oldenburg

5.337.4.2 kick_condition `pthread_cond_t MHAPlugin_Split::posix_threads_t::kick_condition [private]`

The condition for signalling the kicking and termination.

5.337.4.3 catch_condition `pthread_cond_t MHAPlugin_Split::posix_threads_t::catch_condition [private]`

The condition for signalling the processing is finished.

5.337.4.4 attr `pthread_attr_t MHAPlugin_Split::posix_threads_t::attr [private]`

Thread attributes.

5.337.4.5 priority `struct sched_param MHAPlugin_Split::posix_threads_t::priority [private]`

Thread scheduling priority.

5.337.4.6 scheduler `int MHAPlugin_Split::posix_threads_t::scheduler [private]`

5.337.4.7 thread `pthread_t MHAPlugin_Split::posix_threads_t::thread [private]`

The thread object.

5.337.4.8 kicked `bool MHAPlugin_Split::posix_threads_t::kicked [private]`

A flag that is set to true by kick_thread and to false by the thread after it has woken up from the kicking.

© 2005-2021 HörTech gGmbH, Oldenburg

5.337.4.9 processing_done `bool MHAPlugin_Split::posix_threads_t::processing_done [private]`

A flag that is set to true by the thread when it returns from processing and to false by catch_↩ thread after it has waited for that return.

5.337.4.10 termination_request `bool MHAPlugin_Split::posix_threads_t::termination↩ _request [private]`

Set to true by the destructor.

The documentation for this class was generated from the following file:

- **split.cpp**

5.338 MHAPlugin_Split::split_t Class Reference

Implements split plugin.

Inheritance diagram for MHAPlugin_Split::split_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **split_t** (**algo_comm_t** iac, const std::string &configured_name)
  Plugin constructor.
- ~**split_t** ()
  Plugin destructor. Unloads nested plugins.
- void **prepare_** (**mhaconfig_t** &)
  Check signal parameters, prepare chains, and allocate output signal holders.
- void **release_** ()
  Delete output signal holder and release chains.
- template<class SigTypeIn , class SigTypeOut >
  void **process** (SigTypeIn *, SigTypeOut **)
  Let the parallel plugins process channel groups of the input signal.

Private Member Functions

- void **update** ()
  Load plugins in response to a value change in the algos variable.
- void **clear_chains** ()
  Unload the plugins.
- **mha_wave_t** * **copy_output_wave** ()
- **mha_spec_t** * **copy_output_spec** ()
- template<class SigType >
  void **trigger_processing** (SigType *s_in)
  Split the argument input signal to groups of channels for the plugins and initiate signal processing.
- template<class SigType >
  void **collect_result** (SigType *s_out)
  Combine the output signal from the plugins.
- **MHASignal::waveform_t** * **signal_out** (**mha_wave_t** **)
  Waveform domain output signal structure accessor.
- **MHASignal::spectrum_t** * **signal_out** (**mha_spec_t** **)
  Spectrum domain output signal structure. Parameter is ignored.

Private Attributes

- **MHAEvents::patchbay_t**< **split_t** > **patchbay**
  Reload plugins when the algos variable changes.
- **MHAParser::vstring_t algos**
  Vector of plugins to load in parallel.
- **MHAParser::vint_t channels**
  Number of channels to route through each plugin.
- **MHAParser::kw_t thread_platform**
  Thread platform chooser.

© 2005-2021 HörTech gGmbH, Oldenburg

- **MHAParser::kw_t worker_thread_scheduler**
  Scheduler used for worker threads.
- **MHAParser::int_t worker_thread_priority**
  Priority of worker threads.
- **MHAParser::string_mon_t framework_thread_scheduler**
  Scheduler of the signal processing thread.
- **MHAParser::int_mon_t framework_thread_priority**
  Priority of signal processing thread.
- **MHAParser::bool_t delay**
  Switch to activate parallel processing of plugins at the cost of one block of additional delay.
- std::vector< **splitted_part_t** * > **chains**
  Interfaces to parallel plugins.
- **MHASignal::waveform_t** * **wave_out**
  Combined output waveforms structure.
- **MHASignal::spectrum_t** * **spec_out**
  Combined output spectra structure.

Additional Inherited Members

5.338.1 Detailed Description

Implements split plugin.

An instance of class **split_t** (p. 1176) implements the split plugin functionality: The audio channels are splitted and groups of audio channels are processed by different plugins in parallel.

5.338.2 Constructor & Destructor Documentation

5.338.2.1 split_t()

```
MHAPlugin_Split::split_t::split_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Plugin constructor.

5.338.2.2 ~split_t()

```
MHAPlugin_Split::split_t::~split_t ( )
```

Plugin destructor. Unloads nested plugins.

© 2005-2021 HörTech gGmbH, Oldenburg

5.338.3 Member Function Documentation

5.338.3.1 prepare_() `void MHAPlugin_Split::split_t::prepare_ ( mhaconfig_t & signal_parameters )`

Check signal parameters, prepare chains, and allocate output signal holders.

5.338.3.2 release_() `void MHAPlugin_Split::split_t::release_ ( )`

Delete output signal holder and release chains.

5.338.3.3 process() `template<class SigTypeIn , class SigTypeOut >`

```
void MHAPlugin_Split::split_t::process (
            SigTypeIn * s_in,
            SigTypeOut ** s_out )
```

Let the parallel plugins process channel groups of the input signal.

5.338.3.4 update() `void MHAPlugin_Split::split_t::update ( ) [private]`

Load plugins in response to a value change in the algos variable.

5.338.3.5 clear_chains() `void MHAPlugin_Split::split_t::clear_chains ( ) [private]`

Unload the plugins.

5.338.3.6 copy_output_wave() `mha_wave_t* MHAPlugin_Split::split_t::copy_output_wave ( ) [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.338.3.7 copy_output_spec()

`mha_spec_t* MHAPlugin_Split::split_t::copy_output_spec ( ) [private]`

5.338.3.8 trigger_processing()

```
template<class SigType >
void MHAPlugin_Split::split_t::trigger_processing (
            SigType * s_in ) [private]
```

Split the argument input signal to groups of channels for the plugins and initiate signal processing.

5.338.3.9 collect_result()

```
template<class SigType >
void MHAPlugin_Split::split_t::collect_result (
            SigType * s_out ) [private]
```

Combine the output signal from the plugins.

5.338.3.10 signal_out() [1/2]

`MHASignal::waveform_t* MHAPlugin_Split::split_t::signal_out ( mha_wave_t ** ) [inline], [private]`

Waveform domain output signal structure accessor.

Parameter is only for domain disambiguation and is ignored.

5.338.3.11 signal_out() [2/2]

`MHASignal::spectrum_t* MHAPlugin_Split::split_t::signal_out ( mha_spec_t ** ) [inline], [private]`

Spectrum domain output signal structure. Parameter is ignored.

5.338.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.338.4.1 patchbay

`MHAEvents::patchbay_t< split_t> MHAPlugin_Split::split_t↩ ::patchbay [private]`

Reload plugins when the algos variable changes.

5.338.4.2 algos

`MHAParser::vstring_t MHAPlugin_Split::split_t::algos [private]`

Vector of plugins to load in parallel.

5.338.4.3 channels

`MHAParser::vint_t MHAPlugin_Split::split_t::channels [private]`

Number of channels to route through each plugin.

5.338.4.4 thread_platform

`MHAParser::kw_t MHAPlugin_Split::split_t::thread_↩ platform [private]`

Thread platform chooser.

5.338.4.5 worker_thread_scheduler

`MHAParser::kw_t MHAPlugin_Split::split_t↩ ::worker_thread_scheduler [private]`

Scheduler used for worker threads.

5.338.4.6 worker_thread_priority

`MHAParser::int_t MHAPlugin_Split::split_t::worker↩ _thread_priority [private]`

Priority of worker threads.

© 2005-2021 HörTech gGmbH, Oldenburg

5.338.4.7 framework_thread_scheduler

`MHAParser::string_mon_t MHAPlugin_Split::split_t::framework_thread_scheduler [private]`

Scheduler of the signal processing thread.

5.338.4.8 framework_thread_priority

`MHAParser::int_mon_t MHAPlugin_Split::split_t::framework_thread_priority [private]`

Priority of signal processing thread.

5.338.4.9 delay

`MHAParser::bool_t MHAPlugin_Split::split_t::delay [private]`

Switch to activate parallel processing of plugins at the cost of one block of additional delay.

5.338.4.10 chains

`std::vector< splitted_part_t* > MHAPlugin_Split::split_t::chains [private]`

Interfaces to parallel plugins.

5.338.4.11 wave_out

`MHASignal::waveform_t* MHAPlugin_Split::split_t::wave_out [private]`

Combined output waveforms structure.

5.338.4.12 spec_out

`MHASignal::spectrum_t* MHAPlugin_Split::split_t::spec_out [private]`

Combined output spectra structure.

The documentation for this class was generated from the following file:

- **split.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.339 MHAPlugin_Split::splitted_part_t Class Reference

The **splitted_part_t** (p. 1183) instance manages the plugin that performs processing on the reduced set of channels.

Inheritance diagram for MHAPlugin_Split::splitted_part_t:

Public Member Functions

- **splitted_part_t** (const std::string &plugname, **MHAParser::parser_t** ∗parent)

  Load the plugin for this partial signal path.
- **splitted_part_t** (**PluginLoader::fourway_processor_t** ∗plugin)

  Create the handler for the partial signal.
- ∼**splitted_part_t** () throw ()

  *Destructor. Deletes the plugin **plug** (p. 1187).*
- void **prepare** (**mhaconfig_t** &signal_parameters, const std::string &thread_platform, const std::string &thread_scheduler, int thread_priority)

  *Delegates the prepare method to the plugin and allocates a suitable **MHAPlugin_Split::domain_handler_t** (p. 1163) instance.*
- void **release** ()

  *Delegates the release method to the plugin and deletes the **MHAPlugin_Split::domain_handler_t** (p. 1163) instance.*
- std::string **parse** (const std::string &str)

  Delegates parser incovation to plugin.
- template<class SigType >
  unsigned **trigger_processing** (SigType ∗s_in, unsigned start_channel)

  The domain handler copies the input signal channels.
- template<class SigType >
  unsigned **collect_result** (SigType ∗s_out, unsigned start_channel)

  Wait until processing is finished, then copy the output data.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- **splitted_part_t** (const **splitted_part_t** &)
  Disallow copy constructor.
- **splitted_part_t** & **operator=** (const **splitted_part_t** &)
  Disallow assignment operator.

Private Attributes

- **PluginLoader::fourway_processor_t** ∗ **plug**
  The plugin that performs the signal processing on the prepared channels.
- **domain_handler_t** ∗ **domain**
  The domain specific signal handler, allocated from prepare when input and output domains and signal parameters are known.
- **thread_platform_t** ∗ **thread**
  The platform-dependent thread synchronization implementation.

Additional Inherited Members

5.339.1 Detailed Description

The **splitted_part_t** (p. 1183) instance manages the plugin that performs processing on the reduced set of channels.

The signal is split by channels by this instance, but the signal is combined again by the calling class.

5.339.2 Constructor & Destructor Documentation

5.339.2.1 splitted_part_t() [1/3] MHAPlugin_Split::splitted_part_t::splitted_part_t (const **splitted_part_t** &) [private]

Disallow copy constructor.

5.339.2.2 splitted_part_t() [2/3] MHAPlugin_Split::splitted_part_t::splitted_part_t (const std::string & *plugname*, **MHAParser::parser_t** ∗ *parent*)

Load the plugin for this partial signal path.

Loads the MHA plugin for a signal path of these audio channels.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

plugname	The name of the MHA plugin, optionally followed by a colon and the algorithm name.
parent	The parser node where the configuration of the new plugin is inserted. The plugin's parser name is the configured name (colon syntax).

5.339.2.3 splitted_part_t() [3/3] `MHAPlugin_Split::splitted_part_t::splitted_part_t ( PluginLoader::fourway_processor_t * plugin )`

Create the handler for the partial signal.

The plugin is loaded by the caller, but it will be deleted by the destructor of this class. This constructor exists solely for testing purposes.

Parameters

plugin	The plugin used for processing the signal. The new **splitted_part_t** (p. 1184) instance will take ownership of this instance and release it in the destructor.

5.339.2.4 ~splitted_part_t() `MHAPlugin_Split::splitted_part_t::~splitted_part_t ( ) throw ( )`

Destructor. Deletes the plugin **plug** (p. 1187).

5.339.3 Member Function Documentation

5.339.3.1 operator=() `splitted_part_t& MHAPlugin_Split::splitted_part_t::operator= ( const splitted_part_t & ) [private]`

Disallow assignment operator.

© 2005-2021 HörTech gGmbH, Oldenburg

5.339.3.2 prepare()

```
void MHAPlugin_Split::splitted_part_t::prepare (
    mhaconfig_t & signal_parameters,
    const std::string & thread_platform,
    const std::string & thread_scheduler,
    int thread_priority )
```

Delegates the prepare method to the plugin and allocates a suitable **MHAPlugin_Split↩ ::domain_handler_t** (p. 1163) instance.

Prepare the loaded plugin.

Plugin preparation.

Parameters

signal_parameters	The signal description parameters for this path.
thread_platform	The name of the thread platform to use. Possible values: "posix", "win32", "dummy".
thread_scheduler	The name of the scheduler to use. Posix threads support "SCHED_OTHER", "SCHED_RR", "SCHED_FIFO". The other thread platforms do not support different thread schedulers. This value is not used for platforms other than "posix".
thread_priority	The new thread priority. Interpretation and permitted range depend on the thread platform and possibly on the scheduler.

5.339.3.3 release()

```
void MHAPlugin_Split::splitted_part_t::release ( )
```

Delegates the release method to the plugin and deletes the **MHAPlugin_Split::domain_↩ handler_t** (p. 1163) instance.

Release the loaded plugin.

Plugin release.

5.339.3.4 parse()

```
std::string MHAPlugin_Split::splitted_part_t::parse (
    const std::string & str ) [inline]
```

Delegates parser incovation to plugin.

5.339.3.5 trigger_processing()

```
template<class SigType >
unsigned MHAPlugin_Split::splitted_part_t::trigger_processing (
    SigType * s_in,
    unsigned start_channel ) [inline]
```

The domain handler copies the input signal channels.

Then, processing is initiated.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s_in	The combined input signal.
start_channel	The index (0-based) of the first channel in s_in to be copied to the partial input signal.

Returns

The number of channels that were copied from the input signal

5.339.3.6 collect_result()

```
template<class SigType >
unsigned MHAPlugin_Split::splitted_part_t::collect_result (
            SigType * s_out,
            unsigned start_channel ) [inline]
```

Wait until processing is finished, then copy the output data.

Parameters

s_out	The combined waveform output signal.
start_channel	The channel offset (0-based) in s_out.

Returns

The number of channels that were copied to the output signal

5.339.4 Member Data Documentation

5.339.4.1 plug

```
PluginLoader::fourway_processor_t* MHAPlugin_Split::splitted_part↩
_t::plug [private]
```

The plugin that performs the signal processing on the prepared channels.

© 2005-2021 HörTech gGmbH, Oldenburg

5.339.4.2 domain `domain_handler_t* MHAPlugin_Split::splitted_part_t::domain [private]`

The domain specific signal handler, allocated from prepare when input and output domains and signal parameters are known.

5.339.4.3 thread `thread_platform_t* MHAPlugin_Split::splitted_part_t::thread [private]`

The platform-dependent thread synchronization implementation.

The documentation for this class was generated from the following file:

- **split.cpp**

5.340 MHAPlugin_Split::thread_platform_t Class Reference

Basic interface for encapsulating thread creation, thread priority setting, and synchronization on any threading platform (i.e., pthreads or win32threads).

Inheritance diagram for MHAPlugin_Split::thread_platform_t:

Public Member Functions

- **thread_platform_t** (**uni_processor_t** *proc)
  Constructor.
- virtual ~**thread_platform_t** ()
  Make derived classes destructable via pointer to this base class.
- virtual void **kick_thread** ()=0
  Derived classes notify their processing thread that it should call processor->process().
- virtual void **catch_thread** ()=0
  Derived classes wait for their signal processing thread to return from the call to part->process().

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Attributes

- **uni_processor_t** ∗ **processor**
  A pointer to the plugin loader that processes the sound data in the channels for which this thread was created.

Private Member Functions

- **thread_platform_t** (const **thread_platform_t** &)
  Disallow copy constructor.
- **thread_platform_t** & **operator=** (const **thread_platform_t** &)
  Disallow assignment operator.

5.340.1 Detailed Description

Basic interface for encapsulating thread creation, thread priority setting, and synchronization on any threading platform (i.e., pthreads or win32threads).

Derived classes specialize in the actual thread platform.

5.340.2 Constructor & Destructor Documentation

5.340.2.1 thread_platform_t() [1/2]

```
MHAPlugin_Split::thread_platform_t::thread_↩
platform_t (
            const thread_platform_t & ) [private]
```

Disallow copy constructor.

5.340.2.2 thread_platform_t() [2/2]

```
MHAPlugin_Split::thread_platform_t::thread_↩
platform_t (
             uni_processor_t * proc ) [inline]
```

Constructor.

Derived classes create the thread in the constructor.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

proc	Pointer to the associated plugin loader. This plugin loader has to live at least as long as this instance. This instance does not take possession of the plugin loader. In production code, this thread platform and the plugin loader are both created and destroyed by the **MHAPlugin_Split::splitted_part_t** (p. 1183) instance.

5.340.2.3 ~thread_platform_t() `virtual MHAPlugin_Split::thread_platform_t::~thread_platform_t ( ) [inline], [virtual]`

Make derived classes destructable via pointer to this base class.

Derived classes' destructors notify the thread that it should terminate itself, and wait for the termination to occur.

5.340.3 Member Function Documentation

5.340.3.1 operator=() `thread_platform_t& MHAPlugin_Split::thread_platform_t::operator= ( const thread_platform_t & ) [private]`

Disallow assignment operator.

5.340.3.2 kick_thread() `virtual void MHAPlugin_Split::thread_platform_t::kick_thread ( ) [pure virtual]`

Derived classes notify their processing thread that it should call processor->process().

Implemented in **MHAPlugin_Split::posix_threads_t** (p. 1173), and **MHAPlugin_Split::dummy_threads_t** (p. 1170).

© 2005-2021 HörTech gGmbH, Oldenburg

5.340.3.3 catch_thread() `virtual void MHAPlugin_Split::thread_platform_t::catch_thread ( ) [pure virtual]`

Derived classes wait for their signal processing thread to return from the call to part->process().

Implemented in **MHAPlugin_Split::posix_threads_t** (p. 1173), and **MHAPlugin_Split::dummy_threads_t** (p. 1171).

5.340.4 Member Data Documentation

5.340.4.1 processor `uni_processor_t* MHAPlugin_Split::thread_platform_t::processor [protected]`

A pointer to the plugin loader that processes the sound data in the channels for which this thread was created.

Using the **MHAPlugin_Split::uni_processor_t** (p. 1191) interface instead of the mhaplugin-loader class directly for testability (no need to load real plugins for testing the thread platform).

The documentation for this class was generated from the following file:

- **split.cpp**

5.341 MHAPlugin_Split::uni_processor_t Class Reference

An interface to a class that sports a process method with no parameters and no return value.

Inheritance diagram for MHAPlugin_Split::uni_processor_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- virtual void **process** ()=0

  This method uses some input signal, performs processing and stores the output signal somewhere.
- virtual ∼**uni_processor_t** ()

  Classes containing virtual methods need virtual destructors.

5.341.1 Detailed Description

An interface to a class that sports a process method with no parameters and no return value.

No signal transfer occurs through this interface, because the signal transfer is performed in another thread than the processing.

5.341.2 Constructor & Destructor Documentation

5.341.2.1 ∼uni_processor_t() `virtual MHAPlugin_Split::uni_processor_t::∼uni_↩processor_t ( ) [inline], [virtual]`

Classes containing virtual methods need virtual destructors.

5.341.3 Member Function Documentation

5.341.3.1 process() `virtual void MHAPlugin_Split::uni_processor_t::process ( ) [pure virtual]`

This method uses some input signal, performs processing and stores the output signal somewhere.

This method also has to dispatch the process call based on the configured domains.

Signal transfer and domain configuration have to be done in derived class in different methods.

Implemented in **MHAPlugin_Split::domain_handler_t** (p. 1168).

The documentation for this class was generated from the following file:

- **split.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.342 mhaserver_t Class Reference

MHA Framework listening on TCP port for commands.

Inheritance diagram for mhaserver_t:

Classes

- class **tcp_server_t**

Public Member Functions

- **mhaserver_t** (const std::string &ao, const std::string &af, const std::string &lf, bool b_interactive_)
- ~**mhaserver_t** ()
- virtual std::string **on_received_line** (const std::string &line)

  A line of text was received from network client.
- virtual void **acceptor_started** ()

  Notification: "TCP port is open".
- virtual void **send_port_announcement** ()

  sends an announcement which port this MHA is listening on to the creator of the process.
- virtual void **start_stdin_thread** ()

  Starts a separate thread that reads lines from stdin and forwards these lines over TCP to the MHA configuration thread which multiplexes multiple TCP connections.

© 2005-2021 HörTech gGmbH, Oldenburg

- virtual void **set_announce_port** (unsigned short **announce_port**)

  If set to nonzero, the spawning process has asked to be notified of the TCP port used by this process.
- void **logstring** (const std::string &)

  Log a message to log file.
- int **run** (unsigned short **port**, const std::string &_interface)

  Accept network connections and act on commands.

Public Attributes

- **MHAParser::int_t port**

Private Attributes

- std::shared_ptr< **tcp_server_t** > **tcpserver**
- std::string **ack_ok**
- std::string **ack_fail**
- std::string **logfile**
- unsigned short **announce_port**
- bool **b_interactive**
- **MHAParser::int_mon_t pid_mon**

Additional Inherited Members

5.342.1 Detailed Description

MHA Framework listening on TCP port for commands.

5.342.2 Constructor & Destructor Documentation

5.342.2.1 mhaserver_t()

```
mhaserver_t::mhaserver_t (
    const std::string & ao,
    const std::string & af,
    const std::string & lf,
    bool b_interactive_ )
```

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

ao	Acknowledgement string at end of successful command responses
af	Acknowledgement string at end of failed command responses
lf	File system path of file to use as log file. MHA appends.

5.342.2.2 ~mhaserver_t() `mhaserver_t::~mhaserver_t ( )`

5.342.3 Member Function Documentation

5.342.3.1 on_received_line() `std::string mhaserver_t::on_received_line ( const std::string & line ) [virtual]`

A line of text was received from network client.

5.342.3.2 acceptor_started() `void mhaserver_t::acceptor_started ( ) [virtual]`

Notification: "TCP port is open".

5.342.3.3 send_port_announcement() `void mhaserver_t::send_port_announcement ( ) [virtual]`

sends an announcement which port this MHA is listening on to the creator of the process.

See command line option –announce

5.342.3.4 start_stdin_thread() `void mhaserver_t::start_stdin_thread ( ) [virtual]`

Starts a separate thread that reads lines from stdin and forwards these lines over TCP to the MHA configuration thread which multiplexes multiple TCP connections.

Enables users to type mha configuration language commands directly into the terminal where MHA was started, without the need to use third-party tools like nc or putty.

© 2005-2021 HörTech gGmbH, Oldenburg

5.342.3.5 set_announce_port()

```
void mhaserver_t::set_announce_port (
    unsigned short announce_port ) [virtual]
```

If set to nonzero, the spawning process has asked to be notified of the TCP port used by this process.

5.342.3.6 logstring()

```
void mhaserver_t::logstring (
    const std::string & s ) [inline]
```

Log a message to log file.

5.342.3.7 run()

```
int mhaserver_t::run (
    unsigned short port,
    const std::string & _interface )
```

Accept network connections and act on commands.

Calls **acceptor_started()** (p. 1195) when the TCP port is opened. Calls on_received_line for every line received.

Returns

exit code that can be used as process exit code

5.342.4 Member Data Documentation

5.342.4.1 tcpserver

```
std::shared_ptr< tcp_server_t> mhaserver_t::tcpserver [private]
```

5.342.4.2 ack_ok

```
std::string mhaserver_t::ack_ok [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.342.4.3 ack_fail `std::string mhaserver_t::ack_fail [private]`

5.342.4.4 logfile `std::string mhaserver_t::logfile [private]`

5.342.4.5 announce_port `unsigned short mhaserver_t::announce_port [private]`

5.342.4.6 b_interactive `bool mhaserver_t::b_interactive [private]`

5.342.4.7 pid_mon `MHAParser::int_mon_t mhaserver_t::pid_mon [private]`

5.342.4.8 port `MHAParser::int_t mhaserver_t::port`

The documentation for this class was generated from the following file:

- **mhamain.cpp**

5.343 mhaserver_t::tcp_server_t Class Reference

Inheritance diagram for mhaserver_t::tcp_server_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **tcp_server_t** (const std::string &interface, uint16_t **port**, **mhaserver_t** ∗ **mha**)
- virtual bool **on_received_line** (std::shared_ptr< **mha_tcp::buffered_socket_t** > c, const std::string &l) override

  This method is invoked when a line of text is received on one of the accepted connections.

Private Attributes

- **mhaserver_t** ∗ **mha**

5.343.1 Constructor & Destructor Documentation

5.343.1.1 tcp_server_t()

```
mhaserver_t::tcp_server_t::tcp_server_t (
    const std::string & interface,
    uint16_t port,
    mhaserver_t * mha ) [inline]
```

5.343.2 Member Function Documentation

5.343.2.1 on_received_line()

```
virtual bool mhaserver_t::tcp_server_t::on_received_line (
    std::shared_ptr< mha_tcp::buffered_socket_t > c,
    const std::string & l ) [inline], [override], [virtual]
```

This method is invoked when a line of text is received on one of the accepted connections.

Override this method to process the communication with the client.

Parameters

c	the connection that has received this line
l	the line that has been received, without the line ending

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

client should return true when client wants to read another line of text, else false.

Reimplemented from **mha_tcp::server_t** (p. 822).

5.343.3 Member Data Documentation

5.343.3.1 mha `mhaserver_t* mhaserver_t::tcp_server_t::mha [private]`

The documentation for this class was generated from the following file:

- **mhamain.cpp**

5.344 MHASignal::async_rmslevel_t Class Reference

Class for asynchronous level metering.

Inheritance diagram for MHASignal::async_rmslevel_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **async_rmslevel_t** (unsigned int frames, unsigned int **channels**)
  Constructor for level metering class.
- std::vector< float > **rmslevel** () const
  Read-only function for querying the current RMS level.
- std::vector< float > **peaklevel** () const
  Read-only function for querying the current peak level.
- void **process** (**mha_wave_t** *s)
  Function to store a chunk of audio in the level meter.

Private Attributes

- unsigned int **pos**
- unsigned int **filled**

Additional Inherited Members

5.344.1 Detailed Description

Class for asynchronous level metering.

5.344.2 Constructor & Destructor Documentation

5.344.2.1 **async_rmslevel_t()** `MHASignal::async_rmslevel_t::async_rmslevel_t (`

```
unsigned int frames,
unsigned int channels )
```

Constructor for level metering class.

Allocate memory for metering. The RMS integration time corresponds to the number of frames in the buffer.

Parameters

frames	Number of frames to integrate.
channels	Number of channels used for level-metering.

© 2005-2021 HörTech gGmbH, Oldenburg

5.344.3 Member Function Documentation

5.344.3.1 rmslevel() `std::vector< float > MHASignal::async_rmslevel_t::rmslevel ( ) const`

Read-only function for querying the current RMS level.

Returns

Vector of floats, one value for each channel, containing the RMS level in dB (SPL if calibrated properly).

5.344.3.2 peaklevel() `std::vector< float > MHASignal::async_rmslevel_t::peaklevel ( ) const`

Read-only function for querying the current peak level.

Returns

Vector of floats, one value for each channel, containing the peak level in dB (SPL if calibrated properly).

5.344.3.3 process() `void MHASignal::async_rmslevel_t::process ( mha_wave_t * s )`

Function to store a chunk of audio in the level meter.

Parameters

s	Audio chunk (same number of channels required as given in the constructor).

5.344.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.344.4.1 pos `unsigned int MHASignal::async_rmslevel_t::pos [private]`

5.344.4.2 filled `unsigned int MHASignal::async_rmslevel_t::filled [private]`

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.345 MHASignal::delay_spec_t Class Reference

Public Member Functions

- **delay_spec_t** (unsigned int **delay**, unsigned int frames, unsigned int **channels**)
- ∼**delay_spec_t** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

Private Attributes

- unsigned int **delay**
- **MHASignal::spectrum_t** ∗∗ **buffer**
- unsigned int **pos**

5.345.1 Constructor & Destructor Documentation

5.345.1.1 delay_spec_t() `MHASignal::delay_spec_t::delay_spec_t (`

```
unsigned int delay,
unsigned int frames,
unsigned int channels )
```

5.345.1.2 ∼delay_spec_t() `MHASignal::delay_spec_t::~delay_spec_t ( )`

© 2005-2021 HörTech gGmbH, Oldenburg

5.345.2 Member Function Documentation

5.345.2.1 process() `mha_spec_t * MHASignal::delay_spec_t::process ( mha_spec_t * s )`

5.345.3 Member Data Documentation

5.345.3.1 delay `unsigned int MHASignal::delay_spec_t::delay [private]`

5.345.3.2 buffer `MHASignal::spectrum_t** MHASignal::delay_spec_t::buffer [private]`

5.345.3.3 pos `unsigned int MHASignal::delay_spec_t::pos [private]`

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.346 MHASignal::delay_t Class Reference

Class to realize a simple delay of waveform streams.

Inheritance diagram for MHASignal::delay_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **delay_t** (std::vector< int > **delays**, unsigned int **channels**)
  Constructor.
- **mha_wave_t** * **process** (**mha_wave_t** *s)
  Processing method.
- ~**delay_t** ()
- std::string **inspect** () const

Private Attributes

- unsigned int **channels**
- unsigned int * **delays**
- unsigned int * **pos**
- **mha_real_t** ** **buffer**

5.346.1 Detailed Description

Class to realize a simple delay of waveform streams.

5.346.2 Constructor & Destructor Documentation

5.346.2.1 delay_t() `MHASignal::delay_t::delay_t ( std::vector< int > delays, unsigned int channels )`

Constructor.

Parameters

delays	Vector of delays, one entry for each channel.
channels	Number of channels expected.

5.346.2.2 ~delay_t() `MHASignal::delay_t::~delay_t ( )`

© 2005-2021 HörTech gGmbH, Oldenburg

5.346.3 Member Function Documentation

5.346.3.1 process() `mha_wave_t * MHASignal::delay_t::process ( mha_wave_t * s )`

Processing method.

Parameters

s	Input waveform fragment, with number of channels provided in constructor.

Returns

Output waveform fragment.

5.346.3.2 inspect() `std::string MHASignal::delay_t::inspect ( ) const [inline]`

5.346.4 Member Data Documentation

5.346.4.1 channels `unsigned int MHASignal::delay_t::channels [private]`

5.346.4.2 delays `unsigned int* MHASignal::delay_t::delays [private]`

5.346.4.3 pos `unsigned int* MHASignal::delay_t::pos [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.346.4.4 buffer `mha_real_t** MHASignal::delay_t::buffer [private]`

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.347 MHASignal::delay_wave_t Class Reference

Delayline containing wave fragments.

Public Member Functions

- **delay_wave_t** (unsigned int **delay**, unsigned int frames, unsigned int **channels**)
- ~**delay_wave_t** ()
- **mha_wave_t** * **process** (**mha_wave_t** *)

Private Attributes

- unsigned int **delay**
- **MHASignal::waveform_t** ** **buffer**
- unsigned int **pos**

5.347.1 Detailed Description

Delayline containing wave fragments.

The delayline contains waveform fragments. The delay can be configured in integer fragments (sample delay or sub-sample delay is not possible).

5.347.2 Constructor & Destructor Documentation

5.347.2.1 delay_wave_t()

```
MHASignal::delay_wave_t::delay_wave_t (
    unsigned int delay,
    unsigned int frames,
    unsigned int channels )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.347.2.2 ~delay_wave_t() `MHASignal::delay_wave_t::~delay_wave_t ( )`

5.347.3 Member Function Documentation

5.347.3.1 process() `mha_wave_t * MHASignal::delay_wave_t::process ( mha_wave_t * s )`

5.347.4 Member Data Documentation

5.347.4.1 delay `unsigned int MHASignal::delay_wave_t::delay [private]`

5.347.4.2 buffer `MHASignal::waveform_t** MHASignal::delay_wave_t::buffer [private]`

5.347.4.3 pos `unsigned int MHASignal::delay_wave_t::pos [private]`

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.348 MHASignal::doublebuffer_t Class Reference

Double-buffering class.

Inheritance diagram for MHASignal::doublebuffer_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **doublebuffer_t** (unsigned int nchannels_in, unsigned int nchannels_out, unsigned int outer_fragsize, unsigned int inner_fragsize)
  Constructor of double buffer.
- virtual ~**doublebuffer_t** ()
- **mha_wave_t** ∗ **outer_process** (**mha_wave_t** ∗s)
  Method to pass audio fragments into the inner layer.

Protected Member Functions

- virtual **mha_wave_t** ∗ **inner_process** (**mha_wave_t** ∗s)=0
  Method to realize inner processing callback.

Private Member Functions

- unsigned int **min** (unsigned int a, unsigned int b)

Private Attributes

- **waveform_t outer_out**
- **mha_wave_t this_outer_out**
- **waveform_t inner_in**
- **waveform_t inner_out**
- unsigned int **k_inner**
- unsigned int **k_outer**
- unsigned int **ch**

5.348.1 Detailed Description

Double-buffering class.

This class has two layers: The outer layer, with an outer fragment size, and an inner layer, with its own fragment size. Data is passed into the inner layer through the doublebuffer_t↩ ::outr_process() callback. The pure virtual method **doublebuffer_t::inner_process()** (p. 1209) is called whenever enough data is available.

5.348.2 Constructor & Destructor Documentation

5.348.2.1 doublebuffer_t()

```
MHASignal::doublebuffer_t::doublebuffer_t (
    unsigned int nchannels_in,
    unsigned int nchannels_out,
    unsigned int outer_fragsize,
    unsigned int inner_fragsize )
```

Constructor of double buffer.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

nchannels_in	Number of channels at the input (both layers).
nchannels_out	Number of channels at the output (both layers).
outer_fragsize	Fragment size of the outer layer (e.g., hardware fragment size)
inner_fragsize	Fragment size of the inner layer (e.g., software fragment size)

5.348.2.2 ∼doublebuffer_t() `MHASignal::doublebuffer_t::~doublebuffer_t ( ) [virtual]`

5.348.3 Member Function Documentation

5.348.3.1 outer_process() `mha_wave_t * MHASignal::doublebuffer_t::outer_process ( mha_wave_t * s )`

Method to pass audio fragments into the inner layer.

Parameters

s	Pointer to input waveform fragment.

Returns

Pointer to output waveform fragment.

5.348.3.2 inner_process() `virtual mha_wave_t* MHASignal::doublebuffer_t::inner_process ( mha_wave_t * s ) [protected], [pure virtual]`

Method to realize inner processing callback.

To be overwritten by derived classes.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s	Pointer to input waveform fragment.

Returns

Pointer to output waveform fragment.

Implemented in **db_t** (p. 372).

5.348.3.3 min()

```
unsigned int MHASignal::doublebuffer_t::min (
    unsigned int a,
    unsigned int b ) [inline], [private]
```

5.348.4 Member Data Documentation

5.348.4.1 outer_out

waveform_t `MHASignal::doublebuffer_t::outer_out` `[private]`

5.348.4.2 this_outer_out

mha_wave_t `MHASignal::doublebuffer_t::this_outer_out` `[private]`

5.348.4.3 inner_in

waveform_t `MHASignal::doublebuffer_t::inner_in` `[private]`

5.348.4.4 inner_out

waveform_t `MHASignal::doublebuffer_t::inner_out` `[private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.348.4.5 k_inner `unsigned int MHASignal::doublebuffer_t::k_inner [private]`

5.348.4.6 k_outer `unsigned int MHASignal::doublebuffer_t::k_outer [private]`

5.348.4.7 ch `unsigned int MHASignal::doublebuffer_t::ch [private]`

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.349 MHASignal::fft_t Class Reference

Public Member Functions

- **fft_t** (const unsigned int &)
- ∼**fft_t** ()
- void **wave2spec** (const **mha_wave_t** ∗, **mha_spec_t** ∗, bool swap)
  fast fourier transform.
- void **spec2wave** (const **mha_spec_t** ∗, **mha_wave_t** ∗)
- void **spec2wave** (const **mha_spec_t** ∗, **mha_wave_t** ∗, unsigned int offset)
  wave may have fewer number of frames than needed for a complete iFFT.
- void **forward** (**mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)
- void **backward** (**mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)
- void **wave2spec_scale** (const **mha_wave_t** ∗, **mha_spec_t** ∗, bool swap)
- void **spec2wave_scale** (const **mha_spec_t** ∗, **mha_wave_t** ∗)
- void **forward_scale** (**mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)
- void **backward_scale** (**mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)

Private Member Functions

- void **sort_fftw2spec** (fftw_real ∗s_fftw, **mha_spec_t** ∗s_spec, unsigned int ch)
  Arrange the order of an fftw spectrum to the internal order.
- void **sort_spec2fftw** (fftw_real ∗s_fftw, const **mha_spec_t** ∗s_spec, unsigned int ch)
  Arrange the order of an internal spectrum to the fftw order.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- unsigned int **nfft**
- unsigned int **n_re**
- unsigned int **n_im**
- **mha_real_t scale**
- **mha_real_t** * **buf_in**
- **mha_real_t** * **buf_out**
- rfftw_plan **fftw_plan_wave2spec**
- rfftw_plan **fftw_plan_spec2wave**
- fftw_plan **fftw_plan_fft**
- fftw_plan **fftw_plan_ifft**

5.349.1 Constructor & Destructor Documentation

5.349.1.1 fft_t()

```
MHASignal::fft_t::fft_t (
    const unsigned int & n )
```

5.349.1.2 ~fft_t()

```
MHASignal::fft_t::~fft_t ( )
```

5.349.2 Member Function Documentation

5.349.2.1 wave2spec()

```
void MHASignal::fft_t::wave2spec (
    const mha_wave_t * wave,
    mha_spec_t * spec,
    bool swap )
```

fast fourier transform.

if swap is set, the buffer halfes of the wave signal are exchanged before computing the fft.

© 2005-2021 HörTech gGmbH, Oldenburg

5.349.2.2 spec2wave() [1/2]

```
void MHASignal::fft_t::spec2wave (
    const mha_spec_t * spec,
    mha_wave_t * wave )
```

5.349.2.3 spec2wave() [2/2]

```
void MHASignal::fft_t::spec2wave (
    const mha_spec_t * spec,
    mha_wave_t * wave,
    unsigned int offset )
```

wave may have fewer number of frames than needed for a complete iFFT.

Only as many frames are written into wave as fit, starting with offset offset of the complete iFFT.

5.349.2.4 forward()

```
void MHASignal::fft_t::forward (
    mha_spec_t * sIn,
    mha_spec_t * sOut )
```

5.349.2.5 backward()

```
void MHASignal::fft_t::backward (
    mha_spec_t * sIn,
    mha_spec_t * sOut )
```

5.349.2.6 wave2spec_scale()

```
void MHASignal::fft_t::wave2spec_scale (
    const mha_wave_t * wave,
    mha_spec_t * spec,
    bool swap )
```

5.349.2.7 spec2wave_scale()

```
void MHASignal::fft_t::spec2wave_scale (
    const mha_spec_t * spec,
    mha_wave_t * wave )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.349.2.8 forward_scale()

```
void MHASignal::fft_t::forward_scale (
    mha_spec_t * sIn,
    mha_spec_t * sOut )
```

5.349.2.9 backward_scale()

```
void MHASignal::fft_t::backward_scale (
    mha_spec_t * sIn,
    mha_spec_t * sOut )
```

5.349.2.10 sort_fftw2spec()

```
void MHASignal::fft_t::sort_fftw2spec (
    fftw_real * s_fftw,
    mha_spec_t * s_spec,
    unsigned int ch ) [private]
```

Arrange the order of an fftw spectrum to the internal order.

The fftw spectrum is arranged [r0 r1 r2 ... rn-1 in in-1 ... i1], while the interal order is [r0 – r1 i1 r2 i2 ... rn-1 in-1 rn –].

5.349.2.11 sort_spec2fftw()

```
void MHASignal::fft_t::sort_spec2fftw (
    fftw_real * s_fftw,
    const mha_spec_t * s_spec,
    unsigned int ch ) [private]
```

Arrange the order of an internal spectrum to the fftw order.

5.349.3 Member Data Documentation

5.349.3.1 nfft

```
unsigned int MHASignal::fft_t::nfft [private]
```

5.349.3.2 n_re

```
unsigned int MHASignal::fft_t::n_re [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.349.3.3 n_im `unsigned int MHASignal::fft_t::n_im [private]`

5.349.3.4 scale `mha_real_t MHASignal::fft_t::scale [private]`

5.349.3.5 buf_in `mha_real_t* MHASignal::fft_t::buf_in [private]`

5.349.3.6 buf_out `mha_real_t* MHASignal::fft_t::buf_out [private]`

5.349.3.7 fftw_plan_wave2spec `rfftw_plan MHASignal::fft_t::fftw_plan_wave2spec [private]`

5.349.3.8 fftw_plan_spec2wave `rfftw_plan MHASignal::fft_t::fftw_plan_spec2wave [private]`

5.349.3.9 fftw_plan_fft `fftw_plan MHASignal::fft_t::fftw_plan_fft [private]`

5.349.3.10 fftw_plan_ifft `fftw_plan MHASignal::fft_t::fftw_plan_ifft [private]`

The documentation for this class was generated from the following files:

- **mha_signal_fft.h**
- **mha_signal.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.350 MHASignal::hilbert_fftw_t Class Reference

Public Member Functions

- **hilbert_fftw_t** (unsigned int len)
  *C'tor of **hilbert_fftw_t** (p. 1216).*
- ∼**hilbert_fftw_t** ()
  *D'tor of **hilbert_fftw_t** (p. 1216).*
- void **hilbert** (const **mha_wave_t** ∗, **mha_wave_t** ∗)

Private Attributes

- unsigned int **n**
- rfftw_plan **p1**
- fftw_plan **p2**
- fftw_real ∗ **buf_r_in**
- fftw_real ∗ **buf_r_out**
- fftw_complex ∗ **buf_c_in**
- fftw_complex ∗ **buf_c_out**
- **mha_real_t sc**

5.350.1 Constructor & Destructor Documentation

5.350.1.1 hilbert_fftw_t() `MHASignal::hilbert_fftw_t::hilbert_fftw_t ( unsigned int len )`

C'tor of **hilbert_fftw_t** (p. 1216).

Parameters

len	fft length

5.350.1.2 ∼hilbert_fftw_t() `MHASignal::hilbert_fftw_t::~hilbert_fftw_t ( )`

D'tor of **hilbert_fftw_t** (p. 1216).

© 2005-2021 HörTech gGmbH, Oldenburg

5.350.2 Member Function Documentation

5.350.2.1 hilbert()

```
void MHASignal::hilbert_fftw_t::hilbert (
    const mha_wave_t * s_in,
    mha_wave_t * s_out )
```

5.350.3 Member Data Documentation

5.350.3.1 n

```
unsigned int MHASignal::hilbert_fftw_t::n [private]
```

5.350.3.2 p1

```
rfftw_plan MHASignal::hilbert_fftw_t::p1 [private]
```

5.350.3.3 p2

```
fftw_plan MHASignal::hilbert_fftw_t::p2 [private]
```

5.350.3.4 buf_r_in

```
fftw_real* MHASignal::hilbert_fftw_t::buf_r_in [private]
```

5.350.3.5 buf_r_out

```
fftw_real* MHASignal::hilbert_fftw_t::buf_r_out [private]
```

5.350.3.6 buf_c_in

```
fftw_complex* MHASignal::hilbert_fftw_t::buf_c_in [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.350.3.7 buf_c_out `fftw_complex* MHASignal::hilbert_fftw_t::buf_c_out [private]`

5.350.3.8 sc `mha_real_t MHASignal::hilbert_fftw_t::sc [private]`

The documentation for this class was generated from the following file:

- **mha_signal.cpp**

5.351 MHASignal::hilbert_t Class Reference

Hilbert transformation of a waveform segment.

Inheritance diagram for MHASignal::hilbert_t:

Public Member Functions

- **hilbert_t** (unsigned int len)
- ~**hilbert_t** ()
- void **operator()** (const **mha_wave_t** *, **mha_wave_t** *)

  Apply Hilbert transformation on a waveform segment.

Private Attributes

- void * **h**

© 2005-2021 HörTech gGmbH, Oldenburg

5.351.1 Detailed Description

Hilbert transformation of a waveform segment.

Returns the imaginary part of the inverse Fourier transformation of the Fourier transformed input signal with negative frequencies set to zero.

5.351.2 Constructor & Destructor Documentation

5.351.2.1 hilbert_t()

```
MHASignal::hilbert_t::hilbert_t (
    unsigned int len )
```

Parameters

len	Length of waveform segment

5.351.2.2 ~hilbert_t()

```
MHASignal::hilbert_t::~hilbert_t ( )
```

5.351.3 Member Function Documentation

5.351.3.1 operator()()

```
void MHASignal::hilbert_t::operator() (
    const mha_wave_t * s_in,
    mha_wave_t * s_out )
```

Apply Hilbert transformation on a waveform segment.

5.351.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.351.4.1 h `void* MHASignal::hilbert_t::h [private]`

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.352 MHASignal::loop_wavefragment_t Class Reference

Copy a fixed waveform fragment to a series of waveform fragments of other size.

Inheritance diagram for MHASignal::loop_wavefragment_t:

Public Types

- enum **level_mode_t** { **relative**, **peak**, **rms**, **rms_limit40** }
  Switch for playback level mode.
- enum **playback_mode_t** { **add**, **replace**, **input**, **mute** }
  Switch for playback mode.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **loop_wavefragment_t** (const **mha_wave_t** &src, bool loop, **level_mode_t** level_mode, std::vector< int > **channels**, unsigned int startpos=0)

  *Constructor to create an instance of **loop_wavefragment_t** (p. 1220) based on an existing waveform block.*
- std::vector< int > **get_mapping** (unsigned int **channels**)
- void **playback** (**mha_wave_t** *s, **playback_mode_t** pmode, **mha_wave_t** *level_pa, const std::vector< int > & **channels**)

  Add source waveform block to an output block.
- void **playback** (**mha_wave_t** *s, **playback_mode_t** pmode, **mha_wave_t** *level_pa)

  Add source waveform block to an output block.
- void **playback** (**mha_wave_t** *s, **playback_mode_t** pmode)

  Add source waveform block to an output block.
- void **set_level_lin** (**mha_real_t** l)
- void **set_level_db** (**mha_real_t** l)
- void **rewind** ()
- void **locate_end** ()
- bool **is_playback_active** () const

Private Attributes

- std::vector< int > **playback_channels**
- bool **b_loop**
- unsigned int **pos**
- **MHASignal::waveform_t intern_level**

Additional Inherited Members

5.352.1 Detailed Description

Copy a fixed waveform fragment to a series of waveform fragments of other size.

This class is designed to continously play back a waveform to an output stream, with variable output block size.

5.352.2 Member Enumeration Documentation

5.352.2.1 level_mode_t enum MHASignal::loop_wavefragment_t::level_mode_t

Switch for playback level mode.

© 2005-2021 HörTech gGmbH, Oldenburg

Enumerator

relative	The nominal level is applied as a gain to the source signal.
peak	The nominal level is the peak level of source signal in Pascal.
rms	The nominal level is the RMS level of the source signal in Pascal.
rms_limit40	

5.352.2.2 playback_mode_t `enum MHASignal::loop_wavefragment_t::playback_mode_t`

Switch for playback mode.

Enumerator

add	Add source signal to output stream.
replace	Replace output stream by source signal.
input	Do nothing, keep output stream (source position is unchanged).
mute	Mute output stream (source position is unchanged).

5.352.3 Constructor & Destructor Documentation

5.352.3.1 loop_wavefragment_t() `MHASignal::loop_wavefragment_t::loop_wavefragment↩_t (`

```
const mha_wave_t & src,
bool loop,
 level_mode_t level_mode,
std::vector< int > channels,
unsigned int startpos = 0 )
```

Constructor to create an instance of **loop_wavefragment_t** (p. 1220) based on an existing waveform block.

Parameters

src	Waveform block to copy data from.
loop	Flag whether the block should be looped or played once.
level_mode	Configuration of playback level (see **MHASignal::loop_wavefragment_t::level_mode_t** (p. 1221) for details)
channels	Mapping of input to output channels.
startpos	Starting position

© 2005-2021 HörTech gGmbH, Oldenburg

5.352.4 Member Function Documentation

5.352.4.1 get_mapping()

```
std::vector< int > MHASignal::loop_wavefragment_t::get_↩
mapping (
            unsigned int channels )
```

5.352.4.2 playback() [1/3]

```
void MHASignal::loop_wavefragment_t::playback (
            mha_wave_t * s,
            playback_mode_t pmode,
            mha_wave_t * level_pa,
            const std::vector< int > & channels )
```

Add source waveform block to an output block.

Parameters

s	Output block (streamed signal).
pmode	Playback mode (add, replace, input, mute).
level_pa	Linear output level/gain (depending on level_mode parameter in constructor); one value for each sample in output block.
channels	Output channels

5.352.4.3 playback() [2/3]

```
void MHASignal::loop_wavefragment_t::playback (
            mha_wave_t * s,
            playback_mode_t pmode,
            mha_wave_t * level_pa )
```

Add source waveform block to an output block.

Parameters

s	Output block (streamed signal).
pmode	Playback mode (add, replace, input, mute).
level_pa	Linear output level/gain (depending on level_mode parameter in constructor); one value for each sample in output block.

© 2005-2021 HörTech gGmbH, Oldenburg

5.352.4.4 playback() [3/3]

```
void MHASignal::loop_wavefragment_t::playback (
    mha_wave_t * s,
    playback_mode_t pmode )
```

Add source waveform block to an output block.

Parameters

s	Output block (streamed signal).
pmode	Playback mode (add, replace, input, mute).

5.352.4.5 set_level_lin()

```
void MHASignal::loop_wavefragment_t::set_level_lin (
    mha_real_t l )
```

5.352.4.6 set_level_db()

```
void MHASignal::loop_wavefragment_t::set_level_db (
    mha_real_t l )
```

5.352.4.7 rewind()

```
void MHASignal::loop_wavefragment_t::rewind ( ) [inline]
```

5.352.4.8 locate_end()

```
void MHASignal::loop_wavefragment_t::locate_end ( ) [inline]
```

5.352.4.9 is_playback_active()

```
bool MHASignal::loop_wavefragment_t::is_playback_↩
active ( ) const [inline]
```

5.352.5 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.352.5.1 playback_channels `std::vector<int> MHASignal::loop_wavefragment_t::playback_channels [private]`

5.352.5.2 b_loop `bool MHASignal::loop_wavefragment_t::b_loop [private]`

5.352.5.3 pos `unsigned int MHASignal::loop_wavefragment_t::pos [private]`

5.352.5.4 intern_level `MHASignal::waveform_t MHASignal::loop_wavefragment_t::intern_level [private]`

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.353 MHASignal::matrix_t Class Reference

n-dimensional matrix with real or complex floating point values.

Inheritance diagram for MHASignal::matrix_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **matrix_t** (unsigned int nrows, unsigned int ncols, bool b_is_complex=true)
  Create a two-dimensional matrix.
- **matrix_t** (const **mha_spec_t** &spec)
  Create a two-dimensional matrix from a spectrum, copy values.
- **matrix_t** (const **MHASignal::uint_vector_t** & **size**, bool b_is_complex=true)
  Create n-dimensional matrix, descriped by size argument.
- **matrix_t** (const **MHASignal::matrix_t** &)
- **matrix_t** (const uint8_t ∗buf, unsigned int len)
  Construct from memory area.
- ∼**matrix_t** ()
- **MHASignal::matrix_t** & **operator=** (const **MHASignal::matrix_t** &)
- **MHASignal::matrix_t** & **operator=** (const **comm_var_t** &v)
  Fill matrix with data of an AC variable object.
- **comm_var_t** **get_comm_var** ()
  Return a AC communication variable pointing to the data of the current matrix.
- unsigned int **dimension** () const
  Return the dimension of the matrix.
- unsigned int **size** (unsigned int k) const
  Return the size of the matrix.
- unsigned int **get_nelements** () const
  Return total number of elements.
- bool **is_same_size** (const **MHASignal::matrix_t** &)
  Test if matrix has same size as other.
- bool **iscomplex** () const
  Return information about complexity.
- **mha_real_t** & **real** (const **MHASignal::uint_vector_t** &index)
  Access real part of an element in a n-dimensional matrix.
- **mha_real_t** & **imag** (const **MHASignal::uint_vector_t** &index)
  Access imaginary part of an element in a n-dimensional matrix.
- **mha_complex_t** & **operator()** (const **MHASignal::uint_vector_t** &index)
  Access complex value of an element in a n-dimensional matrix.
- const **mha_real_t** & **real** (const **MHASignal::uint_vector_t** &index) const
  Access real part of an element in a n-dimensional matrix.
- const **mha_real_t** & **imag** (const **MHASignal::uint_vector_t** &index) const
  Access imaginary part of an element in a n-dimensional matrix.
- const **mha_complex_t** & **operator()** (const **MHASignal::uint_vector_t** &index) const
  Access complex value of an element in a n-dimensional matrix.
- **mha_real_t** & **real** (unsigned int row, unsigned int col)
  Access real part of an element in a two-dimensional matrix.
- **mha_real_t** & **imag** (unsigned int row, unsigned int col)
  Access imaginary part of an element in a two-dimensional matrix.
- **mha_complex_t** & **operator()** (unsigned int row, unsigned int col)
  Access complex value of an element in a two-dimensional matrix.

© 2005-2021 HörTech gGmbH, Oldenburg

- const **mha_real_t** & **real** (unsigned int row, unsigned int col) const
  Access real part of an element in a two-dimensional matrix.
- const **mha_real_t** & **imag** (unsigned int row, unsigned int col) const
  Access imaginary part of an element in a two-dimensional matrix.
- const **mha_complex_t** & **operator()** (unsigned int row, unsigned int col) const
  Access complex value of an element in a two-dimensional matrix.
- unsigned int **get_nreals** () const
- unsigned int **get_index** (unsigned int row, unsigned int col) const
- unsigned int **get_index** (const **MHASignal::uint_vector_t** &index) const
- unsigned int **numbytes** () const
  Return number of bytes needed to store into memory.
- unsigned int **write** (uint8_t *buf, unsigned int len) const
  Copy to memory area.
- const **mha_real_t** * **get_rdata** () const
  Return pointer of real data.
- const **mha_complex_t** * **get_cdata** () const
  Return pointer of complex data.

Private Attributes

- uint32_t **complex_ofs**
- uint32_t **nelements**
- union {
  mha_real_t * **rdata**
  mha_complex_t * **cdata**
  };

Additional Inherited Members

5.353.1 Detailed Description

n-dimensional matrix with real or complex floating point values.

Warning

The member functions **imag()** (p. 1232) and operator() should only be called if the matrix is defined to hold complex values.

5.353.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.353.2.1 matrix_t() [1/5]

```
MHASignal::matrix_t::matrix_t (
    unsigned int nrows,
    unsigned int ncols,
    bool b_is_complex = true )
```

Create a two-dimensional matrix.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

nrows	Number of rows
ncols	Number of columns
b_is_complex	Add space for complex values

5.353.2.2 matrix_t() [2/5]

```
MHASignal::matrix_t::matrix_t (
    const mha_spec_t & spec )
```

Create a two-dimensional matrix from a spectrum, copy values.

Parameters

spec	Source spectrum structure

5.353.2.3 matrix_t() [3/5]

```
MHASignal::matrix_t::matrix_t (
    const MHASignal::uint_vector_t & size,
    bool b_is_complex = true )
```

Create n-dimensional matrix, descriped by size argument.

Parameters

size	Size vector
b_is_complex	Add space for complex values

5.353.2.4 matrix_t() [4/5]

```
MHASignal::matrix_t::matrix_t (
    const MHASignal::matrix_t & src )
```

5.353.2.5 matrix_t() [5/5]

```
MHASignal::matrix_t::matrix_t (
    const uint8_t * buf,
    unsigned int len )
```

Construct from memory area.

© 2005-2021 HörTech gGmbH, Oldenburg

Warning

This constructor is not real time safe

5.353.2.6 ~matrix_t() `MHASignal::matrix_t::~matrix_t ( )`

5.353.3 Member Function Documentation

5.353.3.1 operator=() [1/2] `matrix_t & MHASignal::matrix_t::operator= ( const MHASignal::matrix_t & src )`

5.353.3.2 operator=() [2/2] `MHASignal::matrix_t & MHASignal::matrix_t::operator= ( const comm_var_t & v )`

Fill matrix with data of an AC variable object.

Parameters

v	Source AC variable (**comm_var_t** (p. 359))

Note

The type and dimension of the AC variable must match the type and dimension of the matrix.

5.353.3.3 get_comm_var() `comm_var_t MHASignal::matrix_t::get_comm_var ( )`

Return a AC communication variable pointing to the data of the current matrix.

Returns

AC variable object (**comm_var_t** (p. 359)), valid for the life time of the matrix.

© 2005-2021 HörTech gGmbH, Oldenburg

5.353.3.4 dimension() `unsigned int MHASignal::matrix_t::dimension ( ) const [inline]`

Return the dimension of the matrix.

Returns

Dimension of the matrix

5.353.3.5 size() `unsigned int MHASignal::matrix_t::size ( unsigned int k ) const [inline]`

Return the size of the matrix.

Parameters

k	Dimension

Returns

Size of the matrix in dimension k

5.353.3.6 get_nelements() `unsigned int MHASignal::matrix_t::get_nelements ( ) const`

Return total number of elements.

5.353.3.7 is_same_size() `bool MHASignal::matrix_t::is_same_size ( const MHASignal::matrix_t & src )`

Test if matrix has same size as other.

5.353.3.8 iscomplex() `bool MHASignal::matrix_t::iscomplex ( ) const [inline]`

Return information about complexity.

5.353.3.9 real() [1/4] `mha_real_t& MHASignal::matrix_t::real ( const MHASignal::uint_vector_t & index ) [inline]`

Access real part of an element in a n-dimensional matrix.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

index	Index vector

5.353.3.10 imag() [1/4]

`mha_real_t& MHASignal::matrix_t::imag ( const MHASignal::uint_vector_t & index ) [inline]`

Access imaginary part of an element in a n-dimensional matrix.

Parameters

index	Index vector

5.353.3.11 operator()() [1/4]

`mha_complex_t& MHASignal::matrix_t::operator() ( const MHASignal::uint_vector_t & index ) [inline]`

Access complex value of an element in a n-dimensional matrix.

Parameters

index	Index vector

5.353.3.12 real() [2/4]

`const mha_real_t& MHASignal::matrix_t::real ( const MHASignal::uint_vector_t & index ) const [inline]`

Access real part of an element in a n-dimensional matrix.

Parameters

index	Index vector

5.353.3.13 imag() [2/4]

`const mha_real_t& MHASignal::matrix_t::imag ( const MHASignal::uint_vector_t & index ) const [inline]`

© 2005-2021 HörTech gGmbH, Oldenburg

Access imaginary part of an element in a n-dimensional matrix.

Parameters

index	Index vector

5.353.3.14 operator()() [2/4]

```
const mha_complex_t& MHASignal::matrix_t::operator()
(
    const MHASignal::uint_vector_t & index ) const [inline]
```

Access complex value of an element in a n-dimensional matrix.

Parameters

index	Index vector

5.353.3.15 real() [3/4]

```
mha_real_t& MHASignal::matrix_t::real (
    unsigned int row,
    unsigned int col ) [inline]
```

Access real part of an element in a two-dimensional matrix.

Parameters

row	Row number of element
col	Column number of element

5.353.3.16 imag() [3/4]

```
mha_real_t& MHASignal::matrix_t::imag (
    unsigned int row,
    unsigned int col ) [inline]
```

Access imaginary part of an element in a two-dimensional matrix.

Parameters

row	Row number of element
col	Column number of element

© 2005-2021 HörTech gGmbH, Oldenburg

5.353.3.17 operator()() [3/4]

```
mha_complex_t& MHASignal::matrix_t::operator() (
unsigned int row,
unsigned int col ) [inline]
```

Access complex value of an element in a two-dimensional matrix.

Parameters

row	Row number of element
col	Column number of element

5.353.3.18 real() [4/4]

```
const mha_real_t& MHASignal::matrix_t::real (
unsigned int row,
unsigned int col ) const [inline]
```

Access real part of an element in a two-dimensional matrix.

Parameters

row	Row number of element
col	Column number of element

5.353.3.19 imag() [4/4]

```
const mha_real_t& MHASignal::matrix_t::imag (
unsigned int row,
unsigned int col ) const [inline]
```

Access imaginary part of an element in a two-dimensional matrix.

Parameters

row	Row number of element
col	Column number of element

© 2005-2021 HörTech gGmbH, Oldenburg

5.353.3.20 operator()() [4/4]

```
const mha_complex_t& MHASignal::matrix_t::operator()
(
    unsigned int row,
    unsigned int col ) const [inline]
```

Access complex value of an element in a two-dimensional matrix.

Parameters

row	Row number of element
col	Column number of element

5.353.3.21 get_nreals()

```
unsigned int MHASignal::matrix_t::get_nreals ( ) const [inline]
```

5.353.3.22 get_index() [1/2]

```
unsigned int MHASignal::matrix_t::get_index (
    unsigned int row,
    unsigned int col ) const
```

5.353.3.23 get_index() [2/2]

```
unsigned int MHASignal::matrix_t::get_index (
    const MHASignal::uint_vector_t & index ) const
```

5.353.3.24 numbytes()

```
unsigned int MHASignal::matrix_t::numbytes ( ) const
```

Return number of bytes needed to store into memory.

5.353.3.25 write()

```
unsigned int MHASignal::matrix_t::write (
    uint8_t * buf,
    unsigned int len ) const
```

Copy to memory area.

© 2005-2021 HörTech gGmbH, Oldenburg

5.353.3.26 get_rdata() `const mha_real_t* MHASignal::matrix_t::get_rdata ( ) const [inline]`

Return pointer of real data.

5.353.3.27 get_cdata() `const mha_complex_t* MHASignal::matrix_t::get_cdata ( ) const [inline]`

Return pointer of complex data.

5.353.4 Member Data Documentation

5.353.4.1 complex_ofs `uint32_t MHASignal::matrix_t::complex_ofs [private]`

5.353.4.2 nelements `uint32_t MHASignal::matrix_t::nelements [private]`

5.353.4.3 rdata `mha_real_t* MHASignal::matrix_t::rdata`

5.353.4.4 cdata `mha_complex_t* MHASignal::matrix_t::cdata`

5.353.4.5 "@1 `union { ... } [private]`

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.354 MHASignal::minphase_t Class Reference

Minimal phase function.

Inheritance diagram for MHASignal::minphase_t:

Public Member Functions

- **minphase_t** (unsigned int fftlen, unsigned int ch)
  Constructor.
- void **operator()** (**mha_spec_t** $*$s)
  Transform input spectrum to a minimal-phase spectrum, discarding the original phase.

Private Attributes

- **MHASignal::waveform_t phase**

Additional Inherited Members

5.354.1 Detailed Description

Minimal phase function.

The output spectrum $Y(f)$ is

$$Y(f) = |X(f)|e^{i\mathcal{H}\{\log|X(f)|\}},$$

with the input spectrum $X(f)$ and the Hilbert transformation $\mathcal{H}\{\cdots\}$.

5.354.2 Constructor & Destructor Documentation

5.354.2.1 minphase_t()

```
MHASignal::minphase_t::minphase_t (
    unsigned int fftlen,
    unsigned int ch )
```

Constructor.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

fftlen	FFT length
ch	Number of channels

5.354.3 Member Function Documentation

5.354.3.1 operator()() `void MHASignal::minphase_t::operator() ( mha_spec_t * s )`

Transform input spectrum to a minimal-phase spectrum, discarding the original phase.

Parameters

s	Spectrum to operate on.

5.354.4 Member Data Documentation

5.354.4.1 phase `MHASignal::waveform_t MHASignal::minphase_t::phase [private]`

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.355 MHASignal::quantizer_t Class Reference

Simple simulation of fixpoint quantization.

Public Member Functions

- **quantizer_t** (unsigned int num_bits)
  Constructor.
- void **operator()** (**mha_wave_t** &s)
  Quantization of a waveform fragment.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- bool **limit**
- **mha_real_t upscale**
- **mha_real_t downscale**
- **mha_real_t up_limit**

5.355.1 Detailed Description

Simple simulation of fixpoint quantization.

5.355.2 Constructor & Destructor Documentation

5.355.2.1 quantizer_t() `MHASignal::quantizer_t::quantizer_t ( unsigned int num_bits )`

Constructor.

Parameters

num_bits	Number of bits to simulate, or zero for limiting to [-1,1] only.

5.355.3 Member Function Documentation

5.355.3.1 operator()() `void MHASignal::quantizer_t::operator() ( mha_wave_t & s )`

Quantization of a waveform fragment.

Parameters

s	Waveform fragment to be quantized.

© 2005-2021 HörTech gGmbH, Oldenburg

5.355.4 Member Data Documentation

5.355.4.1 limit `bool MHASignal::quantizer_t::limit [private]`

5.355.4.2 upscale `mha_real_t MHASignal::quantizer_t::upscale [private]`

5.355.4.3 downscale `mha_real_t MHASignal::quantizer_t::downscale [private]`

5.355.4.4 up_limit `mha_real_t MHASignal::quantizer_t::up_limit [private]`

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.356 MHASignal::ringbuffer_t Class Reference

A ringbuffer class for time domain audio signal, which makes no assumptions with respect to fragment size.

Inheritance diagram for MHASignal::ringbuffer_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **ringbuffer_t** (unsigned frames, unsigned **channels**, unsigned prefilled_frames)
  Creates new ringbuffer for time domain signal.
- unsigned **contained_frames** () const
  number of currently contained frames
- **mha_real_t** & **value** (unsigned frame, unsigned channel)
  Access to value stored in ringbuffer.
- void **discard** (unsigned frames)
  Discards the oldest frames.
- void **write** (**mha_wave_t** &signal)
  Copies the contents of the signal into the ringbuffer if there is enough space.

Private Attributes

- unsigned **next_read_frame_index**
  Index of oldest frame in underlying storage for the ringbuffer.
- unsigned **next_write_frame_index**
  Index of first free frame in underlying storage.

Additional Inherited Members

5.356.1 Detailed Description

A ringbuffer class for time domain audio signal, which makes no assumptions with respect to fragment size.

Blocks of audio signal can be placed into the ringbuffer using the **write** (p. 1243) method. Individual audio samples can be accessed and altered using the **value** (p. 1242) method. Blocks of audio data can be deleted from the ringbuffer using the **discard** (p. 1243) method.

5.356.2 Constructor & Destructor Documentation

5.356.2.1 ringbuffer_t()

```
ringbuffer_t::ringbuffer_t (
    unsigned frames,
    unsigned channels,
    unsigned prefilled_frames )
```

Creates new ringbuffer for time domain signal.

Constructor allocates enough storage so that *frames* audio samples can be stored in the ringbuffer.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

frames	Size of ringbuffer in samples per channel. Maximum number of frames that can be stored in the ringbuffer at one time. This number cannot be changed after instance creation.
channels	Number of audio channels.
prefilled_frames	Number of frames to be prefilled with zero values. Many applications of a ringbuffer require the introduction of a delay. In practice, this delay is achieved by inserting silence audio samples (zeros) into the ringbuffer before the start of the actual signal is inserted for the first time.

Exceptions

MHA_Error (p. 764)	if prefilled_frames > frames

5.356.3 Member Function Documentation

5.356.3.1 contained_frames() `unsigned MHASignal::ringbuffer_t::contained_frames ( ) const [inline]`

number of currently contained frames

5.356.3.2 value() `mha_real_t& MHASignal::ringbuffer_t::value ( unsigned frame, unsigned channel ) [inline]`

Access to value stored in ringbuffer.

frame index is relative to the oldest frame stored in the ringbuffer, therefore, the meaning of the *frame* changes when the **discard** (p. 1243) method is called.

Parameters

frame	frame index, 0 corresponds to oldest frame stored.
channel	audio channel

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

reference to contained sample value

Exceptions

MHA_Error *(p. 764)*	if channel or frame out of bounds.

5.356.3.3 discard() `void MHASignal::ringbuffer_t::discard ( unsigned frames ) [inline]`

Discards the oldest frames.

Makes room for new **write** (p. 1243), alters base frame index for **value** (p. 1242)

Parameters

frames	how many frames to discard.

Exceptions

MHA_Error *(p. 764)*	if frames > **contained_frames** (p. 1242)

5.356.3.4 write() `void MHASignal::ringbuffer_t::write ( mha_wave_t & signal ) [inline]`

Copies the contents of the signal into the ringbuffer if there is enough space.

Parameters

signal	New signal to be appended to the signal already present in the ringbuffer

Exceptions

MHA_Error *(p. 764)*	if there is not enough space or if the channel count mismatches. Nothing is copied if the space is insufficient.

© 2005-2021 HörTech gGmbH, Oldenburg

5.356.4 Member Data Documentation

5.356.4.1 next_read_frame_index `unsigned MHASignal::ringbuffer_t::next_read_frame_index [private]`

Index of oldest frame in underlying storage for the ringbuffer.

This value is added to the frame parameter of the **value** (p. 1242) method, and this value is altered when **discard** (p. 1243) is called.

5.356.4.2 next_write_frame_index `unsigned MHASignal::ringbuffer_t::next_write_frame_index [private]`

Index of first free frame in underlying storage.

Next frame to be stored will be placed here.

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.357 MHASignal::schroeder_t Class Reference

Schroeder tone complex class.

Inheritance diagram for MHASignal::schroeder_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Types

- enum **sign_t** { **up**, **down** }
  Enumerator for sign of Schroeder tone complex sweep direction.
- typedef float(∗ **groupdelay_t**) (float f, float fmin, float fmax)
  Function type for group delay definition.

Public Member Functions

- **schroeder_t** (unsigned int len, unsigned int **channels**=1, **schroeder_t::sign_t** sign= **up**, **mha_real_t** speed=1)
  Constructor.
- **schroeder_t** (unsigned int len, unsigned int **channels**=1, **schroeder_t::groupdelay_t** freqfun= **MHASignal::schroeder_t::identity**, float fmin=0, float fmax=1, float eps=1e-10)
  Construct create Schroeder tone complex from a given frequency function.

Static Public Member Functions

- static float **identity** (float x, float, float)
- static float **log_up** (float x, float fmin, float fmax)
- static float **log_down** (float x, float fmin, float fmax)

Additional Inherited Members

5.357.1 Detailed Description

Schroeder tone complex class.

The Schroeder tone complex is a sweep defined in the sampled spectrum:

$$\Phi(f) = \sigma 2\pi\tau(2f/f_s)^{2\alpha}, \quad S(f) = e^{i\Phi(f)}$$

f is the sampled frequency in Hz, σ is the sign of the sweep (-1 for up sweep, +1 for down sweep), τ is the sweep duration in samples, f_s is the sampling rate in Hz and α is the relative sweep speed.

5.357.2 Member Typedef Documentation

5.357.2.1 groupdelay_t `typedef float(* MHASignal::schroeder_t::groupdelay_t) (float f, float fmin, float fmax)`

Function type for group delay definition.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

f	Frequency relative to Nyquist frequency.
fmin	Minimum frequency relative to Nyquist frequency.
fmax	Maximum frequency relative to Nyquist frequency.

5.357.3 Member Enumeration Documentation

5.357.3.1 sign_t `enum` **`MHASignal::schroeder_t::sign_t`**

Enumerator for sign of Schroeder tone complex sweep direction.

Enumerator

up	Sweep from zero to Nyquist frequency ($\sigma = -1$)
down	Sweep from Nyquist frequency to zero ($\sigma = +1$)

5.357.4 Constructor & Destructor Documentation

5.357.4.1 schroeder_t() [1/2]

```
MHASignal::schroeder_t::schroeder_t (
    unsigned int len,
    unsigned int channels = 1,
    schroeder_t::sign_t sign = up,
    mha_real_t speed = 1 )
```

Constructor.

Parameters of the Schroeder tone complex are configured in the constructor.

Parameters

len	Length τ of the Schroeder tone complex in samples
channels	Number of channels
sign	Sign σ of Schroeder sweep
speed	Relative speed α (curvature of phase function)

© 2005-2021 HörTech gGmbH, Oldenburg

5.357.4.2 schroeder_t() [2/2]

```
MHASignal::schroeder_t::schroeder_t (
    unsigned int len,
    unsigned int channels = 1,
    schroeder_t::groupdelay_t freqfun = MHASignal::schroeder_t::identity,
    float fmin = 0,
    float fmax = 1,
    float eps = 1e-10 )
```

Construct create Schroeder tone complex from a given frequency function.

The frequency function $g(f)$ defines the sweep speed and sign (based on the group delay). It must be defined in the interval [0,1) and should return values in the interval [0,1].

$$\Phi(f) = -4\pi\tau \int_0^\tau g(f)\,\mathrm{d}f, \quad S(f) = e^{i\Phi(f)}$$

Parameters

len	Length τ of the Schroeder tone complex in samples.
channels	Number of channels.
freqfun	Frequency function $g(f)$.
fmin	Start frequency (relative to Nyquist frequency).
fmax	End frequency (relative to Nyquist frequency).
eps	Stability constant for frequency ranges not covered by Schroeder tone complex.

5.357.5 Member Function Documentation

5.357.5.1 identity()

```
static float MHASignal::schroeder_t::identity (
    float x,
    float ,
    float  ) [inline], [static]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.357.5.2 log_up()

```
static float MHASignal::schroeder_t::log_up (
    float x,
    float fmin,
    float fmax ) [inline], [static]
```

5.357.5.3 log_down()

```
static float MHASignal::schroeder_t::log_down (
    float x,
    float fmin,
    float fmax ) [inline], [static]
```

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.358 MHASignal::spectrum_t Class Reference

a signal processing class for spectral data (based on **mha_spec_t** (p. 794))

Inheritance diagram for MHASignal::spectrum_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **spectrum_t** (const unsigned int &frames, const unsigned int & **channels**)
  constructor of spectrum class
- **spectrum_t** (const **mha_spec_t** &)
  Copy constructor.
- **spectrum_t** (const **MHASignal::spectrum_t** &)
  Copy constructor.
- **spectrum_t** (const std::vector< **mha_complex_t** > &)
- virtual ~**spectrum_t** (void)
- **mha_complex_t** & **operator()** (unsigned int f, unsigned int ch)
  Access to element.
- **mha_complex_t** & **operator[]** (unsigned int k)
  Access to a single element, direct index into data buffer.
- **mha_complex_t** & **value** (unsigned int f, unsigned int ch)
  Access to element.
- void **copy** (const **mha_spec_t** &)
  copy all elements from a spectrum
- void **copy_channel** (const **mha_spec_t** &s, unsigned sch, unsigned dch)
  Copy one channel of a given spectrum signal to a target channel.
- void **export_to** (**mha_spec_t** &)
  copy elements to spectrum structure
- void **scale** (const unsigned int &, const unsigned int &, const unsigned int &, const **mha_real_t** &)
  scale section [a,b) in channel "ch" by "val"
- void **scale_channel** (const unsigned int &, const **mha_real_t** &)
  scale all elements in one channel

Additional Inherited Members

5.358.1 Detailed Description

a signal processing class for spectral data (based on **mha_spec_t** (p. 794))

5.358.2 Constructor & Destructor Documentation

5.358.2.1 spectrum_t() [1/4]

```
spectrum_t::spectrum_t (
    const unsigned int & frames,
    const unsigned int & channels )
```

constructor of spectrum class

Allocates buffers and initializes memory to zeros.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

frames	number of frames (fft bins) in one channel. Number of Frames is usually fftlen / 2 + 1
channels	number of channels

5.358.2.2 spectrum_t() [2/4]

```
spectrum_t::spectrum_t (
    const mha_spec_t & src ) [explicit]
```

Copy constructor.

5.358.2.3 spectrum_t() [3/4]

```
spectrum_t::spectrum_t (
    const MHASignal::spectrum_t & src )
```

Copy constructor.

5.358.2.4 spectrum_t() [4/4]

```
spectrum_t::spectrum_t (
    const std::vector< mha_complex_t > & src )
```

5.358.2.5 ~spectrum_t()

```
spectrum_t::~spectrum_t (
    void ) [virtual]
```

Reimplemented in **MHA_AC::spectrum_t** (p. 736).

5.358.3 Member Function Documentation

5.358.3.1 operator()()

```
mha_complex_t& MHASignal::spectrum_t::operator() (
    unsigned int f,
    unsigned int ch ) [inline]
```

Access to element.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

f	Bin number
ch	Channel number

Returns

Reference to element

5.358.3.2 operator[]()

```
mha_complex_t& MHASignal::spectrum_t::operator[] (
    unsigned int k ) [inline]
```

Access to a single element, direct index into data buffer.

Parameters

k	Buffer index

Returns

Reference to element

5.358.3.3 value()

```
mha_complex_t& MHASignal::spectrum_t::value (
    unsigned int f,
    unsigned int ch ) [inline]
```

Access to element.

Parameters

f	Bin number
ch	Channel number

Returns

Reference to element

© 2005-2021 HörTech gGmbH, Oldenburg

5.358.3.4 copy()

```
void spectrum_t::copy (
    const mha_spec_t & src )
```

copy all elements from a spectrum

Parameters

src	input spectrum

5.358.3.5 copy_channel()

```
void spectrum_t::copy_channel (
    const mha_spec_t & s,
    unsigned sch,
    unsigned dch )
```

Copy one channel of a given spectrum signal to a target channel.

Parameters

s	Input spectrum signal
sch	Channel index in source signal
dch	Channel index in destination (this) signal

5.358.3.6 export_to()

```
void spectrum_t::export_to (
    mha_spec_t & dest )
```

copy elements to spectrum structure

Parameters

dest	destination spectrum structure

5.358.3.7 scale()

```
void spectrum_t::scale (
    const unsigned int & a,
    const unsigned int & b,
    const unsigned int & ch,
    const mha_real_t & val )
```

© 2005-2021 HörTech gGmbH, Oldenburg

scale section [a,b) in channel "ch" by "val"

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

a	starting frame
b	end frame (excluded)
ch	channel number
val	scale factor

5.358.3.8 scale_channel()

```
void spectrum_t::scale_channel (
    const unsigned int & ch,
    const mha_real_t & src )
```

scale all elements in one channel

Parameters

ch	channel number
src	scale factor

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.359 MHASignal::stat_t Class Reference

Inheritance diagram for MHASignal::stat_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **stat_t** (const unsigned int &frames, const unsigned int & **channels**)
- void **mean** (**mha_wave_t** &m)
- void **mean_std** (**mha_wave_t** &m, **mha_wave_t** &s)
- void **push** (const **mha_wave_t** &)
- void **push** (const **mha_real_t** &x, const unsigned int &k, const unsigned int &ch)

Private Attributes

- **MHASignal::waveform_t n**
- **MHASignal::waveform_t sum**
- **MHASignal::waveform_t sum2**

5.359.1 Constructor & Destructor Documentation

5.359.1.1 stat_t()

```
MHASignal::stat_t::stat_t (
    const unsigned int & frames,
    const unsigned int & channels )
```

5.359.2 Member Function Documentation

5.359.2.1 mean()

```
void MHASignal::stat_t::mean (
    mha_wave_t & m )
```

5.359.2.2 mean_std()

```
void MHASignal::stat_t::mean_std (
    mha_wave_t & m,
    mha_wave_t & s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.359.2.3 push() [1/2]

```
void MHASignal::stat_t::push (
    const mha_wave_t & x )
```

5.359.2.4 push() [2/2]

```
void MHASignal::stat_t::push (
    const mha_real_t & x,
    const unsigned int & k,
    const unsigned int & ch )
```

5.359.3 Member Data Documentation

5.359.3.1 n

```
MHASignal::waveform_t MHASignal::stat_t::n [private]
```

5.359.3.2 sum

```
MHASignal::waveform_t MHASignal::stat_t::sum [private]
```

5.359.3.3 sum2

```
MHASignal::waveform_t MHASignal::stat_t::sum2 [private]
```

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

5.360 MHASignal::subsample_delay_t Class Reference

implements subsample delay in spectral domain.

Public Member Functions

- **subsample_delay_t** (const std::vector< float > &subsample_delay, unsigned fftlen)
  Constructor computes complex phase factors to apply to achieve subsample delay.
- void **process** (**mha_spec_t** *s)
  Apply the phase_gains to s to achieve the subsample delay.
- void **process** (**mha_spec_t** *s, unsigned idx)
  Apply the pase gains to channel idx in s to achieve the subsample delay in channel idx.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **spectrum_t phase_gains**
  The complex factors to apply to achieve the necessary phase shift.

Private Attributes

- unsigned **last_complex_bin**
  index of the last complex fft bin for the used fft length.

5.360.1 Detailed Description

implements subsample delay in spectral domain.

When transformed back to the time domain, the signal is delayed by the configured fraction of a sample. This operation must not be used in a smoothgains bracket.

5.360.2 Constructor & Destructor Documentation

5.360.2.1 subsample_delay_t()

```
MHASignal::subsample_delay_t::subsample_delay_t (
    const std::vector< float > & subsample_delay,
    unsigned fftlen )
```

Constructor computes complex phase factors to apply to achieve subsample delay.

Parameters

subsample_delay	The subsample delay to apply. -0.5 <= subsample_delay <= 0.5
fftlen	FFT length

Exceptions

MHA_Error *(p. 764)*	if the parameters are out of range

5.360.3 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.360.3.1 process() [1/2] void MHASignal::subsample_delay_t::process (mha_spec_t * s)

Apply the phase_gains to s to achieve the subsample delay.

5.360.3.2 process() [2/2] void MHASignal::subsample_delay_t::process (mha_spec_t * s, unsigned idx)

Apply the pase gains to channel idx in s to achieve the subsample delay in channel idx.

Parameters

s	signal
idx	channel index, 0-based

Exceptions

MHA_Error *(p. 764)*	if idx >= s->num_channels

5.360.4 Member Data Documentation

5.360.4.1 phase_gains spectrum_t MHASignal::subsample_delay_t::phase_gains

The complex factors to apply to achieve the necessary phase shift.

5.360.4.2 last_complex_bin unsigned MHASignal::subsample_delay_t::last_complex_bin [private]

index of the last complex fft bin for the used fft length.

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.361 MHASignal::uint_vector_t Class Reference

Vector of unsigned values, used for size and index description of n-dimensional matrixes.

Inheritance diagram for MHASignal::uint_vector_t:

Public Member Functions

- **uint_vector_t** (unsigned int len)
  Constructor, initializes all elements to zero.
- **uint_vector_t** (const **uint_vector_t** &)
- **uint_vector_t** (const uint8_t *buf, unsigned int len)
  Construct from memory area.
- ~**uint_vector_t** ()
- bool **operator==** (const **uint_vector_t** &) const
  Check for equality.
- **uint_vector_t** & **operator=** (const **uint_vector_t** &)
  *Assign from other **uint_vector_t** (p. 1259).*
- unsigned int **get_length** () const
  Return the length of the vector.
- const uint32_t & **operator[]** (unsigned int k) const
  Read-only access to elements.
- uint32_t & **operator[]** (unsigned int k)
  Access to elements.
- unsigned int **numbytes** () const
  Return number of bytes needed to store into memory.
- unsigned int **write** (uint8_t *buf, unsigned int len) const
  Copy to memory area.
- const uint32_t * **getdata** () const
  Return pointer to the data field.

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Attributes

- uint32_t **length**
- uint32_t * **data**

5.361.1 Detailed Description

Vector of unsigned values, used for size and index description of n-dimensional matrixes.

5.361.2 Constructor & Destructor Documentation

5.361.2.1 uint_vector_t() [1/3]

```
MHASignal::uint_vector_t::uint_vector_t (
    unsigned int len )
```

Constructor, initializes all elements to zero.

Parameters

len	Length of vector.

5.361.2.2 uint_vector_t() [2/3]

```
MHASignal::uint_vector_t::uint_vector_t (
    const uint_vector_t & src )
```

5.361.2.3 uint_vector_t() [3/3]

```
MHASignal::uint_vector_t::uint_vector_t (
    const uint8_t * buf,
    unsigned int len )
```

Construct from memory area.

Warning

This constructor is not real time safe

© 2005-2021 HörTech gGmbH, Oldenburg

5.361.2.4 ~uint_vector_t() `MHASignal::uint_vector_t::~uint_vector_t ( )`

5.361.3 Member Function Documentation

5.361.3.1 operator==() `bool MHASignal::uint_vector_t::operator== ( const uint_vector_t & src ) const`

Check for equality.

5.361.3.2 operator=() `uint_vector_t & MHASignal::uint_vector_t::operator= ( const uint_vector_t & src )`

Assign from other **uint_vector_t** (p. 1259).

Warning

This assignment will fail if the lengths mismatch.

5.361.3.3 get_length() `unsigned int MHASignal::uint_vector_t::get_length ( ) const [inline]`

Return the length of the vector.

5.361.3.4 operator[]() [1/2] `const uint32_t& MHASignal::uint_vector_t::operator[] ( unsigned int k ) const [inline]`

Read-only access to elements.

© 2005-2021 HörTech gGmbH, Oldenburg

5.361.3.5 operator[]() [2/2] uint32_t& MHASignal::uint_vector_t::operator[] (unsigned int *k*) [inline]

Access to elements.

5.361.3.6 numbytes() unsigned int MHASignal::uint_vector_t::numbytes ( ) const

Return number of bytes needed to store into memory.

5.361.3.7 write() unsigned int MHASignal::uint_vector_t::write (uint8_t * *buf*, unsigned int *len*) const

Copy to memory area.

5.361.3.8 getdata() const uint32_t* MHASignal::uint_vector_t::getdata ( ) const [inline]

Return pointer to the data field.

5.361.4 Member Data Documentation

5.361.4.1 length uint32_t MHASignal::uint_vector_t::length [protected]

5.361.4.2 data uint32_t* MHASignal::uint_vector_t::data [protected]

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.362 MHASignal::waveform_t Class Reference

signal processing class for waveform data (based on **mha_wave_t** (p. 840))

Inheritance diagram for MHASignal::waveform_t:

Public Member Functions

- **waveform_t** (const unsigned int &frames, const unsigned int & **channels**)
  *constructor of **waveform_t** (p. 1263)*
- **waveform_t** (const **mhaconfig_t** &cf)
  Constructor to create a waveform from plugin configuration.
- **waveform_t** (const **mha_wave_t** &src)
  *Copy contructor for **mha_wave_t** (p. 840) source.*
- **waveform_t** (const **MHASignal::waveform_t** &src)
  Copy contructor.
- **waveform_t** (const std::vector< **mha_real_t** > &src)
  Copy contructor for std::vector<mha_real_t> source.
- virtual ~**waveform_t** (void)
- std::vector< **mha_real_t** > **flatten** () const
- **operator std::vector**< **mha_real_t** > () const
- void **operator=** (const **mha_real_t** &v)
- **mha_real_t** & **operator[]** (unsigned int k)
- const **mha_real_t** & **operator[]** (unsigned int k) const
- **mha_real_t** & **value** (unsigned int t, unsigned int ch)

© 2005-2021 HörTech gGmbH, Oldenburg

Element accessor.

- **mha_real_t** & **operator()** (unsigned int t, unsigned int ch)

  Element accessor.
- const **mha_real_t** & **value** (unsigned int t, unsigned int ch) const

  Constant element accessor.
- const **mha_real_t** & **operator()** (unsigned int t, unsigned int ch) const

  Constant element accessor.
- **mha_real_t** **sum** (const unsigned int &a, const unsigned int &b)

  sum of all elements between [a,b) in all channels
- **mha_real_t** **sum** (const unsigned int &a, const unsigned int &b, const unsigned int &ch)

  sum of all elements between [a,b) in channel ch
- **mha_real_t** **sum** ()

  sum of all elements
- **mha_real_t** **sumsqr** ()

  sum of square of all elements
- **mha_real_t** **sum_channel** (const unsigned int &)

  return sum of all elements in one channel
- void **assign** (const unsigned int &k, const unsigned int &ch, const **mha_real_t** &val)

  set frame "k" in channel "ch" to value "val"
- void **assign** (const **mha_real_t** &)

  set all elements to value
- void **assign_frame** (const unsigned int &k, const **mha_real_t** &val)

  assign value "val" to frame k in all channels
- void **assign_channel** (const unsigned int &c, const **mha_real_t** &val)

  assign value "val" to channel ch in all frames
- void **copy** (const std::vector< **mha_real_t** > &v)
- void **copy** (const **mha_wave_t** &)

  copy data from source into current waveform
- void **copy** (const **mha_wave_t** *)
- void **copy_channel** (const **mha_wave_t** &, unsigned int, unsigned int)

  Copy one channel of a given waveform signal to a target channel.
- void **copy_from_at** (unsigned int, unsigned int, const **mha_wave_t** &, unsigned int)

  Copy part of the source signal into part of this waveform object.
- void **export_to** (**mha_wave_t** &)

  *copy data into allocated **mha_wave_t*** (p. 840) *structure*
- void **limit** (const **mha_real_t** & **min**, const **mha_real_t** & **max**)

  limit target to range [min,max]
- void **power** (const **waveform_t** &)

  transform waveform signal (in Pa) to squared signal (in W/m^2)
- void **powspec** (const **mha_spec_t** &)

  get the power spectrum (in W/m^2) from a complex spectrum
- void **scale** (const unsigned int &a, const unsigned int &b, const unsigned int &ch, const **mha_real_t** &val)

  scale section [a,b) in channel "ch" by "val"
- void **scale** (const unsigned int &k, const unsigned int &ch, const **mha_real_t** &val)

© 2005-2021 HörTech gGmbH, Oldenburg

scale one element

- void **scale_channel** (const unsigned int &, const **mha_real_t** &)
  scale one channel of target with a scalar
- void **scale_frame** (const unsigned int &, const **mha_real_t** &)
- unsigned int **get_size** () const

Additional Inherited Members

5.362.1 Detailed Description

signal processing class for waveform data (based on **mha_wave_t** (p. 840))

5.362.2 Constructor & Destructor Documentation

5.362.2.1 waveform_t() [1/5]

```
waveform_t::waveform_t (
    const unsigned int & frames,
    const unsigned int & channels )
```

constructor of **waveform_t** (p. 1263)

Allocates buffer memory and initializes values to zero.

Parameters

frames	number of frames in each channel
channels	number of channels

5.362.2.2 waveform_t() [2/5]

```
waveform_t::waveform_t (
    const mhaconfig_t & cf ) [explicit]
```

Constructor to create a waveform from plugin configuration.

Parameters

cf	Plugin configuration

© 2005-2021 HörTech gGmbH, Oldenburg

5.362.2.3 waveform_t() [3/5] `waveform_t::waveform_t ( const mha_wave_t & src ) [explicit]`

Copy contructor for **mha_wave_t** (p. 840) source.

5.362.2.4 waveform_t() [4/5] `waveform_t::waveform_t ( const MHASignal::waveform_t & src )`

Copy contructor.

5.362.2.5 waveform_t() [5/5] `waveform_t::waveform_t ( const std::vector< mha_real_t > & src )`

Copy contructor for std::vector<mha_real_t> source.

A waveform structure with a single channel is created, the length is equal to the number of elements in the source vector.

5.362.2.6 ~waveform_t() `waveform_t::~waveform_t ( void ) [virtual]`

Reimplemented in **MHA_AC::waveform_t** (p. 740).

5.362.3 Member Function Documentation

5.362.3.1 flatten() `std::vector< mha_real_t > waveform_t::flatten ( ) const`

© 2005-2021 HörTech gGmbH, Oldenburg

5.362.3.2 operator std::vector< mha_real_t >()

```
MHASignal::waveform_t::operator std::vector< mha_real_t > ( ) const [explicit]
```

5.362.3.3 operator=()

```
void MHASignal::waveform_t::operator= ( const mha_real_t & v ) [inline]
```

5.362.3.4 operator[]() [1/2]

```
mha_real_t& MHASignal::waveform_t::operator[] ( unsigned int k ) [inline]
```

5.362.3.5 operator[]() [2/2]

```
const mha_real_t& MHASignal::waveform_t::operator[] ( unsigned int k ) const [inline]
```

5.362.3.6 value() [1/2]

```
mha_real_t& MHASignal::waveform_t::value ( unsigned int t, unsigned int ch ) [inline]
```

Element accessor.

Parameters

t	Frame number
ch	Channel number

Returns

Reference to element

5.362.3.7 operator()() [1/2]

```
mha_real_t& MHASignal::waveform_t::operator() ( unsigned int t, unsigned int ch ) [inline]
```

Element accessor.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

t	Frame number
ch	Channel number

Returns

Reference to element

5.362.3.8 value() [2/2]

```
const mha_real_t& MHASignal::waveform_t::value (
    unsigned int t,
    unsigned int ch ) const [inline]
```

Constant element accessor.

Parameters

t	Frame number
ch	Channel number

Returns

Reference to element

5.362.3.9 operator()() [2/2]

```
const mha_real_t& MHASignal::waveform_t::operator() (
    unsigned int t,
    unsigned int ch ) const [inline]
```

Constant element accessor.

Parameters

t	Frame number
ch	Channel number

Returns

Reference to element

© 2005-2021 HörTech gGmbH, Oldenburg

5.362.3.10 sum() [1/3]

```
mha_real_t waveform_t::sum (
    const unsigned int & a,
    const unsigned int & b )
```

sum of all elements between [a,b) in all channels

Parameters

a	starting frame
b	end frame (excluded)

Returns

sum

5.362.3.11 sum() [2/3]

```
mha_real_t waveform_t::sum (
    const unsigned int & a,
    const unsigned int & b,
    const unsigned int & ch )
```

sum of all elements between [a,b) in channel ch

Parameters

a	starting frame
b	end frame (exluded)
ch	channel number

Returns

sum

5.362.3.12 sum() [3/3]

```
mha_real_t waveform_t::sum ( )
```

sum of all elements

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

sum of all elements

5.362.3.13 sumsqr() `mha_real_t waveform_t::sumsqr ( )`

sum of square of all elements

Returns

sum of square of all elements

5.362.3.14 sum_channel() `mha_real_t waveform_t::sum_channel ( const unsigned int & ch )`

return sum of all elements in one channel

Parameters

ch	channel number

Returns

sum

5.362.3.15 assign() [1/2] `void waveform_t::assign ( const unsigned int & k, const unsigned int & ch, const mha_real_t & val )`

set frame "k" in channel "ch" to value "val"

Parameters

k	frame number
ch	channel number
val	new value

© 2005-2021 HörTech gGmbH, Oldenburg

5.362.3.16 assign() [2/2]

```
void waveform_t::assign (
const mha_real_t & val )
```

set all elements to value

Parameters

val	new value

5.362.3.17 assign_frame()

```
void waveform_t::assign_frame (
const unsigned int & k,
const mha_real_t & val )
```

assign value "val" to frame k in all channels

Parameters

k	frame number
val	new value

5.362.3.18 assign_channel()

```
void waveform_t::assign_channel (
const unsigned int & ch,
const mha_real_t & val )
```

assign value "val" to channel ch in all frames

Parameters

ch	channel number
val	new value

5.362.3.19 copy() [1/3]

```
void waveform_t::copy (
const std::vector< mha_real_t > & v )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.362.3.20 copy() [2/3]

```
void waveform_t::copy (
const mha_wave_t & src )
```

copy data from source into current waveform

Parameters

src	input data (need to be same size as target)

5.362.3.21 copy() [3/3]

```
void waveform_t::copy (
const mha_wave_t * src )
```

5.362.3.22 copy_channel()

```
void waveform_t::copy_channel (
const mha_wave_t & src,
unsigned int src_channel,
unsigned int dest_channel )
```

Copy one channel of a given waveform signal to a target channel.

Parameters

src	Input waveform signal
src_channel	Channel in source signal
dest_channel	Channel number in destination signal

5.362.3.23 copy_from_at()

```
void waveform_t::copy_from_at (
unsigned int to_pos,
unsigned int len,
const mha_wave_t & src,
unsigned int from_pos )
```

Copy part of the source signal into part of this waveform object.

Source and target have to have the same number of channels.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

to_pos	Offset in target
len	Number of frames copied
src	Source
from_pos	Offset in source

5.362.3.24 export_to()

```
void waveform_t::export_to (
    mha_wave_t & dest )
```

copy data into allocated **mha_wave_t** (p. 840) structure

Parameters

dest	destination structure

5.362.3.25 limit()

```
void waveform_t::limit (
    const mha_real_t & min,
    const mha_real_t & max )
```

limit target to range [min,max]

Parameters

min	lower limit
max	upper limit

5.362.3.26 power()

```
void waveform_t::power (
    const waveform_t & src )
```

transform waveform signal (in Pa) to squared signal (in W/m^2)

Parameters

src	linear waveform signal (in Pa)

© 2005-2021 HörTech gGmbH, Oldenburg

5.362.3.27 powspec()

```
void waveform_t::powspec (
const mha_spec_t & src )
```

get the power spectrum (in W/m^2) from a complex spectrum

Parameters

src	complex spectrum (normalized to Pa)

5.362.3.28 scale() [1/2]

```
void waveform_t::scale (
const unsigned int & a,
const unsigned int & b,
const unsigned int & ch,
const mha_real_t & val )
```

scale section [a,b) in channel "ch" by "val"

Parameters

a	starting frame
b	end frame (excluded)
ch	channel number
val	scale factor

5.362.3.29 scale() [2/2]

```
void waveform_t::scale (
const unsigned int & k,
const unsigned int & ch,
const mha_real_t & val )
```

scale one element

Parameters

k	frame number
ch	channel number
val	scale factor

© 2005-2021 HörTech gGmbH, Oldenburg

5.362.3.30 scale_channel()

```
void waveform_t::scale_channel (
const unsigned int & ch,
const mha_real_t & src )
```

scale one channel of target with a scalar

Parameters

ch	channel number
src	factor

5.362.3.31 scale_frame()

```
void waveform_t::scale_frame (
const unsigned int & frame,
const mha_real_t & val )
```

5.362.3.32 get_size()

```
unsigned int MHASignal::waveform_t::get_size ( ) const [inline]
```

The documentation for this class was generated from the following files:

- **mha_signal.hh**
- **mha_signal.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.363 MHASndFile::sf_t Class Reference

Inheritance diagram for MHASndFile::sf_t:

Public Member Functions

- **sf_t** (const std::string &fname)
- ∼**sf_t** ()

Public Attributes

- SNDFILE ∗ **sf**

5.363.1 Constructor & Destructor Documentation

5.363.1.1 sf_t() `MHASndFile::sf_t::sf_t ( const std::string & fname )`

5.363.1.2 ∼sf_t() `MHASndFile::sf_t::∼sf_t ( )`

© 2005-2021 HörTech gGmbH, Oldenburg

5.363.2 Member Data Documentation

5.363.2.1 sf `SNDFILE* MHASndFile::sf_t::sf`

The documentation for this class was generated from the following files:

- **mhasndfile.h**
- **mhasndfile.cpp**

5.364 MHASndFile::sf_wave_t Class Reference

Inheritance diagram for MHASndFile::sf_wave_t:

Public Member Functions

- **sf_wave_t** (const std::string &fname, **mha_real_t** peaklevel_db, unsigned int maxlen=std::numeric_limits< unsigned int >:: **max**(), unsigned int startpos=0, std↩ ::vector< int > channel_map=std::vector< int >())

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.364.1 Constructor & Destructor Documentation

5.364.1.1 sf_wave_t()

```
MHASndFile::sf_wave_t::sf_wave_t (
    const std::string & fname,
    mha_real_t peaklevel_db,
    unsigned int maxlen = std::numeric_limits<unsigned int>::  max(),
    unsigned int startpos = 0,
    std::vector< int > channel_map = std::vector<int>() )
```

The documentation for this class was generated from the following files:

- **mhasndfile.h**
- **mhasndfile.cpp**

5.365 MHATableLookup::linear_table_t Class Reference

Class for interpolation with equidistant x values.

Inheritance diagram for MHATableLookup::linear_table_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **linear_table_t** (void)

  *contructor creates an empty **linear_table_t** (p. 1278) object.*
- **mha_real_t lookup** (**mha_real_t** x) const

  look up the y value that is stored for the mesh point where x is lower than or equal to the x value given here.
- **mha_real_t interp** (**mha_real_t** x) const

  interpolate y value for the given x value.
- ∼**linear_table_t** (void)

  destructor
- void **set_xmin** (**mha_real_t xmin**)

  set the x value for the first mesh point.
- void **add_entry** (**mha_real_t** y)

  set the y value for the next mesh point.
- void **set_xmax** (**mha_real_t xmax**)

  this sets the x value for a past-the-end, not added mesh point.
- void **prepare** (void)

  prepare computes the x distance of the mesh points based on the values given to set_xmin, set_xmax, and the number of times that add_entry was called.
- void **clear** (void)

  clear resets the state of this object to the state directly after construction.

Protected Attributes

- **mha_real_t** ∗ **vy**
- unsigned int **len**

Private Attributes

- vector< **mha_real_t** > **vec_y**
- **mha_real_t xmin**
- **mha_real_t xmax**
- **mha_real_t scalefac**

Additional Inherited Members

5.365.1 Detailed Description

Class for interpolation with equidistant x values.

This class can be used for linear interpolation tasks where the mesh points are known for equidistant x values.

Before the class can be used for interpolation, it has to be filled with the y values for the mesh points, the x range has to be specified, and when all values are given, the prepare method has to be called so that the object can determine the distance between x values from the range and the number of mesh points given.

Only after prepare has returned, the object may be used for interpolation.

© 2005-2021 HörTech gGmbH, Oldenburg

5.365.2 Constructor & Destructor Documentation

5.365.2.1 linear_table_t() `linear_table_t::linear_table_t ( void )`

contructor creates an empty **linear_table_t** (p. 1278) object.

add_entry, set_xmin, set_xmax and prepare methods have to be called before the object can be used to lookup and interpolate values.

5.365.2.2 ~linear_table_t() `linear_table_t::~linear_table_t ( void )`

destructor

5.365.3 Member Function Documentation

5.365.3.1 lookup() `mha_real_t linear_table_t::lookup ( mha_real_t x ) const [virtual]`

look up the y value that is stored for the mesh point where x is lower than or equal to the x value given here.

This method does not extrapolate, so for x $<$ xmin, the y value for xmin is returned. For all x greater than the x of the last mesh point, the y value of the last mesh point is returned.

Precondition

prepare must have been called before lookup may be called.

Implements **MHATableLookup::table_t** (p. 1284).

© 2005-2021 HörTech gGmbH, Oldenburg

5.365.3.2 interp()

```
mha_real_t linear_table_t::interp (
    mha_real_t x ) const [virtual]
```

interpolate y value for the given x value.

The y values for the neighbouring mesh points are looked up and linearly interpolated. For x values outside the range of mesh points, the y value is extrapolated from the nearest two mesh points.

Precondition

prepare must have been called before interp may be called.

Implements **MHATableLookup::table_t** (p. 1284).

5.365.3.3 set_xmin()

```
void linear_table_t::set_xmin (
    mha_real_t xmin )
```

set the x value for the first mesh point.

Must be called before prepare can be called.

5.365.3.4 add_entry()

```
void linear_table_t::add_entry (
    mha_real_t y )
```

set the y value for the next mesh point.

Must be called at least twice before prepare can be called.

5.365.3.5 set_xmax()

```
void linear_table_t::set_xmax (
    mha_real_t xmax )
```

this sets the x value for a past-the-end, not added mesh point.

Example:

```
t.set_xmin(100);
t.add_entry(0); // mesh point {100,0}
t.add_entry(1); // mesh point {110,1}
// the next mesh point would be at x=120, but we do not add this
t.set_xmax(120); // the x where the next mesh point would be
t.prepare();
```

now, t.interp(100) == 0; t.interp(110) == 1; t.interp(105) == 0.5;

© 2005-2021 HörTech gGmbH, Oldenburg

5.365.3.6 prepare() `void linear_table_t::prepare ( void )`

prepare computes the x distance of the mesh points based on the values given to set_xmin, set_xmax, and the number of times that add_entry was called.

Precondition

set_xmin, set_xmax, add_entry functions must have been called before calling prepare, add_entry must have been called at least twice.

Only after this method has been called, interp or lookup may be called.

5.365.3.7 clear() `void linear_table_t::clear ( void ) [virtual]`

clear resets the state of this object to the state directly after construction.

mesh entries and x range are deleted.

interp and lookup may not be called after this function has been called unless prepare and before that its precondition methods are called again.

Implements **MHATableLookup::table_t** (p. 1284).

5.365.4 Member Data Documentation

5.365.4.1 vy `mha_real_t* MHATableLookup::linear_table_t::vy [protected]`

5.365.4.2 len `unsigned int MHATableLookup::linear_table_t::len [protected]`

5.365.4.3 vec_y `vector< mha_real_t> MHATableLookup::linear_table_t::vec_y [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.365.4.4 xmin `mha_real_t MHATableLookup::linear_table_t::xmin [private]`

5.365.4.5 xmax `mha_real_t MHATableLookup::linear_table_t::xmax [private]`

5.365.4.6 scalefac `mha_real_t MHATableLookup::linear_table_t::scalefac [private]`

The documentation for this class was generated from the following files:

- **mha_tablelookup.hh**
- **mha_tablelookup.cpp**

5.366 MHATableLookup::table_t Class Reference

Inheritance diagram for MHATableLookup::table_t:

Public Member Functions

- **table_t** (void)
- virtual ~**table_t** (void)
- virtual **mha_real_t lookup** (**mha_real_t**) const =0
- virtual **mha_real_t interp** (**mha_real_t**) const =0

© 2005-2021 HörTech gGmbH, Oldenburg

Protected Member Functions

- virtual void **clear** (void)=0

5.366.1 Constructor & Destructor Documentation

5.366.1.1 table_t() `table_t::table_t ( void )`

5.366.1.2 ~table_t() `table_t::~table_t ( void ) [virtual]`

5.366.2 Member Function Documentation

5.366.2.1 lookup() `virtual` **`mha_real_t`** `MHATableLookup::table_t::lookup (` **`mha_real_t`** `) const [pure virtual]`

Implemented in **MHATableLookup::xy_table_t** (p. 1286), and **MHATableLookup::linear_table_t** (p. 1280).

5.366.2.2 interp() `virtual` **`mha_real_t`** `MHATableLookup::table_t::interp (` **`mha_real_t`** `) const [pure virtual]`

Implemented in **MHATableLookup::xy_table_t** (p. 1287), and **MHATableLookup::linear_table_t** (p. 1280).

© 2005-2021 HörTech gGmbH, Oldenburg

5.366.2.3 clear() `virtual void MHATableLookup::table_t::clear ( void ) [protected], [pure virtual]`

Implemented in **MHATableLookup::linear_table_t** (p. 1282), and **MHATableLookup::xy_table_t** (p. 1288).

The documentation for this class was generated from the following files:

- **mha_tablelookup.hh**
- **mha_tablelookup.cpp**

5.367 MHATableLookup::xy_table_t Class Reference

Class for interpolation with non-equidistant x values.

Inheritance diagram for MHATableLookup::xy_table_t:

Public Member Functions

- **xy_table_t** ()
- **mha_real_t lookup** (**mha_real_t** x) const

  Return the y-value at the position of the nearest x value below input.
- **mha_real_t interp** (**mha_real_t** x) const

  Linear interpolation function.
- void **add_entry** (**mha_real_t** x, **mha_real_t** y)

  Add a single x-y pair entry.
- void **add_entry** (**mha_real_t** *pVX, **mha_real_t** *pVY, unsigned int len)

© 2005-2021 HörTech gGmbH, Oldenburg

Add multiple entries at once.

- void **clear** ()

  Clear the table and transformation functions.
- void **set_xfun** (float(∗pXFun)(float))

  Set transformation function for x values.
- void **set_yfun** (float(∗pYFun)(float))

  Set transformation function for y values during insertion.
- void **set_xyfun** (float(∗pYFun)(float, float))

  Set transformation function for y values during insertion, based on x and y values.
- std::pair< **mha_real_t**, **mha_real_t** > **get_xlimits** () const

  returns the min and max x of all mesh points that are stored in the lookup table, i.e.

Private Attributes

- std::map< **mha_real_t**, **mha_real_t** > **mXY**
- float(∗ **xfun**)(float)
- float(∗ **yfun**)(float)
- float(∗ **xyfun**)(float, float)

Additional Inherited Members

5.367.1 Detailed Description

Class for interpolation with non-equidistant x values.

Linear interpolation of the x-y table is performed. A transformation of x and y-values is possible; if a transformation function is provided for the x-values, the same function is applied to the argument of **xy_table_t::interp()** (p. 1287) and **xy_table_t::lookup()** (p. 1286). The transformation of y values is applied only during insertion into the table. Two functions for y-transformation can be provided: a simple transformation which depends only on the y values, or a transformation which takes both (non-transformed) x and y value as an argument. The two-argument transformation is applied before the one-argument transformation.

5.367.2 Constructor & Destructor Documentation

5.367.2.1 xy_table_t() `xy_table_t::xy_table_t ( )`

5.367.3 Member Function Documentation

5.367.3.1 lookup() `mha_real_t xy_table_t::lookup ( mha_real_t x ) const [virtual]`

Return the y-value at the position of the nearest x value below input.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

x	Input value

Returns

y value at nearest x value below input.

Implements **MHATableLookup::table_t** (p. 1284).

5.367.3.2 interp()

```
mha_real_t xy_table_t::interp (
    mha_real_t x ) const [virtual]
```

Linear interpolation function.

Parameters

x	x value

Returns

interpolated y value

Implements **MHATableLookup::table_t** (p. 1284).

5.367.3.3 add_entry() [1/2]

```
void xy_table_t::add_entry (
    mha_real_t x,
    mha_real_t y )
```

Add a single x-y pair entry.

Parameters

x	x value
y	corresponding y value

© 2005-2021 HörTech gGmbH, Oldenburg

5.367.3.4 add_entry() [2/2]

```
void xy_table_t::add_entry (
    mha_real_t * pVX,
    mha_real_t * pVY,
    unsigned int uLength )
```

Add multiple entries at once.

Parameters

pVX	array of x values
pVY	array of y values
uLength	Length of x and y arrays

5.367.3.5 clear()

```
void xy_table_t::clear ( ) [virtual]
```

Clear the table and transformation functions.

Implements **MHATableLookup::table_t** (p. 1284).

5.367.3.6 set_xfun()

```
void xy_table_t::set_xfun (
    float(*)(float) fun )
```

Set transformation function for x values.

Parameters

fun	Transformation function.

5.367.3.7 set_yfun()

```
void xy_table_t::set_yfun (
    float(*)(float) fun )
```

Set transformation function for y values during insertion.

Parameters

fun	Transformation function.

© 2005-2021 HörTech gGmbH, Oldenburg

5.367.3.8 set_xyfun() `void xy_table_t::set_xyfun ( float(*)(float, float) fun )`

Set transformation function for y values during insertion, based on x and y values.

Parameters

fun	Transformation function.

5.367.3.9 get_xlimits() `std::pair< mha_real_t, mha_real_t> MHATableLookup::xy_table_t::get_xlimits ( ) const [inline]`

returns the min and max x of all mesh points that are stored in the lookup table, i.e.

after transformation with xfun, if any. Not real-time safe

5.367.4 Member Data Documentation

5.367.4.1 mXY `std::map< mha_real_t, mha_real_t> MHATableLookup::xy_table_t::mXY [private]`

5.367.4.2 xfun `float(* MHATableLookup::xy_table_t::xfun) (float) [private]`

5.367.4.3 yfun `float(* MHATableLookup::xy_table_t::yfun) (float) [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.367.4.4 xyfun `float(* MHATableLookup::xy_table_t::xyfun) (float, float) [private]`

The documentation for this class was generated from the following files:

- **mha_tablelookup.hh**
- **mha_tablelookup.cpp**

5.368 MHAWindow::bartlett_t Class Reference

Bartlett window.

Inheritance diagram for MHAWindow::bartlett_t:

Public Member Functions

- **bartlett_t** (unsigned int n)

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.368.1 Detailed Description

Bartlett window.

5.368.2 Constructor & Destructor Documentation

5.368.2.1 bartlett_t() `MHAWindow::bartlett_t::bartlett_t ( unsigned int n ) [inline]`

The documentation for this class was generated from the following file:

- **mha_windowparser.h**

5.369 MHAWindow::base_t Class Reference

Common base for window types.

Inheritance diagram for MHAWindow::base_t:

Public Member Functions

- **base_t** (unsigned int len)
  Constructor.
- **base_t** (const **MHAWindow::base_t** &src)
  Copy constructor.
- void **operator()** (**mha_wave_t** &) const
  Apply window to waveform segment (reference)
- void **operator()** (**mha_wave_t** ∗) const
  Apply window to waveform segment (pointer)
- void **ramp_begin** (**mha_wave_t** &) const
  Apply a ramp at the begining.
- void **ramp_end** (**mha_wave_t** &) const
  Apply a ramp at the end.

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.369.1 Detailed Description

Common base for window types.

5.369.2 Constructor & Destructor Documentation

5.369.2.1 base_t() [1/2] MHAWindow::base_t::base_t (unsigned int *len*)

Constructor.

Parameters

len	Window length in samples.

5.369.2.2 base_t() [2/2] MHAWindow::base_t::base_t (const **MHAWindow::base_t** & *src*)

Copy constructor.

Parameters

src	Source to be copied

5.369.3 Member Function Documentation

5.369.3.1 operator()() [1/2] void MHAWindow::base_t::operator() (**mha_wave_t** & *s*) const

Apply window to waveform segment (reference)

© 2005-2021 HörTech gGmbH, Oldenburg

5.369.3.2 operator()() [2/2] void MHAWindow::base_t::operator() (mha_wave_t * s) const

Apply window to waveform segment (pointer)

5.369.3.3 ramp_begin() void MHAWindow::base_t::ramp_begin (mha_wave_t & s) const

Apply a ramp at the begining.

5.369.3.4 ramp_end() void MHAWindow::base_t::ramp_end (mha_wave_t & s) const

Apply a ramp at the end.

The documentation for this class was generated from the following files:

- **mha_windowparser.h**
- **mha_windowparser.cpp**

5.370 MHAWindow::blackman_t Class Reference

Blackman window.

© 2005-2021 HörTech gGmbH, Oldenburg

Inheritance diagram for MHAWindow::blackman_t:

Public Member Functions

- **blackman_t** (unsigned int n)

Additional Inherited Members

5.370.1 Detailed Description

Blackman window.

5.370.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.370.2.1 blackman_t()

```
MHAWindow::blackman_t::blackman_t (
    unsigned int n ) [inline]
```

The documentation for this class was generated from the following file:

- **mha_windowparser.h**

5.371 MHAWindow::fun_t Class Reference

Generic window based on a generator function.

Inheritance diagram for MHAWindow::fun_t:

Public Member Functions

- **fun_t** (unsigned int n, float(∗fun)(float), float xmin=-1, float xmax=1, bool min_included=true, bool max_included=false)

  Constructor.

Additional Inherited Members

5.371.1 Detailed Description

Generic window based on a generator function.

The generator function should return a valid window function in the interval [-1,1[.

5.371.2 Constructor & Destructor Documentation

5.371.2.1 fun_t()

```
MHAWindow::fun_t::fun_t (
    unsigned int n,
    float(*)(float) fun,
    float xmin = -1,
    float xmax = 1,
    bool min_included = true,
    bool max_included = false )
```

Constructor.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

n	Window length
fun	Generator function, i.e. **MHAWindow::hanning()** (p. 155)
xmin	Start value of window, i.e. -1 for full window or 0 for fade-out ramp.
xmax	Last value of window, i.e. 1 for full window
min_included	Flag if minimum value is included
max_included	Flag if maximum value is included

The documentation for this class was generated from the following files:

- **mha_windowparser.h**
- **mha_windowparser.cpp**

5.372 MHAWindow::hamming_t Class Reference

Hamming window.

Inheritance diagram for MHAWindow::hamming_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **hamming_t** (unsigned int n)

Additional Inherited Members

5.372.1 Detailed Description

Hamming window.

5.372.2 Constructor & Destructor Documentation

5.372.2.1 hamming_t() `MHAWindow::hamming_t::hamming_t ( unsigned int n ) [inline]`

The documentation for this class was generated from the following file:

- **mha_windowparser.h**

5.373 MHAWindow::hanning_t Class Reference

von-Hann window

© 2005-2021 HörTech gGmbH, Oldenburg

Inheritance diagram for MHAWindow::hanning_t:

Public Member Functions

- **hanning_t** (unsigned int n)

Additional Inherited Members

5.373.1 Detailed Description

von-Hann window

5.373.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.373.2.1 hanning_t() `MHAWindow::hanning_t::hanning_t ( unsigned int n ) [inline]`

The documentation for this class was generated from the following file:

- **mha_windowparser.h**

5.374 MHAWindow::rect_t Class Reference

Rectangular window.

Inheritance diagram for MHAWindow::rect_t:

Public Member Functions

- **rect_t** (unsigned int n)

Additional Inherited Members

5.374.1 Detailed Description

Rectangular window.

© 2005-2021 HörTech gGmbH, Oldenburg

5.374.2 Constructor & Destructor Documentation

5.374.2.1 rect_t() `MHAWindow::rect_t::rect_t ( unsigned int n ) [inline]`

The documentation for this class was generated from the following file:

- **mha_windowparser.h**

5.375 MHAWindow::user_t Class Reference

User defined window.

Inheritance diagram for MHAWindow::user_t:

Public Member Functions

- **user_t** (const std::vector< **mha_real_t** > &wnd)
  Constructor.

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.375.1 Detailed Description

User defined window.

5.375.2 Constructor & Destructor Documentation

5.375.2.1 user_t()

```
MHAWindow::user_t::user_t (
    const std::vector< mha_real_t > & wnd )
```

Constructor.

Parameters

wnd	User defined window

The documentation for this class was generated from the following files:

- **mha_windowparser.h**
- **mha_windowparser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.376 multibandcompressor::fftfb_plug_t Class Reference

Inheritance diagram for multibandcompressor::fftfb_plug_t:

Public Member Functions

- **fftfb_plug_t** (**MHAOvlFilter::fftfb_vars_t** &, const **mhaconfig_t** &cfg, **algo_comm_t** ac, std::string alg)
- void **insert** ()

Private Attributes

- **MHA_AC::waveform_t cfv**

  vector of nominal center frequencies / Hz
- **MHA_AC::waveform_t efv**

  vector of edge frequencies / Hz
- **MHA_AC::waveform_t bwv**

  vector of band-weigths (sum of squared fft-bin-weigths)/num_frames

Additional Inherited Members

5.376.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.376.1.1 fftfb_plug_t()

```
multibandcompressor::fftfb_plug_t::fftfb_plug_t (
    MHAOvlFilter::fftfb_vars_t & vars,
    const mhaconfig_t & cfg,
    algo_comm_t ac,
    std::string alg )
```

5.376.2 Member Function Documentation

5.376.2.1 insert()

```
void multibandcompressor::fftfb_plug_t::insert ( )
```

5.376.3 Member Data Documentation

5.376.3.1 cfv

```
MHA_AC::waveform_t multibandcompressor::fftfb_plug_t::cfv [private]
```

vector of nominal center frequencies / Hz

5.376.3.2 efv

```
MHA_AC::waveform_t multibandcompressor::fftfb_plug_t::efv [private]
```

vector of edge frequencies / Hz

5.376.3.3 bwv

```
MHA_AC::waveform_t multibandcompressor::fftfb_plug_t::bwv [private]
```

vector of band-weigths (sum of squared fft-bin-weigths)/num_frames

The documentation for this class was generated from the following file:

- **multibandcompressor.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.377 multibandcompressor::interface_t Class Reference

Inheritance diagram for multibandcompressor::interface_t:

Public Member Functions

- **interface_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHA_AC::int_t num_channels**
- **DynComp::dc_afterburn_t burn**
- **MHAEvents::patchbay_t**< **interface_t** > **patchbay**
- std::string **algo**
- **MHAParser::mhapluginloader_t plug**
- **plugin_signals_t** ∗ **plug_sigs**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.377.1 Constructor & Destructor Documentation

5.377.1.1 interface_t()

```
multibandcompressor::interface_t::interface_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Default values are set and MHA configuration variables registered into the parser.

Parameters

ac↩ _	algorithm communication handle
th	chain name
al	algorithm name

5.377.2 Member Function Documentation

5.377.2.1 prepare()

```
void multibandcompressor::interface_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< fftfb_plug_t >** (p. 1153).

5.377.2.2 release()

```
void multibandcompressor::interface_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< fftfb_plug_t >** (p. 1154).

5.377.2.3 process()

```
mha_spec_t * multibandcompressor::interface_t::process (
    mha_spec_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.377.2.4 update_cfg() `void multibandcompressor::interface_t::update_cfg ( ) [private]`

5.377.3 Member Data Documentation

5.377.3.1 num_channels `MHA_AC::int_t multibandcompressor::interface_t::num_channels [private]`

5.377.3.2 burn `DynComp::dc_afterburn_t multibandcompressor::interface_t::burn [private]`

5.377.3.3 patchbay `MHAEvents::patchbay_t< interface_t> multibandcompressor::interface_t::patchbay [private]`

5.377.3.4 algo `std::string multibandcompressor::interface_t::algo [private]`

5.377.3.5 plug `MHAParser::mhapluginloader_t multibandcompressor::interface_t::plug [private]`

5.377.3.6 plug_sigs `plugin_signals_t* multibandcompressor::interface_t::plug_sigs [private]`

The documentation for this class was generated from the following file:

- **multibandcompressor.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.378 multibandcompressor::plugin_signals_t Class Reference

Public Member Functions

- **plugin_signals_t** (unsigned int **channels**, unsigned int bands)
- void **update_levels** (**MHAOvlFilter::fftfb_t** ∗, **mha_spec_t** ∗s_in)
- void **apply_gains** (**MHAOvlFilter::fftfb_t** ∗, **DynComp::dc_afterburn_t** &burn, **mha↩_spec_t** ∗s_out)

Public Attributes

- **mha_wave_t** ∗ **plug_output**

Private Attributes

- **MHASignal::waveform_t plug_level**
- **MHASignal::waveform_t gain**

5.378.1 Constructor & Destructor Documentation

5.378.1.1 plugin_signals_t()

```
multibandcompressor::plugin_signals_t::plugin_signals↩
_t (
            unsigned int channels,
            unsigned int bands )
```

5.378.2 Member Function Documentation

5.378.2.1 update_levels()

```
void multibandcompressor::plugin_signals_t::update_levels
(
            MHAOvlFilter::fftfb_t * pFb,
            mha_spec_t * s_in )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.378.2.2 apply_gains()

```
void multibandcompressor::plugin_signals_t::apply_gains (
    MHAOvlFilter::fftfb_t * pFb,
    DynComp::dc_afterburn_t & burn,
    mha_spec_t * s_out )
```

5.378.3 Member Data Documentation

5.378.3.1 plug_level

MHASignal::waveform_t multibandcompressor::plugin_signals_t↩ ::plug_level [private]

5.378.3.2 gain

MHASignal::waveform_t multibandcompressor::plugin_signals_t::gain [private]

5.378.3.3 plug_output

mha_wave_t* multibandcompressor::plugin_signals_t::plug_↩ output

The documentation for this class was generated from the following file:

- **multibandcompressor.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.379 nlms_t Class Reference

Inheritance diagram for nlms_t:

Public Member Functions

- **nlms_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

Private Member Functions

- void **update** ()

Private Attributes

- **MHAParser::float_t rho**
- **MHAParser::float_t c**
- **MHAParser::int_t ntaps**
- **MHAParser::string_t name_u**
- **MHAParser::string_t name_d**

© 2005-2021 HörTech gGmbH, Oldenburg

- **MHAParser::kw_t normtype**
- **MHAParser::kw_t estimtype**
- **MHAParser::float_t lambda_smoothing_power**
- **MHAParser::string_t name_e**
- **MHAParser::string_t name_f**
- **MHAParser::int_t n_no_update**
- std::string **algo**
- **MHAEvents::patchbay_t< nlms_t > patchbay**

Additional Inherited Members

5.379.1 Constructor & Destructor Documentation

5.379.1.1 nlms_t()

```
nlms_t::nlms_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.379.2 Member Function Documentation

5.379.2.1 prepare()

```
void nlms_t::prepare (
    mhaconfig_t & cf ) [virtual]
```

Implements **MHAPlugin::plugin_t< rt_nlms_t >** (p. 1153).

5.379.2.2 release()

```
void nlms_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< rt_nlms_t >** (p. 1154).

5.379.2.3 process()

```
mha_wave_t * nlms_t::process (
    mha_wave_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.379.2.4 update() `void nlms_t::update ( ) [private]`

5.379.3 Member Data Documentation

5.379.3.1 rho `MHAParser::float_t nlms_t::rho [private]`

5.379.3.2 c `MHAParser::float_t nlms_t::c [private]`

5.379.3.3 ntaps `MHAParser::int_t nlms_t::ntaps [private]`

5.379.3.4 name_u `MHAParser::string_t nlms_t::name_u [private]`

5.379.3.5 name_d `MHAParser::string_t nlms_t::name_d [private]`

5.379.3.6 normtype `MHAParser::kw_t nlms_t::normtype [private]`

5.379.3.7 estimtype `MHAParser::kw_t nlms_t::estimtype [private]`

5.379.3.8 lambda_smoothing_power `MHAParser::float_t nlms_t::lambda_smoothing↩_power [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.379.3.9 name_e `MHAParser::string_t nlms_t::name_e [private]`

5.379.3.10 name_f `MHAParser::string_t nlms_t::name_f [private]`

5.379.3.11 n_no_update `MHAParser::int_t nlms_t::n_no_update [private]`

5.379.3.12 algo `std::string nlms_t::algo [private]`

5.379.3.13 patchbay `MHAEvents::patchbay_t< nlms_t> nlms_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **nlms_wave.cpp**

5.380 noise_psd_estimator::noise_psd_estimator_if_t Class Reference

Inheritance diagram for noise_psd_estimator::noise_psd_estimator_if_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **noise_psd_estimator_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAParser::float_t alphaPH1mean**
- **MHAParser::float_t alphaPSD**
- **MHAParser::float_t q**
- **MHAParser::float_t xiOptDb**
- std::string **name**
- **MHAEvents::patchbay_t**< **noise_psd_estimator_if_t** > **patchbay**

Additional Inherited Members

5.380.1 Constructor & Destructor Documentation

5.380.1.1 noise_psd_estimator_if_t()

```
noise_psd_estimator::noise_psd_estimator_if_t::noise_psd_estimator_if_t (
            algo_comm_t iac,
            const std::string & configured_name )
```

5.380.2 Member Function Documentation

5.380.2.1 process()

```
mha_spec_t * noise_psd_estimator::noise_psd_estimator_if_t::process (
            mha_spec_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.380.2.2 prepare() `void noise_psd_estimator::noise_psd_estimator_if_t::prepare ( mhaconfig_t & cf ) [virtual]`

Implements **MHAPlugin::plugin_t< noise_psd_estimator_t >** (p. 1153).

5.380.2.3 update_cfg() `void noise_psd_estimator::noise_psd_estimator_if_t::update↩_cfg ( ) [private]`

5.380.3 Member Data Documentation

5.380.3.1 alphaPH1mean `MHAParser::float_t noise_psd_estimator::noise_psd_↩estimator_if_t::alphaPH1mean [private]`

5.380.3.2 alphaPSD `MHAParser::float_t noise_psd_estimator::noise_psd_estimator↩_if_t::alphaPSD [private]`

5.380.3.3 q `MHAParser::float_t noise_psd_estimator::noise_psd_estimator_if_t::q [private]`

5.380.3.4 xiOptDb `MHAParser::float_t noise_psd_estimator::noise_psd_estimator_↩if_t::xiOptDb [private]`

5.380.3.5 name `std::string noise_psd_estimator::noise_psd_estimator_if_t::name [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.380.3.6 patchbay `MHAEvents::patchbay_t< noise_psd_estimator_if_t> noise_psd_estimator::noise_psd_estimator_if_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **noise_psd_estimator.cpp**

5.381 noise_psd_estimator::noise_psd_estimator_t Class Reference

Public Member Functions

- **noise_psd_estimator_t** (const **mhaconfig_t** &cf, **algo_comm_t** ac, const std::string &name, float alphaPH1mean, float alphaPSD, float q, float xiOptDb)
- void **process** (**mha_spec_t** ∗noisyDftFrame)
- void **insert** ()

Private Attributes

- **MHASignal::waveform_t noisyPer**
- **MHASignal::waveform_t PH1mean**
- **MHA_AC::waveform_t noisePow**
- **MHA_AC::waveform_t inputPow**
- **MHA_AC::waveform_t snrPost1Debug**
- **MHA_AC::waveform_t GLRDebug**
- **MHA_AC::waveform_t PH1Debug**
- **MHA_AC::waveform_t estimateDebug**
- **MHA_AC::spectrum_t inputSpec**
- float **alphaPH1mean_**
- float **alphaPSD_**
- float **priorFact**
- float **xiOpt**
- float **logGLRFact**
- float **GLRexp**
- int **frameno**

5.381.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.381.1.1 noise_psd_estimator_t()

```
noise_psd_estimator::noise_psd_estimator_t↩
::noise_psd_estimator_t (
            const  mhaconfig_t & cf,
             algo_comm_t ac,
            const std::string & name,
            float alphaPH1mean,
            float alphaPSD,
            float q,
            float xiOptDb )
```

5.381.2 Member Function Documentation

5.381.2.1 process()

```
void noise_psd_estimator::noise_psd_estimator_t::process (
            mha_spec_t * noisyDftFrame )
```

5.381.2.2 insert()

```
void noise_psd_estimator::noise_psd_estimator_t::insert ( ) [inline]
```

5.381.3 Member Data Documentation

5.381.3.1 noisyPer

```
MHASignal::waveform_t noise_psd_estimator::noise_psd_estimator↩
_t::noisyPer [private]
```

5.381.3.2 PH1mean

```
MHASignal::waveform_t noise_psd_estimator::noise_psd_estimator↩
_t::PH1mean [private]
```

5.381.3.3 noisePow

```
MHA_AC::waveform_t noise_psd_estimator::noise_psd_estimator_↩
t::noisePow [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.381.3.4 inputPow

MHA_AC::waveform_t noise_psd_estimator::noise_psd_estimator_t::inputPow [private]

5.381.3.5 snrPost1Debug

MHA_AC::waveform_t noise_psd_estimator::noise_psd_estimator_t::snrPost1Debug [private]

5.381.3.6 GLRDebug

MHA_AC::waveform_t noise_psd_estimator::noise_psd_estimator_t::GLRDebug [private]

5.381.3.7 PH1Debug

MHA_AC::waveform_t noise_psd_estimator::noise_psd_estimator_t::PH1Debug [private]

5.381.3.8 estimateDebug

MHA_AC::waveform_t noise_psd_estimator::noise_psd_estimator_t::estimateDebug [private]

5.381.3.9 inputSpec

MHA_AC::spectrum_t noise_psd_estimator::noise_psd_estimator_t::inputSpec [private]

5.381.3.10 alphaPH1mean_

float noise_psd_estimator::noise_psd_estimator_t::alphaPH1mean_ [private]

5.381.3.11 alphaPSD_

float noise_psd_estimator::noise_psd_estimator_t::alphaPSD_ [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.381.3.12 priorFact `float noise_psd_estimator::noise_psd_estimator_t::priorFact [private]`

5.381.3.13 xiOpt `float noise_psd_estimator::noise_psd_estimator_t::xiOpt [private]`

5.381.3.14 logGLRFact `float noise_psd_estimator::noise_psd_estimator_t::logGLR↩ Fact [private]`

5.381.3.15 GLRexp `float noise_psd_estimator::noise_psd_estimator_t::GLRexp [private]`

5.381.3.16 frameno `int noise_psd_estimator::noise_psd_estimator_t::frameno [private]`

The documentation for this class was generated from the following file:

- **noise_psd_estimator.cpp**

5.382 noise_t Class Reference

Inheritance diagram for noise_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **noise_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **update_cfg** ()

Private Attributes

- **MHAParser::float_t lev**
- **MHAParser::kw_t mode**
- **MHAParser::float_t frozennoise_length**
- **MHAParser::int_t seed**
- **MHAEvents::patchbay_t**< **noise_t** > **patchbay**

Additional Inherited Members

5.382.1 Constructor & Destructor Documentation

5.382.1.1 noise_t()

```
noise_t::noise_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.382.2 Member Function Documentation

5.382.2.1 process() [1/2]

```
mha_wave_t * noise_t::process (
    mha_wave_t * s )
```

5.382.2.2 process() [2/2]

```
mha_spec_t * noise_t::process (
    mha_spec_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.382.2.3 prepare() `void noise_t::prepare ( mhaconfig_t & tf ) [virtual]`

Implements **MHAPlugin::plugin_t< cfg_t >** (p. 1153).

5.382.2.4 update_cfg() `void noise_t::update_cfg ( )`

5.382.3 Member Data Documentation

5.382.3.1 lev `MHAParser::float_t noise_t::lev [private]`

5.382.3.2 mode `MHAParser::kw_t noise_t::mode [private]`

5.382.3.3 frozennoise_length `MHAParser::float_t noise_t::frozennoise_length [private]`

5.382.3.4 seed `MHAParser::int_t noise_t::seed [private]`

5.382.3.5 patchbay `MHAEvents::patchbay_t< noise_t> noise_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **noise.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.383 osc2ac_t Class Reference

Inheritance diagram for osc2ac_t:

Public Member Functions

- **osc2ac_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗s)
- void **process** ()

Private Member Functions

- void **setlock** (bool b)

Private Attributes

- **MHAParser::string_t host**
- **MHAParser::string_t port**
- **MHAParser::vstring_t vars**
- **MHAParser::vint_t size**
- **MHAEvents::patchbay_t**< **osc2ac_t** > **patchbay**
- std::unique_ptr< **osc_server_t** > **srv**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.383.1 Constructor & Destructor Documentation

5.383.1.1 osc2ac_t()

```
osc2ac_t::osc2ac_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.383.2 Member Function Documentation

5.383.2.1 prepare()

```
void osc2ac_t::prepare (
    mhaconfig_t & ) [virtual]
```

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

5.383.2.2 release()

```
void osc2ac_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< int >** (p. 1154).

5.383.2.3 process() [1/3]

```
mha_wave_t* osc2ac_t::process (
    mha_wave_t * s ) [inline]
```

5.383.2.4 process() [2/3]

```
mha_spec_t* osc2ac_t::process (
    mha_spec_t * s ) [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.383.2.5 process() [3/3] void osc2ac_t::process ( )

5.383.2.6 setlock() void osc2ac_t::setlock (bool *b*) [private]

5.383.3 Member Data Documentation

5.383.3.1 host MHAParser::string_t osc2ac_t::host [private]

5.383.3.2 port MHAParser::string_t osc2ac_t::port [private]

5.383.3.3 vars MHAParser::vstring_t osc2ac_t::vars [private]

5.383.3.4 size MHAParser::vint_t osc2ac_t::size [private]

5.383.3.5 patchbay MHAEvents::patchbay_t< osc2ac_t> osc2ac_t::patchbay [private]

5.383.3.6 srv std::unique_ptr< osc_server_t> osc2ac_t::srv [private]

The documentation for this class was generated from the following file:

- **osc2ac.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.384 osc_server_t Class Reference

OSC receiver implemented using liblo.

Public Member Functions

- **osc_server_t** (const std::string &multicast_addr, const std::string &port)
- ∼**osc_server_t** ()
- void **server_stop** ()
- void **server_start** ()
- void **insert_variable** (const std::string &name, unsigned int **size**, **algo_comm_t** hAC)
- void **sync_osc2ac** ()
- void **ac_insert** ()

Static Public Member Functions

- static void **error_h** (int num, const char ∗msg, const char ∗path)

Private Attributes

- std::vector< std::unique_ptr< **osc_variable_t** > > **pVars**
- lo_server_thread **lost**
- bool **is_running**

5.384.1 Detailed Description

OSC receiver implemented using liblo.

5.384.2 Constructor & Destructor Documentation

5.384.2.1 osc_server_t()

```
osc_server_t::osc_server_t (
    const std::string & multicast_addr,
    const std::string & port )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.384.2.2 ~osc_server_t() `osc_server_t::~osc_server_t ( )`

5.384.3 Member Function Documentation

5.384.3.1 server_stop() `void osc_server_t::server_stop ( )`

5.384.3.2 server_start() `void osc_server_t::server_start ( )`

5.384.3.3 insert_variable()

```
void osc_server_t::insert_variable (
    const std::string & name,
    unsigned int size,
    algo_comm_t hAC )
```

5.384.3.4 sync_osc2ac() `void osc_server_t::sync_osc2ac ( )`

5.384.3.5 ac_insert() `void osc_server_t::ac_insert ( )`

5.384.3.6 error_h()

```
void osc_server_t::error_h (
    int num,
    const char * msg,
    const char * path ) [static]
```

5.384.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.384.4.1 pVars `std::vector<std::unique_ptr< osc_variable_t> > osc_server_t::pVars [private]`

5.384.4.2 lost `lo_server_thread osc_server_t::lost [private]`

5.384.4.3 is_running `bool osc_server_t::is_running [private]`

The documentation for this class was generated from the following file:

- **osc2ac.cpp**

5.385 osc_variable_t Class Reference

Class for converting messages received at a single osc address to a single AC variable.

Public Member Functions

- **osc_variable_t** (const **osc_variable_t** &)=delete

  An instance of this class cannot safely be copied.
- **osc_variable_t** (const std::string &name, unsigned int **size**, **algo_comm_t** hAC, lo_server_thread lost)

  Constructor.
- void **sync_osc2ac** ()

  Copies the latest OSC data from the OSC storage to the AC storage.
- void **ac_insert** ()

  Insert/Re-insert the AC variable into AC space.
- int **handler** (const char *types, lo_arg **argv, int argc)

  Callback function called by network thread managed by liblo when a new OSC message has been received.

Static Public Member Functions

- static int **handler** (const char *path, const char *types, lo_arg **argv, int argc, lo_message msg, void *user_data)

  Callback function called by network thread managed by liblo when a new OSC message has been received.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- std::string **acname**
  Name of the ac variable.
- std::string **oscaddr**
  OSC address.
- **MHA_AC::waveform_t ac_data**
  AC variable storage.
- **MHASignal::waveform_t osc_data**
  OSC variable storage.
- std::string **name_**
  Name of AC variable and OSC address without the initial slash.

5.385.1 Detailed Description

Class for converting messages received at a single osc address to a single AC variable.

OSC variables are received asynchronously in a network thread and must not modify their AC variables directly, because MHA plugins may only access their AC variables while executing their prepare, release, or process callbacks.

One osc2ac plugin uses multiple instances of **osc_variable_t** (p. 1326), one for each mapping of an OSC address to an AC variable.

5.385.2 Constructor & Destructor Documentation

5.385.2.1 osc_variable_t() [1/2]

```
osc_variable_t::osc_variable_t (
    const osc_variable_t & ) [delete]
```

An instance of this class cannot safely be copied.

5.385.2.2 osc_variable_t() [2/2]

```
osc_variable_t::osc_variable_t (
    const std::string & name,
    unsigned int size,
    algo_comm_t hAC,
    lo_server_thread lost )
```

Constructor.

Allocates memory.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

name	The name of the AC variable that stores the latest value.
size	Number of elements to copy from OSC message to AC variable.
hAC	Handle of Algorithm Communication Variable space.
lost	libLO Server Thread.

5.385.3 Member Function Documentation

5.385.3.1 sync_osc2ac() `void osc_variable_t::sync_osc2ac ( ) [inline]`

Copies the latest OSC data from the OSC storage to the AC storage.

To be executed during process callback of osc2ac plugin.

5.385.3.2 ac_insert() `void osc_variable_t::ac_insert ( ) [inline]`

Insert/Re-insert the AC variable into AC space.

Should be done in each process callback.

5.385.3.3 handler() [1/2]

```
int osc_variable_t::handler (
    const char * path,
    const char * types,
    lo_arg ** argv,
    int argc,
    lo_message msg,
    void * user_data ) [static]
```

Callback function called by network thread managed by liblo when a new OSC message has been received.

This static method forwards to the instance method by casting user_data to osc_variable_t*.

Parameters

path	Unused.
types	The OSC data type indicator of the received message.
argv	Array of received OSC data.
argc	Number of elements in array of received OSC data.
msg	Unused.
user_data	Pointer to **osc_variable_t** (p. 1326) instance.

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

1 if the message was accepted, 0 if not.

5.385.3.4 handler() [2/2]

```
int osc_variable_t::handler (
    const char * types,
    lo_arg ** argv,
    int argc )
```

Callback function called by network thread managed by liblo when a new OSC message has been received.

This instance method checks if the received data is of expected length and contains only floats, and if yes, copies the data into the buffer osc_data where the latest received data for this osc address is stored until it is either overwritten by the next data for the same osc address or copied to an AC variable.

Parameters

types	The OSC data type indicator of the received message.
argv	Array of received OSC data.
argc	Number of elements in array of received OSC data.

Returns

1 if the message had correct length and contained only floats, 0 if not.

5.385.4 Member Data Documentation

5.385.4.1 acname

```
std::string osc_variable_t::acname [private]
```

Name of the ac variable.

5.385.4.2 oscaddr

```
std::string osc_variable_t::oscaddr [private]
```

OSC address.

© 2005-2021 HörTech gGmbH, Oldenburg

5.385.4.3 ac_data `MHA_AC::waveform_t osc_variable_t::ac_data [private]`

AC variable storage.

5.385.4.4 osc_data `MHASignal::waveform_t osc_variable_t::osc_data [private]`

OSC variable storage.

5.385.4.5 name_ `std::string osc_variable_t::name_ [private]`

Name of AC variable and OSC address without the initial slash.

The documentation for this class was generated from the following file:

- **osc2ac.cpp**

5.386 overlapadd::overlapadd_if_t Class Reference

Inheritance diagram for overlapadd::overlapadd_if_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **overlapadd_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- ~**overlapadd_if_t** ()=default
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- **mha_wave_t** * **process** (**mha_wave_t** *)

Private Member Functions

- void **update** ()
- void **setlock** (bool b)

  Lock/Unlock all configuration variables.

Private Attributes

- **MHAParser::int_t nfft**

  FFT length to be used, zero-padding is FFT length-wndlength.
- **MHAParser::int_t nwnd**

  Window length to be used (overlap is 1-fragsize/wndlength)
- **MHAParser::float_t wndpos**

  Relative position of zero padding (0 end, 0.5 center, 1 start)
- **MHAParser::window_t window**
- **MHAParser::float_t wndexp**
- **MHAParser::window_t zerowindow**
- **MHAParser::bool_t strict_window_ratio**

  Disallow window sizes that are not a multiple of the hop size ("fragsize" in MHA) a by power of two.
- **MHAParser::mhapluginloader_t plugloader**
- **MHAParser::float_mon_t prescale**
- **MHAParser::float_mon_t postscale**
- std::string **algo**
- **mhaconfig_t cf_in**
- **mhaconfig_t cf_out**

Additional Inherited Members

5.386.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.386.1.1 overlapadd_if_t()

```
overlapadd::overlapadd_if_t::overlapadd_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.386.1.2 ~overlapadd_if_t()

```
overlapadd::overlapadd_if_t::~overlapadd_if_t ( ) [default]
```

5.386.2 Member Function Documentation

5.386.2.1 prepare()

```
void overlapadd::overlapadd_if_t::prepare (
    mhaconfig_t & t ) [virtual]
```

Implements **MHAPlugin::plugin_t< overlapadd_t >** (p. 1153).

5.386.2.2 release()

```
void overlapadd::overlapadd_if_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< overlapadd_t >** (p. 1154).

5.386.2.3 process()

```
mha_wave_t * overlapadd::overlapadd_if_t::process (
    mha_wave_t * wave_in )
```

5.386.2.4 update()

```
void overlapadd::overlapadd_if_t::update ( ) [private]
```

5.386.2.5 setlock()

```
void overlapadd::overlapadd_if_t::setlock (
    bool b ) [inline], [private]
```

Lock/Unlock all configuration variables.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

b	Desired lock state

5.386.3 Member Data Documentation

5.386.3.1 nfft `MHAParser::int_t overlapadd::overlapadd_if_t::nfft [private]`

FFT length to be used, zero-padding is FFT length-wndlength.

5.386.3.2 nwnd `MHAParser::int_t overlapadd::overlapadd_if_t::nwnd [private]`

Window length to be used (overlap is 1-fragsize/wndlength)

5.386.3.3 wndpos `MHAParser::float_t overlapadd::overlapadd_if_t::wndpos [private]`

Relative position of zero padding (0 end, 0.5 center, 1 start)

5.386.3.4 window `MHAParser::window_t overlapadd::overlapadd_if_t::window [private]`

5.386.3.5 wndexp `MHAParser::float_t overlapadd::overlapadd_if_t::wndexp [private]`

5.386.3.6 zerowindow `MHAParser::window_t overlapadd::overlapadd_if_t::zerowindow [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.386.3.7 strict_window_ratio

`MHAParser::bool_t overlapadd::overlapadd_if_t↩ ::strict_window_ratio [private]`

Disallow window sizes that are not a multiple of the hop size ("fragsize" in MHA) a by power of two.

5.386.3.8 plugloader

`MHAParser::mhapluginloader_t overlapadd::overlapadd_if_t↩ ::plugloader [private]`

5.386.3.9 prescale

`MHAParser::float_mon_t overlapadd::overlapadd_if_t::prescale [private]`

5.386.3.10 postscale

`MHAParser::float_mon_t overlapadd::overlapadd_if_t::postscale [private]`

5.386.3.11 algo

`std::string overlapadd::overlapadd_if_t::algo [private]`

5.386.3.12 cf_in

`mhaconfig_t overlapadd::overlapadd_if_t::cf_in [private]`

5.386.3.13 cf_out

`mhaconfig_t overlapadd::overlapadd_if_t::cf_out [private]`

The documentation for this class was generated from the following files:

- **overlapadd.hh**
- **overlapadd.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.387 overlapadd::overlapadd_t Class Reference

Public Member Functions

- **overlapadd_t** (**mhaconfig_t** spar_in, **mhaconfig_t** spar_out, float wexp, float wndpos, const **MHAParser::window_t** &window, const **MHAParser::window_t** &zerowindow, float &prescale_fac, float &postscale_fac)
- ∼**overlapadd_t** ()
- **mha_spec_t** ∗ **wave2spec** (**mha_wave_t** ∗)
- **mha_wave_t** ∗ **spec2wave** (**mha_spec_t** ∗)

Private Member Functions

- void **wave2spec_hop_forward** (**mha_wave_t** ∗)
- void **wave2spec_apply_window** (void)
- **mha_spec_t** ∗ **wave2spec_compute_fft** (void)

Private Attributes

- **mha_fft_t fft**
- **MHAWindow::base_t prewnd**
- **MHAWindow::base_t postwnd**
- **MHASignal::waveform_t wave_in1**
- **MHASignal::waveform_t wave_out1**
- **MHASignal::spectrum_t spec_in**
- **MHASignal::waveform_t calc_out**
- **MHASignal::waveform_t out_buf**
- **MHASignal::waveform_t write_buf**
- unsigned int **n_zero**
- unsigned int **n_pad1**
- unsigned int **n_pad2**

5.387.1 Constructor & Destructor Documentation

5.387.1.1 overlapadd_t()

```
overlapadd::overlapadd_t::overlapadd_t (
    mhaconfig_t spar_in,
    mhaconfig_t spar_out,
    float wexp,
    float wndpos,
    const MHAParser::window_t & window,
    const MHAParser::window_t & zerowindow,
    float & prescale_fac,
    float & postscale_fac )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.387.1.2 ~overlapadd_t() `overlapadd::overlapadd_t::~overlapadd_t ( )`

5.387.2 Member Function Documentation

5.387.2.1 wave2spec() `mha_spec_t * overlapadd::overlapadd_t::wave2spec ( mha_wave_t * s )`

5.387.2.2 spec2wave() `mha_wave_t * overlapadd::overlapadd_t::spec2wave ( mha_spec_t * s )`

5.387.2.3 wave2spec_hop_forward() `void overlapadd::overlapadd_t::wave2spec_hop_forward ( mha_wave_t * s ) [private]`

5.387.2.4 wave2spec_apply_window() `void overlapadd::overlapadd_t::wave2spec_apply_window ( void ) [private]`

5.387.2.5 wave2spec_compute_fft() `mha_spec_t * overlapadd::overlapadd_t::wave2spec_compute_fft ( void ) [private]`

5.387.3 Member Data Documentation

5.387.3.1 fft `mha_fft_t overlapadd::overlapadd_t::fft [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.387.3.2 prewnd

`MHAWindow::base_t overlapadd::overlapadd_t::prewnd [private]`

5.387.3.3 postwnd

`MHAWindow::base_t overlapadd::overlapadd_t::postwnd [private]`

5.387.3.4 wave_in1

`MHASignal::waveform_t overlapadd::overlapadd_t::wave_in1 [private]`

5.387.3.5 wave_out1

`MHASignal::waveform_t overlapadd::overlapadd_t::wave_out1 [private]`

5.387.3.6 spec_in

`MHASignal::spectrum_t overlapadd::overlapadd_t::spec_in [private]`

5.387.3.7 calc_out

`MHASignal::waveform_t overlapadd::overlapadd_t::calc_out [private]`

5.387.3.8 out_buf

`MHASignal::waveform_t overlapadd::overlapadd_t::out_buf [private]`

5.387.3.9 write_buf

`MHASignal::waveform_t overlapadd::overlapadd_t::write_buf [private]`

5.387.3.10 n_zero

`unsigned int overlapadd::overlapadd_t::n_zero [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.387.3.11 n_pad1 `unsigned int overlapadd::overlapadd_t::n_pad1 [private]`

5.387.3.12 n_pad2 `unsigned int overlapadd::overlapadd_t::n_pad2 [private]`

The documentation for this class was generated from the following files:

- **overlapadd.hh**
- **overlapadd.cpp**

5.388 parser_int_dyn Class Reference

Inheritance diagram for parser_int_dyn:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **parser_int_dyn** (const std::string &help_text, const std::string &initial_value, const std↩ ::string & **range**)
- void **set_max_angle_ind** (unsigned int max_ind)

Additional Inherited Members

5.388.1 Constructor & Destructor Documentation

5.388.1.1 parser_int_dyn()

```
parser_int_dyn::parser_int_dyn (
    const std::string & help_text,
    const std::string & initial_value,
    const std::string & range ) [inline]
```

5.388.2 Member Function Documentation

5.388.2.1 set_max_angle_ind()

```
void parser_int_dyn::set_max_angle_ind (
    unsigned int max_ind ) [inline]
```

The documentation for this class was generated from the following file:

- **steerbf.h**

© 2005-2021 HörTech gGmbH, Oldenburg

5.389 plingploing::if_t Class Reference

Plugin class of the plingploing music generator.

Inheritance diagram for plingploing::if_t:

Public Member Functions

- **if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &cf)

Private Member Functions

- void **update** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAEvents::patchbay_t< if_t > patchbay**
- **MHAParser::float_t level**
  Output level in dB SPL.
- **MHAParser::float_t pitch**
  Bass pitch in Hz.
- **MHAParser::float_t fun1_key**
  Key1.
- **MHAParser::float_t fun1_range**
  Range1.
- **MHAParser::float_t fun2_key**
  Key 2.
- **MHAParser::float_t fun2_range**
  Range 2.
- **MHAParser::float_t bpm**
  Speed in beats per minute (bpm)
- **MHAParser::float_t minlen**
  Minimum note length in beats.
- **MHAParser::float_t maxlen**
  Maximum note length in beats.
- **MHAParser::float_t bassmod**
  Bass key modulation depth.
- **MHAParser::float_t bassperiod**
  Bass key modulation period.

Additional Inherited Members

5.389.1 Detailed Description

Plugin class of the plingploing music generator.

5.389.2 Constructor & Destructor Documentation

5.389.2.1 if_t()

```
plingploing::if_t::if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.389.3 Member Function Documentation

5.389.3.1 process() `mha_wave_t * plingploing::if_t::process ( mha_wave_t * s )`

5.389.3.2 prepare() `void plingploing::if_t::prepare ( mhaconfig_t & cf ) [virtual]`

Implements **MHAPlugin::plugin_t< plingploing_t >** (p. 1153).

5.389.3.3 update() `void plingploing::if_t::update ( ) [private]`

5.389.4 Member Data Documentation

5.389.4.1 patchbay `MHAEvents::patchbay_t< if_t> plingploing::if_t::patchbay [private]`

5.389.4.2 level `MHAParser::float_t plingploing::if_t::level [private]`

Output level in dB SPL.

5.389.4.3 pitch `MHAParser::float_t plingploing::if_t::pitch [private]`

Bass pitch in Hz.

© 2005-2021 HörTech gGmbH, Oldenburg

5.389.4.4 fun1_key `MHAParser::float_t plingploing::if_t::fun1_key [private]`

Key1.

5.389.4.5 fun1_range `MHAParser::float_t plingploing::if_t::fun1_range [private]`

Range1.

5.389.4.6 fun2_key `MHAParser::float_t plingploing::if_t::fun2_key [private]`

Key 2.

5.389.4.7 fun2_range `MHAParser::float_t plingploing::if_t::fun2_range [private]`

Range 2.

5.389.4.8 bpm `MHAParser::float_t plingploing::if_t::bpm [private]`

Speed in beats per minute (bpm)

5.389.4.9 minlen `MHAParser::float_t plingploing::if_t::minlen [private]`

Minimum note length in beats.

5.389.4.10 maxlen `MHAParser::float_t plingploing::if_t::maxlen [private]`

Maximum note length in beats.

© 2005-2021 HörTech gGmbH, Oldenburg

5.389.4.11 bassmod `MHAParser::float_t plingploing::if_t::bassmod [private]`

Bass key modulation depth.

5.389.4.12 bassperiod `MHAParser::float_t plingploing::if_t::bassperiod [private]`

Bass key modulation period.

The documentation for this class was generated from the following file:

- **plingploing.cpp**

5.390 plingploing::plingploing_t Class Reference

Run-time configuration of the plingploing music generator.

Public Member Functions

- **plingploing_t** (**mhaconfig_t**, **mha_real_t level**, **mha_real_t** pitch, **mha_real_t** k1, **mha_real_t** k2, **mha_real_t** i1, **mha_real_t** i2, **mha_real_t** bpm, **mha_real_t** minlen, **mha_real_t** maxlen, **mha_real_t** bassmod, **mha_real_t** bassperiod)
- void **process** (**mha_wave_t** $*$)

Private Attributes

- **mhaconfig_t cf**
- **mha_real_t pitch_**
- unsigned int **bt**
- unsigned int **t**
- unsigned int **len**
- **mha_real_t dur_**
- **mha_real_t minlen_**
- **mha_real_t maxlen_**
- **mha_real_t bass**
- **mha_real_t freq**
- **mha_real_t fun1_key**
- **mha_real_t fun1_range**
- **mha_real_t fun1**
- **mha_real_t fun2**
- **mha_real_t fun2_key**

© 2005-2021 HörTech gGmbH, Oldenburg

- **mha_real_t fun2_range**
- **mha_real_t dist**
- **mha_real_t dist1**
- **mha_real_t alph**
- **mha_real_t rms**
- **mha_real_t bassmod_**
- **mha_real_t bassperiod_**
- **MHAWindow::hanning_t hann1**
- **MHAWindow::hanning_t hann2**
- **mha_real_t level**

5.390.1 Detailed Description

Run-time configuration of the plingploing music generator.

5.390.2 Constructor & Destructor Documentation

5.390.2.1 plingploing_t()

```
plingploing::plingploing_t::plingploing_t (
    mhaconfig_t c,
    mha_real_t level,
    mha_real_t pitch,
    mha_real_t k1,
    mha_real_t k2,
    mha_real_t i1,
    mha_real_t i2,
    mha_real_t bpm,
    mha_real_t minlen,
    mha_real_t maxlen,
    mha_real_t bassmod,
    mha_real_t bassperiod )
```

5.390.3 Member Function Documentation

5.390.3.1 process()

```
void plingploing::plingploing_t::process (
    mha_wave_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.390.4 Member Data Documentation

5.390.4.1 cf

`mhaconfig_t` plingploing::plingploing_t::cf [private]

5.390.4.2 pitch_

`mha_real_t` plingploing::plingploing_t::pitch_ [private]

5.390.4.3 bt

unsigned int plingploing::plingploing_t::bt [private]

5.390.4.4 t

unsigned int plingploing::plingploing_t::t [private]

5.390.4.5 len

unsigned int plingploing::plingploing_t::len [private]

5.390.4.6 dur_

`mha_real_t` plingploing::plingploing_t::dur_ [private]

5.390.4.7 minlen_

`mha_real_t` plingploing::plingploing_t::minlen_ [private]

5.390.4.8 maxlen_

`mha_real_t` plingploing::plingploing_t::maxlen_ [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.390.4.9 bass

`mha_real_t plingploing::plingploing_t::bass [private]`

5.390.4.10 freq

`mha_real_t plingploing::plingploing_t::freq [private]`

5.390.4.11 fun1_key

`mha_real_t plingploing::plingploing_t::fun1_key [private]`

5.390.4.12 fun1_range

`mha_real_t plingploing::plingploing_t::fun1_range [private]`

5.390.4.13 fun1

`mha_real_t plingploing::plingploing_t::fun1 [private]`

5.390.4.14 fun2

`mha_real_t plingploing::plingploing_t::fun2 [private]`

5.390.4.15 fun2_key

`mha_real_t plingploing::plingploing_t::fun2_key [private]`

5.390.4.16 fun2_range

`mha_real_t plingploing::plingploing_t::fun2_range [private]`

5.390.4.17 dist

`mha_real_t plingploing::plingploing_t::dist [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.390.4.18 dist1 `mha_real_t` plingploing::plingploing_t::dist1 [private]

5.390.4.19 alph `mha_real_t` plingploing::plingploing_t::alph [private]

5.390.4.20 rms `mha_real_t` plingploing::plingploing_t::rms [private]

5.390.4.21 bassmod_ `mha_real_t` plingploing::plingploing_t::bassmod_ [private]

5.390.4.22 bassperiod_ `mha_real_t` plingploing::plingploing_t::bassperiod_ [private]

5.390.4.23 hann1 `MHAWindow::hanning_t` plingploing::plingploing_t::hann1 [private]

5.390.4.24 hann2 `MHAWindow::hanning_t` plingploing::plingploing_t::hann2 [private]

5.390.4.25 level `mha_real_t` plingploing::plingploing_t::level [private]

The documentation for this class was generated from the following file:

- **plingploing.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.391 plug_t Class Reference

Inheritance diagram for plug_t:

Public Member Functions

- **plug_t** (const std::string & **libname**)
- ∼**plug_t** () throw ()
- **MHAProc_wave2wave_t get_process_wave** ()
- **MHAProc_wave2spec_t get_process_spec** ()
- void ∗ **get_handle** ()
- **algo_comm_t get_ac** ()

Additional Inherited Members

5.391.1 Constructor & Destructor Documentation

5.391.1.1 plug_t() `plug_t::plug_t ( const std::string & libname )`

5.391.1.2 ∼plug_t() `plug_t::∼plug_t ( ) throw ( ) [inline]`

5.391.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.391.2.1 get_process_wave() `MHAProc_wave2wave_t plug_t::get_process_wave ( )`

5.391.2.2 get_process_spec() `MHAProc_wave2spec_t plug_t::get_process_spec ( )`

5.391.2.3 get_handle() `void * plug_t::get_handle ( )`

5.391.2.4 get_ac() `algo_comm_t plug_t::get_ac ( ) [inline]`

The documentation for this class was generated from the following file:

- **analysispath.cpp**

5.392 plug_wrapper Class Reference

Inheritance diagram for plug_wrapper:

Public Member Functions

- **plug_wrapper** (**algo_comm_t** iac, const std::string & **libname**)
- virtual ~**plug_wrapper** ()=default
- virtual std::vector< std::string > **get_categories** ()
- virtual std::string **parse** (const std::string &str)
- virtual bool **has_parser** ()
- virtual std::string **get_documentation** ()
- virtual bool **has_process** (**mha_domain_t** in, **mha_domain_t** out)

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.392.1 Constructor & Destructor Documentation

5.392.1.1 plug_wrapper()

```
plug_wrapper::plug_wrapper (
    algo_comm_t iac,
    const std::string & libname ) [inline]
```

5.392.1.2 ~plug_wrapper()

```
virtual plug_wrapper::~plug_wrapper ( ) [virtual], [default]
```

5.392.2 Member Function Documentation

5.392.2.1 get_categories()

```
virtual std::vector<std::string> plug_wrapper::get_categories ( ) [inline], [virtual]
```

Implements **plug_wrapperI** (p. 1353).

5.392.2.2 parse()

```
virtual std::string plug_wrapper::parse (
    const std::string & str ) [inline], [virtual]
```

Reimplemented from **PluginLoader::mhapluginloader_t** (p. 1371).

5.392.2.3 has_parser()

```
virtual bool plug_wrapper::has_parser ( ) [inline], [virtual]
```

Implements **plug_wrapperI** (p. 1353).

© 2005-2021 HörTech gGmbH, Oldenburg

5.392.2.4 get_documentation() `virtual std::string plug_wrapper::get_documentation ( ) [inline], [virtual]`

Implements **plug_wrapperl** (p. 1353).

5.392.2.5 has_process() `virtual bool plug_wrapper::has_process (`
`mha_domain_t in,`
`mha_domain_t out ) [inline], [virtual]`

Implements **plug_wrapperl** (p. 1354).

The documentation for this class was generated from the following file:

- **generatemhaplugindoc.cpp**

5.393 plug_wrapperl Class Reference

Inheritance diagram for plug_wrapperl:

Public Member Functions

- **plug_wrapperl** ()
- virtual ~**plug_wrapperl** ()=default
- virtual std::vector< std::string > **get_categories** ()=0
- virtual std::string **parse** (const std::string &str)=0
- virtual bool **has_parser** ()=0
- virtual std::string **get_documentation** ()=0
- virtual bool **has_process** (**mha_domain_t**, **mha_domain_t**)=0

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.393.1 Constructor & Destructor Documentation

5.393.1.1 plug_wrapperI() `plug_wrapperI::plug_wrapperI ( ) [inline]`

5.393.1.2 ~plug_wrapperI() `virtual plug_wrapperI::~plug_wrapperI ( ) [virtual], [default]`

5.393.2 Member Function Documentation

5.393.2.1 get_categories() `virtual std::vector<std::string> plug_wrapperI::get_categories ( ) [pure virtual]`

Implemented in **plug_wrapper** (p. 1351), and **io_wrapper** (p. 644).

5.393.2.2 parse() `virtual std::string plug_wrapperI::parse ( const std::string & str ) [pure virtual]`

Reimplemented from **MHAParser::base_t** (p. 1035).

Implemented in **plug_wrapper** (p. 1351), and **io_wrapper** (p. 644).

5.393.2.3 has_parser() `virtual bool plug_wrapperI::has_parser ( ) [pure virtual]`

Implemented in **plug_wrapper** (p. 1351), and **io_wrapper** (p. 644).

© 2005-2021 HörTech gGmbH, Oldenburg

5.393.2.4 get_documentation() `virtual std::string plug_wrapperI::get_documentation ( ) [pure virtual]`

Implemented in **plug_wrapper** (p. 1351), and **io_wrapper** (p. 644).

5.393.2.5 has_process() `virtual bool plug_wrapperI::has_process (`
`mha_domain_t ,`
`mha_domain_t ) [pure virtual]`

Implemented in **plug_wrapper** (p. 1352), and **io_wrapper** (p. 644).

The documentation for this class was generated from the following file:

- **generatemhaplugindoc.cpp**

5.394 plugin_interface_t Class Reference

Inheritance diagram for plugin_interface_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **plugin_interface_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAParser::int_t scale_ch**
- **MHAParser::float_t factor**
- **MHAEvents::patchbay_t**< **plugin_interface_t** > **patchbay**

Additional Inherited Members

5.394.1 Constructor & Destructor Documentation

5.394.1.1 plugin_interface_t()

```
plugin_interface_t::plugin_interface_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.394.2 Member Function Documentation

5.394.2.1 process()

```
mha_spec_t * plugin_interface_t::process (
    mha_spec_t * spec )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.394.2.2 prepare() `void plugin_interface_t::prepare ( mhaconfig_t & tfcfg ) [virtual]`

Implements **MHAPlugin::plugin_t**< **example5_t** > (p. 1153).

5.394.2.3 update_cfg() `void plugin_interface_t::update_cfg ( ) [private]`

5.394.3 Member Data Documentation

5.394.3.1 scale_ch `MHAParser::int_t plugin_interface_t::scale_ch [private]`

5.394.3.2 factor `MHAParser::float_t plugin_interface_t::factor [private]`

5.394.3.3 patchbay `MHAEvents::patchbay_t< plugin_interface_t> plugin_interface↩_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **example5.cpp**

5.395 pluginbrowser_t Class Reference

Public Member Functions

- **pluginbrowser_t** ()
- void **get_paths** ()
- **plugindescription_t** **scan_plugin** (const std::string &name)
- void **add_plugins** ()
- void **clear_plugins** ()
- void **scan_plugins** ()
- void **add_plugin** (const std::string &name)
- std::list< **plugindescription_t** > **get_plugins** () const

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- std::string **plugin_extension**
- std::list< std::string > **library_paths**
- std::list< **plugindescription_t** > **plugins**
- std::map< std::string, **pluginloader_t** * > **p**

5.395.1 Constructor & Destructor Documentation

5.395.1.1 pluginbrowser_t() `pluginbrowser_t::pluginbrowser_t ( )`

5.395.2 Member Function Documentation

5.395.2.1 get_paths() `void pluginbrowser_t::get_paths ( )`

5.395.2.2 scan_plugin() `plugindescription_t pluginbrowser_t::scan_plugin ( const std::string & name )`

5.395.2.3 add_plugins() `void pluginbrowser_t::add_plugins ( )`

5.395.2.4 clear_plugins() `void pluginbrowser_t::clear_plugins ( )`

5.395.2.5 scan_plugins() `void pluginbrowser_t::scan_plugins ( )`

© 2005-2021 HörTech gGmbH, Oldenburg

5.395.2.6 add_plugin()

```
void pluginbrowser_t::add_plugin (
    const std::string & name )
```

5.395.2.7 get_plugins()

```
std::list< plugindescription_t> pluginbrowser_t::get_↩
plugins ( ) const [inline]
```

5.395.3 Member Data Documentation

5.395.3.1 plugin_extension

```
std::string pluginbrowser_t::plugin_extension [private]
```

5.395.3.2 library_paths

```
std::list<std::string> pluginbrowser_t::library_paths [private]
```

5.395.3.3 plugins

```
std::list< plugindescription_t> pluginbrowser_t::plugins [private]
```

5.395.3.4 p

```
std::map<std::string, pluginloader_t*> pluginbrowser_t::p [private]
```

The documentation for this class was generated from the following files:

- **pluginbrowser.h**
- **pluginbrowser.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.396 plugindescription_t Class Reference

Public Attributes

- std::string **name**
- std::string **fullname**
- std::string **documentation**
- std::vector< std::string > **categories**
- bool **wave2wave**
- bool **wave2spec**
- bool **spec2wave**
- bool **spec2spec**
- std::vector< std::string > **query_cmds**
- std::map< std::string, std::string > **queries**

5.396.1 Member Data Documentation

5.396.1.1 name `std::string plugindescription_t::name`

5.396.1.2 fullname `std::string plugindescription_t::fullname`

5.396.1.3 documentation `std::string plugindescription_t::documentation`

5.396.1.4 categories `std::vector<std::string> plugindescription_t::categories`

5.396.1.5 wave2wave `bool plugindescription_t::wave2wave`

© 2005-2021 HörTech gGmbH, Oldenburg

5.396.1.6 wave2spec `bool plugindescription_t::wave2spec`

5.396.1.7 spec2wave `bool plugindescription_t::spec2wave`

5.396.1.8 spec2spec `bool plugindescription_t::spec2spec`

5.396.1.9 query_cmds `std::vector<std::string> plugindescription_t::query_cmds`

5.396.1.10 queries `std::map<std::string,std::string> plugindescription_t::queries`

The documentation for this class was generated from the following file:

- **pluginbrowser.h**

5.397 pluginlib_t Class Reference

Specialisation of **dynamiclib_t** (p. 446) for mha plugin libraries.

Inheritance diagram for pluginlib_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **pluginlib_t** (const std::string &name_)
  C'tor of the wrapper class.
- virtual void * **resolve** (const std::string &name_) override
  Resolves the plugin callback specified by name_, e.g.
- virtual ~**pluginlib_t** ()
  D'tor.

Additional Inherited Members

5.397.1 Detailed Description

Specialisation of **dynamiclib_t** (p. 446) for mha plugin libraries.

5.397.2 Constructor & Destructor Documentation

5.397.2.1 pluginlib_t() `pluginlib_t::pluginlib_t ( const std::string & name_ ) [explicit]`

C'tor of the wrapper class.

Takes a the the file name of a shared libary w/o the suffix as argument, searches for the library in the system-dependent standard paths for libraries and in MHA_LIBRARY_DIR. Calls load_lib for the actual work.

Parameters

*name_*	File name of the shared library, without suffix

5.397.2.2 ~pluginlib_t() `pluginlib_t::~pluginlib_t ( ) [virtual]`

D'tor.

© 2005-2021 HörTech gGmbH, Oldenburg

5.397.3 Member Function Documentation

5.397.3.1 resolve() `void * pluginlib_t::resolve ( const std::string & name_ ) [override], [virtual]`

Resolves the plugin callback specified by name_, e.g.

'process', 'prepare', etc... and returns a pointer to it or a nullptr if the function was not found. Automatically adds the required prefixes and suffixes for dynamically/statically compiled mha plugins

Parameters

*name_*	Name of the function to be resolved

Returns

Pointer to the function

Reimplemented from **dynamiclib_t** (p. 448).

The documentation for this class was generated from the following files:

- **mha_os.h**
- **mha_os.cpp**

5.398 PluginLoader::config_file_splitter_t Class Reference

Inheritance diagram for PluginLoader::config_file_splitter_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **config_file_splitter_t** (const std::string &name)
- const std::string & **get_configname** () const
- const std::string & **get_libname** () const
- const std::string & **get_origname** () const
- const std::string & **get_configfile** () const

Private Attributes

- std::string **libname**
- std::string **configname**
- std::string **origname**
- std::string **configfile**

5.398.1 Constructor & Destructor Documentation

5.398.1.1 config_file_splitter_t()

```
PluginLoader::config_file_splitter_t::config_file↩
_splitter_t (
            const std::string & name )
```

5.398.2 Member Function Documentation

5.398.2.1 get_configname()

```
const std::string& PluginLoader::config_file_splitter↩
_t::get_configname ( ) const [inline]
```

5.398.2.2 get_libname()

```
const std::string& PluginLoader::config_file_splitter_t↩
::get_libname ( ) const [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.398.2.3 get_origname() `const std::string& PluginLoader::config_file_splitter_t↩ ::get_origname ( ) const [inline]`

5.398.2.4 get_configfile() `const std::string& PluginLoader::config_file_splitter_t↩ ::get_configfile ( ) const [inline]`

5.398.3 Member Data Documentation

5.398.3.1 libname `std::string PluginLoader::config_file_splitter_t::libname [private]`

5.398.3.2 configname `std::string PluginLoader::config_file_splitter_t::configname [private]`

5.398.3.3 origname `std::string PluginLoader::config_file_splitter_t::origname [private]`

5.398.3.4 configfile `std::string PluginLoader::config_file_splitter_t::configfile [private]`

The documentation for this class was generated from the following files:

- **mhapluginloader.h**
- **mhapluginloader.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.399 PluginLoader::fourway_processor_t Class Reference

This abstract class defines the interface for classes that implement all types of signal domain processing supported by the MHA: wave2wave, spec2spec, wave2spec, and spec2wave.

Inheritance diagram for PluginLoader::fourway_processor_t:

Public Member Functions

- virtual void **process** (**mha_wave_t** ∗s_in, **mha_wave_t** ∗∗s_out)=0
  Pure waveform processing.
- virtual void **process** (**mha_spec_t** ∗s_in, **mha_spec_t** ∗∗s_out)=0
  Pure spectrum processing.
- virtual void **process** (**mha_wave_t** ∗s_in, **mha_spec_t** ∗∗s_out)=0
  Signal processing with domain transformation from waveform to spectrum.
- virtual void **process** (**mha_spec_t** ∗s_in, **mha_wave_t** ∗∗s_out)=0
  Signal processing with domain transformation from spectrum to waveform.
- virtual void **prepare** (**mhaconfig_t** &settings)=0
  Prepares the processor for signal processing.
- virtual void **release** ()=0
  *Resources allocated for signal processing in **fourway_processor_t::prepare*** (p. 1367) *are released here in **fourway_processor_t::release*** (p. 1368).
- virtual std::string **parse** (const std::string &query)=0
  Parser interface.
- virtual ∼**fourway_processor_t** ()
  Classes with virtual methods need virtual destructor.

5.399.1 Detailed Description

This abstract class defines the interface for classes that implement all types of signal domain processing supported by the MHA: wave2wave, spec2spec, wave2spec, and spec2wave.

For supporting different output domains for the same input domain, the processing methods are overloaded with respect to input domain and output domain.

© 2005-2021 HörTech gGmbH, Oldenburg

5.399.2 Constructor & Destructor Documentation

5.399.2.1 ~fourway_processor_t() `virtual PluginLoader::fourway_processor_t::~fourway↩_processor_t ( ) [inline], [virtual]`

Classes with virtual methods need virtual destructor.

This destructor is empty.

5.399.3 Member Function Documentation

5.399.3.1 process() [1/4] `virtual void PluginLoader::fourway_processor_t::process (`

```
mha_wave_t * s_in,
mha_wave_t ** s_out ) [pure virtual]
```

Pure waveform processing.

Parameters

s_in	input waveform signal
s_out	output waveform signal

Implemented in **PluginLoader::mhapluginloader_t** (p. 1372).

5.399.3.2 process() [2/4] `virtual void PluginLoader::fourway_processor_t::process (`

```
mha_spec_t * s_in,
mha_spec_t ** s_out ) [pure virtual]
```

Pure spectrum processing.

Parameters

s_in	input spectrum signal
s_out	output spectrum signal

© 2005-2021 HörTech gGmbH, Oldenburg

Implemented in **PluginLoader::mhapluginloader_t** (p. 1372).

5.399.3.3 process() [3/4]

```
virtual void PluginLoader::fourway_processor_t::process (
    mha_wave_t * s_in,
    mha_spec_t ** s_out ) [pure virtual]
```

Signal processing with domain transformation from waveform to spectrum.

Parameters

s_in	input waveform signal
s_out	output spectrum signal

Implemented in **PluginLoader::mhapluginloader_t** (p. 1372).

5.399.3.4 process() [4/4]

```
virtual void PluginLoader::fourway_processor_t::process (
    mha_spec_t * s_in,
    mha_wave_t ** s_out ) [pure virtual]
```

Signal processing with domain transformation from spectrum to waveform.

Parameters

s_in	input spectrum signal
s_out	output waveform signal

Implemented in **PluginLoader::mhapluginloader_t** (p. 1372).

5.399.3.5 prepare()

```
virtual void PluginLoader::fourway_processor_t::prepare (
    mhaconfig_t & settings ) [pure virtual]
```

Prepares the processor for signal processing.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

settings	domain and dimensions of the signal. The contents of settings may be modified by the prepare implementation. Upon calling **fourway_processor_t::prepare** (p. 1367), settings reflects domain and dimensions of the input signal. When **fourway_processor_t::prepare** (p. 1367) returns, settings reflects domain and dimensions of the output signal.

Implemented in **PluginLoader::mhapluginloader_t** (p. 1371).

5.399.3.6 release() `virtual void PluginLoader::fourway_processor_t::release ( ) [pure virtual]`

Resources allocated for signal processing in **fourway_processor_t::prepare** (p. 1367) are released here in **fourway_processor_t::release** (p. 1368).

Implemented in **PluginLoader::mhapluginloader_t** (p. 1372).

5.399.3.7 parse() `virtual std::string PluginLoader::fourway_processor_t::parse ( const std::string & query ) [pure virtual]`

Parser interface.

Implemented in **PluginLoader::mhapluginloader_t** (p. 1371), and **plug_wrapper** (p. 1351).

The documentation for this class was generated from the following file:

- **mhapluginloader.h**

5.400 PluginLoader::mhapluginloader_t Class Reference

Inheritance diagram for PluginLoader::mhapluginloader_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- std::string **parse** (const std::string &str) override
- **mhapluginloader_t** (**algo_comm_t** iac, const std::string & **libname**, bool check_↩ version=true)

  Loads and initializes mha plugin and establishes interface.
- ∼**mhapluginloader_t** () throw ()
- bool **has_process** (**mha_domain_t** in, **mha_domain_t** out) const
- bool **has_parser** () const
- **mha_domain_t** **input_domain** () const
- **mha_domain_t** **output_domain** () const
- void **prepare** (**mhaconfig_t** &) override
- void **release** () override
- void **process** (**mha_wave_t** ∗, **mha_wave_t** ∗∗) override
- void **process** (**mha_spec_t** ∗, **mha_spec_t** ∗∗) override
- void **process** (**mha_wave_t** ∗, **mha_spec_t** ∗∗) override
- void **process** (**mha_spec_t** ∗, **mha_wave_t** ∗∗) override
- std::string **getfullname** () const
- std::string **get_documentation** () const
- std::vector< std::string > **get_categories** () const
- bool **is_prepared** () const

Protected Member Functions

- void **test_error** ()
- void **test_version** ()
- void **mha_test_struct_size** (unsigned int s)
- void **resolve_and_init** ()

Protected Attributes

- int **lib_err**
- **algo_comm_t** **ac**
- **pluginlib_t** **lib_handle**
- void ∗ **lib_data**
- **MHAGetVersion_t** **MHAGetVersion_cb**
- **MHAInit_t** **MHAInit_cb**
- **MHADestroy_t** **MHADestroy_cb**
- **MHAPrepare_t** **MHAPrepare_cb**
- **MHARelease_t** **MHARelease_cb**
- **MHAProc_wave2wave_t** **MHAProc_wave2wave_cb**
- **MHAProc_spec2spec_t** **MHAProc_spec2spec_cb**
- **MHAProc_wave2spec_t** **MHAProc_wave2spec_cb**
- **MHAProc_spec2wave_t** **MHAProc_spec2wave_cb**
- **MHASet_t** **MHASet_cb**

© 2005-2021 HörTech gGmbH, Oldenburg

- **MHASetcpp_t MHASetcpp_cb**
- **MHAStrError_t MHAStrError_cb**
- **mhaconfig_t cf_input**
- **mhaconfig_t cf_output**
- std::string **plugin_documentation**
- std::vector< std::string > **plugin_categories**
- bool **b_check_version**
- bool **b_is_prepared**

Additional Inherited Members

5.400.1 Constructor & Destructor Documentation

5.400.1.1 mhapluginloader_t()

```
PluginLoader::mhapluginloader_t::mhapluginloader_t (
    algo_comm_t iac,
    const std::string & libname,
    bool check_version = true )
```

Loads and initializes mha plugin and establishes interface.

Parameters

iac	AC space (algorithm communication variables)
libname	Either file name of MHA plugin without platform-specific extension (i.e. "identity" for "identity.so" or "identity.dll") to be found on the MHA_LIBRARY_PATH (which is an environment variable). Or the same file name without extension followed by a colon ":" followed by the "configuration name" of the MHA plugin, which may be used to differentiate between multiple identical MHA plugins or to give the plugin a self-documenting name that fits its purpose. The library name - configuration name expression can be followed by a "<" followed by a configuration file name, which will be read after initialization of the plugin.

Example: "overlapadd:agc<compression.cfg" will load the plugin "overlapadd.so" or "overlapadd.dll", insert it as the configuration node "agc", and reads the configuration file "compression.cfg" into that node.

Parameters

check_version	Pluginloader will not check that the plugin was built using a known compatible MHA version if this flag is set to false. Disabling version check is discouraged.

© 2005-2021 HörTech gGmbH, Oldenburg

5.400.1.2 ~mhapluginloader_t()

`PluginLoader::mhapluginloader_t::~mhapluginloader_t ( ) throw ( )`

5.400.2 Member Function Documentation

5.400.2.1 parse()

```
std::string PluginLoader::mhapluginloader_t::parse (
    const std::string & str ) [override], [virtual]
```

Implements **PluginLoader::fourway_processor_t** (p. 1368).

Reimplemented in **plug_wrapper** (p. 1351).

5.400.2.2 has_process()

```
bool PluginLoader::mhapluginloader_t::has_process (
    mha_domain_t in,
    mha_domain_t out ) const
```

5.400.2.3 has_parser()

`bool PluginLoader::mhapluginloader_t::has_parser ( ) const`

5.400.2.4 input_domain()

`mha_domain_t PluginLoader::mhapluginloader_t::input_domain ( ) const`

5.400.2.5 output_domain()

`mha_domain_t PluginLoader::mhapluginloader_t::output_domain ( ) const`

© 2005-2021 HörTech gGmbH, Oldenburg

5.400.2.6 prepare()

```
void PluginLoader::mhapluginloader_t::prepare (
    mhaconfig_t & tf ) [override], [virtual]
```

Implements **PluginLoader::fourway_processor_t** (p. 1367).

5.400.2.7 release()

```
void PluginLoader::mhapluginloader_t::release ( ) [override],
[virtual]
```

Implements **PluginLoader::fourway_processor_t** (p. 1368).

5.400.2.8 process() [1/4]

```
void PluginLoader::mhapluginloader_t::process (
    mha_wave_t * s_in,
    mha_wave_t ** s_out ) [override], [virtual]
```

Implements **PluginLoader::fourway_processor_t** (p. 1366).

5.400.2.9 process() [2/4]

```
void PluginLoader::mhapluginloader_t::process (
    mha_spec_t * s_in,
    mha_spec_t ** s_out ) [override], [virtual]
```

Implements **PluginLoader::fourway_processor_t** (p. 1366).

5.400.2.10 process() [3/4]

```
void PluginLoader::mhapluginloader_t::process (
    mha_wave_t * s_in,
    mha_spec_t ** s_out ) [override], [virtual]
```

Implements **PluginLoader::fourway_processor_t** (p. 1367).

5.400.2.11 process() [4/4]

```
void PluginLoader::mhapluginloader_t::process (
    mha_spec_t * s_in,
    mha_wave_t ** s_out ) [override], [virtual]
```

Implements **PluginLoader::fourway_processor_t** (p. 1367).

© 2005-2021 HörTech gGmbH, Oldenburg

5.400.2.12 getfullname()

`std::string PluginLoader::mhapluginloader_t::getfullname ( ) const [inline]`

5.400.2.13 get_documentation()

`std::string PluginLoader::mhapluginloader_t::get_documentation ( ) const [inline]`

5.400.2.14 get_categories()

`std::vector<std::string> PluginLoader::mhapluginloader_t::get_categories ( ) const [inline]`

5.400.2.15 is_prepared()

`bool PluginLoader::mhapluginloader_t::is_prepared ( ) const [inline]`

5.400.2.16 test_error()

`void PluginLoader::mhapluginloader_t::test_error ( ) [protected]`

5.400.2.17 test_version()

`void PluginLoader::mhapluginloader_t::test_version ( ) [protected]`

5.400.2.18 mha_test_struct_size()

```
void PluginLoader::mhapluginloader_t::mha_test_struct_size (
            unsigned int s ) [protected]
```

5.400.2.19 resolve_and_init()

`void PluginLoader::mhapluginloader_t::resolve_and_init ( ) [protected]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.400.3 Member Data Documentation

5.400.3.1 lib_err `int PluginLoader::mhapluginloader_t::lib_err [protected]`

5.400.3.2 ac `algo_comm_t PluginLoader::mhapluginloader_t::ac [protected]`

5.400.3.3 lib_handle `pluginlib_t PluginLoader::mhapluginloader_t::lib_handle [protected]`

5.400.3.4 lib_data `void* PluginLoader::mhapluginloader_t::lib_data [protected]`

5.400.3.5 MHAGetVersion_cb `MHAGetVersion_t PluginLoader::mhapluginloader_t::MHAGetVersion_cb [protected]`

5.400.3.6 MHAInit_cb `MHAInit_t PluginLoader::mhapluginloader_t::MHAInit_cb [protected]`

5.400.3.7 MHADestroy_cb `MHADestroy_t PluginLoader::mhapluginloader_t::MHADestroy_cb [protected]`

5.400.3.8 MHAPrepare_cb `MHAPrepare_t PluginLoader::mhapluginloader_t::MHAPrepare_cb [protected]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.400.3.9 MHARelease_cb

MHARelease_t PluginLoader::mhapluginloader_t::MHA↩ Release_cb [protected]

5.400.3.10 MHAProc_wave2wave_cb

MHAProc_wave2wave_t PluginLoader::mhapluginloader↩ _t::MHAProc_wave2wave_cb [protected]

5.400.3.11 MHAProc_spec2spec_cb

MHAProc_spec2spec_t PluginLoader::mhapluginloader↩ _t::MHAProc_spec2spec_cb [protected]

5.400.3.12 MHAProc_wave2spec_cb

MHAProc_wave2spec_t PluginLoader::mhapluginloader↩ _t::MHAProc_wave2spec_cb [protected]

5.400.3.13 MHAProc_spec2wave_cb

MHAProc_spec2wave_t PluginLoader::mhapluginloader↩ _t::MHAProc_spec2wave_cb [protected]

5.400.3.14 MHASet_cb

MHASet_t PluginLoader::mhapluginloader_t::MHASet_cb [protected]

5.400.3.15 MHASetcpp_cb

MHASetcpp_t PluginLoader::mhapluginloader_t::MHA↩ Setcpp_cb [protected]

5.400.3.16 MHAStrError_cb

MHAStrError_t PluginLoader::mhapluginloader_t::MHA↩ StrError_cb [protected]

© 2005-2021 HörTech gGmbH, Oldenburg

5.400.3.17 cf_input `mhaconfig_t PluginLoader::mhapluginloader_t::cf_input [protected]`

5.400.3.18 cf_output `mhaconfig_t PluginLoader::mhapluginloader_t::cf_output [protected]`

5.400.3.19 plugin_documentation `std::string PluginLoader::mhapluginloader_t↩ ::plugin_documentation [protected]`

5.400.3.20 plugin_categories `std::vector<std::string> PluginLoader::mhapluginloader↩ _t::plugin_categories [protected]`

5.400.3.21 b_check_version `bool PluginLoader::mhapluginloader_t::b_check_version [protected]`

5.400.3.22 b_is_prepared `bool PluginLoader::mhapluginloader_t::b_is_prepared [protected]`

The documentation for this class was generated from the following files:

- **mhapluginloader.h**
- **mhapluginloader.cpp**

5.401 pluginloader_t Class Reference

Inheritance diagram for pluginloader_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **pluginloader_t** (const std::string &name)
- ∼**pluginloader_t** () throw ()

Additional Inherited Members

5.401.1 Constructor & Destructor Documentation

5.401.1.1 pluginloader_t() `pluginloader_t::pluginloader_t ( const std::string & name )`

5.401.1.2 ∼pluginloader_t() `pluginloader_t::∼pluginloader_t ( ) throw ( )`

The documentation for this class was generated from the following files:

- **pluginbrowser.h**
- **pluginbrowser.cpp**

5.402 plugins::hoertech::acrec::acrec_t Class Reference

Plugin interface class of plugin acrec.

© 2005-2021 HörTech gGmbH, Oldenburg

Inheritance diagram for plugins::hoertech::acrec::acrec_t:

Public Member Functions

- template<class mha_signal_t >
  mha_signal_t * **process** (mha_signal_t *s)

  Process callback.
- void **prepare** (**mhaconfig_t** &cf)

  Prepare callback.
- void **release** ()

  Ensure recorded data is flushed to disk.
- **acrec_t** (**algo_comm_t** iac, const std::string &configured_name)

  Plugin interface constructor.

Private Member Functions

- void **start_new_session** ()

  Configuration callback called whenever configuaration variable "record" is written to.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::bool_t record**
- **MHAParser::int_t fifolen**
- **MHAParser::int_t minwrite**
- **MHAParser::string_t prefix**
- **MHAParser::string_t varname**
- **MHAParser::bool_t use_date**
- **MHAEvents::patchbay_t< acrec_t > patchbay**
- **comm_var_t cv**
- **algo_comm_t ac**

Additional Inherited Members

5.402.1 Detailed Description

Plugin interface class of plugin acrec.

5.402.2 Constructor & Destructor Documentation

5.402.2.1 acrec_t()

```
acrec_t::acrec_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Plugin interface constructor.

Parameters

iac	Algorithm communication variable space.

5.402.3 Member Function Documentation

5.402.3.1 process()

```
template<class mha_signal_t >
mha_signal_t * acrec_t::process (
    mha_signal_t * s )
```

© 2005-2021 HörTech gGmbH, Oldenburg

Process callback.

Pushes the data from one AC variable into the fifo.

Returns

the unmodified input signal.

Parameters

s	input signal. The audio signal is not used or modified.

5.402.3.2 prepare() `void acrec_t::prepare ( mhaconfig_t & cf ) [virtual]`

Prepare callback.

acrec does not modify the signal parameters.

Parameters

cf	The signal parameters.

Implements **MHAPlugin::plugin_t< acwriter_t >** (p. 1153).

5.402.3.3 release() `void acrec_t::release ( ) [virtual]`

Ensure recorded data is flushed to disk.

Reimplemented from **MHAPlugin::plugin_t< acwriter_t >** (p. 1154).

5.402.3.4 start_new_session() `void acrec_t::start_new_session ( ) [private]`

Configuration callback called whenever configuaration variable "record" is written to.

© 2005-2021 HörTech gGmbH, Oldenburg

5.402.4 Member Data Documentation

5.402.4.1 record

`MHAParser::bool_t plugins::hoertech::acrec::acrec_t::record [private]`

5.402.4.2 fifolen

`MHAParser::int_t plugins::hoertech::acrec::acrec_t::fifolen [private]`

5.402.4.3 minwrite

`MHAParser::int_t plugins::hoertech::acrec::acrec_t::minwrite [private]`

5.402.4.4 prefix

`MHAParser::string_t plugins::hoertech::acrec::acrec_t::prefix [private]`

5.402.4.5 varname

`MHAParser::string_t plugins::hoertech::acrec::acrec_t::varname [private]`

5.402.4.6 use_date

`MHAParser::bool_t plugins::hoertech::acrec::acrec_t::use_date [private]`

5.402.4.7 patchbay

`MHAEvents::patchbay_t< acrec_t> plugins::hoertech::acrec↩::acrec_t::patchbay [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.402.4.8 cv `comm_var_t` plugins::hoertech::acrec::acrec_t::cv [private]

5.402.4.9 ac `algo_comm_t` plugins::hoertech::acrec::acrec_t::ac [private]

The documentation for this class was generated from the following files:

- **acrec.hh**
- **acrec.cpp**

5.403 plugins::hoertech::acrec::acwriter_t Class Reference

acwriter_t (p. 1382) decouples signal processing from writing to disk.

Public Types

- typedef double **output_type**
  The numeric data type used for outputting the data to disk.

Public Member Functions

- **acwriter_t** (bool **active**, unsigned fifosize, unsigned minwrite, const std::string &prefix, bool use_date, const std::string & **varname**)
  Constructor allocates fifo and disk output buffer.
- ∼**acwriter_t** ()=default
  Deallocates memory but does not terminate the write_thread.
- void **process** (**comm_var_t** ∗)
  Place the data present in the algorithm communication variable into the fifo for output to disk.
- void **exit_request** ()
  Terminate output thread.
- const char ∗ **get_varname** () const
  getter for ac variable name

Private Member Functions

- void **write_thread** ()
  Main method of the disk writer thread.
- void **create_datafile** (const std::string &prefix, bool use_date)
  Open data file for output.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- std::atomic< bool > **close_session**
  cross-thread-synchronization.
- const bool **active**
  The writer thread and the output file will only be created when active is true.
- std::unique_ptr< **mha_fifo_lf_t**< **output_type** > > **fifo**
  Fifo for decoupling signal processing thread from disk writer thread.
- const unsigned int **disk_write_threshold_min_num_samples**
  Minimum number of samples that need to be waiting in the fifo before the disk writer thread writes them to disk.
- std::thread **writethread**
  The thread that writes to disk.
- std::unique_ptr< **output_type**[ ]> **diskbuffer**
  Intermediate buffer to receive data from fifo and store on disk.
- std::fstream **outfile**
  Ouput file.
- unsigned **num_channels** = 0U
  Number of channels of AC variable using stride.
- bool **is_num_channels_known** = false
  The number of channels is determined during the first process callback.
- bool **is_complex** = false
  If the AC variable is of complex valued type or not.
- const std::string **varname**
  The name of the ac variable to publish.

5.403.1 Detailed Description

acwriter_t (p. 1382) decouples signal processing from writing to disk.

Data arriving in numeric AC variables is converted to data type double, placed into a fifo pipeline to transport the data from the signal processing thread to the disk writing thread, and finally written to disk in chunks of at least minwrite numbers. All numbers are written to disk as binary doubles (8 bytes) in host byte order.

5.403.2 Member Typedef Documentation

5.403.2.1 output_type `typedef double` **plugins::hoertech::acrec::acwriter_t::output↩_type**

The numeric data type used for outputting the data to disk.

© 2005-2021 HörTech gGmbH, Oldenburg

5.403.3 Constructor & Destructor Documentation

5.403.3.1 acwriter_t()

```
acwriter_t::acwriter_t (
    bool active,
    unsigned fifosize,
    unsigned minwrite,
    const std::string & prefix,
    bool use_date,
    const std::string & varname )
```

Constructor allocates fifo and disk output buffer.

It spawns a new thread for writing data to disk when active==true. In order to terminate the thread, method exit_request **must** be called before this object is destroyed.

Parameters

active	Only write data to disk when this is true.
fifosize	Capacity of both the fifo pipeline and of the disk buffer.
minwrite	Wait for a fifo fill count of at least minwrite doubles before flushing the contents of the fifo to disk. Fifo is also flushed before this object is destroyed.
prefix	Path and start of output file name. Will be extended with file name extension ".dat".
use_date	When true, the current date and time will be appended to the output file name before the file name extension.
varname	Name of AC variable to save into file. Can be accessed through getter method **get_varname()** (p. 1385). Stored here to avoid races between processing thread and configuration thread.

5.403.3.2 ~acwriter_t()

```
plugins::hoertech::acrec::acwriter_t::~acwriter_t ( ) [default]
```

Deallocates memory but does not terminate the write_thread.

write_thread must be terminated before the destructor executes by calling exit_request.

5.403.4 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.403.4.1 process()

```
void acwriter_t::process (
    comm_var_t * s )
```

Place the data present in the algorithm communication variable into the fifo for output to disk.

5.403.4.2 exit_request()

```
void acwriter_t::exit_request ( )
```

Terminate output thread.

5.403.4.3 get_varname()

```
const char* plugins::hoertech::acrec::acwriter_t::get_↩
varname ( ) const [inline]
```

getter for ac variable name

Returns

name as char∗ as needed by get_var

5.403.4.4 write_thread()

```
void acwriter_t::write_thread ( ) [private]
```

Main method of the disk writer thread.

Periodically wakes up and checks if data needs to be written to disk.

5.403.4.5 create_datafile()

```
void acwriter_t::create_datafile (
    const std::string & prefix,
    bool use_date ) [private]
```

Open data file for output.

Combine prefix, date, and file name extension

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

prefix	Path and start of output file name. Will be extended with file name extension ".dat".
use_date	When true, the current date and time will be appended to the output file name before the file name extension.

5.403.5 Member Data Documentation

5.403.5.1 close_session `std::atomic<bool> plugins::hoertech::acrec::acwriter_t↩ ::close_session [private]`

cross-thread-synchronization.

write_thread() (p. 1385) terminates after this is set to true by **exit_request()** (p. 1385).

5.403.5.2 active `const bool plugins::hoertech::acrec::acwriter_t::active [private]`

The writer thread and the output file will only be created when active is true.

5.403.5.3 fifo `std::unique_ptr< mha_fifo_lf_t< output_type> > plugins::hoertech↩ ::acrec::acwriter_t::fifo [private]`

Fifo for decoupling signal processing thread from disk writer thread.

5.403.5.4 disk_write_threshold_min_num_samples `const unsigned int plugins↩ ::hoertech::acrec::acwriter_t::disk_write_threshold_min_num_samples [private]`

Minimum number of samples that need to be waiting in the fifo before the disk writer thread writes them to disk.

© 2005-2021 HörTech gGmbH, Oldenburg

5.403.5.5 writethread `std::thread plugins::hoertech::acrec::acwriter_t::writethread [private]`

The thread that writes to disk.

5.403.5.6 diskbuffer `std::unique_ptr< output_type[]> plugins::hoertech::acrec↩ ::acwriter_t::diskbuffer [private]`

Intermediate buffer to receive data from fifo and store on disk.

5.403.5.7 outfile `std::fstream plugins::hoertech::acrec::acwriter_t::outfile [private]`

Ouput file.

5.403.5.8 num_channels `unsigned plugins::hoertech::acrec::acwriter_t::num_↩ channels = 0U [private]`

Number of channels of AC variable using stride.

If the number of channels changes during processing, an exception is thrown.

5.403.5.9 is_num_channels_known `bool plugins::hoertech::acrec::acwriter_t::is↩ _num_channels_known = false [private]`

The number of channels is determined during the first process callback.

is_num_channels_known is set to true after the first process callback.

5.403.5.10 is_complex `bool plugins::hoertech::acrec::acwriter_t::is_complex = false [private]`

If the AC variable is of complex valued type or not.

If this changes during processing, then an exception is thrown.

© 2005-2021 HörTech gGmbH, Oldenburg

5.403.5.11 varname `const std::string plugins::hoertech::acrec::acwriter_t::varname [private]`

The name of the ac variable to publish.

The documentation for this class was generated from the following files:

- **acrec.hh**
- **acrec.cpp**

5.404 proc_counter_t Class Reference

Inheritance diagram for proc_counter_t:

Public Member Functions

- **proc_counter_t** (**algo_comm_t** iac, const std::string &configured_name)
- ~**proc_counter_t** ()
- **mha_wave_t** * **process** (**mha_wave_t** *s)
- **mha_spec_t** * **process** (**mha_spec_t** *s)
- void **prepare_** (**mhaconfig_t** &)
- void **release_** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **algo_comm_t ac**

Additional Inherited Members

5.404.1 Constructor & Destructor Documentation

5.404.1.1 proc_counter_t()

```
proc_counter_t::proc_counter_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.404.1.2 ~proc_counter_t()

```
proc_counter_t::~proc_counter_t ( )
```

5.404.2 Member Function Documentation

5.404.2.1 process() [1/2]

```
mha_wave_t * proc_counter_t::process (
    mha_wave_t * s )
```

5.404.2.2 process() [2/2]

```
mha_spec_t * proc_counter_t::process (
    mha_spec_t * s )
```

5.404.2.3 prepare_()

```
void proc_counter_t::prepare_ (
    mhaconfig_t & ) [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.404.2.4 release_() `void proc_counter_t::release_ ( ) [inline]`

5.404.3 Member Data Documentation

5.404.3.1 ac `algo_comm_t proc_counter_t::ac [private]`

The documentation for this class was generated from the following file:

- **proc_counter.cpp**

5.405 rmslevel::mon_t Class Reference

Inheritance diagram for rmslevel::mon_t:

Public Member Functions

- **mon_t** (unsigned int nch, const std::string & **name**, **algo_comm_t** **ac**, const std::string &base, **MHAParser::parser_t** &p, const std::string & **help**)
- **mon_t** (const **mon_t** &)=delete
- **mon_t** (**mon_t** &&)=delete
- **mon_t** & **operator=** (const **mon_t** &)=delete
- **mon_t** & **operator=** (**mon_t** &&)=delete
- void **store** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.405.1 Constructor & Destructor Documentation

5.405.1.1 mon_t() [1/3]

```
rmslevel::mon_t::mon_t (
    unsigned int nch,
    const std::string & name,
    algo_comm_t ac,
    const std::string & base,
    MHAParser::parser_t & p,
    const std::string & help )
```

5.405.1.2 mon_t() [2/3]

```
rmslevel::mon_t::mon_t (
    const mon_t & ) [delete]
```

5.405.1.3 mon_t() [3/3]

```
rmslevel::mon_t::mon_t (
    mon_t && ) [delete]
```

5.405.2 Member Function Documentation

5.405.2.1 operator=() [1/2]

```
mon_t& rmslevel::mon_t::operator= (
    const mon_t & ) [delete]
```

5.405.2.2 operator=() [2/2]

```
mon_t& rmslevel::mon_t::operator= (
    mon_t && ) [delete]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.405.2.3 store() `void rmslevel::mon_t::store ( )`

The documentation for this class was generated from the following file:

- **rmslevel.cpp**

5.406 rmslevel::rmslevel_if_t Class Reference

Interface class of the rmslevel plugin.

Inheritance diagram for rmslevel::rmslevel_if_t:

Public Member Functions

- **rmslevel_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **update** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAEvents::patchbay_t< rmslevel_if_t > patchbay**
- std::string **name**
- **MHAParser::kw_t unit**

Additional Inherited Members

5.406.1 Detailed Description

Interface class of the rmslevel plugin.

5.406.2 Constructor & Destructor Documentation

5.406.2.1 rmslevel_if_t()

```
rmslevel::rmslevel_if_t::rmslevel_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.406.3 Member Function Documentation

5.406.3.1 process() [1/2]

```
mha_spec_t * rmslevel::rmslevel_if_t::process (
    mha_spec_t * s )
```

5.406.3.2 process() [2/2]

```
mha_wave_t * rmslevel::rmslevel_if_t::process (
    mha_wave_t * s )
```

5.406.3.3 prepare()

```
void rmslevel::rmslevel_if_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< rmslevel_t >** (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.406.3.4 update() `void rmslevel::rmslevel_if_t::update ( ) [private]`

5.406.4 Member Data Documentation

5.406.4.1 patchbay `MHAEvents::patchbay_t< rmslevel_if_t> rmslevel::rmslevel_if↩_t::patchbay [private]`

5.406.4.2 name `std::string rmslevel::rmslevel_if_t::name [private]`

5.406.4.3 unit `MHAParser::kw_t rmslevel::rmslevel_if_t::unit [private]`

The documentation for this class was generated from the following file:

- **rmslevel.cpp**

5.407 rmslevel::rmslevel_t Class Reference

Run-time configuration class of the rmslevel plugin.

Public Member Functions

- **rmslevel_t** (const **mhaconfig_t** &tf, **algo_comm_t** ac, const std::string &name, **MH↩AParser::parser_t** &p, **UNIT** unit_)
  C'tor of the runtime configuration class.
- ∼**rmslevel_t** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **fill** (**mha_spec_t** ∗)
- void **sum_and_fill** (**mha_spec_t** ∗)
- void **fill** (**mha_wave_t** ∗)
- void **sum_and_fill** (**mha_wave_t** ∗)
- void **insert** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- std::unordered_map< std::string, std::unique_ptr< **mon_t** > > **monitors**
- unsigned int **fftlen**
- **UNIT unit**
- **mha_domain_t domain**
- std::vector< **mha_real_t** > **freq_offsets**

  freq_offsets provides the conversion of dB(SPL) to dB(HL) for every frequency bin in the stft used by coloured_intensity.

5.407.1 Detailed Description

Run-time configuration class of the rmslevel plugin.

5.407.2 Constructor & Destructor Documentation

5.407.2.1 rmslevel_t()

```
rmslevel::rmslevel_t::rmslevel_t (
    const mhaconfig_t & tf,
    algo_comm_t ac,
    const std::string & name,
    MHAParser::parser_t & p,
    UNIT unit_ )
```

C'tor of the runtime configuration class.

Parameters

tf	Input signal configuration
ac	AC space, needed to publish ac variables
name	Configured name of the plugin within the parser structure
p	Instance of the parent interface class, needed to publish monitor variables
*unit_*	Configured dB scale
*num_channels_*	Number of monitor channels if channel summation shall be used, zero otherwise

5.407.2.2 ~rmslevel_t()

```
rmslevel::rmslevel_t::~rmslevel_t ( ) [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.407.3 Member Function Documentation

5.407.3.1 process() [1/2]

```
mha_spec_t * rmslevel::rmslevel_t::process (
    mha_spec_t * s )
```

5.407.3.2 process() [2/2]

```
mha_wave_t * rmslevel::rmslevel_t::process (
    mha_wave_t * s )
```

5.407.3.3 fill() [1/2]

```
void rmslevel::rmslevel_t::fill (
    mha_spec_t *  )
```

5.407.3.4 sum_and_fill() [1/2]

```
void rmslevel::rmslevel_t::sum_and_fill (
    mha_spec_t *  )
```

5.407.3.5 fill() [2/2]

```
void rmslevel::rmslevel_t::fill (
    mha_wave_t *  )
```

5.407.3.6 sum_and_fill() [2/2]

```
void rmslevel::rmslevel_t::sum_and_fill (
    mha_wave_t *  )
```

5.407.3.7 insert()

```
void rmslevel::rmslevel_t::insert ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.407.4 Member Data Documentation

5.407.4.1 monitors `std::unordered_map<std::string,std::unique_ptr< mon_t> > rmslevel::rmslevel_t::monitors [private]`

5.407.4.2 fftlen `unsigned int rmslevel::rmslevel_t::fftlen [private]`

5.407.4.3 unit `UNIT rmslevel::rmslevel_t::unit [private]`

5.407.4.4 domain `mha_domain_t rmslevel::rmslevel_t::domain [private]`

5.407.4.5 freq_offsets `std::vector< mha_real_t> rmslevel::rmslevel_t::freq_offsets [private]`

freq_offsets provides the conversion of dB(SPL) to dB(HL) for every frequency bin in the stft used by coloured_intensity.

Unused when not in spectral domain and unit=hl.

The documentation for this class was generated from the following file:

- **rmslevel.cpp**

5.408 rohBeam::configOptions Struct Reference

Public Attributes

- bool **enable_adaptive_beam**
- int **binaural_type_index**
- float **alpha_postfilter**
- float **alpha_blocking_XkXi**
- float **alpha_blocking_XkY**

© 2005-2021 HörTech gGmbH, Oldenburg

5.408.1 Member Data Documentation

5.408.1.1 enable_adaptive_beam `bool rohBeam::configOptions::enable_adaptive_beam`

5.408.1.2 binaural_type_index `int rohBeam::configOptions::binaural_type_index`

5.408.1.3 alpha_postfilter `float rohBeam::configOptions::alpha_postfilter`

5.408.1.4 alpha_blocking_XkXi `float rohBeam::configOptions::alpha_blocking_XkXi`

5.408.1.5 alpha_blocking_XkY `float rohBeam::configOptions::alpha_blocking_XkY`

The documentation for this struct was generated from the following file:

- **rohBeam.hh**

© 2005-2021 HörTech gGmbH, Oldenburg

5.409 rohBeam::rohBeam Class Reference

Inheritance diagram for rohBeam::rohBeam:

Public Member Functions

- **rohBeam** (**algo_comm_t** iac, const std::string &configured_name)
- ∼**rohBeam** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** (void)

Private Member Functions

- void **update_cfg** ()
- float **compute_head_model_T** (float)
- float **compute_head_model_alpha** (float)
- Eigen::MatrixXcf ∗ **compute_head_model_mat** (float src_az_degrees)
- **MHASignal::matrix_t** ∗ **compute_delaycomp_vec** (Eigen::MatrixXcf ∗headModel)
- std::vector< Eigen::MatrixXcf > ∗ **noise_integrate_hrtf** ()
- Eigen::VectorXcf **solve_MVDR** (Eigen::VectorXcf propVec, Eigen::MatrixXcf noiseM)
- const Eigen::MatrixXf **compute_uncorr** (float w)
- const Eigen::MatrixXf **compute_diff2D** (float)

© 2005-2021 HörTech gGmbH, Oldenburg

- const Eigen::MatrixXf **compute_diff3D** (float)
- **MHASignal::matrix_t** ∗ **compute_beamW** (Eigen::MatrixXcf ∗)
- float **compute_wng** (Eigen::VectorXcf freqRes, Eigen::VectorXcf propVec)
- void **export_beam_design** (const **MHASignal::matrix_t** &beamW, const Eigen::↩ MatrixXcf &headModel)
- **noiseFuncPtr get_noise_model_func** (void)
- void **on_model_param_valuechanged** ()

Private Attributes

- const typedef Eigen::MatrixXf(rohBeam::∗ **noiseFuncPtr**)(float)
- **MHAParser::kw_t prop_type**
- **MHAParser::string_t sampled_hrir_path**
- **MHAParser::float_t source_azimuth_degrees**
- **MHAParser::vfloat_t mic_azimuth_degrees_vec**
- **MHAParser::float_t head_model_sphere_radius_cm**
- **MHAParser::mfloat_t intermic_distance_cm**
- **MHAParser::kw_t noise_field_model**
- **MHAParser::bool_t enable_adaptive_beam**
- **MHAParser::kw_t binaural_type**
- **MHAParser::float_t diag_loading_mu**
- **MHAParser::bool_t enable_export**
- **MHAParser::bool_t enable_wng_optimization**
- **MHAParser::float_t tau_postfilter_ms**
- **MHAParser::float_t tau_blocking_XkXi_ms**
- **MHAParser::float_t tau_blocking_XkY_ms**
- **MHAEvents::patchbay_t**< **rohBeam** > **patchbay**
- bool **prepared**
- **MHA_AC::spectrum_t** ∗ **beamExport**
- **MHA_AC::waveform_t** ∗ **noiseModelExport**
- **MHA_AC::spectrum_t** ∗ **propExport**

Additional Inherited Members

5.409.1 Constructor & Destructor Documentation

5.409.1.1 rohBeam()

```
rohBeam::rohBeam::rohBeam (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.409.1.2 ~rohBeam() `rohBeam::rohBeam::~rohBeam ( )`

5.409.2 Member Function Documentation

5.409.2.1 process() `mha_spec_t* rohBeam::rohBeam::process ( mha_spec_t * )`

5.409.2.2 prepare() `void rohBeam::rohBeam::prepare ( mhaconfig_t & ) [virtual]`

Implements **MHAPlugin::plugin_t< rohConfig >** (p. 1153).

5.409.2.3 release() `void rohBeam::rohBeam::release ( void ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t< rohConfig >** (p. 1154).

5.409.2.4 update_cfg() `void rohBeam::rohBeam::update_cfg ( ) [private]`

5.409.2.5 compute_head_model_T() `float rohBeam::rohBeam::compute_head_model_T ( float ) [private]`

5.409.2.6 compute_head_model_alpha() `float rohBeam::rohBeam::compute_head_model_alpha ( float ) [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.409.2.7 compute_head_model_mat()

```
Eigen::MatrixXcf* rohBeam::rohBeam::compute_head_model_mat (
        float src_az_degrees ) [private]
```

5.409.2.8 compute_delaycomp_vec()

```
MHASignal::matrix_t* rohBeam::rohBeam::compute_delaycomp_vec (
        Eigen::MatrixXcf * headModel ) [private]
```

5.409.2.9 noise_integrate_hrtf()

```
std::vector<Eigen::MatrixXcf>* rohBeam::rohBeam::noise_integrate_hrtf ( ) [private]
```

5.409.2.10 solve_MVDR()

```
Eigen::VectorXcf rohBeam::rohBeam::solve_MVDR (
        Eigen::VectorXcf propVec,
        Eigen::MatrixXcf noiseM ) [private]
```

5.409.2.11 compute_uncorr()

```
const Eigen::MatrixXf rohBeam::rohBeam::compute_uncorr (
        float w ) [private]
```

5.409.2.12 compute_diff2D()

```
const Eigen::MatrixXf rohBeam::rohBeam::compute_diff2D (
        float ) [private]
```

5.409.2.13 compute_diff3D()

```
const Eigen::MatrixXf rohBeam::rohBeam::compute_diff3D (
        float ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.409.2.14 compute_beamW()

```
MHASignal::matrix_t* rohBeam::rohBeam::compute_beamW (
    Eigen::MatrixXcf * ) [private]
```

5.409.2.15 compute_wng()

```
float rohBeam::rohBeam::compute_wng (
    Eigen::VectorXcf freqRes,
    Eigen::VectorXcf propVec ) [private]
```

5.409.2.16 export_beam_design()

```
void rohBeam::rohBeam::export_beam_design (
    const MHASignal::matrix_t & beamW,
    const Eigen::MatrixXcf & headModel ) [private]
```

5.409.2.17 get_noise_model_func()

```
noiseFuncPtr rohBeam::rohBeam::get_noise_↩
model_func (
    void ) [private]
```

5.409.2.18 on_model_param_valuechanged()

```
void rohBeam::rohBeam::on_model_↩
param_valuechanged ( ) [private]
```

5.409.3 Member Data Documentation

5.409.3.1 noiseFuncPtr

```
const typedef Eigen::MatrixXf(rohBeam::* rohBeam::rohBeam↩
::noiseFuncPtr) (float) [private]
```

5.409.3.2 prop_type

```
MHAParser::kw_t rohBeam::rohBeam::prop_type [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.409.3.3 sampled_hrir_path

MHAParser::string_t rohBeam::rohBeam::sampled_hrir_path [private]

5.409.3.4 source_azimuth_degrees

MHAParser::float_t rohBeam::rohBeam::source_azimuth_degrees [private]

5.409.3.5 mic_azimuth_degrees_vec

MHAParser::vfloat_t rohBeam::rohBeam::mic_azimuth_degrees_vec [private]

5.409.3.6 head_model_sphere_radius_cm

MHAParser::float_t rohBeam::rohBeam::head_model_sphere_radius_cm [private]

5.409.3.7 intermic_distance_cm

MHAParser::mfloat_t rohBeam::rohBeam::intermic_distance_cm [private]

5.409.3.8 noise_field_model

MHAParser::kw_t rohBeam::rohBeam::noise_field_model [private]

5.409.3.9 enable_adaptive_beam

MHAParser::bool_t rohBeam::rohBeam::enable_adaptive_beam [private]

5.409.3.10 binaural_type

MHAParser::kw_t rohBeam::rohBeam::binaural_type [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.409.3.11 diag_loading_mu

MHAParser::float_t rohBeam::rohBeam::diag_loading_mu [private]

5.409.3.12 enable_export

MHAParser::bool_t rohBeam::rohBeam::enable_export [private]

5.409.3.13 enable_wng_optimization

MHAParser::bool_t rohBeam::rohBeam::enable_↩wng_optimization [private]

5.409.3.14 tau_postfilter_ms

MHAParser::float_t rohBeam::rohBeam::tau_postfilter↩_ms [private]

5.409.3.15 tau_blocking_XkXi_ms

MHAParser::float_t rohBeam::rohBeam::tau_↩blocking_XkXi_ms [private]

5.409.3.16 tau_blocking_XkY_ms

MHAParser::float_t rohBeam::rohBeam::tau_blocking↩_XkY_ms [private]

5.409.3.17 patchbay

MHAEvents::patchbay_t< **rohBeam**> rohBeam::rohBeam::patchbay [private]

5.409.3.18 prepared

bool rohBeam::rohBeam::prepared [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.409.3.19 beamExport `MHA_AC::spectrum_t* rohBeam::rohBeam::beamExport [private]`

5.409.3.20 noiseModelExport `MHA_AC::waveform_t* rohBeam::rohBeam::noiseModel↩ Export [private]`

5.409.3.21 propExport `MHA_AC::spectrum_t* rohBeam::rohBeam::propExport [private]`

The documentation for this class was generated from the following file:

- **rohBeam.hh**

5.410 rohBeam::rohConfig Class Reference

Public Member Functions

- **rohConfig** (const **mhaconfig_t** **in_cfg**, const **mhaconfig_t** **out_cfg**, std::unique_ptr< Eigen::MatrixXcf > headModel_, std::unique_ptr< **MHASignal::matrix_t** > beamW_, std::unique_ptr< **MHASignal::matrix_t** > delayComp_, const **configOptions** &options)
- **rohConfig** (**rohConfig** *lastConfig, const **mhaconfig_t**, const **mhaconfig_t** **out_↩ cfg**, std::unique_ptr< Eigen::MatrixXcf > headModel_, std::unique_ptr< **MHASignal↩ ::matrix_t** > beamW_, std::unique_ptr< **MHASignal::matrix_t** > delayComp_, const **configOptions** &options)
- ~**rohConfig** ()
- **rohConfig** (const **rohConfig** &)=delete
- **rohConfig** & **operator=** (const **rohConfig** &)=delete
- **mha_spec_t** * **process** (**mha_spec_t** *)
- void **init_dynamic** ()

Private Member Functions

- void **phasereconstruction** (**MHASignal::spectrum_t** *)
- void **postfilter** (**mha_spec_t** *, **MHASignal::spectrum_t** *)
- void **copyfixedbfoutput** (**MHASignal::spectrum_t** *)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- int **nfreq**
- int **nchan_block**
- **mhaconfig_t in_cfg**
- **mhaconfig_t out_cfg**
- bool **enable_adaptive_beam**
- int **binaural_type_index**
- std::unique_ptr< Eigen::MatrixXcf > **headModel**
- std::unique_ptr< **MHASignal::matrix_t** > **beamW**
- std::unique_ptr< **MHASignal::matrix_t** > **delayComp**
- **MHASignal::spectrum_t** * **beam1**
- **MHASignal::spectrum_t** * **beamA**
- **MHASignal::spectrum_t** * **blockSpec**
- **MHASignal::spectrum_t** * **outSpec**
- float **alpha_postfilter**
- float **alpha_blocking_XkXi**
- float **alpha_blocking_XkY**
- std::vector< Eigen::MatrixXcf > **corrXpXp**
- std::vector< Eigen::VectorXcf > **corrXpYf**
- Eigen::VectorXf **corrZZ**
- Eigen::VectorXf **corrLL**
- Eigen::VectorXf **corrRR**
- Eigen::HouseholderQR< Eigen::MatrixXcf > **hhCorrXpXp**
- Eigen::VectorXcf **nextXpYf**
- Eigen::VectorXcf **blockXp**
- Eigen::VectorXcf **freqResp**
- Eigen::ArrayXf **magResp**
- float **minLim**
- float **maxLim**

5.410.1 Constructor & Destructor Documentation

5.410.1.1 rohConfig() [1/3]

```
rohBeam::rohConfig::rohConfig (
    const mhaconfig_t in_cfg,
    const mhaconfig_t out_cfg,
    std::unique_ptr< Eigen::MatrixXcf > headModel_,
    std::unique_ptr< MHASignal::matrix_t > beamW_,
    std::unique_ptr< MHASignal::matrix_t > delayComp_,
    const configOptions & options )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.410.1.2 rohConfig() [2/3]

```
rohBeam::rohConfig::rohConfig (
    rohConfig * lastConfig,
    const mhaconfig_t,
    const mhaconfig_t out_cfg,
    std::unique_ptr< Eigen::MatrixXcf > headModel_,
    std::unique_ptr< MHASignal::matrix_t > beamW_,
    std::unique_ptr< MHASignal::matrix_t > delayComp_,
    const configOptions & options )
```

5.410.1.3 ~rohConfig()

```
rohBeam::rohConfig::~rohConfig ( )
```

5.410.1.4 rohConfig() [3/3]

```
rohBeam::rohConfig::rohConfig (
    const rohConfig & ) [delete]
```

5.410.2 Member Function Documentation

5.410.2.1 operator=()

```
rohConfig& rohBeam::rohConfig::operator= (
    const rohConfig & ) [delete]
```

5.410.2.2 process()

```
mha_spec_t * rohBeam::rohConfig::process (
    mha_spec_t * inSpec )
```

5.410.2.3 init_dynamic()

```
void rohBeam::rohConfig::init_dynamic ( )
```

5.410.2.4 phasereconstruction()

```
void rohBeam::rohConfig::phasereconstruction (
    MHASignal::spectrum_t * prevSpecPost ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.410.2.5 postfilter()

```
void rohBeam::rohConfig::postfilter (
    mha_spec_t * inSpec,
    MHASignal::spectrum_t * prevSpecPost ) [private]
```

5.410.2.6 copyfixedbfoutput()

```
void rohBeam::rohConfig::copyfixedbfoutput (
    MHASignal::spectrum_t * prevSpecPost ) [private]
```

5.410.3 Member Data Documentation

5.410.3.1 nfreq

```
int rohBeam::rohConfig::nfreq [private]
```

5.410.3.2 nchan_block

```
int rohBeam::rohConfig::nchan_block [private]
```

5.410.3.3 in_cfg

```
mhaconfig_t rohBeam::rohConfig::in_cfg [private]
```

5.410.3.4 out_cfg

```
mhaconfig_t rohBeam::rohConfig::out_cfg [private]
```

5.410.3.5 enable_adaptive_beam

```
bool rohBeam::rohConfig::enable_adaptive_beam
[private]
```

5.410.3.6 binaural_type_index

```
int rohBeam::rohConfig::binaural_type_index [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.410.3.7 headModel

`std::unique_ptr<Eigen::MatrixXcf> rohBeam::rohConfig::headModel [private]`

5.410.3.8 beamW

`std::unique_ptr< MHASignal::matrix_t> rohBeam::rohConfig::beamW [private]`

5.410.3.9 delayComp

`std::unique_ptr< MHASignal::matrix_t> rohBeam::rohConfig::delayComp [private]`

5.410.3.10 beam1

`MHASignal::spectrum_t* rohBeam::rohConfig::beam1 [private]`

5.410.3.11 beamA

`MHASignal::spectrum_t* rohBeam::rohConfig::beamA [private]`

5.410.3.12 blockSpec

`MHASignal::spectrum_t* rohBeam::rohConfig::blockSpec [private]`

5.410.3.13 outSpec

`MHASignal::spectrum_t* rohBeam::rohConfig::outSpec [private]`

5.410.3.14 alpha_postfilter

`float rohBeam::rohConfig::alpha_postfilter [private]`

5.410.3.15 alpha_blocking_XkXi

`float rohBeam::rohConfig::alpha_blocking_XkXi [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.410.3.16 alpha_blocking_XkY `float rohBeam::rohConfig::alpha_blocking_XkY [private]`

5.410.3.17 corrXpXp `std::vector<Eigen::MatrixXcf> rohBeam::rohConfig::corrXpXp [private]`

5.410.3.18 corrXpYf `std::vector<Eigen::VectorXcf> rohBeam::rohConfig::corrXpYf [private]`

5.410.3.19 corrZZ `Eigen::VectorXf rohBeam::rohConfig::corrZZ [private]`

5.410.3.20 corrLL `Eigen::VectorXf rohBeam::rohConfig::corrLL [private]`

5.410.3.21 corrRR `Eigen::VectorXf rohBeam::rohConfig::corrRR [private]`

5.410.3.22 hhCorrXpXp `Eigen::HouseholderQR<Eigen::MatrixXcf> rohBeam::rohConfig↩ ::hhCorrXpXp [private]`

5.410.3.23 nextXpYf `Eigen::VectorXcf rohBeam::rohConfig::nextXpYf [private]`

5.410.3.24 blockXp `Eigen::VectorXcf rohBeam::rohConfig::blockXp [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.410.3.25 freqResp `Eigen::VectorXcf rohBeam::rohConfig::freqResp [private]`

5.410.3.26 magResp `Eigen::ArrayXf rohBeam::rohConfig::magResp [private]`

5.410.3.27 minLim `float rohBeam::rohConfig::minLim [private]`

5.410.3.28 maxLim `float rohBeam::rohConfig::maxLim [private]`

The documentation for this class was generated from the following files:

- **rohBeam.hh**
- **rohBeam.cpp**

5.411 route::interface_t Class Reference

Inheritance diagram for route::interface_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **interface_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- **mha_wave_t** $*$ **process** (**mha_wave_t** $*$)
- **mha_spec_t** $*$ **process** (**mha_spec_t** $*$)

Private Member Functions

- void **update** ()

Private Attributes

- **MHAEvents::patchbay_t**$<$ **route::interface_t** $>$ **patchbay**
- **MHAParser::vstring_t route_out**
- **MHAParser::vstring_t route_ac**
- **mhaconfig_t cfin**
- **mhaconfig_t cfout**
- **mhaconfig_t cfac**
- bool **prepared**
- bool **stopped**
- std::string **algo**

Additional Inherited Members

5.411.1 Constructor & Destructor Documentation

5.411.1.1 interface_t()

```
route::interface_t::interface_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.411.2 Member Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.411.2.1 prepare() `void route::interface_t::prepare ( mhaconfig_t & cf ) [virtual]`

Implements **MHAPlugin::plugin_t< route::process_t >** (p. 1153).

5.411.2.2 release() `void route::interface_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t< route::process_t >** (p. 1154).

5.411.2.3 process() [1/2] `mha_wave_t * route::interface_t::process ( mha_wave_t * s )`

5.411.2.4 process() [2/2] `mha_spec_t * route::interface_t::process ( mha_spec_t * s )`

5.411.2.5 update() `void route::interface_t::update ( ) [private]`

5.411.3 Member Data Documentation

5.411.3.1 patchbay `MHAEvents::patchbay_t< route::interface_t> route::interface_t::patchbay [private]`

5.411.3.2 route_out `MHAParser::vstring_t route::interface_t::route_out [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.411.3.3 route_ac `MHAParser::vstring_t route::interface_t::route_ac [private]`

5.411.3.4 cfin `mhaconfig_t route::interface_t::cfin [private]`

5.411.3.5 cfout `mhaconfig_t route::interface_t::cfout [private]`

5.411.3.6 cfac `mhaconfig_t route::interface_t::cfac [private]`

5.411.3.7 prepared `bool route::interface_t::prepared [private]`

5.411.3.8 stopped `bool route::interface_t::stopped [private]`

5.411.3.9 algo `std::string route::interface_t::algo [private]`

The documentation for this class was generated from the following file:

- **route.cpp**

5.412 route::process_t Class Reference

Public Member Functions

- **process_t** (**algo_comm_t** iac, const std::string acname, const std::vector< std::string > &r_out, const std::vector< std::string > &r_ac, const **mhaconfig_t** &cf_in, const **mhaconfig_t** &cf_out, const **mhaconfig_t** &cf_ac, bool sync)
- **mha_wave_t** * **process** (**mha_wave_t** *)
- **mha_spec_t** * **process** (**mha_spec_t** *)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAMultiSrc::waveform_t wout**
- **MHAMultiSrc::spectrum_t sout**
- **MHAMultiSrc::waveform_t wout_ac**
- **MHAMultiSrc::spectrum_t sout_ac**

5.412.1 Constructor & Destructor Documentation

5.412.1.1 process_t()

```
route::process_t::process_t (
    algo_comm_t iac,
    const std::string acname,
    const std::vector< std::string > & r_out,
    const std::vector< std::string > & r_ac,
    const mhaconfig_t & cf_in,
    const mhaconfig_t & cf_out,
    const mhaconfig_t & cf_ac,
    bool sync )
```

5.412.2 Member Function Documentation

5.412.2.1 process() [1/2]

```
mha_wave_t * route::process_t::process (
    mha_wave_t * s )
```

5.412.2.2 process() [2/2]

```
mha_spec_t * route::process_t::process (
    mha_spec_t * s )
```

5.412.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.412.3.1 wout `MHAMultiSrc::waveform_t route::process_t::wout [private]`

5.412.3.2 sout `MHAMultiSrc::spectrum_t route::process_t::sout [private]`

5.412.3.3 wout_ac `MHAMultiSrc::waveform_t route::process_t::wout_ac [private]`

5.412.3.4 sout_ac `MHAMultiSrc::spectrum_t route::process_t::sout_ac [private]`

The documentation for this class was generated from the following file:

- **route.cpp**

5.413 rt_nlms_t Class Reference

Public Member Functions

- **rt_nlms_t** (**algo_comm_t** iac, const std::string &name, const **mhaconfig_t** &cfg, unsigned int ntaps_, const std::string &name_u, const std::string &name_d, const std::string &name_e, const std::string &name_f, const int n_no_update)
- ∼**rt_nlms_t** ()
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗sUD, **mha_real_t** rho, **mha_real_t** c, unsigned int norm_type, unsigned int estim_type, **mha_real_t** lambda_smooth)
- void **insert** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **algo_comm_t ac**
- unsigned int **ntaps**
- unsigned int **frames**
- unsigned int **channels**
- **MHA_AC::waveform_t F**
- **MHASignal::waveform_t U**

  Input signal cache.
- **MHASignal::waveform_t Uflt**

  Input signal cache (second filter)
- **MHASignal::waveform_t Pu**

  Power of input signal delayline.
- **MHASignal::waveform_t fu**

  Filtered input signal.
- **MHASignal::waveform_t fuflt**

  Filtered input signal.
- **MHASignal::waveform_t fu_previous**
- **MHASignal::waveform_t y_previous**
- **MHASignal::waveform_t P_Sum**
- std::string **name_u_**
- std::string **name_d_**
- std::string **name_e_**
- int **n_no_update_**
- int **no_iter**
- **mha_wave_t s_E**

5.413.1 Constructor & Destructor Documentation

5.413.1.1 rt_nlms_t()

```
rt_nlms_t::rt_nlms_t (
    algo_comm_t iac,
    const std::string & name,
    const mhaconfig_t & cfg,
    unsigned int ntaps_,
    const std::string & name_u,
    const std::string & name_d,
    const std::string & name_e,
    const std::string & name_f,
    const int n_no_update )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.413.1.2 ~rt_nlms_t() `rt_nlms_t::~rt_nlms_t ( ) [inline]`

5.413.2 Member Function Documentation

5.413.2.1 process()

```
mha_wave_t * rt_nlms_t::process (
    mha_wave_t * sUD,
    mha_real_t rho,
    mha_real_t c,
    unsigned int norm_type,
    unsigned int estim_type,
    mha_real_t lambda_smooth )
```

5.413.2.2 insert() `void rt_nlms_t::insert ( )`

5.413.3 Member Data Documentation

5.413.3.1 ac `algo_comm_t rt_nlms_t::ac [private]`

5.413.3.2 ntaps `unsigned int rt_nlms_t::ntaps [private]`

5.413.3.3 frames `unsigned int rt_nlms_t::frames [private]`

5.413.3.4 channels `unsigned int rt_nlms_t::channels [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.413.3.5 F `MHA_AC::waveform_t rt_nlms_t::F [private]`

5.413.3.6 U `MHASignal::waveform_t rt_nlms_t::U [private]`

Input signal cache.

5.413.3.7 Uflt `MHASignal::waveform_t rt_nlms_t::Uflt [private]`

Input signal cache (second filter)

5.413.3.8 Pu `MHASignal::waveform_t rt_nlms_t::Pu [private]`

Power of input signal delayline.

5.413.3.9 fu `MHASignal::waveform_t rt_nlms_t::fu [private]`

Filtered input signal.

5.413.3.10 fuflt `MHASignal::waveform_t rt_nlms_t::fuflt [private]`

Filtered input signal.

5.413.3.11 fu_previous `MHASignal::waveform_t rt_nlms_t::fu_previous [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.413.3.12 y_previous `MHASignal::waveform_t rt_nlms_t::y_previous [private]`

5.413.3.13 P_Sum `MHASignal::waveform_t rt_nlms_t::P_Sum [private]`

5.413.3.14 name_u_ `std::string rt_nlms_t::name_u_ [private]`

5.413.3.15 name_d_ `std::string rt_nlms_t::name_d_ [private]`

5.413.3.16 name_e_ `std::string rt_nlms_t::name_e_ [private]`

5.413.3.17 n_no_update_ `int rt_nlms_t::n_no_update_ [private]`

5.413.3.18 no_iter `int rt_nlms_t::no_iter [private]`

5.413.3.19 s_E `mha_wave_t rt_nlms_t::s_E [private]`

The documentation for this class was generated from the following file:

- **nlms_wave.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.414 save_spec_t Class Reference

Inheritance diagram for save_spec_t:

Public Member Functions

- **save_spec_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗s)
- void **prepare** (**mhaconfig_t** &tf)

Private Attributes

- std::string **basename**

Additional Inherited Members

5.414.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.414.1.1 save_spec_t()

```
save_spec_t::save_spec_t (
    algo_comm_t iac,
    const std::string & configured_name ) [inline]
```

5.414.2 Member Function Documentation

5.414.2.1 process()

```
mha_spec_t* save_spec_t::process (
    mha_spec_t * s ) [inline]
```

5.414.2.2 prepare()

```
void save_spec_t::prepare (
    mhaconfig_t & tf ) [inline], [virtual]
```

Implements **MHAPlugin::plugin_t**< **MHA_AC::spectrum_t** > (p. 1153).

5.414.3 Member Data Documentation

5.414.3.1 basename

```
std::string save_spec_t::basename [private]
```

The documentation for this class was generated from the following file:

- **save_spec.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.415 save_wave_t Class Reference

Inheritance diagram for save_wave_t:

Public Member Functions

- **save_wave_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s)
- void **prepare** (**mhaconfig_t** &tf)

Private Attributes

- std::string **basename**

Additional Inherited Members

5.415.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.415.1.1 save_wave_t()

```
save_wave_t::save_wave_t (
    algo_comm_t iac,
    const std::string & configured_name ) [inline]
```

5.415.2 Member Function Documentation

5.415.2.1 process()

```
mha_wave_t* save_wave_t::process (
    mha_wave_t * s ) [inline]
```

5.415.2.2 prepare()

```
void save_wave_t::prepare (
    mhaconfig_t & tf ) [inline], [virtual]
```

Implements **MHAPlugin::plugin_t**< **MHA_AC::waveform_t** > (p. 1153).

5.415.3 Member Data Documentation

5.415.3.1 basename

```
std::string save_wave_t::basename [private]
```

The documentation for this class was generated from the following file:

- **save_wave.cpp**

5.416 shadowfilter_begin::cfg_t Class Reference

Public Member Functions

- **cfg_t** (int nfft, int inch, int outch, **algo_comm_t** ac, std::string name)
- **mha_spec_t** * **process** (**mha_spec_t** *)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHA_AC::spectrum_t in_spec_copy**
- **MHASignal::spectrum_t out_spec**
- **MHA_AC::int_t nch**
- **MHA_AC::int_t ntracks**

5.416.1 Constructor & Destructor Documentation

5.416.1.1 cfg_t()

```
cfg_t::cfg_t (
    int nfft,
    int inch,
    int outch,
    algo_comm_t ac,
    std::string name )
```

5.416.2 Member Function Documentation

5.416.2.1 process()

```
mha_spec_t * cfg_t::process (
    mha_spec_t * s )
```

5.416.3 Member Data Documentation

5.416.3.1 in_spec_copy

MHA_AC::spectrum_t shadowfilter_begin::cfg_t::in_spec_copy [private]

5.416.3.2 out_spec

MHASignal::spectrum_t shadowfilter_begin::cfg_t::out_spec [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.416.3.3 nch

`MHA_AC::int_t shadowfilter_begin::cfg_t::nch [private]`

5.416.3.4 ntracks

`MHA_AC::int_t shadowfilter_begin::cfg_t::ntracks [private]`

The documentation for this class was generated from the following file:

- **shadowfilter_begin.cpp**

5.417 shadowfilter_begin::shadowfilter_begin_t Class Reference

Inheritance diagram for shadowfilter_begin::shadowfilter_begin_t:

Public Member Functions

- **shadowfilter_begin_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- void **prepare** (**mhaconfig_t** &)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- std::string **basename**
- **MHAParser::int_t nch**
- **MHAParser::int_t ntracks**

Additional Inherited Members

5.417.1 Constructor & Destructor Documentation

5.417.1.1 shadowfilter_begin_t()

```
shadowfilter_begin::shadowfilter_begin_t::shadowfilter↩
_begin_t (
            algo_comm_t iac,
           const std::string & configured_name )
```

5.417.2 Member Function Documentation

5.417.2.1 process()

```
mha_spec_t * shadowfilter_begin::shadowfilter_begin_t::process
(
            mha_spec_t * s )
```

5.417.2.2 prepare()

```
void shadowfilter_begin::shadowfilter_begin_t::prepare (
            mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< cfg_t >** (p. 1153).

5.417.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.417.3.1 basename `std::string shadowfilter_begin::shadowfilter_begin_t::basename [private]`

5.417.3.2 nch `MHAParser::int_t shadowfilter_begin::shadowfilter_begin_t::nch [private]`

5.417.3.3 ntracks `MHAParser::int_t shadowfilter_begin::shadowfilter_begin_t↩ ::ntracks [private]`

The documentation for this class was generated from the following file:

- **shadowfilter_begin.cpp**

5.418 shadowfilter_end::cfg_t Class Reference

Public Member Functions

- **cfg_t** (int nfft_, **algo_comm_t** ac_, std::string name_)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

Private Attributes

- **algo_comm_t ac**
- std::string **name**
- int **nfft**
- int **ntracks**
- int **nch_out**
- **mha_spec_t in_spec**
- **MHASignal::spectrum_t out_spec**
- **MHA_AC::spectrum_t gains**

5.418.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.418.1.1 cfg_t()

```
cfg_t::cfg_t (
    int nfft_,
    algo_comm_t ac_,
    std::string name_ )
```

5.418.2 Member Function Documentation

5.418.2.1 process()

```
mha_spec_t * cfg_t::process (
    mha_spec_t * s )
```

5.418.3 Member Data Documentation

5.418.3.1 ac

```
algo_comm_t shadowfilter_end::cfg_t::ac [private]
```

5.418.3.2 name

```
std::string shadowfilter_end::cfg_t::name [private]
```

5.418.3.3 nfft

```
int shadowfilter_end::cfg_t::nfft [private]
```

5.418.3.4 ntracks

```
int shadowfilter_end::cfg_t::ntracks [private]
```

5.418.3.5 nch_out

```
int shadowfilter_end::cfg_t::nch_out [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.418.3.6 in_spec `mha_spec_t shadowfilter_end::cfg_t::in_spec [private]`

5.418.3.7 out_spec `MHASignal::spectrum_t shadowfilter_end::cfg_t::out_spec [private]`

5.418.3.8 gains `MHA_AC::spectrum_t shadowfilter_end::cfg_t::gains [private]`

The documentation for this class was generated from the following file:

- **shadowfilter_end.cpp**

5.419 shadowfilter_end::shadowfilter_end_t Class Reference

Inheritance diagram for shadowfilter_end::shadowfilter_end_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **shadowfilter_end_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
- void **prepare** (**mhaconfig_t** &)

Private Attributes

- **MHAParser::string_t basename**

Additional Inherited Members

5.419.1 Constructor & Destructor Documentation

5.419.1.1 shadowfilter_end_t()

```
shadowfilter_end::shadowfilter_end_t::shadowfilter↩
_end_t (
            algo_comm_t iac,
            const std::string & configured_name )
```

5.419.2 Member Function Documentation

5.419.2.1 process()

```
mha_spec_t * shadowfilter_end::shadowfilter_end_t::process (
            mha_spec_t * s )
```

5.419.2.2 prepare()

```
void shadowfilter_end::shadowfilter_end_t::prepare (
            mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t**< **cfg_t** > (p. 1153).

5.419.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.419.3.1 basename `MHAParser::string_t` `shadowfilter_end::shadowfilter_end_t↩ ::basename [private]`

The documentation for this class was generated from the following file:

- **shadowfilter_end.cpp**

5.420 sine_cfg_t Struct Reference

Runtime configuration of the sine plugin.

Public Member Functions

- **sine_cfg_t** (double sampling_rate, **mha_real_t** frequency, **mha_real_t** newlev, int _↩ mix, const std::vector< int > &_channels)

  Constructor computes data members from input parameters.

Public Attributes

- double **phase_increment_div_2pi**

  Phase increment per sample, divided by 2 pi for easier phase wrapping.
- double **amplitude**

  Amplitude of the sinusoid in Pascal.
- int **mix**

  0 for mode replace, 1 for mode mix. Used as factor on input signal.
- const std::vector< int > **channels**

  Indices of affected audio channels.

5.420.1 Detailed Description

Runtime configuration of the sine plugin.

5.420.2 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.420.2.1 sine_cfg_t()

```
sine_cfg_t::sine_cfg_t (
    double sampling_rate,
     mha_real_t frequency,
     mha_real_t newlev,
    int _mix,
    const std::vector< int > & _channels ) [inline]
```

Constructor computes data members from input parameters.

5.420.3 Member Data Documentation

5.420.3.1 phase_increment_div_2pi

```
double sine_cfg_t::phase_increment_div_2pi
```

Phase increment per sample, divided by 2 pi for easier phase wrapping.

5.420.3.2 amplitude

```
double sine_cfg_t::amplitude
```

Amplitude of the sinusoid in Pascal.

5.420.3.3 mix

```
int sine_cfg_t::mix
```

0 for mode replace, 1 for mode mix. Used as factor on input signal.

5.420.3.4 channels

```
const std::vector<int> sine_cfg_t::channels
```

Indices of affected audio channels.

The documentation for this struct was generated from the following file:

- **sine.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.421 sine_t Class Reference

Interface class of plugin `sine`, a sinusoid generator plugin.

Inheritance diagram for sine_t:

Public Member Functions

- **sine_t** (**algo_comm_t** iac, const std::string &configured_name)

  Constructor initializes and connects configuration variables.
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)

  Computes sinusoid and mixes/replaces input signal.
- void **prepare** (**mhaconfig_t** &)

  Adapts range of channel variable and prepares.

Private Member Functions

- void **update_cfg** ()

  Computes new runtime configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::float_t lev**
- **MHAParser::float_t frequency**
- **MHAParser::kw_t mode**
- **MHAParser::vint_t channels**
- double **phase_div_2pi**
- **MHAEvents::patchbay_t< sine_t > patchbay**

Additional Inherited Members

5.421.1 Detailed Description

Interface class of plugin `sine`, a sinusoid generator plugin.

5.421.2 Constructor & Destructor Documentation

5.421.2.1 sine_t()

```
sine_t::sine_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructor initializes and connects configuration variables.

5.421.3 Member Function Documentation

5.421.3.1 process()

```
mha_wave_t * sine_t::process (
    mha_wave_t * s )
```

Computes sinusoid and mixes/replaces input signal.

If the amplitude has changed since the last process callback, spread out the amplitude change linearly across all samples of the buffer to avoid clicks.

© 2005-2021 HörTech gGmbH, Oldenburg

5.421.3.2 prepare() `void sine_t::prepare ( mhaconfig_t & tf ) [virtual]`

Adapts range of channel variable and prepares.

Implements **MHAPlugin::plugin_t< sine_cfg_t >** (p. 1153).

5.421.3.3 update_cfg() `void sine_t::update_cfg ( ) [private]`

Computes new runtime configuration.

5.421.4 Member Data Documentation

5.421.4.1 lev `MHAParser::float_t sine_t::lev [private]`

5.421.4.2 frequency `MHAParser::float_t sine_t::frequency [private]`

5.421.4.3 mode `MHAParser::kw_t sine_t::mode [private]`

5.421.4.4 channels `MHAParser::vint_t sine_t::channels [private]`

5.421.4.5 phase_div_2pi `double sine_t::phase_div_2pi [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.421.4.6 patchbay `MHAEvents::patchbay_t< sine_t> sine_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **sine.cpp**

5.422 smooth_cepstrum::smooth_cepstrum_if_t Class Reference

Inheritance diagram for smooth_cepstrum::smooth_cepstrum_if_t:

Public Member Functions

- **smooth_cepstrum_if_t** (**algo_comm_t** iac, const std::string &configured_name)

  Constructs the beamforming plugin.
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

  This plugin implements noise reduction using spectral subtraction: by nonnegative subtraction from the output magnitude of the estimated noise magnitude spectrum.
- void **prepare** (**mhaconfig_t** &)

  Plugin preparation.
- void **release** (void)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **update_cfg** ()
- void **on_model_param_valuechanged** ()

Private Attributes

- **MHAParser::float_t xi_min_db**
- **MHAParser::float_t f0_low**
- **MHAParser::float_t f0_high**
- **MHAParser::float_t delta_pitch**
- **MHAParser::float_t lambda_thresh**
- **MHAParser::float_t alpha_pitch**
- **MHAParser::float_t beta_const**
- **MHAParser::float_t kappa_const**
- **MHAParser::float_t gain_min_db**
- **MHAParser::vfloat_t win_f0**
- **MHAParser::vfloat_t alpha_const_vals**
- **MHAParser::vfloat_t alpha_const_limits_hz**
- **MHAParser::string_t noisePow_name**
- **MHAParser::parser_t spp**
- **MHAParser::float_t prior_q**
- **MHAParser::float_t xi_opt_db**
- **MHAEvents::patchbay_t**< **smooth_cepstrum_if_t** > **patchbay**
- bool **prepared**

Additional Inherited Members

5.422.1 Constructor & Destructor Documentation

5.422.1.1 smooth_cepstrum_if_t()

```
smooth_cepstrum::smooth_cepstrum_if_t::smooth_↩
cepstrum_if_t (
            algo_comm_t iac,
            const std::string & configured_name )
```

Constructs the beamforming plugin.

5.422.2 Member Function Documentation

5.422.2.1 process()

```
mha_spec_t * smooth_cepstrum::smooth_cepstrum_if_t::process (
            mha_spec_t * signal )
```

This plugin implements noise reduction using spectral subtraction: by nonnegative subtraction from the output magnitude of the estimated noise magnitude spectrum.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal	Pointer to the input signal structure.

Returns

Returns a pointer to the input signal structure, with a the signal modified by this plugin.

5.422.2.2 prepare() `void smooth_cepstrum::smooth_cepstrum_if_t::prepare ( mhaconfig_t & signal_info ) [virtual]`

Plugin preparation.

This plugin checks that the input signal has the spectral domain and contains at least one channel

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< smooth_cepstrum_t >** (p. 1153).

5.422.2.3 release() `void smooth_cepstrum::smooth_cepstrum_if_t::release ( void ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t< smooth_cepstrum_t >** (p. 1154).

5.422.2.4 update_cfg() `void smooth_cepstrum::smooth_cepstrum_if_t::update_cfg ( ) [private]`

5.422.2.5 on_model_param_valuechanged() `void smooth_cepstrum::smooth_cepstrum↩_if_t::on_model_param_valuechanged ( ) [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.422.3 Member Data Documentation

5.422.3.1 xi_min_db

MHAParser::float_t smooth_cepstrum::smooth_cepstrum_if_t↩ ::xi_min_db [private]

5.422.3.2 f0_low

MHAParser::float_t smooth_cepstrum::smooth_cepstrum_if_t::f0_low [private]

5.422.3.3 f0_high

MHAParser::float_t smooth_cepstrum::smooth_cepstrum_if_t::f0_↩ high [private]

5.422.3.4 delta_pitch

MHAParser::float_t smooth_cepstrum::smooth_cepstrum_if_t↩ ::delta_pitch [private]

5.422.3.5 lambda_thresh

MHAParser::float_t smooth_cepstrum::smooth_cepstrum_if↩ _t::lambda_thresh [private]

5.422.3.6 alpha_pitch

MHAParser::float_t smooth_cepstrum::smooth_cepstrum_if_t↩ ::alpha_pitch [private]

5.422.3.7 beta_const

MHAParser::float_t smooth_cepstrum::smooth_cepstrum_if_t↩ ::beta_const [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.422.3.8 kappa_const

MHAParser::float_t smooth_cepstrum::smooth_cepstrum_if_t↩ ::kappa_const [private]

5.422.3.9 gain_min_db

MHAParser::float_t smooth_cepstrum::smooth_cepstrum_if_t↩ ::gain_min_db [private]

5.422.3.10 win_f0

MHAParser::vfloat_t smooth_cepstrum::smooth_cepstrum_if_t↩ ::win_f0 [private]

5.422.3.11 alpha_const_vals

MHAParser::vfloat_t smooth_cepstrum::smooth_cepstrum↩ _if_t::alpha_const_vals [private]

5.422.3.12 alpha_const_limits_hz

MHAParser::vfloat_t smooth_cepstrum::smooth_↩ cepstrum_if_t::alpha_const_limits_hz [private]

5.422.3.13 noisePow_name

MHAParser::string_t smooth_cepstrum::smooth_cepstrum↩ _if_t::noisePow_name [private]

5.422.3.14 spp

MHAParser::parser_t smooth_cepstrum::smooth_cepstrum_if_t::spp [private]

5.422.3.15 prior_q

MHAParser::float_t smooth_cepstrum::smooth_cepstrum_if_t↩ ::prior_q [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.422.3.16 xi_opt_db

`MHAParser::float_t smooth_cepstrum::smooth_cepstrum_if_t↩ ::xi_opt_db [private]`

5.422.3.17 patchbay

`MHAEvents::patchbay_t< smooth_cepstrum_if_t> smooth_cepstrum↩ ::smooth_cepstrum_if_t::patchbay [private]`

5.422.3.18 prepared

`bool smooth_cepstrum::smooth_cepstrum_if_t::prepared [private]`

The documentation for this class was generated from the following files:

- **smooth_cepstrum.hh**
- **smooth_cepstrum.cpp**

5.423 smooth_cepstrum::smooth_cepstrum_t Class Reference

Public Member Functions

- **smooth_cepstrum_t** (**algo_comm_t** & **ac**, **smooth_params** & **params**)
- **smooth_cepstrum_t** (const **smooth_cepstrum_t** &)=delete
- **smooth_cepstrum_t** & **operator=** (const **smooth_cepstrum_t** &)=delete
- ∼**smooth_cepstrum_t** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

Private Attributes

- **algo_comm_t ac**
- **smooth_params params**
- unsigned int **fftlen**
- **mha_fft_t mha_fft**
- unsigned int **nfreq**
- unsigned int **nchan**
- float **ola_powspec_scale**
- float **q_low**
- float **q_high**
- **MHASignal::waveform_t winF0**
- float **xi_min**
- float **gain_min**
- **MHASignal::waveform_t alpha_const**

© 2005-2021 HörTech gGmbH, Oldenburg

- **MHASignal::waveform_t alpha_prev**
- **MHASignal::waveform_t noisePow**
- **MHASignal::waveform_t powSpec**
- **MHASignal::waveform_t gamma_post**
- **MHASignal::waveform_t xi_ml**
- **MHASignal::spectrum_t lambda_ml_full**
- **MHASignal::spectrum_t lambda_ml_ceps**
- **MHASignal::waveform_t lambda_ml_smooth**
- **MHASignal::waveform_t alpha_hat**
- **MHASignal::waveform_t alpha_frame**
- **MHASignal::spectrum_t lambda_ceps**
- **MHASignal::waveform_t lambda_ceps_prev**
- **MHASignal::spectrum_t log_lambda_spec**
- **MHASignal::waveform_t lambda_spec**
- **MHASignal::waveform_t xi_est**
- **MHASignal::waveform_t gain_wiener**
- **MHASignal::spectrum_t spec_out**
- double * **max_val**
- int * **max_q**
- int * **pitch_set_first**
- int * **pitch_set_last**
- float **priorFact**
- float **xiOpt**
- float **logGLRFact**
- float **GLRexp**
- **MHASignal::waveform_t GLR**

5.423.1 Constructor & Destructor Documentation

5.423.1.1 smooth_cepstrum_t() [1/2]

```
smooth_cepstrum::smooth_cepstrum_t::smooth_↩
cepstrum_t (
            algo_comm_t & ac,
            smooth_params & params )
```

5.423.1.2 smooth_cepstrum_t() [2/2]

```
smooth_cepstrum::smooth_cepstrum_t::smooth_↩
cepstrum_t (
            const smooth_cepstrum_t & ) [delete]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.423.1.3 ~smooth_cepstrum_t()

`smooth_cepstrum::smooth_cepstrum_t::~smooth_cepstrum_t ( )`

5.423.2 Member Function Documentation

5.423.2.1 operator=()

`smooth_cepstrum_t& smooth_cepstrum::smooth_cepstrum_t::operator= ( const smooth_cepstrum_t & ) [delete]`

5.423.2.2 process()

`mha_spec_t * smooth_cepstrum::smooth_cepstrum_t::process ( mha_spec_t * noisyFrame )`

5.423.3 Member Data Documentation

5.423.3.1 ac

`algo_comm_t smooth_cepstrum::smooth_cepstrum_t::ac [private]`

5.423.3.2 params

`smooth_params smooth_cepstrum::smooth_cepstrum_t::params [private]`

5.423.3.3 fftlen

`unsigned int smooth_cepstrum::smooth_cepstrum_t::fftlen [private]`

5.423.3.4 mha_fft

`mha_fft_t smooth_cepstrum::smooth_cepstrum_t::mha_fft [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.423.3.5 nfreq `unsigned int smooth_cepstrum::smooth_cepstrum_t::nfreq [private]`

5.423.3.6 nchan `unsigned int smooth_cepstrum::smooth_cepstrum_t::nchan [private]`

5.423.3.7 ola_powspec_scale `float smooth_cepstrum::smooth_cepstrum_t::ola_powspec↩_scale [private]`

5.423.3.8 q_low `float smooth_cepstrum::smooth_cepstrum_t::q_low [private]`

5.423.3.9 q_high `float smooth_cepstrum::smooth_cepstrum_t::q_high [private]`

5.423.3.10 winF0 **`MHASignal::waveform_t`** `smooth_cepstrum::smooth_cepstrum_t::winF0 [private]`

5.423.3.11 xi_min `float smooth_cepstrum::smooth_cepstrum_t::xi_min [private]`

5.423.3.12 gain_min `float smooth_cepstrum::smooth_cepstrum_t::gain_min [private]`

5.423.3.13 alpha_const **`MHASignal::waveform_t`** `smooth_cepstrum::smooth_cepstrum_t↩::alpha_const [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.423.3.14 alpha_prev

MHASignal::waveform_t smooth_cepstrum::smooth_cepstrum_t↩ ::alpha_prev [private]

5.423.3.15 noisePow

MHASignal::waveform_t smooth_cepstrum::smooth_cepstrum_t↩ ::noisePow [private]

5.423.3.16 powSpec

MHASignal::waveform_t smooth_cepstrum::smooth_cepstrum_t↩ ::powSpec [private]

5.423.3.17 gamma_post

MHASignal::waveform_t smooth_cepstrum::smooth_cepstrum_↩ t::gamma_post [private]

5.423.3.18 xi_ml

MHASignal::waveform_t smooth_cepstrum::smooth_cepstrum_t::xi_ml [private]

5.423.3.19 lambda_ml_full

MHASignal::spectrum_t smooth_cepstrum::smooth_cepstrum↩ _t::lambda_ml_full [private]

5.423.3.20 lambda_ml_ceps

MHASignal::spectrum_t smooth_cepstrum::smooth_cepstrum↩ _t::lambda_ml_ceps [private]

5.423.3.21 lambda_ml_smooth

MHASignal::waveform_t smooth_cepstrum::smooth_↩ cepstrum_t::lambda_ml_smooth [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.423.3.22 alpha_hat

MHASignal::waveform_t smooth_cepstrum::smooth_cepstrum_t↩ ::alpha_hat [private]

5.423.3.23 alpha_frame

MHASignal::waveform_t smooth_cepstrum::smooth_cepstrum_t↩ ::alpha_frame [private]

5.423.3.24 lambda_ceps

MHASignal::spectrum_t smooth_cepstrum::smooth_cepstrum_↩ t::lambda_ceps [private]

5.423.3.25 lambda_ceps_prev

MHASignal::waveform_t smooth_cepstrum::smooth_↩ cepstrum_t::lambda_ceps_prev [private]

5.423.3.26 log_lambda_spec

MHASignal::spectrum_t smooth_cepstrum::smooth_↩ cepstrum_t::log_lambda_spec [private]

5.423.3.27 lambda_spec

MHASignal::waveform_t smooth_cepstrum::smooth_cepstrum_↩ t::lambda_spec [private]

5.423.3.28 xi_est

MHASignal::waveform_t smooth_cepstrum::smooth_cepstrum_t::xi_est [private]

5.423.3.29 gain_wiener

MHASignal::waveform_t smooth_cepstrum::smooth_cepstrum_t↩ ::gain_wiener [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.423.3.30 spec_out

`MHASignal::spectrum_t` `smooth_cepstrum::smooth_cepstrum_t↩::spec_out` [private]

5.423.3.31 max_val

`double* smooth_cepstrum::smooth_cepstrum_t::max_val` [private]

5.423.3.32 max_q

`int* smooth_cepstrum::smooth_cepstrum_t::max_q` [private]

5.423.3.33 pitch_set_first

`int* smooth_cepstrum::smooth_cepstrum_t::pitch_set_first` [private]

5.423.3.34 pitch_set_last

`int* smooth_cepstrum::smooth_cepstrum_t::pitch_set_last` [private]

5.423.3.35 priorFact

`float smooth_cepstrum::smooth_cepstrum_t::priorFact` [private]

5.423.3.36 xiOpt

`float smooth_cepstrum::smooth_cepstrum_t::xiOpt` [private]

5.423.3.37 logGLRFact

`float smooth_cepstrum::smooth_cepstrum_t::logGLRFact` [private]

5.423.3.38 GLRexp

`float smooth_cepstrum::smooth_cepstrum_t::GLRexp` [private]

© 2005-2021 HörTech gGmbH, Oldenburg

5.423.3.39 GLR `MHASignal::waveform_t smooth_cepstrum::smooth_cepstrum_t::GLR [private]`

The documentation for this class was generated from the following files:

- **smooth_cepstrum.hh**
- **smooth_cepstrum.cpp**

5.424 smooth_cepstrum::smooth_params Class Reference

Public Member Functions

- **smooth_params** (const **mhaconfig_t** &_in_cfg, float _xi_min_db, float _f0_low, float _f0_high, float _delta_pitch, float _lambda_thresh, float _alpha_pitch, float _beta_const, float _kappa_const, float _prior_q, float _xi_opt_db, float _gain_min_db, std::vector< float > &_winF0, std::vector< float > &_alpha_const_vals, std::vector< float > &_alpha_↩const_limits_hz, std::string &_noisePow_name)

Public Attributes

- const **mhaconfig_t in_cfg**
- float **xi_min_db**
- float **f0_low**
- float **f0_high**
- float **delta_pitch**
- float **lambda_thresh**
- float **alpha_pitch**
- float **beta_const**
- float **kappa_const**
- float **prior_q**
- float **xi_opt_db**
- float **gain_min_db**
- std::vector< float > **winF0**
- std::vector< float > **alpha_const_vals**
- std::vector< float > **alpha_const_limits_hz**
- std::string **noisePow_name**

5.424.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.424.1.1 smooth_params()

```
smooth_cepstrum::smooth_params::smooth_params (
    const mhaconfig_t & _in_cfg,
    float _xi_min_db,
    float _f0_low,
    float _f0_high,
    float _delta_pitch,
    float _lambda_thresh,
    float _alpha_pitch,
    float _beta_const,
    float _kappa_const,
    float _prior_q,
    float _xi_opt_db,
    float _gain_min_db,
    std::vector< float > & _winF0,
    std::vector< float > & _alpha_const_vals,
    std::vector< float > & _alpha_const_limits_hz,
    std::string & _noisePow_name ) [inline]
```

5.424.2 Member Data Documentation

5.424.2.1 in_cfg

```
const mhaconfig_t smooth_cepstrum::smooth_params::in_cfg
```

5.424.2.2 xi_min_db

```
float smooth_cepstrum::smooth_params::xi_min_db
```

5.424.2.3 f0_low

```
float smooth_cepstrum::smooth_params::f0_low
```

5.424.2.4 f0_high

```
float smooth_cepstrum::smooth_params::f0_high
```

5.424.2.5 delta_pitch

```
float smooth_cepstrum::smooth_params::delta_pitch
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.424.2.6 lambda_thresh `float smooth_cepstrum::smooth_params::lambda_thresh`

5.424.2.7 alpha_pitch `float smooth_cepstrum::smooth_params::alpha_pitch`

5.424.2.8 beta_const `float smooth_cepstrum::smooth_params::beta_const`

5.424.2.9 kappa_const `float smooth_cepstrum::smooth_params::kappa_const`

5.424.2.10 prior_q `float smooth_cepstrum::smooth_params::prior_q`

5.424.2.11 xi_opt_db `float smooth_cepstrum::smooth_params::xi_opt_db`

5.424.2.12 gain_min_db `float smooth_cepstrum::smooth_params::gain_min_db`

5.424.2.13 winF0 `std::vector<float> smooth_cepstrum::smooth_params::winF0`

5.424.2.14 alpha_const_vals `std::vector<float> smooth_cepstrum::smooth_params↩ ::alpha_const_vals`

© 2005-2021 HörTech gGmbH, Oldenburg

5.424.2.15 alpha_const_limits_hz `std::vector<float> smooth_cepstrum::smooth_params::alpha_const_limits_hz`

5.424.2.16 noisePow_name `std::string smooth_cepstrum::smooth_params::noisePow_name`

The documentation for this class was generated from the following file:

- **smooth_cepstrum.hh**

5.425 smoothgains_bridge::overlapadd_if_t Class Reference

Inheritance diagram for smoothgains_bridge::overlapadd_if_t:

Public Member Functions

- **overlapadd_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- ~**overlapadd_if_t** ()
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- **mha_spec_t** * **process** (**mha_spec_t** *)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **update** ()

Private Attributes

- **MHAEvents::patchbay_t**< **overlapadd_if_t** > **patchbay**
- **MHAParser::kw_t mode**
- **MHAParser::window_t irswnd**
- **MHAParser::float_t epsilon**
- **MHAParser::mhapluginloader_t plugloader**
- std::string **algo**
- **mhaconfig_t cf_in**
- **mhaconfig_t cf_out**

Additional Inherited Members

5.425.1 Constructor & Destructor Documentation

5.425.1.1 overlapadd_if_t()

```
smoothgains_bridge::overlapadd_if_t::overlapadd_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.425.1.2 ~overlapadd_if_t()

```
smoothgains_bridge::overlapadd_if_t::~overlapadd_if_t
( )
```

5.425.2 Member Function Documentation

5.425.2.1 prepare()

```
void smoothgains_bridge::overlapadd_if_t::prepare (
    mhaconfig_t & t ) [virtual]
```

Implements **MHAPlugin::plugin_t**< **smoothspec_wrap_t** > (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.425.2.2 release() `void smoothgains_bridge::overlapadd_if_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t< smoothspec_wrap_t >** (p. 1154).

5.425.2.3 process() `mha_spec_t * smoothgains_bridge::overlapadd_if_t::process ( mha_spec_t * spec )`

5.425.2.4 update() `void smoothgains_bridge::overlapadd_if_t::update ( ) [private]`

5.425.3 Member Data Documentation

5.425.3.1 patchbay `MHAEvents::patchbay_t< overlapadd_if_t> smoothgains_bridge↩ ::overlapadd_if_t::patchbay [private]`

5.425.3.2 mode `MHAParser::kw_t smoothgains_bridge::overlapadd_if_t::mode [private]`

5.425.3.3 irswnd `MHAParser::window_t smoothgains_bridge::overlapadd_if_t::irswnd [private]`

5.425.3.4 epsilon `MHAParser::float_t smoothgains_bridge::overlapadd_if_t::epsilon [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.425.3.5 plugloader `MHAParser::mhapluginloader_t smoothgains_bridge::overlapadd↩_if_t::plugloader [private]`

5.425.3.6 algo `std::string smoothgains_bridge::overlapadd_if_t::algo [private]`

5.425.3.7 cf_in `mhaconfig_t smoothgains_bridge::overlapadd_if_t::cf_in [private]`

5.425.3.8 cf_out `mhaconfig_t smoothgains_bridge::overlapadd_if_t::cf_out [private]`

The documentation for this class was generated from the following file:

- **smoothgains_bridge.cpp**

5.426 smoothgains_bridge::smoothspec_wrap_t Class Reference

Public Member Functions

- **smoothspec_wrap_t** (**mhaconfig_t** spar_in, **mhaconfig_t** spar_out, const **MH↩AParser::kw_t** &mode, const **MHAParser::window_t** &irswnd, const **MHAParser↩::float_t** &epsilon)
- **mha_spec_t** ∗ **proc_1** (**mha_spec_t** ∗)
- **mha_spec_t** ∗ **proc_2** (**mha_spec_t** ∗)

Private Attributes

- **MHASignal::spectrum_t spec_in_copy**
  Copy of input spectrum for smoothspec.
- **MHAFilter::smoothspec_t smoothspec**
  Smoothspec calculator.
- bool **use_smoothspec**
- float **smoothspec_epsilon**

© 2005-2021 HörTech gGmbH, Oldenburg

5.426.1 Constructor & Destructor Documentation

5.426.1.1 smoothspec_wrap_t()

```
smoothgains_bridge::smoothspec_wrap_t::smoothspec↩
_wrap_t (
            mhaconfig_t spar_in,
            mhaconfig_t spar_out,
           const  MHAParser::kw_t & mode,
           const  MHAParser::window_t & irswnd,
           const  MHAParser::float_t & epsilon )
```

5.426.2 Member Function Documentation

5.426.2.1 proc_1()

```
mha_spec_t * smoothgains_bridge::smoothspec_wrap_t::proc_1 (
            mha_spec_t * s )
```

5.426.2.2 proc_2()

```
mha_spec_t * smoothgains_bridge::smoothspec_wrap_t::proc_2 (
            mha_spec_t * s )
```

5.426.3 Member Data Documentation

5.426.3.1 spec_in_copy

```
MHASignal::spectrum_t smoothgains_bridge::smoothspec_↩
wrap_t::spec_in_copy [private]
```

Copy of input spectrum for smoothspec.

5.426.3.2 smoothspec

```
MHAFilter::smoothspec_t smoothgains_bridge::smoothspec_↩
wrap_t::smoothspec [private]
```

Smoothspec calculator.

© 2005-2021 HörTech gGmbH, Oldenburg

5.426.3.3 use_smoothspec `bool smoothgains_bridge::smoothspec_wrap_t::use_smoothspec [private]`

5.426.3.4 smoothspec_epsilon `float smoothgains_bridge::smoothspec_wrap_t::smoothspec↩_epsilon [private]`

The documentation for this class was generated from the following file:

- **smoothgains_bridge.cpp**

5.427 softclip_t Class Reference

Inheritance diagram for softclip_t:

Public Member Functions

- **softclip_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **update** ()

Private Attributes

- **mhaconfig_t tftype**
- **MHAParser::float_t attack**
- **MHAParser::float_t decay**
- **MHAParser::float_t start_limit**
- **MHAParser::float_t slope_db**
- **MHAEvents::patchbay_t< softclip_t > patchbay**

Additional Inherited Members

5.427.1 Constructor & Destructor Documentation

5.427.1.1 softclip_t()

```
softclip_t::softclip_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.427.2 Member Function Documentation

5.427.2.1 process()

```
mha_wave_t * softclip_t::process (
    mha_wave_t * s )
```

5.427.2.2 prepare()

```
void softclip_t::prepare (
    mhaconfig_t & tf ) [virtual]
```

Implements **MHAPlugin::plugin_t< cfg_t >** (p. 1153).

5.427.2.3 update()

```
void softclip_t::update ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.427.3 Member Data Documentation

5.427.3.1 tftype `mhaconfig_t softclip_t::tftype [private]`

5.427.3.2 attack `MHAParser::float_t softclip_t::attack [private]`

5.427.3.3 decay `MHAParser::float_t softclip_t::decay [private]`

5.427.3.4 start_limit `MHAParser::float_t softclip_t::start_limit [private]`

5.427.3.5 slope_db `MHAParser::float_t softclip_t::slope_db [private]`

5.427.3.6 patchbay `MHAEvents::patchbay_t< softclip_t> softclip_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **softclip.cpp**

5.428 softclipper_t Class Reference

Soft clipper signal processing implementation.

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **softclipper_t** (const **softclipper_variables_t** &v, const **mhaconfig_t** &tf)
  Constructor, copies information from parameters and initializes state.
- **mha_real_t process** (**mha_wave_t** ∗s)
  Process one block of audio signal.

Private Attributes

- **MHAFilter::o1flt_lowpass_t attack**
  Attack filter.
- **MHAFilter::o1flt_maxtrack_t decay**
  Decay filter.
- **MHAFilter::o1flt_lowpass_t clipmeter**
  Clipping ratio filter.
- **mha_real_t threshold**
  Compression onset value.
- **mha_real_t hardlimit**
  Maximum output amplitude of softclipper.
- **mha_real_t slope**
  Compression slope.
- bool **linear**
  Is compression done on linear or log scale.

5.428.1 Detailed Description

Soft clipper signal processing implementation.

5.428.2 Constructor & Destructor Documentation

5.428.2.1 softclipper_t()

```
softclipper_t::softclipper_t (
    const softclipper_variables_t & v,
    const mhaconfig_t & tf )
```

Constructor, copies information from parameters and initializes state.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

v	Configuration variables of the softclipper
tf	Signal dimensions

5.428.3 Member Function Documentation

5.428.3.1 process() `mha_real_t softclipper_t::process ( mha_wave_t * s )`

Process one block of audio signal.

Parameters

in,out	*s*	Input signal which is modified in-place

5.428.4 Member Data Documentation

5.428.4.1 attack `MHAFilter::o1flt_lowpass_t softclipper_t::attack [private]`

Attack filter.

5.428.4.2 decay `MHAFilter::o1flt_maxtrack_t softclipper_t::decay [private]`

Decay filter.

5.428.4.3 clipmeter `MHAFilter::o1flt_lowpass_t softclipper_t::clipmeter [private]`

Clipping ratio filter.

© 2005-2021 HörTech gGmbH, Oldenburg

5.428.4.4 threshold `mha_real_t softclipper_t::threshold [private]`

Compression onset value.

5.428.4.5 hardlimit `mha_real_t softclipper_t::hardlimit [private]`

Maximum output amplitude of softclipper.

5.428.4.6 slope `mha_real_t softclipper_t::slope [private]`

Compression slope.

5.428.4.7 linear `bool softclipper_t::linear [private]`

Is compression done on linear or log scale.

The documentation for this class was generated from the following file:

- **transducers.cpp**

5.429 softclipper_variables_t Class Reference

Parser aggregate of all configuration variables for the output soft clipper.

Inheritance diagram for softclipper_variables_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **softclipper_variables_t** ()
  Constructor, initializes all variables and inserts them into ∗this.

Public Attributes

- **MHAParser::float_t tau_attack**
- **MHAParser::float_t tau_decay**
- **MHAParser::float_t tau_clip**
- **MHAParser::float_t threshold**
- **MHAParser::float_t hardlimit**
- **MHAParser::float_t slope**
- **MHAParser::bool_t linear**
- **MHAParser::float_mon_t clipped**
- **MHAParser::float_t max_clipped**

Additional Inherited Members

5.429.1 Detailed Description

Parser aggregate of all configuration variables for the output soft clipper.

5.429.2 Constructor & Destructor Documentation

5.429.2.1 softclipper_variables_t() `softclipper_variables_t::softclipper_variables_t ( )`

Constructor, initializes all variables and inserts them into ∗this.

5.429.3 Member Data Documentation

5.429.3.1 tau_attack `MHAParser::float_t softclipper_variables_t::tau_attack`

© 2005-2021 HörTech gGmbH, Oldenburg

5.429.3.2 tau_decay `MHAParser::float_t softclipper_variables_t::tau_decay`

5.429.3.3 tau_clip `MHAParser::float_t softclipper_variables_t::tau_clip`

5.429.3.4 threshold `MHAParser::float_t softclipper_variables_t::threshold`

5.429.3.5 hardlimit `MHAParser::float_t softclipper_variables_t::hardlimit`

5.429.3.6 slope `MHAParser::float_t softclipper_variables_t::slope`

5.429.3.7 linear `MHAParser::bool_t softclipper_variables_t::linear`

5.429.3.8 clipped `MHAParser::float_mon_t softclipper_variables_t::clipped`

5.429.3.9 max_clipped `MHAParser::float_t softclipper_variables_t::max_clipped`

The documentation for this class was generated from the following file:

- **transducers.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.430 spec2wave_if_t Class Reference

Inheritance diagram for spec2wave_if_t:

Public Member Functions

- **spec2wave_if_t** (**algo_comm_t** iac, const std::string &configured_name)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()
- **mha_wave_t** ∗ **process** (**mha_spec_t** ∗)

Private Member Functions

- void **update** ()
- void **setlock** (bool b)

Private Attributes

- **MHAParser::float_t ramplen**
- **windowselector_t window_config**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.430.1 Constructor & Destructor Documentation

5.430.1.1 spec2wave_if_t()

```
spec2wave_if_t::spec2wave_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.430.2 Member Function Documentation

5.430.2.1 prepare()

```
void spec2wave_if_t::prepare (
    mhaconfig_t & t ) [virtual]
```

Implements **MHAPlugin::plugin_t< spec2wave_t >** (p. 1153).

5.430.2.2 release()

```
void spec2wave_if_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< spec2wave_t >** (p. 1154).

5.430.2.3 process()

```
mha_wave_t * spec2wave_if_t::process (
    mha_spec_t * spec_in )
```

5.430.2.4 update()

```
void spec2wave_if_t::update ( ) [private]
```

5.430.2.5 setlock()

```
void spec2wave_if_t::setlock (
    bool b ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.430.3 Member Data Documentation

5.430.3.1 ramplen `MHAParser::float_t spec2wave_if_t::ramplen [private]`

5.430.3.2 window_config `windowselector_t spec2wave_if_t::window_config [private]`

The documentation for this class was generated from the following file:

- **spec2wave.cpp**

5.431 spec2wave_t Class Reference

Public Member Functions

- **spec2wave_t** (unsigned int nfft_, unsigned int nwnd_, unsigned int nwndshift_, unsigned int nch, **mha_real_t** ramplen, const **MHAWindow::base_t** &postwin)
- ∼**spec2wave_t** ()
- **mha_wave_t** ∗ **process** (**mha_spec_t** ∗)

Private Attributes

- **mha_fft_t ft**
  FFT class.
- unsigned int **npad1**
  length of zero padding before window
- unsigned int **npad2**
  length of zero padding after window
- **hanning_ramps_t ramps**
- **MHASignal::waveform_t calc_out**
- **MHASignal::waveform_t out_buf**
- **MHASignal::waveform_t write_buf**
- **mha_real_t sc**
- unsigned int **nfft**
- unsigned int **nwndshift**
- **MHAWindow::base_t postwindow**

© 2005-2021 HörTech gGmbH, Oldenburg

5.431.1 Constructor & Destructor Documentation

5.431.1.1 spec2wave_t()

```
spec2wave_t::spec2wave_t (
    unsigned int nfft_,
    unsigned int nwnd_,
    unsigned int nwndshift_,
    unsigned int nch,
    mha_real_t ramplen,
    const MHAWindow::base_t & postwin )
```

5.431.1.2 ~spec2wave_t()

```
spec2wave_t::~spec2wave_t ( )
```

5.431.2 Member Function Documentation

5.431.2.1 process()

```
mha_wave_t * spec2wave_t::process (
    mha_spec_t * spec_in )
```

5.431.3 Member Data Documentation

5.431.3.1 ft

```
mha_fft_t spec2wave_t::ft [private]
```

FFT class.

5.431.3.2 npad1

```
unsigned int spec2wave_t::npad1 [private]
```

length of zero padding before window

© 2005-2021 HörTech gGmbH, Oldenburg

5.431.3.3 npad2 `unsigned int spec2wave_t::npad2 [private]`

length of zero padding after window

5.431.3.4 ramps `hanning_ramps_t spec2wave_t::ramps [private]`

5.431.3.5 calc_out `MHASignal::waveform_t spec2wave_t::calc_out [private]`

5.431.3.6 out_buf `MHASignal::waveform_t spec2wave_t::out_buf [private]`

5.431.3.7 write_buf `MHASignal::waveform_t spec2wave_t::write_buf [private]`

5.431.3.8 sc `mha_real_t spec2wave_t::sc [private]`

5.431.3.9 nfft `unsigned int spec2wave_t::nfft [private]`

5.431.3.10 nwndshift `unsigned int spec2wave_t::nwndshift [private]`

5.431.3.11 postwindow `MHAWindow::base_t spec2wave_t::postwindow [private]`

The documentation for this class was generated from the following file:

- **spec2wave.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.432 spec_fader_t Class Reference

Public Member Functions

- **spec_fader_t** (unsigned int ch, **mha_real_t fr**, **MHAParser::vfloat_t** &ng, **MHA↩ Parser::float_t** &t)
- ∼**spec_fader_t** ()

Public Attributes

- unsigned int **nch**
- **mha_real_t** ∗ **gains**
- unsigned int **fr**

5.432.1 Constructor & Destructor Documentation

5.432.1.1 spec_fader_t()

```
spec_fader_t::spec_fader_t (
    unsigned int ch,
    mha_real_t fr,
    MHAParser::vfloat_t & ng,
    MHAParser::float_t & t )
```

5.432.1.2 ∼spec_fader_t() `spec_fader_t::∼spec_fader_t ( ) [inline]`

5.432.2 Member Data Documentation

5.432.2.1 nch `unsigned int spec_fader_t::nch`

5.432.2.2 gains `mha_real_t* spec_fader_t::gains`

© 2005-2021 HörTech gGmbH, Oldenburg

5.432.2.3 fr `unsigned int spec_fader_t::fr`

The documentation for this class was generated from the following file:

- **fader_spec.cpp**

5.433 speechnoise_t Class Reference

Inheritance diagram for speechnoise_t:

Public Types

- enum **noise_type_t** {
  mha, **olnoise**, **LTASS_combined**, **LTASS_female**,
  LTASS_male, **white**, **pink**, **brown**,
  TEN_SPL, **TEN_SPL_250_8k**, **TEN_SPL_50_16k**, **sin125**,
  sin250, **sin500**, **sin1k**, **sin2k**,
  sin4k, **sin8k** }

Public Member Functions

- **speechnoise_t** (float duration, float srate, unsigned int **channels**, **speechnoise_t↩ ::noise_type_t** noise_type= **speechnoise_t::mha**)
- **speechnoise_t** (unsigned int length_samples, float srate, unsigned int **channels**, **speechnoise_t::noise_type_t** noise_type= **speechnoise_t::mha**)

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **creator** (**speechnoise_t::noise_type_t** noise_type, float srate)

Additional Inherited Members

5.433.1 Member Enumeration Documentation

5.433.1.1 noise_type_t `enum` **`speechnoise_t::noise_type_t`**

Enumerator

mha	
olnoise	
LTASS_combined	
LTASS_female	
LTASS_male	
white	
pink	
brown	
TEN_SPL	
TEN_SPL_250_8k	
TEN_SPL_50_16k	
sin125	
sin250	
sin500	
sin1k	
sin2k	
sin4k	
sin8k	

5.433.2 Constructor & Destructor Documentation

5.433.2.1 speechnoise_t() [1/2] `speechnoise_t::speechnoise_t (`

`float` *`duration,`*

© 2005-2021 HörTech gGmbH, Oldenburg

```
float srate,
unsigned int channels,
 speechnoise_t::noise_type_t noise_type =  speechnoise_t::mha )
```

5.433.2.2 speechnoise_t() [2/2]

```
speechnoise_t::speechnoise_t (
unsigned int length_samples,
float srate,
unsigned int channels,
 speechnoise_t::noise_type_t noise_type =  speechnoise_t::mha )
```

5.433.3 Member Function Documentation

5.433.3.1 creator()

```
void speechnoise_t::creator (
 speechnoise_t::noise_type_t noise_type,
float srate ) [private]
```

The documentation for this class was generated from the following files:

- **speechnoise.h**
- **speechnoise.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.434 steerbf Class Reference

Inheritance diagram for steerbf:

Public Member Functions

- **steerbf** (**algo_comm_t** iac, const std::string &configured_name)
  Constructs our plugin.
- ∼**steerbf** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)
  Defers to configuration class.
- void **prepare** (**mhaconfig_t** &)
  Plugin preparation.
- void **release** (void)

Public Attributes

- **MHAParser::string_t bf_src**
- **parser_int_dyn angle_ind**
- **MHAParser::string_t angle_src**

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **update_cfg** ()

Private Attributes

- **MHAEvents::patchbay_t< steerbf > patchbay**

Additional Inherited Members

5.434.1 Constructor & Destructor Documentation

5.434.1.1 steerbf()

```
steerbf::steerbf (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructs our plugin.

5.434.1.2 ~steerbf()

```
steerbf::~steerbf ( )
```

5.434.2 Member Function Documentation

5.434.2.1 process()

```
mha_spec_t * steerbf::process (
    mha_spec_t * signal )
```

Defers to configuration class.

5.434.2.2 prepare()

```
void steerbf::prepare (
    mhaconfig_t & signal_info ) [virtual]
```

Plugin preparation.

An opportunity to validate configuration parameters before instantiating a configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	Structure containing a description of the form of the signal (domain, number of channels, frames per block, sampling rate.

Implements **MHAPlugin::plugin_t< steerbf_config >** (p. 1153).

5.434.2.3 release() `void steerbf::release ( void ) [inline], [virtual]`

Reimplemented from **MHAPlugin::plugin_t< steerbf_config >** (p. 1154).

5.434.2.4 update_cfg() `void steerbf::update_cfg ( ) [private]`

5.434.3 Member Data Documentation

5.434.3.1 bf_src `MHAParser::string_t steerbf::bf_src`

5.434.3.2 angle_ind `parser_int_dyn steerbf::angle_ind`

5.434.3.3 angle_src `MHAParser::string_t steerbf::angle_src`

5.434.3.4 patchbay `MHAEvents::patchbay_t< steerbf> steerbf::patchbay [private]`

The documentation for this class was generated from the following files:

- **steerbf.h**
- **steerbf.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.435 steerbf_config Class Reference

Public Member Functions

- **steerbf_config** (**algo_comm_t** & **ac**, const **mhaconfig_t** in_cfg, **steerbf** ∗ **steerbf**)
- ∼**steerbf_config** ()
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗)

Private Attributes

- unsigned int **nchan**
- unsigned int **nfreq**
- **MHASignal::spectrum_t outSpec**
- **mha_spec_t bf_vec**
- unsigned int **nangle**
- **steerbf** ∗ **_steerbf**
- **algo_comm_t** & **ac**
- std::string **bf_src_copy**

5.435.1 Constructor & Destructor Documentation

5.435.1.1 steerbf_config()

```
steerbf_config::steerbf_config (
    algo_comm_t & ac,
  const  mhaconfig_t in_cfg,
    steerbf * steerbf )
```

5.435.1.2 ∼steerbf_config()

```
steerbf_config::~steerbf_config ( )
```

5.435.2 Member Function Documentation

5.435.2.1 process()

```
mha_spec_t * steerbf_config::process (
    mha_spec_t * inSpec )
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.435.3 Member Data Documentation

5.435.3.1 nchan `unsigned int steerbf_config::nchan [private]`

5.435.3.2 nfreq `unsigned int steerbf_config::nfreq [private]`

5.435.3.3 outSpec `MHASignal::spectrum_t steerbf_config::outSpec [private]`

5.435.3.4 bf_vec `mha_spec_t steerbf_config::bf_vec [private]`

5.435.3.5 nangle `unsigned int steerbf_config::nangle [private]`

5.435.3.6 _steerbf `steerbf* steerbf_config::_steerbf [private]`

5.435.3.7 ac `algo_comm_t& steerbf_config::ac [private]`

5.435.3.8 bf_src_copy `std::string steerbf_config::bf_src_copy [private]`

The documentation for this class was generated from the following files:

- **steerbf.h**
- **steerbf.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.436 testplugin::ac_parser_t Class Reference

Inheritance diagram for testplugin::ac_parser_t:

Public Types

- enum **data_type_t** {
  _MHA_AC_CHAR, **_MHA_AC_INT**, **_MHA_AC_MHAREAL**, **_MHA_AC_FLOAT**, **_MHA_AC_DOUBLE**, **_MHA_AC_MHACOMPLEX**, **_unknown** }

Public Member Functions

- **ac_parser_t** ()
- void **do_insert_var** ()
  Insert variable into AC space.
- void **do_get_var** ()

Public Attributes

- **MHAParser::string_t insert_var**
- **MHAParser::string_t get_var**
- **MHAParser::kw_t data_type**
- **MHAParser::int_t num_entries**
- **MHAParser::int_t stride**
- **MHAParser::string_t char_data**
- **MHAParser::vint_t int_data**
- **MHAParser::vfloat_t float_data**
- **MHAParser::vcomplex_t complex_data**
- **MHAEvents::patchbay_t**< **ac_parser_t** > **patchbay**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.436.1 Member Enumeration Documentation

5.436.1.1 data_type_t `enum` **`testplugin::ac_parser_t::data_type_t`**

Enumerator

_MHA_AC_CHAR	
_MHA_AC_INT	
_MHA_AC_MHAREAL	
_MHA_AC_FLOAT	
_MHA_AC_DOUBLE	
_MHA_AC_MHACOMPLEX	
_unknown	

5.436.2 Constructor & Destructor Documentation

5.436.2.1 ac_parser_t() `testplugin::ac_parser_t::ac_parser_t ( ) [inline]`

5.436.3 Member Function Documentation

5.436.3.1 do_insert_var() `void testplugin::ac_parser_t::do_insert_var ( ) [inline]`

Insert variable into AC space.

This leaks memory by design, as the plugin is for testing only

5.436.3.2 do_get_var() `void testplugin::ac_parser_t::do_get_var ( ) [inline]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.436.4 Member Data Documentation

5.436.4.1 insert_var `MHAParser::string_t` `testplugin::ac_parser_t::insert_var`

5.436.4.2 get_var `MHAParser::string_t` `testplugin::ac_parser_t::get_var`

5.436.4.3 data_type `MHAParser::kw_t` `testplugin::ac_parser_t::data_type`

5.436.4.4 num_entries `MHAParser::int_t` `testplugin::ac_parser_t::num_entries`

5.436.4.5 stride `MHAParser::int_t` `testplugin::ac_parser_t::stride`

5.436.4.6 char_data `MHAParser::string_t` `testplugin::ac_parser_t::char_data`

5.436.4.7 int_data `MHAParser::vint_t` `testplugin::ac_parser_t::int_data`

5.436.4.8 float_data `MHAParser::vfloat_t` `testplugin::ac_parser_t::float_data`

© 2005-2021 HörTech gGmbH, Oldenburg

5.436.4.9 complex_data `MHAParser::vcomplex_t testplugin::ac_parser_t::complex_data`

5.436.4.10 patchbay `MHAEvents::patchbay_t< ac_parser_t> testplugin::ac_parser_t::patchbay`

The documentation for this class was generated from the following file:

- **testplugin.cpp**

5.437 testplugin::config_parser_t Class Reference

Inheritance diagram for testplugin::config_parser_t:

Public Member Functions

- void **setlock** (const bool &b)
- **config_parser_t** ()
- **mhaconfig_t get** () const
- void **set** (**mhaconfig_t** c)

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **MHAParser::int_t channels**
- **MHAParser::kw_t domain**
- **MHAParser::int_t fragsize**
- **MHAParser::int_t wndlen**
- **MHAParser::int_t fftlen**
- **MHAParser::float_t srate**

Additional Inherited Members

5.437.1 Constructor & Destructor Documentation

5.437.1.1 config_parser_t() `testplugin::config_parser_t::config_parser_t ( ) [inline]`

5.437.2 Member Function Documentation

5.437.2.1 setlock() `void testplugin::config_parser_t::setlock ( const bool & b ) [inline]`

5.437.2.2 get() `mhaconfig_t testplugin::config_parser_t::get ( ) const [inline]`

5.437.2.3 set() `void testplugin::config_parser_t::set ( mhaconfig_t c ) [inline]`

5.437.3 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.437.3.1 channels `MHAParser::int_t testplugin::config_parser_t::channels`

5.437.3.2 domain `MHAParser::kw_t testplugin::config_parser_t::domain`

5.437.3.3 fragsize `MHAParser::int_t testplugin::config_parser_t::fragsize`

5.437.3.4 wndlen `MHAParser::int_t testplugin::config_parser_t::wndlen`

5.437.3.5 fftlen `MHAParser::int_t testplugin::config_parser_t::fftlen`

5.437.3.6 srate `MHAParser::float_t testplugin::config_parser_t::srate`

The documentation for this class was generated from the following file:

- **testplugin.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.438 testplugin::if_t Class Reference

Inheritance diagram for testplugin::if_t:

Public Member Functions

- **if_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗s_in)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗s_in)
- void **prepare** (**mhaconfig_t** &)

Private Member Functions

- void **test_prepare** ()
- void **test_process** ()

Private Attributes

- **config_parser_t config_in**
- **config_parser_t config_out**
- **ac_parser_t ac**
- **signal_parser_t signal**
- **MHAParser::bool_t _prepare**
- **MHAEvents::patchbay_t**< **if_t** > **patchbay**
- **MHAParser::mhapluginloader_t plug**

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.438.1 Constructor & Destructor Documentation

5.438.1.1 if_t()

```
testplugin::if_t::if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.438.2 Member Function Documentation

5.438.2.1 process() [1/2]

```
mha_spec_t* testplugin::if_t::process (
    mha_spec_t * s_in ) [inline]
```

5.438.2.2 process() [2/2]

```
mha_wave_t* testplugin::if_t::process (
    mha_wave_t * s_in ) [inline]
```

5.438.2.3 prepare()

```
void testplugin::if_t::prepare (
    mhaconfig_t & ) [inline], [virtual]
```

Implements **MHAPlugin::plugin_t< int >** (p. 1153).

5.438.2.4 test_prepare()

```
void testplugin::if_t::test_prepare ( ) [private]
```

5.438.2.5 test_process()

```
void testplugin::if_t::test_process ( ) [private]
```

© 2005-2021 HörTech gGmbH, Oldenburg

5.438.3 Member Data Documentation

5.438.3.1 config_in

`config_parser_t testplugin::if_t::config_in [private]`

5.438.3.2 config_out

`config_parser_t testplugin::if_t::config_out [private]`

5.438.3.3 ac

`ac_parser_t testplugin::if_t::ac [private]`

5.438.3.4 signal

`signal_parser_t testplugin::if_t::signal [private]`

5.438.3.5 _prepare

`MHAParser::bool_t testplugin::if_t::_prepare [private]`

5.438.3.6 patchbay

`MHAEvents::patchbay_t< if_t> testplugin::if_t::patchbay [private]`

5.438.3.7 plug

`MHAParser::mhapluginloader_t testplugin::if_t::plug [private]`

The documentation for this class was generated from the following file:

- **testplugin.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.439 testplugin::signal_parser_t Class Reference

Inheritance diagram for testplugin::signal_parser_t:

Public Member Functions

- **signal_parser_t** ()

Public Attributes

- **MHAParser::mfloat_t input_wave**
- **MHAParser::mcomplex_t input_spec**
- **MHAParser::mfloat_mon_t output_wave**
- **MHAParser::mcomplex_mon_t output_spec**

Additional Inherited Members

5.439.1 Constructor & Destructor Documentation

5.439.1.1 signal_parser_t() `testplugin::signal_parser_t::signal_parser_t ( ) [inline]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.439.2 Member Data Documentation

5.439.2.1 input_wave

MHAParser::mfloat_t testplugin::signal_parser_t::input_wave

5.439.2.2 input_spec

MHAParser::mcomplex_t testplugin::signal_parser_t::input_↩spec

5.439.2.3 output_wave

MHAParser::mfloat_mon_t testplugin::signal_parser_t::output↩_wave

5.439.2.4 output_spec

MHAParser::mcomplex_mon_t testplugin::signal_parser_t↩::output_spec

The documentation for this class was generated from the following file:

- **testplugin.cpp**

5.440 us_t Class Reference

Inheritance diagram for us_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **us_t** (**algo_comm_t** iac, const std::string &configured_name)
- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &)
- void **release** ()

Private Attributes

- **MHAParser::int_t ratio**
- **MHAFilter::iir_filter_t antialias**

Additional Inherited Members

5.440.1 Constructor & Destructor Documentation

5.440.1.1 us_t()

```
us_t::us_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.440.2 Member Function Documentation

5.440.2.1 process()

```
mha_wave_t * us_t::process (
    mha_wave_t * s )
```

5.440.2.2 prepare()

```
void us_t::prepare (
    mhaconfig_t & cf ) [virtual]
```

Implements **MHAPlugin::plugin_t**< **MHASignal::waveform_t** > (p. 1153).

© 2005-2021 HörTech gGmbH, Oldenburg

5.440.2.3 release() `void us_t::release ( ) [virtual]`

Reimplemented from **MHAPlugin::plugin_t**< **MHASignal::waveform_t** > (p. 1154).

5.440.3 Member Data Documentation

5.440.3.1 ratio `MHAParser::int_t us_t::ratio [private]`

5.440.3.2 antialias `MHAFilter::iir_filter_t us_t::antialias [private]`

The documentation for this class was generated from the following file:

- **upsample.cpp**

5.441 wave2spec_if_t Class Reference

Plugin wave2spec interface class, uses **wave2spec_t** (p. 1497) as runtime configuration.

Inheritance diagram for wave2spec_if_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **wave2spec_if_t** (**algo_comm_t** iac, const std::string &configured_name)
  Constructor of wave2spec plugin, sets up configuration variables and callbacks.
- void **prepare** (**mhaconfig_t** &t)
  prepare for signal processing
- void **release** ()
  Unprepare signal processing.
- void **process** (**mha_wave_t** ∗wave_in, **mha_spec_t** ∗∗sout)
  processing callback used for domain transformation
- void **process** (**mha_wave_t** ∗wave_in, **mha_wave_t** ∗∗sout)
  processing callback used if output of original waveform is requested.

Private Member Functions

- void **update** ()
  Create a new runtime configuration from configuration parameters when the plugin is prepared, or when the window position or other window parameters change.
- void **setlock** (bool b)
  Lock/Unlock all configuration variables.

Private Attributes

- **MHAParser::int_t nfft**
  FFT length selector.
- **MHAParser::int_t nwnd**
  Window length selector.
- **MHAParser::float_t wndpos**
  Window position selector.
- **windowselector_t window_config**
- **MHAParser::bool_t strict_window_ratio**
  Switch to disallow window sizes that are not a multiple of the fragsize a by power of two.
- **MHAParser::bool_t return_wave**
  Switch to select return domain.
- std::string **algo**
  configured name this plugin, used to name the AC variables
- **MHAParser::vfloat_mon_t zeropadding**

Additional Inherited Members

5.441.1 Detailed Description

Plugin wave2spec interface class, uses **wave2spec_t** (p. 1497) as runtime configuration.

© 2005-2021 HörTech gGmbH, Oldenburg

5.441.2 Constructor & Destructor Documentation

5.441.2.1 wave2spec_if_t()

```
wave2spec_if_t::wave2spec_if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructor of wave2spec plugin, sets up configuration variables and callbacks.

Parameters

iac	algorithm communication storage accessor
ialg	configured name of this plugin, used to name the AC variables published by wave2spec

5.441.3 Member Function Documentation

5.441.3.1 prepare()

```
void wave2spec_if_t::prepare (
    mhaconfig_t & t ) [virtual]
```

prepare for signal processing

Parameters

in,out	*t*	signal dimenstions, modified by prepare as determined by the STFT configuration

Implements **MHAPlugin::plugin_t< wave2spec_t >** (p. 1153).

5.441.3.2 release()

```
void wave2spec_if_t::release ( ) [virtual]
```

Unprepare signal processing.

Reimplemented from **MHAPlugin::plugin_t< wave2spec_t >** (p. 1154).

© 2005-2021 HörTech gGmbH, Oldenburg

5.441.3.3 process() [1/2]

```
void wave2spec_if_t::process (
    mha_wave_t * wave_in,
    mha_spec_t ** sout )
```

processing callback used for domain transformation

Parameters

wave_in	latest block of audio signal (hop size samples per channel)
sout	output spectrum pointer

5.441.3.4 process() [2/2]

```
void wave2spec_if_t::process (
    mha_wave_t * wave_in,
    mha_wave_t ** sout )
```

processing callback used if output of original waveform is requested.

The STFT spectrum is computed and can only be accessed by downstream plugins through the AC variable published by this plugin.

Parameters

wave_in	latest block of audio signal (hop size samples per channel)
sout	output waveform pointer (FFT length samples per channel)

5.441.3.5 update()

```
void wave2spec_if_t::update ( ) [private]
```

Create a new runtime configuration from configuration parameters when the plugin is prepared, or when the window position or other window parameters change.

Exceptions

MHA_Error *(p. 764)*	if the configuration change is not compatible with the current input and FFT length constraints.

© 2005-2021 HörTech gGmbH, Oldenburg

5.441.3.6 setlock() `void wave2spec_if_t::setlock ( bool b ) [private]`

Lock/Unlock all configuration variables.

Parameters

b	Desired lock state

5.441.4 Member Data Documentation

5.441.4.1 nfft `MHAParser::int_t wave2spec_if_t::nfft [private]`

FFT length selector.

5.441.4.2 nwnd `MHAParser::int_t wave2spec_if_t::nwnd [private]`

Window length selector.

5.441.4.3 wndpos `MHAParser::float_t wave2spec_if_t::wndpos [private]`

Window position selector.

5.441.4.4 window_config `windowselector_t wave2spec_if_t::window_config [private]`

5.441.4.5 strict_window_ratio `MHAParser::bool_t wave2spec_if_t::strict_window_↩ ratio [private]`

Switch to disallow window sizes that are not a multiple of the fragsize a by power of two.

© 2005-2021 HörTech gGmbH, Oldenburg

5.441.4.6 return_wave `MHAParser::bool_t wave2spec_if_t::return_wave [private]`

Switch to select return domain.

5.441.4.7 algo `std::string wave2spec_if_t::algo [private]`

configured name this plugin, used to name the AC variables

5.441.4.8 zeropadding `MHAParser::vfloat_mon_t wave2spec_if_t::zeropadding [private]`

The documentation for this class was generated from the following files:

- **wave2spec.hh**
- **wave2spec.cpp**

5.442 wave2spec_t Class Reference

Runtime configuration class for plugin wave2spec.

Inheritance diagram for wave2spec_t:

© 2005-2021 HörTech gGmbH, Oldenburg

Public Member Functions

- **wave2spec_t** (unsigned int nfft, unsigned int nwnd_, unsigned int nwndshift_, unsigned int nch, **mha_real_t** wndpos, const **MHAWindow::base_t** & **window**, **algo_comm_t ac**, std::string algo)

  Constructor computes window and zeropadding, allocates storage and FFT plan.
- void **publish_ac_variables** ()

  Insert two AC variables into AC space:
- **mha_spec_t** ∗ **process** (**mha_wave_t** ∗wave_in)

  Perform signal shift, windowing, zero-padding and FFT.
- ∼**wave2spec_t** ()

  Destructor removes AC variables from AC space and deallocates memory.
- unsigned **get_zeropadding** (bool after) const

  Getter method to read zeropadding computed for the STFT configuration parameters.

Private Member Functions

- void **calc_pre_wnd** (**MHASignal::waveform_t** &dest, const **MHASignal::waveform↩_t** &src)

  Applies analysis window weights to current input signal and writes windowed signal to correct place in FFT buffer.

Private Attributes

- unsigned int **nwnd**

  window length
- unsigned int **nwndshift**

  window shift or hop size
- **mha_fft_t ft**

  FFT instance used for transformation.
- unsigned int **npad1**

  length of zero padding before window
- unsigned int **npad2**

  length of zero padding after window
- **MHAWindow::base_t window**

  Analysis window.
- **MHASignal::waveform_t calc_in**

  waveform buffer with FFT length samples per channel
- **MHASignal::waveform_t in_buf**

  waveform buffer with window length samples per channel
- **MHASignal::spectrum_t spec_in**

  spectrum buffer containing only the positive frequency bins
- std::string **ac_wndshape_name**

  name of window shape AC variable

© 2005-2021 HörTech gGmbH, Oldenburg

Additional Inherited Members

5.442.1 Detailed Description

Runtime configuration class for plugin wave2spec.

Manages window shift, windowing, zero-padding, and FFT. Inserts current window shape and current STFT spectrum into AC space.

5.442.2 Constructor & Destructor Documentation

5.442.2.1 wave2spec_t()

```
wave2spec_t::wave2spec_t (
    unsigned int nfft,
    unsigned int nwnd_,
    unsigned int nwndshift_,
    unsigned int nch,
    mha_real_t wndpos,
    const MHAWindow::base_t & window,
    algo_comm_t ac,
    std::string algo )
```

Constructor computes window and zeropadding, allocates storage and FFT plan.

Parameters

nfft	FFT length
*nwnd_*	window length in samples
*nwndshift_*	window shift (hop size) in samples
nch	number of audio channels
wndpos	for cases nfft > nwnd_, where to place the window inside the FFT buffer: 0 = at start, 1 = at end, 0.5 = centered. Position is rounded to full samples and determines zero-padding
window	Analysis window shape
ac	algorithm communication storage accessor
algo	configured name of this plugin, used to name the AC variables published by wave2spec

© 2005-2021 HörTech gGmbH, Oldenburg

5.442.2.2 ~wave2spec_t() `wave2spec_t::~wave2spec_t ( )`

Destructor removes AC variables from AC space and deallocates memory.

5.442.3 Member Function Documentation

5.442.3.1 publish_ac_variables() `void wave2spec_t::publish_ac_variables ( )`

Insert two AC variables into AC space:

- <configured_name>: Contains the current STFT spectrum.
- <configured_name>_wnd: Contains the window shape as individual weights.

5.442.3.2 process() `mha_spec_t * wave2spec_t::process ( mha_wave_t * wave_in )`

Perform signal shift, windowing, zero-padding and FFT.

Parameters

wave_in	latest block of audio signal (hop size samples per channel)

Returns

pointer to current STFT spectrum. Storage is managed by this object. Downstream plug-ins may modify the signal in place.

5.442.3.3 get_zeropadding() `unsigned wave2spec_t::get_zeropadding ( bool after ) const [inline]`

Getter method to read zeropadding computed for the STFT configuration parameters.

Result is only valid after prepare() has been called.

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

Computed zeropadding before or after the analysis window in number of samples.

Parameters

after	When false, return length of zeropadding before the analysis window. When true, return length of zeropadding in samples after the analysis window.

5.442.3.4 calc_pre_wnd()

```
void wave2spec_t::calc_pre_wnd (
    MHASignal::waveform_t & dest,
    const MHASignal::waveform_t & src ) [private]
```

Applies analysis window weights to current input signal and writes windowed signal to correct place in FFT buffer.

Ensures zero-padding regions contain only zeros. To be invoked before applying FFT.

Parameters

out	*dest*	waveform buffer with FFT length audio samples, completely overwritten by this method
in	*src*	waveform buffer with window length audio samples, these samples are written to dest after window shape has been applied to the individual samples.

5.442.4 Member Data Documentation

5.442.4.1 nwnd

```
unsigned int wave2spec_t::nwnd [private]
```

window length

5.442.4.2 nwndshift

```
unsigned int wave2spec_t::nwndshift [private]
```

window shift or hop size

© 2005-2021 HörTech gGmbH, Oldenburg

5.442.4.3 ft `mha_fft_t` `wave2spec_t::ft` [private]

FFT instance used for transformation.

5.442.4.4 npad1 `unsigned int wave2spec_t::npad1` [private]

length of zero padding before window

5.442.4.5 npad2 `unsigned int wave2spec_t::npad2` [private]

length of zero padding after window

5.442.4.6 window `MHAWindow::base_t` `wave2spec_t::window` [private]

Analysis window.

5.442.4.7 calc_in `MHASignal::waveform_t` `wave2spec_t::calc_in` [private]

waveform buffer with FFT length samples per channel

5.442.4.8 in_buf `MHASignal::waveform_t` `wave2spec_t::in_buf` [private]

waveform buffer with window length samples per channel

5.442.4.9 spec_in `MHASignal::spectrum_t` `wave2spec_t::spec_in` [private]

spectrum buffer containing only the positive frequency bins

© 2005-2021 HörTech gGmbH, Oldenburg

5.442.4.10 ac_wndshape_name `std::string wave2spec_t::ac_wndshape_name [private]`

name of window shape AC variable

The documentation for this class was generated from the following files:

- **wave2spec.hh**
- **wave2spec.cpp**

5.443 wavrec_t Class Reference

Inheritance diagram for wavrec_t:

Public Member Functions

- **mha_wave_t** ∗ **process** (**mha_wave_t** ∗)
- void **prepare** (**mhaconfig_t** &cf)
- void **release** ()
- **wavrec_t** (**algo_comm_t** iac, const std::string &configured_name)

Private Member Functions

- void **start_new_session** ()

© 2005-2021 HörTech gGmbH, Oldenburg

Private Attributes

- **MHAParser::bool_t record**
- **MHAParser::int_t fifolen**
- **MHAParser::int_t minwrite**
- **MHAParser::string_t prefix**
- **MHAParser::bool_t use_date**
- **MHAParser::kw_t output_sample_format**
- **MHAEvents::patchbay_t< wavrec_t > patchbay**

Additional Inherited Members

5.443.1 Constructor & Destructor Documentation

5.443.1.1 wavrec_t()

```
wavrec_t::wavrec_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

5.443.2 Member Function Documentation

5.443.2.1 process()

```
mha_wave_t * wavrec_t::process (
    mha_wave_t * s )
```

5.443.2.2 prepare()

```
void wavrec_t::prepare (
    mhaconfig_t & cf ) [virtual]
```

Implements **MHAPlugin::plugin_t< wavwriter_t >** (p. 1153).

5.443.2.3 release()

```
void wavrec_t::release ( ) [virtual]
```

Reimplemented from **MHAPlugin::plugin_t< wavwriter_t >** (p. 1154).

© 2005-2021 HörTech gGmbH, Oldenburg

5.443.2.4 start_new_session() `void wavrec_t::start_new_session ( ) [private]`

5.443.3 Member Data Documentation

5.443.3.1 record `MHAParser::bool_t wavrec_t::record [private]`

5.443.3.2 fifolen `MHAParser::int_t wavrec_t::fifolen [private]`

5.443.3.3 minwrite `MHAParser::int_t wavrec_t::minwrite [private]`

5.443.3.4 prefix `MHAParser::string_t wavrec_t::prefix [private]`

5.443.3.5 use_date `MHAParser::bool_t wavrec_t::use_date [private]`

5.443.3.6 output_sample_format `MHAParser::kw_t wavrec_t::output_sample_format [private]`

5.443.3.7 patchbay `MHAEvents::patchbay_t< wavrec_t> wavrec_t::patchbay [private]`

The documentation for this class was generated from the following file:

- **wavrec.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.444 wavwriter_t Class Reference

Public Member Functions

- **wavwriter_t** (bool active, const **mhaconfig_t** &cf, unsigned int fifosize, unsigned int minwrite, const std::string &prefix, bool use_date, const std::string &format_name_)
- ∼**wavwriter_t** ()
- void **process** (**mha_wave_t** ∗)
- void **exit_request** ()

Private Member Functions

- void **write_thread** ()
- void **create_soundfile** (const std::string &prefix, bool use_date)
- void **set_format** (SF_INFO &sf_info)

  Converts the format_name string to the corresponding int according to libsndfile and writes it into the format field of sf_info throws if no format of this name is available.

Static Private Member Functions

- static void ∗ **write_thread** (void ∗this_)

Private Attributes

- std::atomic< bool > **close_session**
- bool **act_**
- **mhaconfig_t** **cf_**
- SNDFILE ∗ **sf**
- **mha_fifo_lf_t**< **mha_real_t** > **fifo**
- unsigned int **minw_**
- pthread_t **writethread**
- float ∗ **data**
- std::string **format_name**

5.444.1 Constructor & Destructor Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.444.1.1 wavwriter_t()

```
wavwriter_t::wavwriter_t (
    bool active,
    const mhaconfig_t & cf,
    unsigned int fifosize,
    unsigned int minwrite,
    const std::string & prefix,
    bool use_date,
    const std::string & format_name_ )
```

5.444.1.2 ~wavwriter_t()

```
wavwriter_t::~wavwriter_t ( )
```

5.444.2 Member Function Documentation

5.444.2.1 process()

```
void wavwriter_t::process (
    mha_wave_t * s )
```

5.444.2.2 exit_request()

```
void wavwriter_t::exit_request ( )
```

5.444.2.3 write_thread() [1/2]

```
static void* wavwriter_t::write_thread (
    void * this_ ) [inline], [static], [private]
```

5.444.2.4 write_thread() [2/2]

```
void wavwriter_t::write_thread ( ) [private]
```

5.444.2.5 create_soundfile()

```
void wavwriter_t::create_soundfile (
    const std::string & prefix,
    bool use_date ) [private]
```

5.444.2.6 set_format()

```
void wavwriter_t::set_format (
    SF_INFO & sf_info ) [private]
```

Converts the format_name string to the corresponding int according to libsndfile and writes it into the format field of sf_info throws if no format of this name is available.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

sf_info	Destination sf_info struct for the format

Exceptions

MHA_Error *(p. 764)*	If no sample format of name format_name is offered by libsndfile

5.444.3 Member Data Documentation

5.444.3.1 close_session `std::atomic<bool> wavwriter_t::close_session [private]`

5.444.3.2 act_ `bool wavwriter_t::act_ [private]`

5.444.3.3 cf_ `mhaconfig_t wavwriter_t::cf_ [private]`

5.444.3.4 sf `SNDFILE* wavwriter_t::sf [private]`

5.444.3.5 fifo `mha_fifo_lf_t< mha_real_t> wavwriter_t::fifo [private]`

5.444.3.6 minw_ `unsigned int wavwriter_t::minw_ [private]`

© 2005-2021 HörTech gGmbH, Oldenburg

5.444.3.7 writethread `pthread_t wavwriter_t::writethread [private]`

5.444.3.8 data `float* wavwriter_t::data [private]`

5.444.3.9 format_name `std::string wavwriter_t::format_name [private]`

The documentation for this class was generated from the following file:

- **wavrec.cpp**

5.445 windnoise::cfg_t Class Reference

Runtime config class for windnoise plugin.

Public Member Functions

- **cfg_t** (const **mhaconfig_t** &signal_info, bool **UseChannel_LF_attenuation**, float tau↩_Lowpass, float LowPassCutOffFrequency, float LowPassFraction_dB, float LowPass↩WindGain_dB)

  constructor translates configuration variables to runtime config
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗signal, std::vector< int > &detected, std↩::vector< float > &lowpass_quotient)

  Detect windnoise.
- void **update_PSD_Lowpass** (const **mha_spec_t** ∗signal)

  Low-pass filters the power spectrum.
- void **threshold_compare** (std::vector< int > &detected, std::vector< float > &lowpass↩_quotient)

  Wind noise detection by comparing low-frequency intensity with broadband intensity.
- int **remapping** (const std::vector< float > &lowpass_quotient)
- void **compensation** (**mha_spec_t** ∗signal, int best_signal_channel_index)

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- bool **UseChannel_LF_attenuation** = false

  FIXME: documentation for UseChannel_LF_attenuation.
- float **alpha_Lowpass** = 0

  Filter coefficient for low-pass filtering each bin in the power spectrum with a first-order recursive low-pass filter.
- unsigned **FrequencyBinLowPass** = 0

  Only smoothed power spectrum bins < FrequencyBinLowPass are added to the low-pass intensity.
- float **LowPassFraction** = 1

  The wind noise detection threshold: We have wind noise if lowFreqIntensity / broadBand↩ Intensity > LowPassFraction.
- float **LowPassWindGain** = 1

  FIXME: documentation for LowPassWindGain.
- **MHASignal::waveform_t PSD_Lowpass**

  The smoothed-over-time power spectrum.
- **MHASignal::waveform_t powspec**

  Temporary storage for the power spectrum of the current input spectrum.

5.445.1 Detailed Description

Runtime config class for windnoise plugin.

Computes power spectra of incoming STFT spectra, smoothes the power spectrum over time by low-pass filtering the intensities of each bin over time, then detects wind noise presence by comparing intensity at low frequency bins to broadband intensity.

5.445.2 Constructor & Destructor Documentation

5.445.2.1 cfg_t()

```
cfg_t::cfg_t (
    const mhaconfig_t & signal_info,
    bool UseChannel_LF_attenuation,
    float tau_Lowpass,
    float LowPassCutOffFrequency,
    float LowPassFraction_dB,
    float LowPassWindGain_dB )
```

constructor translates configuration variables to runtime config

© 2005-2021 HörTech gGmbH, Oldenburg

5.445.3 Member Function Documentation

5.445.3.1 process()

```
mha_spec_t * cfg_t::process (
    mha_spec_t * signal,
    std::vector< int > & detected,
    std::vector< float > & lowpass_quotient )
```

Detect windnoise.

FIXME: cancel it. The process method calls update_PSD_Lowpass and threshold_compare to do its work.

Parameters

in,out	*signal*	The current STFT spectrum.
out	*detected*	This Method changes the elements of the vector but not its size. Each element is set to 1 or 0, depending on windnoise being detected in the corresponding audio channel.
out	*lowpass_quotient*	This Method changes elements of the vector but not its size. Each element is set to the ratio between intensity of the signal, at low frequencies and overall intensity, in the corresponding audio channel.

Exceptions

MHA_Error *(p. 764)*	if windnoise_indicators.size() != signal.num_channels.

5.445.3.2 update_PSD_Lowpass()

```
void cfg_t::update_PSD_Lowpass (
    const mha_spec_t * signal )
```

Low-pass filters the power spectrum.

5.445.3.3 threshold_compare()

```
void cfg_t::threshold_compare (
    std::vector< int > & detected,
    std::vector< float > & lowpass_quotient )
```

Wind noise detection by comparing low-frequency intensity with broadband intensity.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

out	*detected*	This Method changes the elements of the vector but not its size. Each element is set to 1 or 0, depending on windnoise being detected in the corresponding audio channel.
out	*lowpass_quotient*	This Method changes elements of the vector but not its size. Each element is set to the ratio between intensity of the signal, at low frequencies and overall intensity, in the corresponding audio channel.

5.445.3.4 remapping()

```
int cfg_t::remapping (
    const std::vector< float > & lowpass_quotient )
```

5.445.3.5 compensation()

```
void cfg_t::compensation (
    mha_spec_t * signal,
    int best_signal_channel_index )
```

5.445.4 Member Data Documentation

5.445.4.1 UseChannel_LF_attenuation

```
bool windnoise::cfg_t::UseChannel_LF_attenuation
= false
```

FIXME: documentation for UseChannel_LF_attenuation.

5.445.4.2 alpha_Lowpass

```
float windnoise::cfg_t::alpha_Lowpass = 0
```

Filter coefficient for low-pass filtering each bin in the power spectrum with a first-order recursive low-pass filter.

© 2005-2021 HörTech gGmbH, Oldenburg

5.445.4.3 FrequencyBinLowPass `unsigned windnoise::cfg_t::FrequencyBinLowPass = 0`

Only smoothed power spectrum bins $<$ FrequencyBinLowPass are added to the low-pass intensity.

5.445.4.4 LowPassFraction `float windnoise::cfg_t::LowPassFraction = 1`

The wind noise detection threshold: We have wind noise if lowFreqIntensity / broadBand↩ Intensity $>$ LowPassFraction.

5.445.4.5 LowPassWindGain `float windnoise::cfg_t::LowPassWindGain = 1`

FIXME: documentation for LowPassWindGain.

5.445.4.6 PSD_Lowpass `MHASignal::waveform_t windnoise::cfg_t::PSD_Lowpass`

The smoothed-over-time power spectrum.

5.445.4.7 powspec `MHASignal::waveform_t windnoise::cfg_t::powspec`

Temporary storage for the power spectrum of the current input spectrum.

Only needed to hold the newest squared magnitudes until they are filtered into PSD_Lowpass.

The documentation for this class was generated from the following files:

- **windnoise.hh**
- **windnoise.cpp**

© 2005-2021 HörTech gGmbH, Oldenburg

5.446 windnoise::if_t Class Reference

interface class for windnoise plugin

Inheritance diagram for windnoise::if_t:

Public Member Functions

- **if_t** (**algo_comm_t** iac, const std::string &configured_name)

  Constructor instantiates one windnoise plugin.
- void **prepare** (**mhaconfig_t** &signal_info) override

  Prepare windnoise plugin for signal processing.
- void **release** (void) override

  Nothing needs to be deallocated on release.
- **mha_spec_t** ∗ **process** (**mha_spec_t** ∗signal)

  *signal processing, delegates to **cfg_t::process*** (p. 1511)
- void **update** (void)

  update runtime config when configuration parameters have changed
- void **insert** ()

  inserts the windnoise detection vector into AC space

© 2005-2021 HörTech gGmbH, Oldenburg

Public Attributes

- **MHAParser::bool_t UseChannel_LF_attenuation**
- **MHAParser::float_t tau_Lowpass**
- **MHAParser::float_t LowPassCutOffFrequency**
- **MHAParser::float_t LowPassFraction**
- **MHAParser::float_t LowPassWindGain**
- **MHAParser::kw_t WindNoiseDetector**
- **MHAParser::vint_mon_t detected**
- **MHAParser::vfloat_mon_t lowpass_quotient**
- const std::string **detected_acname**

  Name of AC variable mirroring the configuration monitor variable "detector".
- const std::string **lowpass_quotient_acname**

  Name of AC variable mirroring the configuration monitor variable "lowpass_quotient".

Private Attributes

- **MHAEvents::patchbay_t< if_t > patchbay**

  The Event connector.

Additional Inherited Members

5.446.1 Detailed Description

interface class for windnoise plugin

5.446.2 Constructor & Destructor Documentation

5.446.2.1 if_t()

```
windnoise::if_t::if_t (
    algo_comm_t iac,
    const std::string & configured_name )
```

Constructor instantiates one windnoise plugin.

5.446.3 Member Function Documentation

5.446.3.1 prepare()

```
void windnoise::if_t::prepare (
    mhaconfig_t & signal_info ) [override], [virtual]
```

Prepare windnoise plugin for signal processing.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

signal_info	signal dimensions, not changed by this plugin

Implements **MHAPlugin::plugin_t**< **cfg_t** > (p. 1153).

5.446.3.2 release()

```
void windnoise::if_t::release ( void ) [inline], [override], [virtual]
```

Nothing needs to be deallocated on release.

Reimplemented from **MHAPlugin::plugin_t**< **cfg_t** > (p. 1154).

5.446.3.3 process()

```
mha_spec_t * windnoise::if_t::process ( mha_spec_t * signal )
```

signal processing, delegates to **cfg_t::process** (p. 1511)

5.446.3.4 update()

```
void windnoise::if_t::update ( void )
```

update runtime config when configuration parameters have changed

5.446.3.5 insert()

```
void windnoise::if_t::insert ( ) [inline]
```

inserts the windnoise detection vector into AC space

5.446.4 Member Data Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

5.446.4.1 patchbay

`MHAEvents::patchbay_t< if_t> windnoise::if_t::patchbay [private]`

The Event connector.

5.446.4.2 UseChannel_LF_attenuation

`MHAParser::bool_t windnoise::if_t::UseChannel_LF_attenuation`

5.446.4.3 tau_Lowpass

`MHAParser::float_t windnoise::if_t::tau_Lowpass`

5.446.4.4 LowPassCutOffFrequency

`MHAParser::float_t windnoise::if_t::LowPassCutOffFrequency`

5.446.4.5 LowPassFraction

`MHAParser::float_t windnoise::if_t::LowPassFraction`

5.446.4.6 LowPassWindGain

`MHAParser::float_t windnoise::if_t::LowPassWindGain`

5.446.4.7 WindNoiseDetector

`MHAParser::kw_t windnoise::if_t::WindNoiseDetector`

5.446.4.8 detected

`MHAParser::vint_mon_t windnoise::if_t::detected`

5.446.4.9 lowpass_quotient

`MHAParser::vfloat_mon_t windnoise::if_t::lowpass_quotient`

© 2005-2021 HörTech gGmbH, Oldenburg

5.446.4.10 detected_acname `const std::string windnoise::if_t::detected_acname`

Name of AC variable mirroring the configuration monitor variable "detector".

Usually "windnoise_detected".

5.446.4.11 lowpass_quotient_acname `const std::string windnoise::if_t::lowpass_quotient_acname`

Name of AC variable mirroring the configuration monitor variable "lowpass_quotient".

Usually "windnoise_lowpass_quotient".

The documentation for this class was generated from the following files:

- **windnoise.hh**
- **windnoise.cpp**

5.447 windowselector_t Class Reference

A combination of mha parser variables to describe an overalapadd analysis window.

Public Member Functions

- **windowselector_t** (const std::string &default_type)
  constructor creates the mha parser variables that describe an overlapadd analysis window.
- ∼**windowselector_t** ()
  destructor frees window data that were allocated
- const **MHAWindow::base_t** & **get_window_data** (unsigned length)
  re-computes the window if required.
- void **insert_items** (**MHAParser::parser_t** ∗p)
  insert the window parameters "wndtype", "wndexp", and "userwnd" as mha configuration parameters into the given mha configuration parser.
- void **setlock** (bool b_)
  Lock/Unlock variables.

Public Attributes

- **MHAEvents::emitter_t updated**
  A collector event that fires when any of the window parameters managed here is written to.

© 2005-2021 HörTech gGmbH, Oldenburg

Private Member Functions

- void **invalidate_window_data** ()
  invalidates any allocated window samples.
- void **update_parser** ()
  invoked when a parser parameter changes.

Private Attributes

- **MHAWindow::base_t** $*$ **wnd**
  *Storage for the window data returned by **get_window_data()*** (p. 1520)
- **MHAParser::kw_t wndtype**
  parser variable for window type
- **MHAParser::float_t wndexp**
  parser variable for window exponent
- **MHAParser::vfloat_t userwnd**
  parser variable for user window samples to use
- **MHAEvents::patchbay_t**$<$ **windowselector_t** $>$ **patchbay**
  patchbay to watch for changes for the parser variables

5.447.1 Detailed Description

A combination of mha parser variables to describe an overalapadd analysis window.

Provides a method to get the window samples as an instance of **MHAWindow::base_t** (p. 1291) when needed.

5.447.2 Constructor & Destructor Documentation

5.447.2.1 windowselector_t() `windowselector_t::windowselector_t ( const std::string & default_type )`

constructor creates the mha parser variables that describe an overlapadd analysis window.

Parameters

default_type	name of the default analysis window type. Must be one of: "rect", "bartlett", "hanning", "hamming", "blackman"

© 2005-2021 HörTech gGmbH, Oldenburg

5.447.2.2 ~windowselector_t() `windowselector_t::~windowselector_t ( )`

destructor frees window data that were allocated

5.447.3 Member Function Documentation

5.447.3.1 get_window_data() `const MHAWindow::base_t & windowselector_t::get_window_data ( unsigned length )`

re-computes the window if required.

Parameters

length	the desired window length in samples return the window's samples as a constref to **MHAWindow::base_t** (p. 1291) instance. The referenced instance lives until the window parameters are changed, or this **windowselector_t** (p. 1518) instance is destroyed.

5.447.3.2 insert_items() `void windowselector_t::insert_items ( MHAParser::parser_t * p )`

insert the window parameters "wndtype", "wndexp", and "userwnd" as mha configuration parameters into the given mha configuration parser.

Parameters

p	The configuration parser where to insert the window parameters. E.g. the plugin wave2spec's interface class.

5.447.3.3 setlock() `void windowselector_t::setlock ( bool b_ )`

© 2005-2021 HörTech gGmbH, Oldenburg

Lock/Unlock variables.

Parameters

b↩ _↩	Desired lock state

5.447.3.4 invalidate_window_data() `void windowselector_t::invalidate_window_data ( ) [private]`

invalidates any allocated window samples.

5.447.3.5 update_parser() `void windowselector_t::update_parser ( ) [private]`

invoked when a parser parameter changes.

Calls **invalidate_window_data()** (p. 1521) and emits the updated event.

5.447.4 Member Data Documentation

5.447.4.1 updated `MHAEvents::emitter_t windowselector_t::updated`

A collector event that fires when any of the window parameters managed here is written to.

5.447.4.2 wnd `MHAWindow::base_t* windowselector_t::wnd [private]`

Storage for the window data returned by **get_window_data()** (p. 1520)

© 2005-2021 HörTech gGmbH, Oldenburg

5.447.4.3 wndtype `MHAParser::kw_t windowselector_t::wndtype [private]`

parser variable for window type

5.447.4.4 wndexp `MHAParser::float_t windowselector_t::wndexp [private]`

parser variable for window exponent

5.447.4.5 userwnd `MHAParser::vfloat_t windowselector_t::userwnd [private]`

parser variable for user window samples to use

5.447.4.6 patchbay `MHAEvents::patchbay_t< windowselector_t> windowselector_t↩ ::patchbay [private]`

patchbay to watch for changes for the parser variables

The documentation for this class was generated from the following files:

- **windowselector.h**
- **windowselector.cpp**

6 File Documentation

6.1 ac2lsl.cpp File Reference

Classes

- struct **ac2lsl::type_info**
- class **ac2lsl::save_var_base_t**
  Interface for ac to lsl bridge variable.
- class **ac2lsl::save_var_t< T >**
  Implementation for all ac to lsl bridges except complex types.
- class **ac2lsl::save_var_t< mha_complex_t >**
  *Template specialization of the **ac2lsl** (p. 78) bridge to take care of complex numbers.*
- class **ac2lsl::cfg_t**
  *Runtime configuration class of the **ac2lsl** (p. 78) plugin.*
- class **ac2lsl::ac2lsl_t**
  *Plugin class of **ac2lsl** (p. 78).*

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **ac2lsl**
  *All types for the **ac2lsl** (p. 78) plugins live in this namespace.*

Variables

- const std::map< int, type_info > **ac2lsl::types**

6.2 ac2osc.cpp File Reference

Classes

- class **ac2osc_t**
  Plugin class of the ac2osc plugin.

6.3 ac2wave.cpp File Reference

Classes

- class **ac2wave_t**
- class **ac2wave_if_t**

6.4 ac_monitor_type.cpp File Reference

6.5 ac_monitor_type.hh File Reference

Classes

- class **acmon::ac_monitor_t**
  A class for converting AC variables to Parser monitors of correct type.

Namespaces

- **acmon**
  Namespace for displaying ac variables as parser monitors.

© 2005-2021 HörTech gGmbH, Oldenburg

6.6 ac_mul.cpp File Reference

6.7 ac_mul.hh File Reference

Classes

- class **ac_mul_t**
  *The class which implements the **ac_mul_t*** (p. 191) *plugin.*

Enumerations

- enum **arg_type_t** { **ARG_RR**, **ARG_RC**, **ARG_CR**, **ARG_CC** }
  Indicates whether the factors of the product are real or complex valued.
- enum **val_type_t** { **VAL_REAL**, **VAL_COMPLEX** }
  Indicates whether an AC variable contains real or complex values.

6.7.1 Enumeration Type Documentation

6.7.1.1 arg_type_t `enum` `arg_type_t`

Indicates whether the factors of the product are real or complex valued.

Enumerator

ARG_RR	Both factors are real.
ARG_RC	First factor is real, second is complex.
ARG_CR	First factor is complex, second is real.
ARG_CC	Both factors are complex.

6.7.1.2 val_type_t `enum` `val_type_t`

Indicates whether an AC variable contains real or complex values.

© 2005-2021 HörTech gGmbH, Oldenburg

Enumerator

VAL_REAL	AC variable contains real values.
VAL_COMPLEX	AC variable contains complex values.

6.8 ac_proc.cpp File Reference

Classes

- class **ac_proc::interface_t**

Namespaces

- **ac_proc**

6.9 acConcat_wave.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **acConcat_wave::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

6.9.1 Macro Definition Documentation

6.9.1.1 PATCH_VAR

```
#define PATCH_VAR(
    var ) patchbay.connect(&var.valuechanged, this, & acConcat_wave::update_cfg)
```

6.9.1.2 INSERT_PATCH

```
#define INSERT_PATCH(
    var ) insert_member(var); PATCH_VAR(var)
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.10 acConcat_wave.h File Reference

Classes

- class **acConcat_wave_config**
- class **acConcat_wave**

6.11 acmon.cpp File Reference

Classes

- class **acmon::acmon_t**

Namespaces

- **acmon**
  Namespace for displaying ac variables as parser monitors.

6.12 acPooling_wave.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **acPooling_wave::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

6.12.1 Macro Definition Documentation

6.12.1.1 PATCH_VAR

```
#define PATCH_VAR(
    var ) patchbay.connect(&var.valuechanged, this, & acPooling_wave::update_cfg)
```

6.12.1.2 INSERT_PATCH

```
#define INSERT_PATCH(
    var ) insert_member(var); PATCH_VAR(var)
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.13 acPooling_wave.h File Reference

Classes

- class **acPooling_wave_config**
- class **acPooling_wave**

6.14 acrec.cpp File Reference

6.15 acrec.hh File Reference

Classes

- class **plugins::hoertech::acrec::acwriter_t**

  acwriter_t *(p. 1382) decouples signal processing from writing to disk.*
- class **plugins::hoertech::acrec::acrec_t**

  Plugin interface class of plugin acrec.

Namespaces

- **plugins**
- **plugins::hoertech**
- **plugins::hoertech::acrec**

Functions

- std::string **plugins::hoertech::acrec::to_iso8601** (time_t tm)

6.16 acsave.cpp File Reference

Classes

- class **acsave::save_var_t**
- class **acsave::cfg_t**
- class **acsave::acsave_t**
- struct **acsave::mat4head_t**

Namespaces

- **acsave**

© 2005-2021 HörTech gGmbH, Oldenburg

Macros

- #define **ACSAVE_FMT_TXT** 0
- #define **ACSAVE_SFMT_TXT** "txt"
- #define **ACSAVE_FMT_MAT4** 1
- #define **ACSAVE_SFMT_MAT4** "mat4"
- #define **ACSAVE_FMT_M** 2
- #define **ACSAVE_SFMT_M** "m"

6.16.1 Macro Definition Documentation

6.16.1.1 ACSAVE_FMT_TXT `#define ACSAVE_FMT_TXT 0`

6.16.1.2 ACSAVE_SFMT_TXT `#define ACSAVE_SFMT_TXT "txt"`

6.16.1.3 ACSAVE_FMT_MAT4 `#define ACSAVE_FMT_MAT4 1`

6.16.1.4 ACSAVE_SFMT_MAT4 `#define ACSAVE_SFMT_MAT4 "mat4"`

6.16.1.5 ACSAVE_FMT_M `#define ACSAVE_FMT_M 2`

6.16.1.6 ACSAVE_SFMT_M `#define ACSAVE_SFMT_M "m"`

© 2005-2021 HörTech gGmbH, Oldenburg

6.17 acSteer.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **acSteer↩ ::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

6.17.1 Macro Definition Documentation

6.17.1.1 PATCH_VAR

```
#define PATCH_VAR(
    var ) patchbay.connect(&var.valuechanged, this, & acSteer::update_cfg)
```

6.17.1.2 INSERT_PATCH

```
#define INSERT_PATCH(
    var ) insert_member(var); PATCH_VAR(var)
```

6.18 acSteer.h File Reference

Classes

- class **acSteer_config**
- class **acSteer**

6.19 acTransform_wave.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **acTransform↩ _wave::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

6.19.1 Macro Definition Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.19.1.1 PATCH_VAR

```
#define PATCH_VAR(
    var ) patchbay.connect(&var.valuechanged, this, & acTransform_wave↩
::update_cfg)
```

6.19.1.2 INSERT_PATCH

```
#define INSERT_PATCH(
    var ) insert_member(var); PATCH_VAR(var)
```

6.20 acTransform_wave.h File Reference

Classes

- class **acTransform_wave_config**
- class **acTransform_wave**

6.21 adaptive_feedback_canceller.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **adaptive_↩ feedback_canceller::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

Functions

- void **make_friendly_number_by_limiting** (**mha_real_t** &x)

6.21.1 Macro Definition Documentation

6.21.1.1 PATCH_VAR

```
#define PATCH_VAR(
    var ) patchbay.connect(&var.valuechanged, this, & adaptive_feedback_↩
canceller::update_cfg)
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.21.1.2 INSERT_PATCH

```
#define INSERT_PATCH( var ) insert_member(var); PATCH_VAR(var)
```

6.21.2 Function Documentation

6.21.2.1 make_friendly_number_by_limiting()

```
void make_friendly_number_by_limiting ( mha_real_t & x ) [inline]
```

6.22 adaptive_feedback_canceller.h File Reference

Classes

- class **adaptive_feedback_canceller_config**
- class **adaptive_feedback_canceller**

6.23 addsndfile.cpp File Reference

Classes

- class **addsndfile::waveform_proxy_t**

  Class helps to specify which instance of MHASignal_waveform_t parent instance is meant in ***resampled_soundfile_t*** *(p. 257).*
- class **addsndfile::resampled_soundfile_t**

  Reads sound from file and resamples it if necessary and wanted.
- class **addsndfile::sndfile_t**
- class **addsndfile::level_adapt_t**
- class **addsndfile::addsndfile_if_t**

Namespaces

- **addsndfile**

Macros

- #define **DEBUG**(x) std::cerr << __FILE__ << ":" << __LINE__ << " " #x "=" << x << std::endl

© 2005-2021 HörTech gGmbH, Oldenburg

Typedefs

- typedef **MHAPlugin::config_t**< level_adapt_t > **addsndfile::level_adaptor**
- typedef **MHAPlugin::plugin_t**< sndfile_t > **addsndfile::wave_reader**

Enumerations

- enum **addsndfile::addsndfile_resampling_mode_t** { **addsndfile::DONT_RESAMP↩LE_PERMISSIVE**, **addsndfile::DONT_RESAMPLE_STRICT**, **addsndfile::DO_RES↩AMPLE** }

  Specifies the resampling mode in resampled_soundfile_t.

Functions

- static unsigned **addsndfile::resampled_num_frames** (unsigned num_source_frames, float source_rate, float target_rate, addsndfile_resampling_mode_t resampling_mode)

6.23.1 Macro Definition Documentation

6.23.1.1 DEBUG `#define DEBUG( x ) std::cerr << __FILE__ << ":" << __LINE__ << " " #x "=" << x << std::endl`

6.24 adm.cpp File Reference

Classes

- class **adm_rtconfig_t**
- class **adm_if_t**

Functions

- **MHASignal::waveform_t** ∗ **adm_fir_lp** (unsigned int fs, unsigned f_pass, unsigned int f_stop, unsigned int order)
- **MHASignal::waveform_t** ∗ **adm_fir_decomb** (unsigned int fs, float dist_m, unsigned int order)

© 2005-2021 HörTech gGmbH, Oldenburg

6.24.1 Function Documentation

6.24.1.1 adm_fir_lp()

```
MHASignal::waveform_t* adm_fir_lp (
    unsigned int fs,
    unsigned f_pass,
    unsigned int f_stop,
    unsigned int order )
```

6.24.1.2 adm_fir_decomb()

```
MHASignal::waveform_t* adm_fir_decomb (
    unsigned int fs,
    float dist_m,
    unsigned int order )
```

6.25 adm.hh File Reference

Classes

- class **ADM::Linearphase_FIR**< **F** >

  An efficient linear-phase fir filter implementation.
- class **ADM::Delay**< **F** >

  A delay-line class.
- class **ADM::ADM**< **F** >

  Adaptive differential microphone, working for speech frequency range.

Namespaces

- **ADM**

Functions

- static double **ADM::subsampledelay_coeff** (double samples, double f_design, double fs=1.0)

  compute IIR coefficient for subsample delay

© 2005-2021 HörTech gGmbH, Oldenburg

Variables

- const double **ADM::PI** = 3.14159265358979312
- const double **ADM::C** = 340
- const double **ADM::DELAY_FREQ** = 2000
- const double **ADM::START_BETA** = 0.5

6.26 altconfig.cpp File Reference

6.27 altconfig.hh File Reference

Classes

- class **altconfig_t**
  Single class implementing plugin altconfig.

Macros

- #define **MHAPLUGIN_OVERLOAD_OUTDOMAIN**

6.27.1 Macro Definition Documentation

6.27.1.1 MHAPLUGIN_OVERLOAD_OUTDOMAIN `#define MHAPLUGIN_OVERLOAD_OUTDOM↩AIN`

6.28 altplugs.cpp File Reference

Classes

- class **mhaplug_cfg_t**
- class **altplugs_t**

Macros

- #define **MHAPLUGIN_OVERLOAD_OUTDOMAIN**

© 2005-2021 HörTech gGmbH, Oldenburg

6.28.1 Macro Definition Documentation

6.28.1.1 MHAPLUGIN_OVERLOAD_OUTDOMAIN `#define MHAPLUGIN_OVERLOAD_OUTDOMAIN`

6.29 analysemhaplugin.cpp File Reference

Functions

- std::string **strdom** (**mha_domain_t** d)
- void **print_ac** (**MHAKernel::algo_comm_class_t** &ac, std::string txt)
- int **document_plugin** (**MHAKernel::algo_comm_class_t** &ac, **PluginLoader::mhapluginloader_t** &load, int argc, char ∗∗argv)
- void **document_io_plugin** (char ∗lib_name)
- int **main** (int argc, char ∗∗argv)

6.29.1 Function Documentation

6.29.1.1 strdom()

```
std::string strdom (
    mha_domain_t d )
```

6.29.1.2 print_ac()

```
void print_ac (
    MHAKernel::algo_comm_class_t & ac,
    std::string txt )
```

6.29.1.3 document_plugin()

```
int document_plugin (
    MHAKernel::algo_comm_class_t & ac,
    PluginLoader::mhapluginloader_t & load,
    int argc,
    char ** argv )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.29.1.4 document_io_plugin()

```
void document_io_plugin (
    char * lib_name )
```

6.29.1.5 main()

```
int main (
    int argc,
    char ** argv )
```

6.30 analysispath.cpp File Reference

Classes

- class **analysepath_t**
- class **plug_t**
- class **analysispath_if_t**

Functions

- static void * **thread_start** (void *instance)

6.30.1 Function Documentation

6.30.1.1 thread_start()

```
static void* thread_start (
    void * instance )  [static]
```

6.31 attenuate20.cpp File Reference

Classes

- class **attenuate20_t**

© 2005-2021 HörTech gGmbH, Oldenburg

6.32 audiometerbackend.cpp File Reference

Classes

- class **audiometerbackend::lnn3rdoct_t**
- class **audiometerbackend::sine_t**
- class **audiometerbackend::signal_gen_t**
- class **audiometerbackend::level_adapt_t**
- class **audiometerbackend::audiometer_if_t**

Namespaces

- **audiometerbackend**

Macros

- #define **DEBUG**(x) std::cerr << __FILE__ << ":" << __LINE__ << " " #x "=" << x << std::endl

Typedefs

- typedef **MHAPlugin::config_t**< level_adapt_t > **audiometerbackend::level_adaptor**
- typedef **MHAPlugin::plugin_t**< signal_gen_t > **audiometerbackend::generator**

Functions

- static unsigned int **audiometerbackend::gcd** (unsigned int a, unsigned int b)
- **MHASignal::waveform_t audiometerbackend::return_sig** (unsigned int sigtype, unsigned int fs, unsigned int f)

6.32.1 Macro Definition Documentation

6.32.1.1 DEBUG

```
#define DEBUG(
        x ) std::cerr << __FILE__ << ":" << __LINE__ << " " #x "=" << x <<
std::endl
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.33 auditory_profile.cpp File Reference

6.34 auditory_profile.h File Reference

Classes

- class **AuditoryProfile::fmap_t**
  A class to store frequency dependent data (e.g., HTL and UCL).
- class **AuditoryProfile::profile_t**
  The Auditory Profile class.
- class **AuditoryProfile::profile_t::ear_t**
  Class for ear-dependent parameters, e.g., audiograms or unilateral loudness scaling.
- class **AuditoryProfile::parser_t**
  Class to make the auditory profile accessible through the parser interface.
- class **AuditoryProfile::parser_t::fmap_t**
- class **AuditoryProfile::parser_t::ear_t**

Namespaces

- **AuditoryProfile**
  Namespace for classes and functions around the auditory profile (e.g., audiogram handling)

6.35 browsemhaplugins.cpp File Reference

Macros

- #define **DEBUG**(x) std::cerr << __FILE__ << ":" << __LINE__ << " " << #x<<"="<<x << std::endl

Functions

- int **main** (int argc, char **argv)

6.35.1 Macro Definition Documentation

6.35.1.1 DEBUG

```
#define DEBUG(
        x ) std::cerr << __FILE__ << ":" << __LINE__ << " " << #x<<"="<<x
<< std::endl
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.35.2 Function Documentation

6.35.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

6.36 coherence.cpp File Reference

Classes

- class **coherence::vars_t**
- class **coherence::cohflt_t**
- class **coherence::cohflt_if_t**

Namespaces

- **coherence**

Functions

- void **coherence::getcipd** (**mha_complex_t** &c, **mha_real_t** &a, const **mha_complex_t** &xl, const **mha_complex_t** &xr)

6.37 combinechannels.cpp File Reference

Classes

- class **combc_t**
- class **combc_if_t**

© 2005-2021 HörTech gGmbH, Oldenburg

6.38 compiler_id.cpp File Reference

6.39 compiler_id.hh File Reference

Macros

- #define **COMPILER_ID_VENDOR** "gcc"
- #define **COMPILER_ID_MAJOR** __GNUC__
- #define **COMPILER_ID_MINOR** __GNUC_MINOR__
- #define **COMPILER_ID_PATCH** __GNUC_PATCHLEVEL__
- #define **COMPILER_ID_VERSION_HELPER2**(x, y, z) #x "." #y "." #z
- #define **COMPILER_ID_VERSION_HELPER1**(x, y, z) **COMPILER_ID_VERSION_H↩ELPER2**(x,y,z)
- #define **COMPILER_ID_VERSION**
- #define **COMPILER_ID** **COMPILER_ID_VENDOR** "-" **COMPILER_ID_VERSION** "-" COMPILER_ID_STANDARD

6.39.1 Macro Definition Documentation

6.39.1.1 COMPILER_ID_VENDOR `#define COMPILER_ID_VENDOR "gcc"`

6.39.1.2 COMPILER_ID_MAJOR `#define COMPILER_ID_MAJOR __GNUC__`

6.39.1.3 COMPILER_ID_MINOR `#define COMPILER_ID_MINOR __GNUC_MINOR__`

6.39.1.4 COMPILER_ID_PATCH `#define COMPILER_ID_PATCH __GNUC_PATCHLEVEL__`

© 2005-2021 HörTech gGmbH, Oldenburg

6.39.1.5 COMPILER_ID_VERSION_HELPER2

```
#define COMPILER_ID_VERSION_HELPER2(
    x,
    y,
    z ) #x "." #y "." #z
```

6.39.1.6 COMPILER_ID_VERSION_HELPER1

```
#define COMPILER_ID_VERSION_HELPER1(
    x,
    y,
    z ) COMPILER_ID_VERSION_HELPER2(x,y,z)
```

6.39.1.7 COMPILER_ID_VERSION

```
#define COMPILER_ID_VERSION
```

6.39.1.8 COMPILER_ID

```
#define COMPILER_ID COMPILER_ID_VENDOR "-" COMPILER_ID_↩
VERSION "-" COMPILER_ID_STANDARD
```

6.40 complex_filter.cpp File Reference

6.41 complex_filter.h File Reference

Classes

- class **MHAFilter::complex_bandpass_t**
  Complex bandpass filter.
- class **MHAFilter::gamma_flt_t**
  Class for gammatone filter.
- class **MHAFilter::thirdoctave_analyzer_t**

Namespaces

- **MHAFilter**
  Namespace for IIR and FIR filter classes.

© 2005-2021 HörTech gGmbH, Oldenburg

6.42 complex_scale_channel.cpp File Reference

Classes

- class **cfg_t**
- class **complex_scale_channel_t**

6.43 cpuload.cpp File Reference

Classes

- class **cpuload::cpuload_cfg_t**
- class **cpuload::cpuload_if_t**

Namespaces

- **cpuload**

6.44 db.cpp File Reference

Classes

- class **db_t**
- class **db_if_t**

6.45 dbasync.cpp File Reference

Classes

- class **dbasync_native::delay_check_t**
- class **dbasync_native::dbasync_t**
- class **dbasync_native::db_if_t**

Namespaces

- **dbasync_native**

© 2005-2021 HörTech gGmbH, Oldenburg

Enumerations

- enum { **dbasync_native::INVALID_THREAD_PRIORITY** = 999999999 }

Functions

- static void * **dbasync_native::thread_start** (void *instance)
- static unsigned **dbasync_native::gcd** (unsigned a, unsigned b)

6.46 dc.cpp File Reference

Macros

- #define **DUPVEC**(x) v.x.data = **MHASignal::dupvec_chk**(v.x.data,s)

Functions

- unsigned int **get_audiochannels** (unsigned int totalchannels, std::string acname, **algo_comm_t** ac)

6.46.1 Macro Definition Documentation

6.46.1.1 DUPVEC

```
#define DUPVEC(
    x ) v.x.data = MHASignal::dupvec_chk(v.x.data,s)
```

6.46.2 Function Documentation

6.46.2.1 get_audiochannels()

```
unsigned int get_audiochannels (
    unsigned int totalchannels,
    std::string acname,
    algo_comm_t ac )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.47 dc.hh File Reference

Classes

- class **dc::dc_vars_t**
- class **dc::dc_vars_validator_t**
- class **dc::dc_t**
- class **dc::dc_if_t**

Namespaces

- **dc**

6.48 dc_afterburn.cpp File Reference

Namespaces

- **DynComp**
  dynamic compression related classes and functions

Functions

- float **mylogf** (float x)

6.48.1 Function Documentation

6.48.1.1 mylogf() `float mylogf ( float x )`

6.49 dc_afterburn.h File Reference

Classes

- class **DynComp::dc_afterburn_vars_t**
  *Variables for **dc_afterburn_t** (p. 453) class.*
- class **DynComp::dc_afterburn_rt_t**
  Real-time class for after burn effect.
- class **DynComp::dc_afterburn_t**
  Afterburn class, to be defined as a member of compressors.

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **DynComp**
  dynamic compression related classes and functions

6.50 dc_simple.cpp File Reference

Namespaces

- **dc_simple**

Functions

- void **dc_simple::test_fail** (const std::vector< float > &v, unsigned int s, const std::string &name)
  Checks size of vector.
- std::vector< float > **dc_simple::force_resize** (const std::vector< float > &v, unsigned int s, const std::string &name)
  Creates a copy of vector v with s elements, provided that \v has either s elements or 1 elements.
- **mha_real_t dc_simple::not_zero** (**mha_real_t** x, const std::string &comment)
  Helper function to throw an error if x is 0.

6.51 dc_simple.hh File Reference

Classes

- class **dc_simple::dc_vars_t**
  *class for **dc_simple** (p. 86) plugin which registers variables to **MHAParser** (p. 122).*
- class **dc_simple::dc_vars_validator_t**
  Helper class to check sizes of configuration variable vectors.
- class **dc_simple::level_smoother_t**
  Class which computes smoothed input levels on individual bands, using an attack and release filter, which are a first order low pass filter and a maximum tracker filter, respectively.
- class **dc_simple::dc_t**
  *Runtime config class for **dc_simple** (p. 86) plugin.*
- class **dc_simple::dc_t::line_t**
  Helper class for usage in computing compression, expansion and limiting.
- class **dc_simple::dc_if_t**
  *interface class for **dc_simple** (p. 86)*

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **dc_simple**

Typedefs

- typedef **MHAPlugin::plugin_t**< dc_t > **dc_simple::DC**

  Define alternate name for runtime_cfg_t.
- typedef **MHAPlugin::config_t**< level_smoother_t > **dc_simple::LEVEL**

  Define alternate name for config_t.

Functions

- void **dc_simple::test_fail** (const std::vector< float > &v, unsigned int s, const std::string &name)

  Checks size of vector.
- std::vector< float > **dc_simple::force_resize** (const std::vector< float > &v, unsigned int s, const std::string &name)

  Creates a copy of vector `v` with `s` elements, provided that \v has either s elements or 1 elements.
- **mha_real_t dc_simple::not_zero** (**mha_real_t** x, const std::string &comment)

  Helper function to throw an error if `x` is 0.

6.52 delay.cpp File Reference

Namespaces

- **delay**

6.53 delay.hh File Reference

Classes

- class **delay::interface_t**

Namespaces

- **delay**

© 2005-2021 HörTech gGmbH, Oldenburg

6.54 delaysum_spec.cpp File Reference

Classes

- class **delaysum_spec::delaysum_t**
- class **delaysum_spec::delaysum_spec_if_t**

Namespaces

- **delaysum_spec**

6.55 delaysum_wave.cpp File Reference

Classes

- class **delaysum::delaysum_wave_t**
  Runtime configuration of the delaysum_wave plugin.
- class **delaysum::delaysum_wave_if_t**
  Interface class for the delaysum plugin.

Namespaces

- **delaysum**
  This namespace contains the delaysum plugin.

6.56 doasvm_classification.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **doasvm_↩classification::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

6.56.1 Macro Definition Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.56.1.1 PATCH_VAR

```
#define PATCH_VAR(
    var ) patchbay.connect(&var.valuechanged, this, & doasvm_classification↩
::update_cfg)
```

6.56.1.2 INSERT_PATCH

```
#define INSERT_PATCH(
    var ) insert_member(var); PATCH_VAR(var)
```

6.57 doasvm_classification.h File Reference

Classes

- class **doasvm_classification_config**
- class **doasvm_classification**

6.58 doasvm_feature_extraction.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **doasvm_↩ feature_extraction::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

6.58.1 Macro Definition Documentation

6.58.1.1 PATCH_VAR

```
#define PATCH_VAR(
    var ) patchbay.connect(&var.valuechanged, this, & doasvm_feature_↩
extraction::update_cfg)
```

6.58.1.2 INSERT_PATCH

```
#define INSERT_PATCH(
    var ) insert_member(var); PATCH_VAR(var)
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.59 doasvm_feature_extraction.h File Reference

Classes

- class **doasvm_feature_extraction_config**
- class **doasvm_feature_extraction**

6.60 doc_appendix.h File Reference

6.61 doc_examples.h File Reference

6.62 doc_frameworks.h File Reference

6.63 doc_general.h File Reference

6.64 doc_kernel.h File Reference

6.65 doc_matlab.h File Reference

6.66 doc_mhamain.h File Reference

6.67 doc_parser.h File Reference

6.68 doc_plugins.h File Reference

6.69 doc_system.h File Reference

6.70 doc_toolbox.h File Reference

6.71 double2acvar.cpp File Reference

Classes

- class **double2acvar::double2acvar_t**

  *Plugin interface class for **double2acvar*** (p. 90).

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **double2acvar**

6.72 downsample.cpp File Reference

Classes

- class **ds_t**

6.73 dropgen.cpp File Reference

Classes

- class **dropgen_t**

6.74 droptect.cpp File Reference

Classes

- class **droptect_t**
  Detect dropouts in a signal with a constant spectrum.

6.75 equalize.cpp File Reference

Classes

- class **equalize::cfg_t**
- class **equalize::freqgains_t**

Namespaces

- **equalize**

6.76 example1.cpp File Reference

Classes

- class **example1_t**
  This C++ class implements the simplest example plugin for the step-by-step tutorial.

© 2005-2021 HörTech gGmbH, Oldenburg

6.77 example2.cpp File Reference

Classes

- class **example2_t**
  This C++ class implements the second example plugin for the step-by-step tutorial.

6.78 example3.cpp File Reference

Classes

- class **example3_t**
  A Plugin class using the openMHA Event mechanism.

6.79 example4.cpp File Reference

Classes

- class **example4_t**
  A Plugin class using the spectral signal.

6.80 example5.cpp File Reference

Classes

- class **example5_t**
- class **plugin_interface_t**

6.81 example6.cpp File Reference

Classes

- class **cfg_t**
- class **example6_t**

© 2005-2021 HörTech gGmbH, Oldenburg

6.82 example7.cpp File Reference

6.83 example7.hh File Reference

Classes

- class **example7_t**

6.84 fader_spec.cpp File Reference

Classes

- class **spec_fader_t**
- class **fader_if_t**

6.85 fader_wave.cpp File Reference

Classes

- class **fader_wave::level_adapt_t**
- class **fader_wave::fader_wave_if_t**

Namespaces

- **fader_wave**

Macros

- #define **DEBUG**(x) std::cerr << __FILE__ << ":" << __LINE__ << " " #x "=" << x << std::endl

Typedefs

- typedef **MHAPlugin::plugin_t**< level_adapt_t > **fader_wave::level_adaptor**

6.85.1 Macro Definition Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.85.1.1 DEBUG

```
#define DEBUG(
            x ) std::cerr << __FILE__ << ":" << __LINE__ << " " #x "=" << x <<
std::endl
```

6.86 ffttfbpow.cpp File Reference

Classes

- class **ffttfbpow::ffttfbpow_t**
  Run time configuration for the ffttfbpow plugin.
- class **ffttfbpow::ffttfbpow_interface_t**
  Interface class for ffttfbpow plugin.

Namespaces

- **ffttfbpow**
  Namespace for the ffttfbpow plugin.

6.87 ffttfilter.cpp File Reference

Classes

- class **ffttfilter::ffttfilter_t**
- class **ffttfilter::interface_t**

Namespaces

- **ffttfilter**

Functions

- unsigned int **ffttfilter::irs_length** (const **MHAParser::mfloat_t** &irs)
- unsigned int **ffttfilter::irs_validator** (const **MHAParser::mfloat_t** &irs, const unsigned int & **channels**, const unsigned int &fragsize, const unsigned int &fftlen)

6.88 ffttfilterbank.cpp File Reference

Classes

- class **ffttfilterbank::ffttfb_plug_t**
- class **ffttfilterbank::ffttfb_interface_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **fftfilterbank**

6.89 fshift.cpp File Reference

6.90 fshift.hh File Reference

Classes

- class **fshift::fshift_config_t**
  fshift runtime config class
- class **fshift::fshift_t**
  fshift plugin interface class

Namespaces

- **fshift**
  All types for the fshift plugin live in this namespace.

Functions

- int **fshift::fft_find_bin** (**mha_real_t** frequency, unsigned fftlen, **mha_real_t** srate)
  Finds bin number of FFT bin nearest to the given frequency.

6.91 fshift_hilbert.cpp File Reference

Classes

- class **fshift_hilbert::hilbert_shifter_t**
- class **fshift_hilbert::frequency_translator_t**

Namespaces

- **fshift_hilbert**
  All types for the hilbert frequency shifter live in this namespace.

© 2005-2021 HörTech gGmbH, Oldenburg

6.92 gain.cpp File Reference

Classes

- class **gain::scaler_t**
- class **gain::gain_if_t**

Namespaces

- **gain**

6.93 gaintable.cpp File Reference

Functions

- std::vector< **mha_real_t** > **convert_f2logf** (const std::vector< **mha_real_t** > &vF)
- bool **isempty** (const std::vector< std::vector< **mha_real_t** > > &arg)

6.93.1 Function Documentation

6.93.1.1 convert_f2logf()

```
std::vector< mha_real_t> convert_f2logf (
    const std::vector< mha_real_t > & vF )
```

6.93.1.2 isempty()

```
bool isempty (
    const std::vector< std::vector< mha_real_t > > & arg )
```

6.94 gaintable.h File Reference

Classes

- class **DynComp::gaintable_t**

  Gain table class.

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **DynComp**

  dynamic compression related classes and functions

Functions

- **mha_real_t DynComp::interp1** (const std::vector< **mha_real_t** > &vX, const std↩ ::vector< **mha_real_t** > &vY, **mha_real_t** X)

  One-dimensional linear interpolation.
- **mha_real_t DynComp::interp2** (const std::vector< **mha_real_t** > &vX, const std↩ ::vector< **mha_real_t** > &vY, const std::vector< std::vector< **mha_real_t** > > &mZ, **mha_real_t** X, **mha_real_t** Y)

  Linear interpolation in a two-dimensional field.

6.95 generatemhaplugindoc.cpp File Reference

Classes

- class **plug_wrapperl**
- class **io_wrapper**
- class **plug_wrapper**
- class **latex_doc_t**

  Class to access the information stored in the plugin source code's MHAPLUGIN_DOCUME↩ NTATION macro.

Functions

- std::string **conv2latex** (std::string s, bool iscolored=false)

  Escapes various character sequences in texts not intended to be processed by LaTeX for processing by LaTeX.
- static void **print_plugin_references** (const **std::set**< std::string > &all_categories, std::map< std::string, std::vector< std::string > > main_category_plugins, std::map< std::string, std::vector< std::string > > additional_category_plugins, std::ofstream &ofile, const std::string &category_macro)

  Function prints an overview of all categories and their associated plugins into the document.
- std::vector< std::string > **create_latex_doc** (std::map< std::string, std::string > &doc, const std::string &plugname, const std::string &plugin_macro)

  Loads the plugin, creates the latex documentation for the plugin, and adds the latex documentation for this plugin to the plugin's main category entry in doc.
- int **main** (int argc, char **argv)

© 2005-2021 HörTech gGmbH, Oldenburg

6.95.1 Function Documentation

6.95.1.1 conv2latex()

```
std::string conv2latex (
    std::string s,
    bool iscolored = false )
```

Escapes various character sequences in texts not intended to be processed by LaTeX for processing by LaTeX.

Focus is on correct display of symbols contained in these texts. E.g. the help texts of MHA configuration variables can be processed by this function. The contents of the MHAPLUGIN↩ _DOCUMENTATION is already in LaTeX format and should not be processed by this function.

Returns

A copy of s with various symbols escaped for LaTeX processing

Parameters

s	Text not ready for LaTeX
iscolored	if true, the complete returned text is surrounded with "\\color{monitorcolor}{" and "}"

6.95.1.2 print_plugin_references()

```
static void print_plugin_references (
    const std::set< std::string > & all_categories,
    std::map< std::string, std::vector< std::string > > main_category_↩
plugins,
    std::map< std::string, std::vector< std::string > > additional_category↩
_plugins,
    std::ofstream & ofile,
    const std::string & category_macro ) [static]
```

Function prints an overview of all categories and their associated plugins into the document.

Parameters

all_categories	A sorted container with all category names
main_category_plugins	map of main categories to plugin names
additional_category_plugins	map of tags to plugin names
ofile	Latex document is produced by writing output to this stream

© 2005-2021 HörTech gGmbH, Oldenburg

6.95.1.3 create_latex_doc()

```
std::vector<std::string> create_latex_doc (
    std::map< std::string, std::string > & doc,
    const std::string & plugname,
    const std::string & plugin_macro )
```

Loads the plugin, creates the latex documentation for the plugin, and adds the latex documentation for this plugin to the plugin's main category entry in doc.

Returns

the vector of all categories.

Parameters

doc	map of main categories to a string containint the documentation of all plugins in that categories. The documentation of the current plugin will be appended to the existing documentation of its main category. Will be created if non-existant.
plugname	Name of the MHA plugin to process
plugin_macro	name of the LaTeX section macro that documents a single plugin (e.g. "section", "subsection", "subsubsection", ...)

6.95.1.4 main()

```
int main (
    int argc,
    char ** argv )
```

6.96 gsc_adaptive_stage.cpp File Reference

6.97 gsc_adaptive_stage.hh File Reference

Classes

- class **gsc_adaptive_stage::gsc_adaptive_stage**

Namespaces

- **gsc_adaptive_stage**

© 2005-2021 HörTech gGmbH, Oldenburg

Variables

- constexpr **mha_real_t gsc_adaptive_stage::DELT** =1e-12
  Small constant to ensure no division by zero occurs.

6.98 gsc_adaptive_stage_if.cpp File Reference

6.99 gsc_adaptive_stage_if.hh File Reference

Classes

- class **gsc_adaptive_stage::gsc_adaptive_stage_if**
  Plugin interface class.

Namespaces

- **gsc_adaptive_stage**

6.100 gtfb_analyzer.cpp File Reference

Gammatone Filterbank Analyzer Plugin.

Classes

- struct **gtfb_analyzer::gtfb_analyzer_cfg_t**
  Configuration for Gammatone Filterbank Analyzer.
- class **gtfb_analyzer::gtfb_analyzer_t**
  Gammatone Filterbank Analyzer Plugin.

Namespaces

- **gtfb_analyzer**

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- static const **mha_complex_t** & **filter_complex** (const **mha_complex_t** &input, const **mha_complex_t** &coeff, **mha_complex_t** ∗states, unsigned orders)

  Filters a complex input sample with the given filter coefficient.
- static const **mha_complex_t** & **filter_real** (**mha_real_t** input, **mha_complex_t** &tmp↩_complex, const **mha_complex_t** &coeff, **mha_complex_t** ∗states, unsigned orders, const **mha_complex_t** &normphase)

  Filters a real input sample with the given filter coefficient and applies the given normalization with phase correction.

6.100.1 Detailed Description

Gammatone Filterbank Analyzer Plugin.

6.100.2 Function Documentation

6.100.2.1 filter_complex()

```
static const mha_complex_t& filter_complex (
    const mha_complex_t & input,
    const mha_complex_t & coeff,
    mha_complex_t * states,
    unsigned orders ) [inline], [static]
```

Filters a complex input sample with the given filter coefficient.

No normalization takes place. The implementation is tail-recursive and to exploit compiler optimization.

Parameters

input	The complex input sample
coeff	The complex filter coefficient
states	Pointer to the array of complex filter states.
orders	The filter order

Returns

A const ref to the filtered sample

© 2005-2021 HörTech gGmbH, Oldenburg

6.100.2.2 filter_real()

```
static const mha_complex_t& filter_real (
    mha_real_t input,
    mha_complex_t & tmp_complex,
    const mha_complex_t & coeff,
    mha_complex_t * states,
    unsigned orders,
    const mha_complex_t & normphase ) [inline], [static]
```

Filters a real input sample with the given filter coefficient and applies the given normalization with phase correction.

Parameters

input	The real input sample
tmp_complex	A reference to a **mha_complex_t** (p. 745) used for intermediate results. No assumptions should be made about the state of tmp_complex after the return of filter_real. This is an optimization to reduce the number of dtor/ctor calls of **mha_complex_t** (p. 745)
coeff	The complex filter coefficient
states	Pointer to the array of complex filter states.
orders	The filter order
normphase	Normalization coefficient including the phase correction

Returns

A const ref to the filtered sample

6.101 gtfb_simd.cpp File Reference

Gammatone Filterbank Analyzer Plugin using SIMD.

Classes

- class **gtfb_simd_cfg_t**
- class **gtfb_simd_t**

Macros

- #define **add4f**(a, b) __builtin_ia32_addps(a,b)
- #define **sub4f**(a, b) __builtin_ia32_subps(a,b)
- #define **mul4f**(a, b) __builtin_ia32_mulps(a,b)
- #define **MXCSR_DAZ** (1 << 6) /* Enable denormals are zero mode */
- #define **MXCSR_FTZ** (1 << 15) /* Enable flush to zero mode */
- #define **check_alignment**(ptr, alignment)

  Checks alignment of pointer address.

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- void **filter_sisd_complex** (const unsigned bands, const unsigned order, const **mha_complex_t** ∗inputs, **mha_complex_t** ∗outputs, const **mha_complex_t** ∗coefficients, **mha_complex_t** ∗states)

  Filters one sample per band, using SISD operations and the mha_complex operations.
- void **filter_sisd_real** (const unsigned bands, const unsigned order, const **mha_complex_t** ∗inputs, **mha_complex_t** ∗outputs, const **mha_complex_t** ∗coefficients, **mha_complex_t** ∗states)

  Filters one sample per band, using SISD operations and real operations (operating on real and imaginary part as necessary).
- void **filter_simd** (const unsigned bands, const unsigned order, const **mha_real_t** ∗rinputs, const **mha_real_t** ∗iinputs, **mha_real_t** ∗routputs, **mha_real_t** ∗ioutputs, const **mha_real_t** ∗rcoefficients, const **mha_real_t** ∗icoefficients, **mha_real_t** ∗rstates, **mha_real_t** ∗istates)

  Filters one sample per band, using simd operations on float32 To process more than one sample, the function must be called repeatedly in correct order (from oldest sample to newest sample), and the calling function must preserve the filter state (parameters rstates and istates).

6.101.1 Detailed Description

Gammatone Filterbank Analyzer Plugin using SIMD.

A single-instruction-multiple-data (SIMD) implementation of a gammatone filterbank (GTFB)

Not all functions in this file are actually used. Read the functions in this file as a path to convert an algorithm, here complex-valued gammatone filtering as introduced in Hohmann 2002, from a single-instruction-single-data (SISD) implementation that uses complex arithmetic operations to a SIMD implementation of the same, splitting the complex arithmetic operations into their defining real operations, i.e (a+b).real==a.real+b.real, (a+b).imag==a.imag+b.imag, (a∗b).real==a.real∗b.real-a.imag∗b.imag, (a∗b).imag==a.real∗b.imag+a.imag∗b.real.

6.101.2 Macro Definition Documentation

6.101.2.1 add4f

```
#define add4f(
    a,
    b ) __builtin_ia32_addps(a,b)
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.101.2.2 sub4f

```
#define sub4f(
    a,
    b ) __builtin_ia32_subps(a,b)
```

6.101.2.3 mul4f

```
#define mul4f(
    a,
    b ) __builtin_ia32_mulps(a,b)
```

6.101.2.4 MXCSR_DAZ

```
#define MXCSR_DAZ (1 << 6) /* Enable denormals are zero mode */
```

6.101.2.5 MXCSR_FTZ

```
#define MXCSR_FTZ (1 << 15) /* Enable flush to zero mode */
```

6.101.2.6 check_alignment

```
#define check_alignment(
    ptr,
    alignment )
```

Checks alignment of pointer address.

Parameters

ptr	pointer to check
alignment	required alignment

Exceptions

MHA_Error *(p. 764)*	if ptr is not aligned as required.

6.101.3 Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.101.3.1 filter_sisd_complex()

```
void filter_sisd_complex (
    const unsigned bands,
    const unsigned order,
    const mha_complex_t * inputs,
    mha_complex_t * outputs,
    const mha_complex_t * coefficients,
    mha_complex_t * states )
```

Filters one sample per band, using SISD operations and the mha_complex operations.

To process more than one sample, the function must be called repeatedly in correct order (from oldest sample to newest sample), and the calling function must preserve the filter state (parameter states).

This function is not actually used in this plugin, but can be used for testing. It implements the Hohmann 2002 filtering in the most readable form, and is translated towards a SIMD implementation in the following functions.

Parameters

bands	Number of total bands to compute (i.e. input_channels ∗ num_frequencies)
order	Gammatone filter order
inputs	Pointer to array of complex input samples, only 1 sample per band
outputs	Pointer to array with space for output samples, 1 complex sample per band
coefficients	Pointer to array of recursive filter coefficient, 1 coefficient per band. The same coefficient is reused for all filter orders.
states	Pointer to array of complex filter states. Array size is bands∗order. Initialize all elements with zeros before filtering the first sample. Filter state values will be modified by the function. For filtering the next sample, this function needs the filter state array from filtering the previous sample again, unmodified. The filter state of band b, order o can be found at index [b+o∗bands]

6.101.3.2 filter_sisd_real()

```
void filter_sisd_real (
    const unsigned bands,
    const unsigned order,
    const mha_complex_t * inputs,
    mha_complex_t * outputs,
    const mha_complex_t * coefficients,
    mha_complex_t * states ) [inline]
```

Filters one sample per band, using SISD operations and real operations (operating on real and imaginary part as necessary).

To process more than one sample, the function must be called repeatedly in correct order (from oldest sample to newest sample), and the calling function must preserve the filter state (parameter states).

© 2005-2021 HörTech gGmbH, Oldenburg

This function is not actually used in this plugin, but can be used for testing. It reimplements filter_sisd_complex, but expands the complex operations into real arithmetics.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

bands	Number of total bands to compute (i.e. input_channels ∗ num_frequencies)
order	Gammatone filter order
inputs	Pointer to array of complex input samples, only 1 sample per band
outputs	Pointer to array with space for output samples, 1 complex sample per band
coefficients	Pointer to array of recursive filter coefficient, 1 coefficient per band. The same coefficient is reused for all filter orders.
states	Pointer to array of complex filter states. Array size is bands∗order. Initialize all elements with zeros before filtering the first sample. Filter state values will be modified by the function. For filtering the next sample, this function needs the filter state array from filtering the previous sample again, unmodified. The filter state of band b, order o can be found at index [b+o∗bands]

6.101.3.3 filter_simd()

```
void filter_simd (
const unsigned bands,
const unsigned order,
const mha_real_t * rinputs,
const mha_real_t * iinputs,
mha_real_t * routputs,
mha_real_t * ioutputs,
const mha_real_t * rcoefficients,
const mha_real_t * icoefficients,
mha_real_t * rstates,
mha_real_t * istates ) [inline]
```

Filters one sample per band, using simd operations on float32 To process more than one sample, the function must be called repeatedly in correct order (from oldest sample to newest sample), and the calling function must preserve the filter state (parameters rstates and istates).

This reimplements filter_sisd_real, but uses the CPU's vector registers for the arithmetics, and is actually used by this plugin.

Parameters

bands	Number of total bands to compute (i.e. input_channels ∗ num_frequencies) bands is also the size of the arrays pointed to by rinputs, iinputs, routputs, ioutputs, rcoefficients, icoefficients.
order	Gammatone filter order
rinputs	Pointer to array of the real part of input samples
iinputs	Pointer to array of the imaginary part of input samples
routputs	Pointer to array with space for the real parts of the output samples
routputs	Pointer to array with space for the real parts of the output samples
rcoefficients	Pointer to array of the real parts of the recursive filter coefficients

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

icoefficients	Pointer to array of the imaginary parts of the filter coefficients
rstates	Pointer to array of real parts of filter states. Array size is bands∗order. Initialize all elements with zeros before filtering the first sample. Filter state values will be modified by the function. For filtering the next sample, this function needs the filter state arrays from filtering the previous sample again, unmodified. The real part of the filter state of band b, order o can be found at index [b+o∗bands]
istates	Pointer to array of imaginary parts of filter states. Array size is bands∗order. Initialize all elements with zeros before filtering the first sample. Filter state values will be modified by the function. For filtering the next sample, this function needs the filter state arrays from filtering the previous sample again, unmodified. The imaginary part of the filter state of band b, order o can be found at index [b+o∗bands]

6.102 gtfb_simple_bridge.cpp File Reference

Classes

- class **gtfb_simple_rt_t**
  Runtime configuration class of gtfb_simple_bridge plugin.
- class **gtfb_simple_t**
  Interface class of gtfb_simple_bridge plugin.

6.103 hann.cpp File Reference

Macros

- #define **PI** 3.14159265358979323846

Functions

- float ∗ **hannf** (const unsigned int N)
- double ∗ **hann** (const unsigned int N)

6.103.1 Macro Definition Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.103.1.1 PI `#define PI 3.14159265358979323846`

6.103.2 Function Documentation

6.103.2.1 hannf() `float* hannf ( const unsigned int N )`

6.103.2.2 hann() `double* hann ( const unsigned int N )`

6.104 hann.h File Reference

Functions

- float * **hannf** (const unsigned int N)
- double * **hann** (const unsigned int N)

6.104.1 Function Documentation

6.104.1.1 hannf() `float* hannf ( const unsigned int N )`

6.104.1.2 hann() `double* hann ( const unsigned int N )`

6.105 identity.cpp File Reference

Classes

- class **identity_t**

© 2005-2021 HörTech gGmbH, Oldenburg

6.106 ifftshift.cpp File Reference

Functions

- void **ifftshift** (**mha_wave_t** ∗spec)

6.106.1 Function Documentation

6.106.1.1 ifftshift()

```
void ifftshift (
    mha_wave_t * spec )
```

6.107 ifftshift.h File Reference

Functions

- void **ifftshift** (**mha_wave_t** ∗spec)

6.107.1 Function Documentation

6.107.1.1 ifftshift()

```
void ifftshift (
    mha_wave_t * spec )
```

6.108 iirfilter.cpp File Reference

Classes

- class **iirfilter_t**

6.109 level_matching.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **level_matching::level_matching_t::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

© 2005-2021 HörTech gGmbH, Oldenburg

6.109.1 Macro Definition Documentation

6.109.1.1 PATCH_VAR

```
#define PATCH_VAR(
    var ) patchbay.connect(&var.valuechanged, this, & level_matching::level↩
_matching_t::update_cfg)
```

6.109.1.2 INSERT_PATCH

```
#define INSERT_PATCH(
    var ) insert_member(var); PATCH_VAR(var)
```

6.110 level_matching.hh File Reference

Classes

- class **level_matching::channel_pair**
- class **level_matching::level_matching_config_t**
- class **level_matching::level_matching_t**

Namespaces

- **level_matching**

6.111 levelmeter.cpp File Reference

Classes

- class **levelmeter_t**

Macros

- #define **PASCALE** 93.979400086720374929

6.111.1 Macro Definition Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.111.1.1 PASCALE `#define PASCALE 93.979400086720374929`

6.112 lpc.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **lpc::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

Functions

- void **Levinson2** (unsigned int P, const std::vector< **mha_real_t** > &R, std::vector< **mha_real_t** > &A)

6.112.1 Macro Definition Documentation

6.112.1.1 PATCH_VAR

```
#define PATCH_VAR(
    var ) patchbay.connect(&var.valuechanged, this, & lpc::update_cfg)
```

6.112.1.2 INSERT_PATCH

```
#define INSERT_PATCH(
    var ) insert_member(var); PATCH_VAR(var)
```

6.112.2 Function Documentation

6.112.2.1 Levinson2()

```
void Levinson2 (
    unsigned int P,
    const std::vector< mha_real_t > & R,
    std::vector< mha_real_t > & A )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.113 lpc.h File Reference

Classes

- class **lpc_config**
- class **lpc**

6.114 lpc_bl_predictor.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **lpc_bl_predictor::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

6.114.1 Macro Definition Documentation

6.114.1.1 PATCH_VAR

```
#define PATCH_VAR(
    var ) patchbay.connect(&var.valuechanged, this, & lpc_bl_predictor↩
::update_cfg)
```

6.114.1.2 INSERT_PATCH

```
#define INSERT_PATCH(
    var )  insert_member(var);  PATCH_VAR(var)
```

6.115 lpc_bl_predictor.h File Reference

Classes

- class **lpc_bl_predictor_config**
- class **lpc_bl_predictor**

Macros

- #define **EPSILON** 1e-10

© 2005-2021 HörTech gGmbH, Oldenburg

6.115.1 Macro Definition Documentation

6.115.1.1 EPSILON `#define EPSILON 1e-10`

6.116 lpc_burg-lattice.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **lpc_burglattice::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

6.116.1 Macro Definition Documentation

6.116.1.1 PATCH_VAR

```
#define PATCH_VAR(
    var ) patchbay.connect(&var.valuechanged, this, & lpc_burglattice↩
::update_cfg)
```

6.116.1.2 INSERT_PATCH

```
#define INSERT_PATCH(
    var ) insert_member(var); PATCH_VAR(var)
```

6.117 lpc_burg-lattice.h File Reference

Classes

- class **lpc_burglattice_config**
- class **lpc_burglattice**

Macros

- #define **EPSILON** 1e-10

© 2005-2021 HörTech gGmbH, Oldenburg

6.117.1 Macro Definition Documentation

6.117.1.1 EPSILON `#define EPSILON 1e-10`

6.118 lsl2ac.cpp File Reference

6.119 lsl2ac.hh File Reference

Classes

- class **lsl2ac::save_var_t**
  LSL to AC bridge variable.
- class **lsl2ac::cfg_t**
  *Runtime configuration class of the **lsl2ac*** (p. 96) *plugin.*
- class **lsl2ac::lsl2ac_t**
  *Plugin class of **lsl2ac*** (p. 96).

Namespaces

- **lsl2ac**

Enumerations

- enum **lsl2ac::overrun_behavior** { **lsl2ac::overrun_behavior::Discard** =0, **lsl2ac↩ ::overrun_behavior::Ignore** }

6.120 matlab_wrapper.cpp File Reference

6.121 matlab_wrapper.hh File Reference

Classes

- struct **matlab_wrapper::types**< **T** >
- struct **matlab_wrapper::types**< **MHA_WAVEFORM** >
- struct **matlab_wrapper::types**< **MHA_SPECTRUM** >
- class **matlab_wrapper::matlab_wrapper_rt_cfg_t**
  Thin wrapper around the emxArray containing the user defined configuration variables.
- class **matlab_wrapper::callback**
  Utility class connecting a user_config_t instance to its corresponding configuration parser.
- class **matlab_wrapper::matlab_wrapper_t**
  Matlab wraper plugin interface class.
- class **matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t**
  Wrapper class around the matlab-generated library.

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **matlab_wrapper**
  Namespace where all classes of the matlab wrapper plugin live.

Macros

- #define **MHAPLUGIN_OVERLOAD_OUTDOMAIN**

6.121.1 Macro Definition Documentation

6.121.1.1 MHAPLUGIN_OVERLOAD_OUTDOMAIN `#define MHAPLUGIN_OVERLOAD_OUTDO↩MAIN`

6.122 matrixmixer.cpp File Reference

Classes

- class **matrixmixer::cfg_t**
- class **matrixmixer::matmix_t**

Namespaces

- **matrixmixer**

6.123 mconv.cpp File Reference

Classes

- class **mconv::MConv**

Namespaces

- **mconv**

© 2005-2021 HörTech gGmbH, Oldenburg

6.124 mha.cpp File Reference

Functions

- int **mhamain** (int argc, char ∗argv[ ])
- int **main** (int argc, char ∗argv[ ])

6.124.1 Function Documentation

6.124.1.1 mhamain()

```
int mhamain (
    int argc,
    char * argv[] )
```

6.124.1.2 main()

```
int main (
    int argc,
    char * argv[] )
```

6.125 mha.hh File Reference

common types for MHA kernel, MHA framework applications and external plugins

Classes

- struct **mha_complex_t**

  Type for complex floating point values.
- struct **mha_complex_test_array_t**

  *Several places in MHA rely on the fact that you can cast an array of **mha_complex_t*** (p. 745) *c[] to an array of mha_real_t r[] with r[0] == c[0].re r[1] == c[0].im r[2] == c[1].re ...*
- struct **mha_real_test_array_t**
- struct **mha_direction_t**

  Channel source direction structure.
- struct **mha_channel_info_t**

  Channel information structure.
- struct **mha_wave_t**

  Waveform signal structure.
- struct **mha_spec_t**

© 2005-2021 HörTech gGmbH, Oldenburg

- struct **mha_audio_descriptor_t**

  *Description of an audio fragment (planned as a replacement of **mhaconfig_t** (p. 851)).*
- struct **mha_audio_t**

  *An audio fragment in the openMHA (planned as a replacement of **mha_wave_t** (p. 840) and **mha_spec_t** (p. 794)).*
- struct **mhaconfig_t**

  MHA prepare configuration structure.
- struct **comm_var_t**

  Algorithm communication variable structure.
- struct **algo_comm_t**

  A reference handle for algorithm communication variables.

Macros

- #define **MHA_CALLBACK_TEST**(x)

  Test macro to compare function type definition and declaration.
- #define **MHA_CALLBACK_TEST_PREFIX**(prefix, x)
- #define **MHA_XSTRF**(x) **MHA_STRF**(x)
- #define **MHA_STRF**(x) #x
- #define **MHA_VERSION_MAJOR** 4

  Major version number of MHA.
- #define **MHA_VERSION_MINOR** 16

  Minor version number of MHA.
- #define **MHA_VERSION_RELEASE** 1

  Release number of MHA.
- #define **MHA_VERSION_BUILD** 0

  Build number of MHA (currently unused)
- #define **MHA_STRUCT_SIZEMATCH** (unsigned int)((sizeof(**mha_real_t**)==4)+2∗(sizeof(**mha_complex_t**)==8)+4∗(sizeof(**mha_wave_t**)==8+2∗sizeof(void∗))+8∗(sizeof(**mha_spec_t**)==8+2∗sizeof(void∗))+16∗(sizeof(**mhaconfig_t**)==24))

  Test number for structure sizes.
- #define **MHA_VERSION** (unsigned int)((**MHA_STRUCT_SIZEMATCH** | (**MHA_VERSION_RELEASE** << 8) | (**MHA_VERSION_MINOR** << 16) | (**MHA_VERSION_MAJOR** << 24)))

  Full version number of MHA kernel.
- #define **MHA_VERSION_STRING** **MHA_XSTRF**(**MHA_VERSION_MAJOR**) "." **MHA_XSTRF**(**MHA_VERSION_MINOR**)

  Version string of MHA kernel (major.minor)
- #define **MHA_RELEASE_VERSION_STRING** **MHA_XSTRF**(**MHA_VERSION_MAJOR**) "." **MHA_XSTRF**(**MHA_VERSION_MINOR**) "." **MHA_XSTRF**(**MHA_VERSION_RELEASE**)

  Version string of MHA kernel (major.minor.release)
- #define **MHA_WAVEFORM** 0
- #define **MHA_SPECTRUM** 1
- #define **MHA_DOMAIN_MAX** 2

© 2005-2021 HörTech gGmbH, Oldenburg

- #define **MHA_DOMAIN_UNKNOWN** **MHA_DOMAIN_MAX**
- #define **MHA_AC_UNKNOWN** 0
- #define **MHA_AC_CHAR** 1
- #define **MHA_AC_INT** 2
- #define **MHA_AC_MHAREAL** 3
- #define **MHA_AC_FLOAT** 4
- #define **MHA_AC_DOUBLE** 5
- #define **MHA_AC_MHACOMPLEX** 6
- #define **MHA_AC_VEC_FLOAT** 51
- #define **MHA_AC_USER** 1000

Typedefs

- typedef unsigned int **mha_domain_t**
- typedef float **mha_real_t**

  openMHA type for real numbers
- typedef void * **mha_fft_t**

  Handle for an FFT object.
- typedef struct **algo_comm_t** **algo_comm_t**
- typedef unsigned int(* **MHAGetVersion_t**) (void)
- typedef int(* **MHAInit_t**) (**algo_comm_t** algo_comm, const char *algo_name, void **h)
- typedef int(* **MHAPrepare_t**) (void *h, **mhaconfig_t** *cfg)
- typedef int(* **MHARelease_t**) (void *h)
- typedef void(* **MHADestroy_t**) (void *h)
- typedef int(* **MHASet_t**) (void *h, const char *cmd, char *retval, unsigned int len)
- typedef std::string(* **MHASetcpp_t**) (void *h, const std::string &command)
- typedef int(* **MHAProc_wave2wave_t**) (void *h, **mha_wave_t** *sIn, **mha_wave_t** **sOut)
- typedef int(* **MHAProc_wave2spec_t**) (void *h, **mha_wave_t** *sIn, **mha_spec_t** **sOut)
- typedef int(* **MHAProc_spec2wave_t**) (void *h, **mha_spec_t** *sIn, **mha_wave_t** **sOut)
- typedef int(* **MHAProc_spec2spec_t**) (void *h, **mha_spec_t** *sIn, **mha_spec_t** **sOut)

Variables

- const typedef char *(* **MHAStrError_t**)(void *h, int err)
- const typedef char *(* **MHAPluginDocumentation_t**)(void)
- const typedef char *(* **MHAPluginCategory_t**)(void)

6.125.1 Detailed Description

common types for MHA kernel, MHA framework applications and external plugins

© 2005-2021 HörTech gGmbH, Oldenburg

6.125.2 Macro Definition Documentation

6.125.2.1 MHA_CALLBACK_TEST

```
#define MHA_CALLBACK_TEST(
    x )
```

Test macro to compare function type definition and declaration.

6.125.2.2 MHA_CALLBACK_TEST_PREFIX

```
#define MHA_CALLBACK_TEST_PREFIX(
    prefix,
    x )
```

6.125.2.3 MHA_XSTRF

```
#define MHA_XSTRF(
    x )  MHA_STRF( x )
```

6.125.2.4 MHA_STRF

```
#define MHA_STRF(
    x ) #x
```

6.125.2.5 MHA_VERSION_MAJOR

```
#define MHA_VERSION_MAJOR 4
```

Major version number of MHA.

6.125.2.6 MHA_VERSION_MINOR

```
#define MHA_VERSION_MINOR 16
```

Minor version number of MHA.

© 2005-2021 HörTech gGmbH, Oldenburg

6.125.2.7 MHA_VERSION_RELEASE `#define MHA_VERSION_RELEASE 1`

Release number of MHA.

6.125.2.8 MHA_VERSION_BUILD `#define MHA_VERSION_BUILD 0`

Build number of MHA (currently unused)

6.125.2.9 MHA_STRUCT_SIZEMATCH `#define MHA_STRUCT_SIZEMATCH (unsigned int)((sizeof( mha_real_t)==4)+2*(sizeof( mha_complex_t)==8)+4*(sizeof( mha_wave_t)==8+2*sizeof(void*))+8*(sizeof mha_spec_t)==8+2*sizeof(void*))+16*(sizeof( mhaconfig_t)==24))`

Test number for structure sizes.

6.125.2.10 MHA_VERSION `#define MHA_VERSION (unsigned int)(( MHA_STRUCT_SIZEMA↩ TCH | ( MHA_VERSION_RELEASE << 8) | ( MHA_VERSION_MINOR << 16) | ( MHA_VERSION_MAJOR << 24)))`

Full version number of MHA kernel.

6.125.2.11 MHA_VERSION_STRING `#define MHA_VERSION_STRING MHA_XSTRF( MHA_VE↩ RSION_MAJOR) "." MHA_XSTRF( MHA_VERSION_MINOR)`

Version string of MHA kernel (major.minor)

6.125.2.12 MHA_RELEASE_VERSION_STRING `#define MHA_RELEASE_VERSION_STRI↩ NG MHA_XSTRF( MHA_VERSION_MAJOR) "." MHA_XSTRF( MHA_VERSION_MINOR) "." MHA_XSTRF( MHA_VERSION_RELEASE)`

Version string of MHA kernel (major.minor.release)

© 2005-2021 HörTech gGmbH, Oldenburg

6.125.2.13 MHA_WAVEFORM `#define MHA_WAVEFORM 0`

6.125.2.14 MHA_SPECTRUM `#define MHA_SPECTRUM 1`

6.125.2.15 MHA_DOMAIN_MAX `#define MHA_DOMAIN_MAX 2`

6.125.2.16 MHA_DOMAIN_UNKNOWN `#define MHA_DOMAIN_UNKNOWN` **MHA_DOMAIN_MAX**

6.125.2.17 MHA_AC_UNKNOWN `#define MHA_AC_UNKNOWN 0`

6.125.2.18 MHA_AC_CHAR `#define MHA_AC_CHAR 1`

6.125.2.19 MHA_AC_INT `#define MHA_AC_INT 2`

6.125.2.20 MHA_AC_MHAREAL `#define MHA_AC_MHAREAL 3`

6.125.2.21 MHA_AC_FLOAT `#define MHA_AC_FLOAT 4`

© 2005-2021 HörTech gGmbH, Oldenburg

6.125.2.22 MHA_AC_DOUBLE `#define MHA_AC_DOUBLE 5`

6.125.2.23 MHA_AC_MHACOMPLEX `#define MHA_AC_MHACOMPLEX 6`

6.125.2.24 MHA_AC_VEC_FLOAT `#define MHA_AC_VEC_FLOAT 51`

6.125.2.25 MHA_AC_USER `#define MHA_AC_USER 1000`

6.125.3 Typedef Documentation

6.125.3.1 mha_domain_t `typedef unsigned int` **`mha_domain_t`**

6.125.3.2 algo_comm_t `typedef struct` **`algo_comm_t`** **`algo_comm_t`**

6.125.3.3 MHAGetVersion_t `typedef unsigned int(* MHAGetVersion_t) (void)`

6.125.3.4 MHAInit_t `typedef int(* MHAInit_t) (` **`algo_comm_t`** `algo_comm, const char *algo_name, void **h)`

© 2005-2021 HörTech gGmbH, Oldenburg

6.125.3.5 MHAPrepare_t

`typedef int(* MHAPrepare_t) (void *h, mhaconfig_t *cfg)`

6.125.3.6 MHARelease_t

`typedef int(* MHARelease_t) (void *h)`

6.125.3.7 MHADestroy_t

`typedef void(* MHADestroy_t) (void *h)`

6.125.3.8 MHASet_t

`typedef int(* MHASet_t) (void *h, const char *cmd, char *retval, unsigned int len)`

6.125.3.9 MHASetcpp_t

`typedef std::string(* MHASetcpp_t) (void *h, const std::string &command)`

6.125.3.10 MHAProc_wave2wave_t

`typedef int(* MHAProc_wave2wave_t) (void *h, mha_wave_t *sIn, mha_wave_t **sOut)`

6.125.3.11 MHAProc_wave2spec_t

`typedef int(* MHAProc_wave2spec_t) (void *h, mha_wave_t *sIn, mha_spec_t **sOut)`

6.125.3.12 MHAProc_spec2wave_t

`typedef int(* MHAProc_spec2wave_t) (void *h, mha_spec_t *sIn, mha_wave_t **sOut)`

6.125.3.13 MHAProc_spec2spec_t

`typedef int(* MHAProc_spec2spec_t) (void *h, mha_spec_t *sIn, mha_spec_t **sOut)`

© 2005-2021 HörTech gGmbH, Oldenburg

6.125.4 Variable Documentation

6.125.4.1 MHAStrError_t `const typedef char*(* MHAStrError_t) (void *h, int err)`

6.125.4.2 MHAPluginDocumentation_t `const typedef char*(* MHAPluginDocumentation↩ _t) (void)`

6.125.4.3 MHAPluginCategory_t `const typedef char*(* MHAPluginCategory_t) (void)`

6.126 mha_algo_comm.cpp File Reference

Macros

- #define **AC_SUCCESS** 0
- #define **AC_INVALID_HANDLE** -1
- #define **AC_INVALID_NAME** -2
- #define **AC_STRING_TRUNCATED** -3
- #define **AC_INVALID_OUTPTR** -4
- #define **AC_TYPE_MISMATCH** -5
- #define **AC_DIM_MISMATCH** -6

Variables

- **algo_comm_t algo_comm_default**

6.126.1 Macro Definition Documentation

6.126.1.1 AC_SUCCESS `#define AC_SUCCESS 0`

© 2005-2021 HörTech gGmbH, Oldenburg

6.126.1.2 AC_INVALID_HANDLE `#define AC_INVALID_HANDLE -1`

6.126.1.3 AC_INVALID_NAME `#define AC_INVALID_NAME -2`

6.126.1.4 AC_STRING_TRUNCATED `#define AC_STRING_TRUNCATED -3`

6.126.1.5 AC_INVALID_OUTPTR `#define AC_INVALID_OUTPTR -4`

6.126.1.6 AC_TYPE_MISMATCH `#define AC_TYPE_MISMATCH -5`

6.126.1.7 AC_DIM_MISMATCH `#define AC_DIM_MISMATCH -6`

6.126.2 Variable Documentation

6.126.2.1 algo_comm_default `algo_comm_t algo_comm_default`

6.127 mha_algo_comm.h File Reference

Header file for Algorithm Communication.

© 2005-2021 HörTech gGmbH, Oldenburg

Classes

- class **MHA_AC::spectrum_t**

  *Insert a **MHASignal::spectrum_t** (p. 1248) class into the AC space.*
- class **MHA_AC::waveform_t**

  *Insert a **MHASignal::waveform_t** (p. 1263) class into the AC space.*
- class **MHA_AC::int_t**

  Insert a integer variable into the AC space.
- class **MHA_AC::float_t**

  Insert a float point variable into the AC space.
- class **MHA_AC::double_t**

  Insert a double precision floating point variable into the AC space.
- class **MHA_AC::stat_t**
- class **MHA_AC::ac2matrix_helper_t**
- class **MHA_AC::ac2matrix_t**

  Copy AC variable to a matrix.
- class **MHA_AC::acspace2matrix_t**

  Copy all or a subset of all numeric AC variables into an array of matrixes.

Namespaces

- **MHA_AC**

  Functions and classes for Algorithm Communication (AC) support.

Functions

- **mha_spec_t** **MHA_AC::get_var_spectrum** (**algo_comm_t** ac, const std::string &name)

  Convert an AC variable into a spectrum.
- **mha_wave_t** **MHA_AC::get_var_waveform** (**algo_comm_t** ac, const std::string &name)

  Convert an AC variable into a waveform.
- int **MHA_AC::get_var_int** (**algo_comm_t** ac, const std::string &name)

  Return value of an integer scalar AC variable.
- float **MHA_AC::get_var_float** (**algo_comm_t** ac, const std::string &name)

  Return value of an floating point scalar AC variable.
- std::vector< float > **MHA_AC::get_var_vfloat** (**algo_comm_t** ac, const std::string &name)

  Return value of an floating point vector AC variable as standard vector of floats.

6.127.1 Detailed Description

Header file for Algorithm Communication.

© 2005-2021 HörTech gGmbH, Oldenburg

6.128 mha_algo_comm.hh File Reference

Classes

- class **MHAKernel::comm_var_map_t**
- class **MHAKernel::algo_comm_class_t**

Namespaces

- **MHAKernel**

Macros

- #define **ALGO_COMM_ID_STR** "MFVK3jL5rmeus1XtggEI971aXCR/GU7RRehKz4k↩ Qtrg="

Functions

- algo_comm_class_t ∗ **MHAKernel::algo_comm_safe_cast** (void ∗)

Variables

- **algo_comm_t algo_comm_default**

6.128.1 Macro Definition Documentation

6.128.1.1 ALGO_COMM_ID_STR `#define ALGO_COMM_ID_STR "MFVK3jL5rmeus1XtggE↩ I971aXCR/GU7RRehKz4kQtrg="`

6.128.2 Variable Documentation

6.128.2.1 algo_comm_default `algo_comm_t algo_comm_default`

© 2005-2021 HörTech gGmbH, Oldenburg

6.129 mha_defs.h File Reference

Preprocessor definitions common to all MHA components.

Macros

- #define **__MHA_FUN__** __FUNC__
- #define **CHECK_EXPR**(x) {if(!(x)){throw **MHA_Error**(__FILE__,__LINE__,"The expression \"" #x "\" is invalid.");}}
- #define **CHECK_VAR**(x) {if(!(x)){throw **MHA_Error**(__FILE__,__LINE__,"The variable \"" #x "\" is not defined.");}}
- #define **__declspec**(p)
- #define **M_PI** 3.14159265358979323846

  Define pi if it is not defined yet.
- #define **MIN**(a, b) (((a)<(b))?(a):(b))

  Macro for minimum function.
- #define **MAX**(a, b) (((a)>(b))?(a):(b))

  Macro for maximum function.
- #define **MHA_EAR_LEFT** 0
- #define **MHA_EAR_RIGHT** 1
- #define **MHA_EAR_MAX** 2

6.129.1 Detailed Description

Preprocessor definitions common to all MHA components.

This file contains all preprocessor and type definitions which are common to all Master Hearing Aid components.

6.129.2 Macro Definition Documentation

6.129.2.1 __MHA_FUN__ `#define __MHA_FUN__ __FUNC__`

6.129.2.2 CHECK_EXPR `#define CHECK_EXPR(`

```
x ) {if(!(x)){throw MHA_Error(__FILE__,__LINE__,"The expression \"" #x "\" is invalid.");}}
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.129.2.3 CHECK_VAR

```
#define CHECK_VAR(
    x ) {if(!(x)){throw MHA_Error(__FILE__,__LINE__,"The variable \"" #x
"\" is not defined.");}}
```

6.129.2.4 __declspec

```
#define __declspec(
    p )
```

6.129.2.5 M_PI

```
#define M_PI 3.14159265358979323846
```

Define pi if it is not defined yet.

6.129.2.6 MIN

```
#define MIN(
    a,
    b ) (((a)<(b))?(a):(b))
```

Macro for minimum function.

6.129.2.7 MAX

```
#define MAX(
    a,
    b ) (((a)>(b))?(a):(b))
```

Macro for maximum function.

6.129.2.8 MHA_EAR_LEFT

```
#define MHA_EAR_LEFT 0
```

6.129.2.9 MHA_EAR_RIGHT

```
#define MHA_EAR_RIGHT 1
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.129.2.10 MHA_EAR_MAX `#define MHA_EAR_MAX 2`

6.130 mha_errno.c File Reference

Macros

- #define **STRLEN** 0x1000

Functions

- const char * **mha_strerror** (int mhaerrno)
- void **mha_set_user_error** (const char *str)

Variables

- char **next_except_str** [**STRLEN**] = ""
- const char * **cstr_strerror** [**MHA_ERR_USER**]

6.130.1 Macro Definition Documentation

6.130.1.1 STRLEN `#define STRLEN 0x1000`

6.130.2 Function Documentation

6.130.2.1 mha_strerror() `const char* mha_strerror ( int mhaerrno )`

6.130.2.2 mha_set_user_error() `void mha_set_user_error ( const char * str )`

© 2005-2021 HörTech gGmbH, Oldenburg

6.130.3 Variable Documentation

6.130.3.1 next_except_str `char next_except_str[ STRLEN] = ""`

6.130.3.2 cstr_strerror `const char* cstr_strerror[ MHA_ERR_USER]`

6.131 mha_errno.h File Reference

Macros

- #define **MHA_ERR_SUCCESS** 0
- #define **MHA_ERR_UNKNOWN** 1
- #define **MHA_ERR_INVALID_HANDLE** 2
- #define **MHA_ERR_NULL** 3
- #define **MHA_ERR_VARRANGE** 4
- #define **MHA_ERR_VARFMT** 5
- #define **MHA_ERR_USER** 10000

Functions

- const char * **mha_strerror** (int mhaerrno)
- void **mha_set_user_error** (const char *str)

6.131.1 Macro Definition Documentation

6.131.1.1 MHA_ERR_SUCCESS `#define MHA_ERR_SUCCESS 0`

6.131.1.2 MHA_ERR_UNKNOWN `#define MHA_ERR_UNKNOWN 1`

© 2005-2021 HörTech gGmbH, Oldenburg

6.131.1.3 MHA_ERR_INVALID_HANDLE `#define MHA_ERR_INVALID_HANDLE 2`

6.131.1.4 MHA_ERR_NULL `#define MHA_ERR_NULL 3`

6.131.1.5 MHA_ERR_VARRANGE `#define MHA_ERR_VARRANGE 4`

6.131.1.6 MHA_ERR_VARFMT `#define MHA_ERR_VARFMT 5`

6.131.1.7 MHA_ERR_USER `#define MHA_ERR_USER 10000`

6.131.2 Function Documentation

6.131.2.1 mha_strerror() `const char* mha_strerror ( int mhaerrno )`

6.131.2.2 mha_set_user_error() `void mha_set_user_error ( const char * str )`

6.132 mha_error.cpp File Reference

Implementation of openMHA error handling.

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **mha_error_helpers**

Functions

- unsigned **mha_error_helpers::digits** (unsigned n)

  Compute number of decimal digits required to represent an unsigned integer.
- unsigned **mha_error_helpers::snprintf_required_length** (const char ∗formatstring,...)

  snprintf_required_length Compute the number of bytes (excluding the terminating nul) required to store the result of an snprintf.
- void **mha_debug** (const char ∗fmt,...)

6.132.1 Detailed Description

Implementation of openMHA error handling.

This file forms a seperate library.

6.132.2 Function Documentation

6.132.2.1 mha_debug()

```
void mha_debug (
    const char * fmt,
    ... )
```

6.133 mha_error.hh File Reference

Classes

- class **MHA_Error**

  Error reporting exception class.

Namespaces

- **mha_error_helpers**

© 2005-2021 HörTech gGmbH, Oldenburg

Macros

- #define **Getmsg**(e) ((e).get_msg())
- #define **MHA_ErrorMsg**(x) **MHA_Error**(__FILE__,__LINE__,"%s",x)

  Throw an openMHA error with a text message.
- #define **MHA_assert**(x) if(!(x)) throw **MHA_Error**(__FILE__,__LINE__,"\"%s\" is false.",#x)

  *Assertion macro, which throws an **MHA_Error** (p. 764).*
- #define **MHA_assert_equal**(a, b) if(a != b) throw **MHA_Error**(__FILE__,__LINE__↩ ,"\"%s == %s\" is false (%s = %g, %s = %g).",#a,#b,#a,(double)(a),#b,(double)(b))

  *Equality assertion macro, which throws an **MHA_Error** (p. 764) with the values.*

Functions

- void **mha_debug** (const char ∗fmt,...) __attribute__((__format__(printf

  Print an info message (stderr on Linux, OutputDebugString in Windows).
- unsigned **mha_error_helpers::digits** (unsigned n)

  Compute number of decimal digits required to represent an unsigned integer.
- unsigned **mha_error_helpers::snprintf_required_length** (const char ∗formatstring,...)

  snprintf_required_length Compute the number of bytes (excluding the terminating nul) required to store the result of an snprintf.

6.133.1 Macro Definition Documentation

6.133.1.1 Getmsg `#define Getmsg( e ) ((e).get_msg())`

6.134 mha_event_emitter.h File Reference

Classes

- class **MHAEvents::connector_base_t**
- class **MHAEvents::emitter_t**

  Class for emitting openMHA events.

Namespaces

- **MHAEvents**

  Collection of event handling classes.

© 2005-2021 HörTech gGmbH, Oldenburg

6.135 mha_events.cpp File Reference

6.136 mha_events.h File Reference

Classes

- class **MHAEvents::connector_t**< **receiver_t** >
- class **MHAEvents::patchbay_t**< **receiver_t** >

  Patchbay which connects any event emitter with any member function of the parameter class.

Namespaces

- **MHAEvents**

  Collection of event handling classes.

6.137 mha_fftfb.cpp File Reference

Classes

- class **MHAOvlFilter::barkscale::hz2bark_t**
- class **MHAOvlFilter::barkscale::bark2hz_t**

Namespaces

- **MHAOvlFilter**

  Namespace for overlapping FFT based filter bank classes and functions.
- **MHAOvlFilter::barkscale**
- **MHAOvlFilter::FreqScaleFun**

  Transform functions from linear scale in Hz to new frequency scales.
- **MHAOvlFilter::ShapeFun**

  Shape functions for overlapping filters.

Macros

- #define **BARKSCALE_ENTRIES** 50

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- **mha_real_t MHAOvlFilter::FreqScaleFun::hz2hz** (**mha_real_t** x)
  Dummy scale transformation Hz to Hz.
- **mha_real_t MHAOvlFilter::FreqScaleFun::hz2khz** (**mha_real_t** x)
- **mha_real_t MHAOvlFilter::FreqScaleFun::hz2octave** (**mha_real_t** x)
- **mha_real_t MHAOvlFilter::FreqScaleFun::hz2third_octave** (**mha_real_t** x)
- **mha_real_t MHAOvlFilter::FreqScaleFun::hz2bark** (**mha_real_t** x)
  Transformation to bark scale.
- **mha_real_t MHAOvlFilter::FreqScaleFun::hz2bark_analytic** (**mha_real_t**)
- **mha_real_t MHAOvlFilter::FreqScaleFun::hz2erb** (**mha_real_t**)
- **mha_real_t MHAOvlFilter::FreqScaleFun::hz2erb_glasberg1990** (**mha_real_t**)
- **mha_real_t MHAOvlFilter::FreqScaleFun::hz2log** (**mha_real_t** x)
  Third octave frequency scale.
- **mha_real_t MHAOvlFilter::FreqScaleFun::inv_scale** (**mha_real_t**, **mha_real_t**(∗)(**mha_real_t**))
- **mha_real_t MHAOvlFilter::ShapeFun::rect** (**mha_real_t** x)
  Filter shape function for rectangular filters.
- **mha_real_t MHAOvlFilter::ShapeFun::linear** (**mha_real_t** x)
  Filter shape function for sawtooth filters.
- **mha_real_t MHAOvlFilter::ShapeFun::hann** (**mha_real_t** x)
  Filter shape function for hanning shaped filters.
- **mha_real_t MHAOvlFilter::ShapeFun::expflt** (**mha_real_t**)
- **mha_real_t MHAOvlFilter::ShapeFun::gauss** (**mha_real_t**)
- **mha_real_t filtershapefun** (**mha_real_t** f, **MHAOvlFilter::band_descriptor_t** b, **mha_real_t** plateau)
  Transform the test frequency into the relative position on the filter flank of the given frequency band.

Variables

- **mha_real_t MHAOvlFilter::barkscale::vfreq** [**BARKSCALE_ENTRIES**]
- **mha_real_t MHAOvlFilter::barkscale::vbark** [**BARKSCALE_ENTRIES**]

6.137.1 Macro Definition Documentation

6.137.1.1 BARKSCALE_ENTRIES `#define BARKSCALE_ENTRIES 50`

6.137.2 Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.137.2.1 filtershapefun()

```
mha_real_t filtershapefun (
    mha_real_t f,
    MHAOvlFilter::band_descriptor_t b,
    mha_real_t plateau )
```

Transform the test frequency into the relative position on the filter flank of the given frequency band.

Parameters

f	Test frequency in units corresponding to the chosen frequency scale
b	Descriptor of a single filter bank band: E.g. contains center frequencies of this and the two adjacent bands, and the crossover ("edge") frequencies of this band.
plateau	For non-rectangular filter shapes, specifies what frequency portion of the band around its center frequency should have no attenuation applied.

Precondition

0 <= plateau <= 1

Returns

The position of frequency f on the filter flank as follows: A returned position of 0 means that f is equal to the band's center frequency, or should be treated the same as the center frequency (i.e. is within the band's plateau). A returned position of -1 means that f is <= the lowest frequency of the filter flank (or is an even lower frequency). A returned value of -0.5 means that f is equal to the lower edge frequency. Positive returned values have equivalent meanings for the high half of the filter flank.

6.138 mha_fttfb.hh File Reference

Classes

- class **MHAOvlFilter::band_descriptor_t**
- class **MHAOvlFilter::scale_var_t**
- class **MHAOvlFilter::fscale_t**
- class **MHAOvlFilter::fscale_bw_t**
- class **MHAOvlFilter::fttfb_vars_t**

  Set of configuration variables for FFT-based overlapping filters.
- class **MHAOvlFilter::fspacing_t**

  Class for frequency spacing, used by filterbank shape generator class.
- class **MHAOvlFilter::fttfb_t**

  FFT based overlapping filter bank.
- class **MHAOvlFilter::overlap_save_filterbank_t**

  *A time-domain minimal phase filter bank with frequency shapes from **MHAOvlFilter::fttfb_t** (p. 1006).*
- class **MHAOvlFilter::overlap_save_filterbank_t::vars_t**
- class **MHAOvlFilter::overlap_save_filterbank_analytic_t**
- class **MHAOvlFilter::fttfb_ac_info_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **MHAOvlFilter**
  Namespace for overlapping FFT based filter bank classes and functions.

Typedefs

- typedef **mha_real_t**() **MHAOvlFilter::scale_fun_t**(**mha_real_t**)

6.139 mha_fifo.cpp File Reference

6.140 mha_fifo.h File Reference

Classes

- class **mha_fifo_t**< **T** >
  A FIFO class.
- class **mha_fifo_lf_t**< **T** >
  A lock-free FIFO class for transferring data from a producer thread to a consumer thread.
- class **mha_drifter_fifo_t**< **T** >
  A FIFO class for blocksize adaptation without Synchronization.
- class **mha_fifo_thread_platform_t**
  Abstract base class for synchronizing multithreaded (producer/consumer) fifo operations.
- class **mha_fifo_posix_threads_t**
- class **mha_fifo_thread_guard_t**
  Simple Mutex Guard Class.
- class **mha_fifo_lw_t**< **T** >
  This FIFO uses locks to synchronize access.
- class **mha_dblbuf_t**< **FIFO** >
  The doublebuffer adapts blocksizes between an outer process, which provides input data and takes output data, and an inner process, which processes the input signal and generates output data using a different block size than the outer process.
- class **mha_rt_fifo_element_t**< **T** >
  *Object wrapper for **mha_rt_fifo_t** (p. 790).*
- class **mha_rt_fifo_t**< **T** >
  Template class for thread safe, half real time safe fifo without explixit locks.

Macros

- #define **mha_fifo_thread_platform_implementation_t** **mha_fifo_posix_threads_t**

© 2005-2021 HörTech gGmbH, Oldenburg

6.140.1 Macro Definition Documentation

6.140.1.1 mha_fifo_thread_platform_implementation_t `#define mha_fifo_thread_platform_implementation_t` **mha_fifo_posix_threads_t**

6.141 mha_filter.cpp File Reference

Functions

- std::vector< **mha_real_t** > **diff_coeffs** ()

6.141.1 Function Documentation

6.141.1.1 diff_coeffs() `std::vector<` **mha_real_t**`> diff_coeffs ( )`

6.142 mha_filter.hh File Reference

Header file for IIR filter classes.

Classes

- class **MHAFilter::filter_t**
  Generic IIR filter class.
- class **MHAFilter::diff_t**
  Differentiator class (non-normalized)
- class **MHAFilter::o1_ar_filter_t**
  First order attack-release lowpass filter.
- class **MHAFilter::o1flt_lowpass_t**
  First order low pass filter.
- class **MHAFilter::o1flt_maxtrack_t**
  First order maximum tracker.
- class **MHAFilter::o1flt_mintrack_t**
  First order minimum tracker.

© 2005-2021 HörTech gGmbH, Oldenburg

- class **MHAFilter::iir_filter_state_t**
- class **MHAFilter::iir_filter_t**

  IIR filter class wrapper for integration into parser structure.
- class **MHAFilter::adapt_filter_state_t**
- class **MHAFilter::adapt_filter_param_t**
- class **MHAFilter::adapt_filter_t**

  Adaptive filter.
- class **MHAFilter::fftfilter_t**

  FFT based FIR filter implementation.
- class **MHAFilter::fftfilterbank_t**

  FFT based FIR filterbank implementation.
- struct **MHAFilter::transfer_function_t**

  a structure containing a source channel number, a target channel number, and an impulse response.
- struct **MHAFilter::transfer_matrix_t**

  A sparse matrix of transfer function partitionss.
- class **MHAFilter::partitioned_convolution_t**

  A filter class for partitioned convolution.
- struct **MHAFilter::partitioned_convolution_t::index_t**

  Bookkeeping class.
- class **MHAFilter::smoothspec_t**

  Smooth spectral gains, create a windowed impulse response.
- class **MHAFilter::resampling_filter_t**

  Hann shaped low pass filter for resampling.
- class **MHAFilter::polyphase_resampling_t**

  A class that performs polyphase resampling.
- class **MHAFilter::blockprocessing_polyphase_resampling_t**

  A class that does polyphase resampling and takes into account block processing.
- class **MHAFilter::iir_ord1_real_t**

  First order recursive filter.

Namespaces

- **MHAFilter**

  Namespace for IIR and FIR filter classes.

Functions

- template<typename T , typename std::enable_if< std::is_floating_point< T >::value, T >::type * = nullptr> void **MHAFilter::make_friendly_number** (T &x)
- void **MHAFilter::o1_lp_coeffs** (const **mha_real_t** tau, const **mha_real_t** fs, **mha_real_t** &c1, **mha_real_t** &c2)

  Set first order filter coefficients from time constant and sampling rate.

© 2005-2021 HörTech gGmbH, Oldenburg

- void **MHAFilter::butter_stop_ord1** (double ∗A, double ∗B, double f1, double f2, double fs)

  Setup a first order butterworth band stop filter.
- std::vector< float > **MHAFilter::fir_lp** (float f_pass_, float f_stop_, float fs_, unsigned order_)

  Setup a nth order fir low pass filter.
- **MHASignal::waveform_t** ∗ **MHAFilter::spec2fir** (const **mha_spec_t** ∗spec, const unsigned int fftlen, const **MHAWindow::base_t** &window, const bool minphase)

  Create a windowed impulse response/FIR filter coefficients from a spectrum.
- unsigned **MHAFilter::gcd** (unsigned a, unsigned b)

  greatest common divisor
- double **MHAFilter::sinc** (double x)

  sin(x)/x function, coping with x=0.
- std::pair< unsigned, unsigned > **MHAFilter::resampling_factors** (float source_↩ sampling_rate, float target_sampling_rate, float factor=1.0f)

  Computes rational resampling factor from two sampling rates.

6.142.1 Detailed Description

Header file for IIR filter classes.

6.143 mha_generic_chain.cpp File Reference

Functions

- void **mhaconfig_compare** (**mhaconfig_t** req, **mhaconfig_t** avail, const char ∗cpref)

6.143.1 Function Documentation

6.143.1.1 mhaconfig_compare()

```
void mhaconfig_compare (
    mhaconfig_t req,
    mhaconfig_t avail,
    const char * cpref )
```

6.144 mha_generic_chain.h File Reference

Classes

- class **mhachain::plugs_t**
- class **mhachain::chain_base_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **mhachain**

Macros

- #define **MHAPLUGIN_OVERLOAD_OUTDOMAIN**

6.144.1 Macro Definition Documentation

6.144.1.1 MHAPLUGIN_OVERLOAD_OUTDOMAIN `#define MHAPLUGIN_OVERLOAD_OUTDO↩ MAIN`

6.145 mha_git_commit_hash.cpp File Reference

Macros

- #define **GITCOMMITHASH** "independent-plugin-build"

Variables

- const char * **mha_git_commit_hash**
  store git commit hash in every binary plgin to support reproducible research

6.145.1 Macro Definition Documentation

6.145.1.1 GITCOMMITHASH `#define GITCOMMITHASH "independent-plugin-build"`

6.145.2 Variable Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.145.2.1 mha_git_commit_hash `const char* mha_git_commit_hash`

store git commit hash in every binary plgin to support reproducible research

6.146 mha_git_commit_hash.hh File Reference

Variables

- const char * **mha_git_commit_hash**
  store git commit hash in every binary plgin to support reproducible research

6.146.1 Variable Documentation

6.146.1.1 mha_git_commit_hash `const char* mha_git_commit_hash`

store git commit hash in every binary plgin to support reproducible research

6.147 mha_io_ifc.h File Reference

Macros

- #define **MHAIO_DOCUMENTATION_PREFIX**(prefix, cat, doc)
- #define **MHAIO_DOCUMENTATION**(plugname, cat, doc) **MHAIO_DOCUMENTATIO↩N_PREFIX**(MHA_STATIC_ ## plugname ## _,cat,doc)

Typedefs

- typedef int(* **IOProcessEvent_t**) (void *handle, **mha_wave_t** *sIn, **mha_wave_t** **s↩Out)
  Event handler for signal stream.
- typedef void(* **IOStoppedEvent_t**) (void *handle, int proc_err, int io_err)
  Event handler for stop event.
- typedef void(* **IOStartedEvent_t**) (void *handle)
  Event handler for start event.
- typedef int(* **IOInit_t**) (int fragsize, float samplerate, **IOProcessEvent_t** proc_event, void *proc_handle, **IOStartedEvent_t** start_event, void *start_handle, **IOStopped↩Event_t** stop_event, void *stop_handle, void **handle)
- typedef int(* **IOPrepare_t**) (void *handle, int num_inchannels, int num_outchannels)
- typedef int(* **IOStart_t**) (void *handle)
- typedef int(* **IOStop_t**) (void *handle)
- typedef int(* **IORelease_t**) (void *handle)
- typedef int(* **IOSetVar_t**) (void *handle, const char *cmd, char *retval, unsigned int len)
- typedef void(* **IODestroy_t**) (void *handle)

© 2005-2021 HörTech gGmbH, Oldenburg

Variables

- const typedef char *(* **IOStrError_t**)(void *handle, int err)

6.147.1 Macro Definition Documentation

6.147.1.1 MHAIO_DOCUMENTATION_PREFIX

```
#define MHAIO_DOCUMENTATION_PREFIX(
    prefix,
    cat,
    doc )
```

6.147.1.2 MHAIO_DOCUMENTATION

```
#define MHAIO_DOCUMENTATION(
    plugname,
    cat,
    doc )  MHAIO_DOCUMENTATION_PREFIX(MHA_STATIC_ ## plugname ## _,cat,doc)
```

6.147.2 Typedef Documentation

6.147.2.1 IOProcessEvent_t

```
typedef int(* IOProcessEvent_t) (void *handle,  mha_↩
wave_t *sIn,  mha_wave_t **sOut)
```

Event handler for signal stream.

This event handler needs to be realtime compatible. All signal path processing will be performed in this callback.

6.147.2.2 IOStoppedEvent_t

```
typedef void(* IOStoppedEvent_t) (void *handle, int
proc_err, int io_err)
```

Event handler for stop event.

This event handler needs to be realtime compatible. The function must return immediatly.

© 2005-2021 HörTech gGmbH, Oldenburg

6.147.2.3 IOStartedEvent_t `typedef void(* IOStartedEvent_t) (void *handle)`

Event handler for start event.

This event handler needs to be realtime compatible. The function must return immediatly.

6.147.2.4 IOInit_t `typedef int(* IOInit_t) (int fragsize, float samplerate, IOProcessEvent_t proc_event, void *proc_handle, IOStartedEvent_t start_event, void *start_handle, IOStoppedEvent_t stop_event, void *stop_handle, void **handle)`

6.147.2.5 IOPrepare_t `typedef int(* IOPrepare_t) (void *handle, int num_inchannels, int num_outchannels)`

6.147.2.6 IOStart_t `typedef int(* IOStart_t) (void *handle)`

6.147.2.7 IOStop_t `typedef int(* IOStop_t) (void *handle)`

6.147.2.8 IORelease_t `typedef int(* IORelease_t) (void *handle)`

6.147.2.9 IOSetVar_t `typedef int(* IOSetVar_t) (void *handle, const char *cmd, char *retval, unsigned int len)`

6.147.2.10 IODestroy_t `typedef void(* IODestroy_t) (void *handle)`

6.147.3 Variable Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.147.3.1 IOStrError_t `const typedef char*(* IOStrError_t) (void *handle, int err)`

6.148 mha_io_utils.cpp File Reference

6.149 mha_io_utils.hh File Reference

Namespaces

- **mhaioutils**

Functions

- template<typename T >
  T **mhaioutils::to_int_clamped** (float val)

6.150 mha_multisrc.cpp File Reference

Namespaces

- **MHAMultiSrc**
  Collection of classes for selecting audio chunks from multiple sources.

6.151 mha_multisrc.h File Reference

Classes

- class **MHAMultiSrc::channel_t**
- class **MHAMultiSrc::channels_t**
- class **MHAMultiSrc::base_t**
  Base class for source selection.
- class **MHAMultiSrc::waveform_t**
- class **MHAMultiSrc::spectrum_t**

Namespaces

- **MHAMultiSrc**
  Collection of classes for selecting audio chunks from multiple sources.

© 2005-2021 HörTech gGmbH, Oldenburg

6.152 mha_os.cpp File Reference

Functions

- bool **mha_hasenv** (const std::string &envvar)
  Checks if environment variable exists.
- std::string **mha_getenv** (const std::string &envvar)
  Get value of environment variable.
- void **mha_delenv** (const std::string &envvar)
  Deletes environment variable from process environment if it exists.
- int **mha_setenv** (const std::string &envvar, const std::string & **value**)
  Set value of environment variable.
- std::list< std::string > **mha_library_paths** ()
- std::list< std::string > **list_dir** (const std::string &path, const std::string &pattern)

6.152.1 Function Documentation

6.152.1.1 mha_hasenv() `bool mha_hasenv ( const std::string & envvar )`

Checks if environment variable exists.

Parameters

envvar	Name of environment variable to check

Returns

true if the environment has a variable of this name

6.152.1.2 mha_getenv() `std::string mha_getenv ( const std::string & envvar )`

Get value of environment variable.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

envvar	Name of environment variable to retrieve

Returns

content of environment variable if it exists, empty string if the environment variable does not exist

6.152.1.3 mha_delenv()

```
void mha_delenv (
    const std::string & envvar )
```

Deletes environment variable from process environment if it exists.

Parameters

envvar	Name of environment variable to delete

6.152.1.4 mha_setenv()

```
int mha_setenv (
    const std::string & envvar,
    const std::string & value )
```

Set value of environment variable.

Parameters

envvar	Name of environment variable to set
value	New content for environment variable

Returns

error code: 0 on success, OS dependent error code on failure

6.152.1.5 mha_library_paths()

```
std::list<std::string> mha_library_paths ( )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.152.1.6 list_dir()

```
std::list<std::string> list_dir (
    const std::string & path,
    const std::string & pattern )
```

6.153 mha_os.h File Reference

Classes

- class **mha_stash_environment_variable_t**

  This class changes the value of an environment variable when constructed and restores the original state of the environment variable when destroyed.
- class **dynamiclib_t**

  Wrapper class around a shared library.
- class **pluginlib_t**

  *Specialisation of **dynamiclib_t** (p. 446) for mha plugin libraries.*

Macros

- #define **mha_loadlib**(x) dlopen(x,RTLD_NOW)
- #define **mha_freelib**(x) dlclose(x)
- #define **mha_freelib_success**(x) (x == 0)
- #define **mha_getlibfun**(h, x) x ## _cb = (x ## _t)dlsym(h,#x)
- #define **mha_getlibfun_checked**(h, x) x ## _cb = (x ## _t)dlsym(h,#x);if(! x ## _cb) throw **MHA_Error**(__FILE__,__LINE__,"Function " #x " is undefined.")
- #define **mha_loadlib_error**(x) dlerror()
- #define **mha_lib_extension** ".so"
- #define **mha_msleep**(milliseconds) usleep((milliseconds)∗1000)
- #define **FMTsz** "%zu"

  printf modifier to print integers of type size_t
- #define **MHA_RESOLVE**(h, t) t ## _cb = (t ## _t)(h->resolve(#t))
- #define **MHA_RESOLVE_CHECKED**(h, t) t ## _cb = (t ## _t)(h->resolve_checked(#t))

Typedefs

- typedef void ∗ **mha_libhandle_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- std::string **mha_getenv** (const std::string &envvar)
  Get value of environment variable.
- bool **mha_hasenv** (const std::string &envvar)
  Checks if environment variable exists.
- int **mha_setenv** (const std::string &envvar, const std::string & **value**)
  Set value of environment variable.
- void **mha_delenv** (const std::string &envvar)
  Deletes environment variable from process environment if it exists.
- std::list< std::string > **mha_library_paths** ()
- std::list< std::string > **list_dir** (const std::string &path, const std::string &pattern)
- void **mha_hton** (float *data, unsigned int len)
- void **mha_ntoh** (float *data, unsigned int len)
- void **mha_hton** (uint32_t *data, unsigned int len)
- void **mha_ntoh** (uint32_t *data, unsigned int len)
- void **mha_hton** (int32_t *data, unsigned int len)
- void **mha_ntoh** (int32_t *data, unsigned int len)

6.153.1 Macro Definition Documentation

6.153.1.1 mha_loadlib

```
#define mha_loadlib(
x ) dlopen(x,RTLD_NOW)
```

6.153.1.2 mha_freelib

```
#define mha_freelib(
x ) dlclose(x)
```

6.153.1.3 mha_freelib_success

```
#define mha_freelib_success(
x ) (x == 0)
```

6.153.1.4 mha_getlibfun

```
#define mha_getlibfun(
h,
x ) x ## _cb = (x ## _t)dlsym(h,#x)
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.153.1.5 mha_getlibfun_checked

```
#define mha_getlibfun_checked(
    h,
    x ) x ## _cb = (x ## _t)dlsym(h,#x);if(!  x ## _cb) throw  MHA_Error(_↩
_FILE__,__LINE__,"Function " #x " is undefined.")
```

6.153.1.6 mha_loadlib_error

```
#define mha_loadlib_error(
    x ) dlerror()
```

6.153.1.7 mha_lib_extension

```
#define mha_lib_extension ".so"
```

6.153.1.8 mha_msleep

```
#define mha_msleep(
    milliseconds ) usleep((milliseconds)*1000)
```

6.153.1.9 FMTsz

```
#define FMTsz "%zu"
```

printf modifier to print integers of type size_t

6.153.1.10 MHA_RESOLVE

```
#define MHA_RESOLVE(
    h,
    t ) t ## _cb = (t ## _t)(h->resolve(#t))
```

6.153.1.11 MHA_RESOLVE_CHECKED

```
#define MHA_RESOLVE_CHECKED(
    h,
    t ) t ## _cb = (t ## _t)(h->resolve_checked(#t))
```

6.153.2 Typedef Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.153.2.1 mha_libhandle_t `typedef void* mha_libhandle_t`

6.153.3 Function Documentation

6.153.3.1 mha_getenv() `std::string mha_getenv ( const std::string & envvar )`

Get value of environment variable.

Parameters

envvar	Name of environment variable to retrieve

Returns

content of environment variable if it exists, empty string if the environment variable does not exist

6.153.3.2 mha_hasenv() `bool mha_hasenv ( const std::string & envvar )`

Checks if environment variable exists.

Parameters

envvar	Name of environment variable to check

Returns

true if the environment has a variable of this name

6.153.3.3 mha_setenv() `int mha_setenv ( const std::string & envvar, const std::string & value )`

Set value of environment variable.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

envvar	Name of environment variable to set
value	New content for environment variable

Returns

error code: 0 on success, OS dependent error code on failure

6.153.3.4 mha_delenv()

```
void mha_delenv (
    const std::string & envvar )
```

Deletes environment variable from process environment if it exists.

Parameters

envvar	Name of environment variable to delete

6.153.3.5 mha_library_paths()

```
std::list<std::string> mha_library_paths ( )
```

6.153.3.6 list_dir()

```
std::list<std::string> list_dir (
    const std::string & path,
    const std::string & pattern )
```

6.153.3.7 mha_hton() [1/3]

```
void mha_hton (
    float * data,
    unsigned int len ) [inline]
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.153.3.8 mha_ntoh() [1/3]

```
void mha_ntoh (
    float * data,
    unsigned int len ) [inline]
```

6.153.3.9 mha_hton() [2/3]

```
void mha_hton (
    uint32_t * data,
    unsigned int len ) [inline]
```

6.153.3.10 mha_ntoh() [2/3]

```
void mha_ntoh (
    uint32_t * data,
    unsigned int len ) [inline]
```

6.153.3.11 mha_hton() [3/3]

```
void mha_hton (
    int32_t * data,
    unsigned int len ) [inline]
```

6.153.3.12 mha_ntoh() [3/3]

```
void mha_ntoh (
    int32_t * data,
    unsigned int len ) [inline]
```

6.154 mha_parser.cpp File Reference

Namespaces

- **MHAParser**
  Name space for the openMHA-Parser configuration language.
- **MHAParser::StrCnv**
  String converter namespace.

Macros

- #define **MHAPLATFORM** "undefined-linux"

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- int **MHAParser::get_precision** ()
- int **MHAParser::StrCnv::num_brackets** (const std::string &s)
  count number of brackets
- int **MHAParser::StrCnv::bracket_balance** (const std::string &s)
- static std::ostream & **write_float** (std::ostream &o, const float &f)
- static std::string **parse_1_float** (const std::string &s, **mha_real_t** &v)
  This internal function parses a floating point number from the beginning of a string.
- static void **check_parenthesis_complex** (const std::string &str)
  This function checks for unbalanced parenthesis in the string containing complex number.
- static int **check_sign_complex** (const std::string &str)
  This function checks for valid sign (b/w real & img.
- static std::string **parse_1_complex** (const std::string &s, **mha_complex_t** &v)
  This internal function parses a complex number from the beginning of a string.

6.154.1 Macro Definition Documentation

6.154.1.1 MHAPLATFORM `#define MHAPLATFORM "undefined-linux"`

6.154.2 Function Documentation

6.154.2.1 write_float()

```
static std::ostream& write_float (
    std::ostream & o,
    const float & f ) [inline], [static]
```

6.154.2.2 parse_1_float()

```
static std::string parse_1_float (
    const std::string & s,
    mha_real_t & v ) [static]
```

This internal function parses a floating point number from the beginning of a string.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

s	The string to parse

Precondition

s.size() $>$ 0

Parameters

v	The float variable to fill with a value

Returns

The rest of the string.

6.154.2.3 check_parenthesis_complex() `static void check_parenthesis_complex ( const std::string & str ) [static]`

This function checks for unbalanced parenthesis in the string containing complex number.

Parameters

str	The string to check. This function returns normally only when the string starts with an opening bracket and ends with a closing bracket.

Precondition

str.size() $>$ 0

Exceptions

MHA_ErrorMsg	This function raises an error with an appropriate error message if the string does not start with an opening bracket '(' or if it does not end with a closing bracket ')'.

6.154.2.4 check_sign_complex() `static int check_sign_complex (`

© 2005-2021 HörTech gGmbH, Oldenburg

```
const std::string & str ) [static]
```

This function checks for valid sign (b/w real & img.

parts of complex number). It also checks if real part is missing in complex number.

Parameters

str	String containing sign and onward imaginary part.

Precondition

str.size() $>$ 0

Exceptions

This	function raises an error if wrong sign (other than '+' or '-') is found. It also raises error, if it finds 'i' instead of sign. This can happen when real part is missing in complex number and 'i' (of imaginary part) is found where sign was expected.

Returns

+1 for '+' sign and -1 for '-' sign

6.154.2.5 parse_1_complex()

```
static std::string parse_1_complex (
const std::string & s,
mha_complex_t & v ) [static]
```

This internal function parses a complex number from the beginning of a string.

Parameters

s	The string to parse
v	The complex variable to fill with a value

Exceptions

MHA_ErrorMsg	This function raises an error if s does not have characters except spaces, tabs etc

© 2005-2021 HörTech gGmbH, Oldenburg

Returns

The rest of the string.

6.155 mha_parser.hh File Reference

Header file for the MHA-Parser script language.

Classes

- class **MHAParser::keyword_list_t**
  Keyword list class.
- class **MHAParser::expression_t**
- class **MHAParser::entry_t**
- class **MHAParser::base_t**
  Base class for all parser items.
- class **MHAParser::base_t::replace_t**
- class **MHAParser::parser_t**
  Parser node class.
- class **MHAParser::c_ifc_parser_t**
- class **MHAParser::monitor_t**
  Base class for monitors and variable nodes.
- class **MHAParser::variable_t**
  Base class for variable nodes.
- class **MHAParser::range_var_t**
  Base class for all variables with a numeric value range.
- class **MHAParser::kw_t**
  Variable with keyword list value.
- class **MHAParser::string_t**
  Variable with a string value.
- class **MHAParser::vstring_t**
  Vector variable with string values.
- class **MHAParser::bool_t**
  Variable with a boolean value ("yes"/"no")
- class **MHAParser::int_t**
  Variable with integer value.
- class **MHAParser::float_t**
  Variable with float value.
- class **MHAParser::complex_t**
  Variable with complex value.
- class **MHAParser::vint_t**
  Variable with vector<int> value.
- class **MHAParser::vfloat_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Vector variable with float value.

- class **MHAParser::vcomplex_t**

  Vector variable with complex value.
- class **MHAParser::mint_t**

  Matrix variable with int value.
- class **MHAParser::mfloat_t**

  Matrix variable with float value.
- class **MHAParser::mcomplex_t**

  Matrix variable with complex value.
- class **MHAParser::int_mon_t**

  Monitor variable with int value.
- class **MHAParser::bool_mon_t**

  Monitor with string value.
- class **MHAParser::string_mon_t**

  Monitor with string value.
- class **MHAParser::vstring_mon_t**

  Vector of monitors with string value.
- class **MHAParser::vint_mon_t**

  Vector of ints monitor.
- class **MHAParser::mint_mon_t**

  Matrix of ints monitor.
- class **MHAParser::vfloat_mon_t**

  Vector of floats monitor.
- class **MHAParser::mfloat_mon_t**

  Matrix of floats monitor.
- class **MHAParser::float_mon_t**

  Monitor with float value.
- class **MHAParser::complex_mon_t**

  Monitor with complex value.
- class **MHAParser::vcomplex_mon_t**

  Monitor with vector of complex values.
- class **MHAParser::mcomplex_mon_t**

  Matrix of complex numbers monitor.
- class **MHAParser::commit_t**$<$ **receiver_t** $>$

  Parser variable with event-emission functionality.
- class **MHAParser::mhaconfig_mon_t**

Namespaces

- **MHAParser**

  Name space for the openMHA-Parser configuration language.
- **MHAParser::StrCnv**

  String converter namespace.

© 2005-2021 HörTech gGmbH, Oldenburg

Macros

- #define **DEFAULT_RETSIZE** 0x100000
- #define **insert_member**(x) insert_item(#x,&x)

  Macro to insert a member variable into a parser.

Typedefs

- typedef std::string(base_t::∗ **MHAParser::opact_t**) (**expression_t** &)
- typedef std::string(base_t::∗ **MHAParser::query_t**) (const std::string &)
- typedef std::map< std::string, opact_t > **MHAParser::opact_map_t**
- typedef std::map< std::string, query_t > **MHAParser::query_map_t**
- typedef std::list< entry_t > **MHAParser::entry_map_t**
- typedef int(∗ **MHAParser::c_parse_cmd_t**) (void ∗, const char ∗, char ∗, unsigned int)

Functions

- std::string **MHAParser::commentate** (const std::string &s)
- void **MHAParser::trim** (std::string &s)
- std::string **MHAParser::cfg_dump** (base_t ∗, const std::string &)
- std::string **MHAParser::cfg_dump_short** (base_t ∗, const std::string &)
- std::string **MHAParser::all_dump** (base_t ∗, const std::string &)
- std::string **MHAParser::mon_dump** (base_t ∗, const std::string &)
- std::string **MHAParser::all_ids** (base_t ∗, const std::string &, const std::string &="")
- void **MHAParser::strreplace** (std::string &, const std::string &, const std::string &)

  string replace function
- void **MHAParser::envreplace** (std::string &s)
- void **MHAParser::StrCnv::str2val** (const std::string &, bool &)

  Convert from string.
- void **MHAParser::StrCnv::str2val** (const std::string &, float &)

  Convert from string.
- void **MHAParser::StrCnv::str2val** (const std::string &, **mha_complex_t** &)

  Convert from string.
- void **MHAParser::StrCnv::str2val** (const std::string &, int &)

  Convert from string.
- void **MHAParser::StrCnv::str2val** (const std::string &, keyword_list_t &)

  Convert from string.
- void **MHAParser::StrCnv::str2val** (const std::string &, std::string &)

  Convert from string.
- template<class arg_t >
  void **MHAParser::StrCnv::str2val** (const std::string &s, std::vector< arg_t > &val)

  Converter for vector types.
- template<> void **MHAParser::StrCnv::str2val**< **mha_real_t** > (const std::string &s, std::vector< **mha_real_t** > &v)

© 2005-2021 HörTech gGmbH, Oldenburg

Converter for vector<mha_real_t> with Matlab-style expansion.

- template<class arg_t >
  void **MHAParser::StrCnv::str2val** (const std::string &s, std::vector< std::vector< arg_t > > &val)

  Converter for matrix types.
- std::string **MHAParser::StrCnv::val2str** (const bool &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const float &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const **mha_complex_t** &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const int &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const keyword_list_t &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const std::string &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const std::vector< float > &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const std::vector< **mha_complex_t** > &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const std::vector< int > &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const std::vector< std::vector< int > > &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const std::vector< std::string > &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const std::vector< std::vector< float > > &)

  Convert to string.
- std::string **MHAParser::StrCnv::val2str** (const std::vector< std::vector< **mha_complex_t** > > &)

  Convert to string.
- int **MHAParser::StrCnv::num_brackets** (const std::string &s)

  count number of brackets

Variables

- const typedef char *(* **MHAParser::c_parse_err_t**)(void *, int)

6.155.1 Detailed Description

Header file for the MHA-Parser script language.

© 2005-2021 HörTech gGmbH, Oldenburg

6.155.2 Macro Definition Documentation

6.155.2.1 DEFAULT_RETSIZE `#define DEFAULT_RETSIZE 0x100000`

6.155.2.2 insert_member `#define insert_member( x ) insert_item(#x,&x)`

Macro to insert a member variable into a parser.

Parameters

x	Member variable to be inserted. Name of member variable will be used as configuration name.

See also **MHAParser::parser_t::insert_item()** (p. 1104).

6.156 mha_plugin.cpp File Reference

6.157 mha_plugin.hh File Reference

Header file for MHA C++ plugin class templates.

Classes

- class **MHAPlugin::cfg_node_t**< **runtime_cfg_t** >

  A node class for storing MHA plugin runtime configurations as a singly linked list, where the nodes store besides the usual "next" and "data" pointers also a flag that indicates whether this node can be deleted.
- class **MHAPlugin::config_t**< **runtime_cfg_t** >

  Template class for thread safe configuration.
- class **MHAPlugin::plugin_t**< **runtime_cfg_t** >

  The template class for C++ openMHA plugins.

Namespaces

- **MHAPlugin**

  Namespace for openMHA plugin class templates and thread-safe runtime configurations.

© 2005-2021 HörTech gGmbH, Oldenburg

Macros

- #define **__declspec**(p)
- #define **WINAPI**
- #define **HINSTANCE** int
- #define **MHAPLUGIN_PROC_CALLBACK_PREFIX**(prefix, classname, indom, outdom)
- #define **MHAPLUGIN_SETCPP_CALLBACK_PREFIX**(prefix, classname)
- #define **MHAPLUGIN_INIT_CALLBACKS_PREFIX**(prefix, classname)
- #define **MHAPLUGIN_CALLBACKS_PREFIX**(prefix, classname, indom, outdom)

  C++ wrapper macro for the plugin interface.
- #define **MHAPLUGIN_DOCUMENTATION_PREFIX**(prefix, cat, doc)
- #define **MHAPLUGIN_PROC_CALLBACK**(plugname, classname, indom, outdom) **MHAPLUGIN_PROC_CALLBACK_PREFIX**(MHA_STATIC_ ## plugname ## _,classname,indom,outdom)
- #define **MHAPLUGIN_INIT_CALLBACKS**(plugname, classname) **MHAPLUGIN_INI↩T_CALLBACKS_PREFIX**(MHA_STATIC_ ## plugname ## _,classname)
- #define **MHAPLUGIN_CALLBACKS**(plugname, classname, indom, outdom) **MHAPL↩UGIN_CALLBACKS_PREFIX**(MHA_STATIC_ ## plugname ## _,classname,indom,outdom)

  C++ wrapper macro for the plugin interface.
- #define **MHAPLUGIN_DOCUMENTATION**(plugname, cat, doc) **MHAPLUGIN_DOC↩UMENTATION_PREFIX**(MHA_STATIC_ ## plugname ## _,cat,doc)

  Wrapper macro for the plugin documentation interface.

6.157.1 Detailed Description

Header file for MHA C++ plugin class templates.

This file defines useful macros and template classes for the development of MHA plugins. A set of macros wraps a C++ interface around the ANSI-C plugin interface. The plugin_t template class defines a corresponding C++ class with all required members. This class can make use of thread safe configurations (config_t).

6.157.2 Macro Definition Documentation

6.157.2.1 __declspec `#define __declspec(p)`

6.157.2.2 WINAPI `#define WINAPI`

6.157.2.3 HINSTANCE `#define HINSTANCE int`

6.157.2.4 MHAPLUGIN_PROC_CALLBACK_PREFIX

```
#define MHAPLUGIN_PROC_CALLBAC↩
K_PREFIX(
            prefix,
            classname,
            indom,
            outdom )
```

6.157.2.5 MHAPLUGIN_SETCPP_CALLBACK_PREFIX

```
#define MHAPLUGIN_SETCPP_CAL↩
LBACK_PREFIX(
            prefix,
            classname )
```

6.157.2.6 MHAPLUGIN_INIT_CALLBACKS_PREFIX

```
#define MHAPLUGIN_INIT_CALLBACK↩
S_PREFIX(
            prefix,
            classname )
```

6.157.2.7 MHAPLUGIN_CALLBACKS_PREFIX

```
#define MHAPLUGIN_CALLBACKS_PREFIX(
            prefix,
            classname,
            indom,
            outdom )
```

C++ wrapper macro for the plugin interface.

Parameters

classname	The name of the plugin class
indom	Input domain (`wave` or `spec`)
outdom	Output domain (`wave` or `spec`)

© 2005-2021 HörTech gGmbH, Oldenburg

This macro defines all required openMHA Plugin interface functions and passes calls of these functions to the corresponding member functions of the class `'classname'`. The parameters `'indom'` and `'outdom'` specify the input and output domain of the processing method. The `MHAInit()` and `MHADestroy()` functions will create or destroy an instance of the class. The approriate member functions have to be defined in the class. It is suggested to make usage of the **MHAPlugin::plugin_t** (p. 1151) template class. Exceptions of type **MHA_Error** (p. 764) are caught and transformed into apropriate error codes with their corresponding error messages.

6.157.2.8 MHAPLUGIN_DOCUMENTATION_PREFIX

```
#define MHAPLUGIN_DOCUMENTATIO↩
N_PREFIX(
            prefix,
            cat,
            doc )
```

6.157.2.9 MHAPLUGIN_PROC_CALLBACK

```
#define MHAPLUGIN_PROC_CALLBACK(
            plugname,
            classname,
            indom,
            outdom )  MHAPLUGIN_PROC_CALLBACK_PREFIX(MHA_STATIC_ ## plugname ## _↩
,classname,indom,outdom)
```

6.157.2.10 MHAPLUGIN_INIT_CALLBACKS

```
#define MHAPLUGIN_INIT_CALLBACKS(
            plugname,
            classname )  MHAPLUGIN_INIT_CALLBACKS_PREFIX(MHA_STATIC_ ## plugname ##
_,classname)
```

6.157.2.11 MHAPLUGIN_CALLBACKS

```
#define MHAPLUGIN_CALLBACKS(
            plugname,
            classname,
            indom,
            outdom )  MHAPLUGIN_CALLBACKS_PREFIX(MHA_STATIC_ ## plugname ## _↩
,classname,indom,outdom)
```

C++ wrapper macro for the plugin interface.

Parameters

plugname	The file name of the plugin without the .so or .dll extension
classname	The name of the plugin class
indom	Input domain (`wave` or `spec`)
outdom	Output domain (`wave` or `spec`)

© 2005-2021 HörTech gGmbH, Oldenburg

This macro defines all required openMHA Plugin interface functions and passes calls of these functions to the corresponding member functions of the class 'classname'. The parameters 'indom' and 'outdom' specify the input and output domain of the processing method. The MHAInit() and MHADestroy() functions will create or destroy an instance of the class. The approriate member functions have to be defined in the class. It is suggested to make usage of the **MHAPlugin::plugin_t** (p. 1151) template class. Exceptions of type **MHA_Error** (p. 764) are caught and transformed into apropriate error codes with their corresponding error messages.

6.157.2.12 MHAPLUGIN_DOCUMENTATION

```
#define MHAPLUGIN_DOCUMENTATION(
    plugname,
    cat,
    doc )  MHAPLUGIN_DOCUMENTATION_PREFIX(MHA_STATIC_ ## plugname ## _↩
,cat,doc)
```

Wrapper macro for the plugin documentation interface.

Parameters

plugin	The file name of the plugin without the .so or .dll extension
cat	Space separated list of categories to which belong the plugin (as const char∗)
doc	Documentation of the plugin (as const char∗)

This macro defines the openMHA Plugin interface function for the documentation. The categories can be any space seperated list of category names. An empty string will categorize the plugin in the category 'other'.

The documentation should contain a description of the plugin including a description of the underlying models, and a paragraph containing hints for usage. The text should be LaTeX compatible (e.g., avoid or quote underscores in the text part); equations should be formatted as LaTeX.

6.158 mha_profiling.c File Reference

Functions

- void **mha_tic** (**mha_tictoc_t** ∗t)
- void **mha_platform_tic** (**mha_platform_tictoc_t** ∗t)
- float **mha_toc** (**mha_tictoc_t** ∗t)
- float **mha_platform_toc** (**mha_platform_tictoc_t** ∗t)

6.158.1 Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.158.1.1 mha_tic()

```
void mha_tic (
    mha_tictoc_t * t )
```

6.158.1.2 mha_platform_tic()

```
void mha_platform_tic (
    mha_platform_tictoc_t * t )
```

6.158.1.3 mha_toc()

```
float mha_toc (
    mha_tictoc_t * t )
```

6.158.1.4 mha_platform_toc()

```
float mha_platform_toc (
    mha_platform_tictoc_t * t )
```

6.159 mha_profiling.h File Reference

Classes

- struct **mha_tictoc_t**

Typedefs

- typedef **mha_tictoc_t** **mha_platform_tictoc_t**

Functions

- void **mha_platform_tic** (**mha_platform_tictoc_t** ∗t)
- float **mha_platform_toc** (**mha_platform_tictoc_t** ∗t)

6.159.1 Typedef Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.159.1.1 mha_platform_tictoc_t `typedef` **`mha_tictoc_t`** **`mha_platform_tictoc_t`**

6.159.2 Function Documentation

6.159.2.1 mha_platform_tic() `void mha_platform_tic (`
`mha_platform_tictoc_t` `* t )`

6.159.2.2 mha_platform_toc() `float mha_platform_toc (`
`mha_platform_tictoc_t` `* t )`

6.160 mha_ruby.cpp File Reference

Typedefs

- typedef VALUE(∗ **rb_f_t**) (...)

Functions

- static void **mha_free** (void ∗mha)
- static VALUE **mha_alloc** (VALUE klass)
- static VALUE **mha_exit_request** (VALUE self)
- static VALUE **mha_parse** (VALUE self, VALUE request)
- void **Init_mha_ruby** ()

6.160.1 Typedef Documentation

6.160.1.1 rb_f_t `typedef VALUE(* rb_f_t) (...)`

6.160.2 Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.160.2.1 mha_free()

```
static void mha_free (
    void * mha ) [static]
```

6.160.2.2 mha_alloc()

```
static VALUE mha_alloc (
    VALUE klass ) [static]
```

6.160.2.3 mha_exit_request()

```
static VALUE mha_exit_request (
    VALUE self ) [static]
```

6.160.2.4 mha_parse()

```
static VALUE mha_parse (
    VALUE self,
    VALUE request ) [static]
```

6.160.2.5 Init_mha_ruby()

```
void Init_mha_ruby ( )
```

6.161 mha_signal.cpp File Reference

Classes

- class **MHASignal::hilbert_fftw_t**

Namespaces

- **MHASignal**
  Namespace for audio signal handling and processing classes.

Macros

- #define **MHA_ID_UINT_VECTOR** "MHASignal::uint_vector_t"
- #define **MHA_ID_MATRIX** "MHASignal::matrix_t"
- #define **ASSERT_EQUAL_DIM**(a, b)
- #define **ASSERT_EQUAL_DIM_PTR**(a, b)

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- void **set_minabs** (**mha_spec_t** &self, const **mha_real_t** &m)
- **mha_wave_t** & **operator+=** (**mha_wave_t** &self, const **mha_real_t** &v)

  Addition operator.
- **mha_wave_t** & **operator∗=** (**mha_wave_t** &self, const **mha_real_t** &v)

  Element-wise multiplication operator.
- **mha_spec_t** & **operator∗=** (**mha_spec_t** &self, const **mha_real_t** &v)

  Element-wise multiplication operator.
- **mha_wave_t** & **operator∗=** (**mha_wave_t** &self, const **mha_wave_t** &v)

  Element-wise multiplication operator.
- **mha_spec_t** & **operator∗=** (**mha_spec_t** &self, const **mha_wave_t** &v)

  Element-wise multiplication operator.
- **mha_spec_t** & **operator∗=** (**mha_spec_t** &self, const **mha_spec_t** &v)

  Element-wise multiplication operator.
- **mha_spec_t** & **safe_div** (**mha_spec_t** &self, const **mha_spec_t** &v, **mha_real_t** eps)

  In-Place division with lower limit on divisor.
- **mha_spec_t** & **operator/=** (**mha_spec_t** &self, const **mha_spec_t** &v)

  Element-wise division operator.
- **mha_wave_t** & **operator/=** (**mha_wave_t** &self, const **mha_wave_t** &v)

  Element-wise division operator.
- **mha_spec_t** & **operator+=** (**mha_spec_t** &self, const **mha_spec_t** &v)

  Addition operator.
- **mha_spec_t** & **operator+=** (**mha_spec_t** &self, const **mha_real_t** &v)

  Addition operator.
- **mha_wave_t** & **operator+=** (**mha_wave_t** &self, const **mha_wave_t** &v)

  Addition operator.
- **mha_wave_t** & **operator-=** (**mha_wave_t** &self, const **mha_wave_t** &v)

  Subtraction operator.
- **mha_spec_t** & **operator-=** (**mha_spec_t** &self, const **mha_spec_t** &v)

  Subtraction operator.
- **mha_fft_t** **mha_fft_new** (unsigned int n)

  Create a new instance of an FFT object.
- void **mha_fft_free** (**mha_fft_t** h)

  Remove an FFT object.
- void **mha_fft_wave2spec** (**mha_fft_t** h, const **mha_wave_t** ∗in, **mha_spec_t** ∗out)

  Perform an FFT on each channel of input waveform signal.
- void **mha_fft_wave2spec** (**mha_fft_t** h, const **mha_wave_t** ∗in, **mha_spec_t** ∗out, bool swap)

  Tranform waveform segment into spectrum.
- void **mha_fft_spec2wave** (**mha_fft_t** h, const **mha_spec_t** ∗in, **mha_wave_t** ∗out)

  Perform an inverse FFT on each channel of input spectrum.
- void **mha_fft_spec2wave** (**mha_fft_t** h, const **mha_spec_t** ∗in, **mha_wave_t** ∗out, unsigned int offset)

  Perform an inverse FFT on each channel of input spectrum.

© 2005-2021 HörTech gGmbH, Oldenburg

- void **mha_fft_forward** (**mha_fft_t** h, **mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)

  Complex to complex FFT (forward).
- void **mha_fft_backward** (**mha_fft_t** h, **mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)

  Complex to complex FFT (backward).
- void **mha_fft_forward_scale** (**mha_fft_t** h, **mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)

  Complex to complex FFT (forward).
- void **mha_fft_backward_scale** (**mha_fft_t** h, **mha_spec_t** ∗sIn, **mha_spec_t** ∗sOut)

  Complex to complex FFT (backward).
- void **mha_fft_wave2spec_scale** (**mha_fft_t** h, const **mha_wave_t** ∗in, **mha_spec_t** ∗out)

  Tranform waveform segment into spectrum.
- void **mha_fft_spec2wave_scale** (**mha_fft_t** h, const **mha_spec_t** ∗in, **mha_wave_t** ∗out)

  Tranform spectrum into waveform segment.
- std::vector< float > **std_vector_float** (const **mha_wave_t** &w)

  *Converts a **mha_wave_t** (p. 840) structure into a std::vector<float> (interleaved order).*
- std::vector< std::vector< float > > **std_vector_vector_float** (const **mha_wave_t** &w)

  *Converts a **mha_wave_t** (p. 840) structure into a std::vector< std::vector<float> > (outer vector represents channels).*
- std::vector< std::vector< **mha_complex_t** > > **std_vector_vector_complex** (const **mha_spec_t** &w)

  *Converts a **mha_spec_t** (p. 794) structure into a std::vector< std::vector<mha_complex_t> > (outer vector represents channels).*
- static **mha_real_t** **intensity** (const **mha_spec_t** &s, unsigned int channel, unsigned int fftlen, **mha_real_t** ∗sqfreq_response=0)
- void **integrate** (**mha_wave_t** &s)

  Numeric integration of a signal vector (real values)
- void **integrate** (**mha_spec_t** &s)

  Numeric integration of a signal vector (complex values)
- **mha_wave_t** & **operator^=** (**mha_wave_t** &self, const **mha_real_t** &arg)

  Exponent operator.
- **mha_wave_t** **range** (**mha_wave_t** s, unsigned int k0, unsigned int len)

  Return a time interval from a waveform chunk.
- **mha_spec_t** **channels** (**mha_spec_t** s, unsigned int ch_start, unsigned int nch)

  Return a channel interval from a spectrum.
- void **assign** (**mha_wave_t** self, const **mha_wave_t** &val)

  Set all values of waveform 'self' to 'val'.
- void **assign** (**mha_spec_t** self, const **mha_spec_t** &val)

  Set all values of spectrum 'self' to 'val'.
- void **timeshift** (**mha_wave_t** &self, int shift)

  Time shift of waveform chunk.

6.161.1 Macro Definition Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.161.1.1 MHA_ID_UINT_VECTOR

```
#define MHA_ID_UINT_VECTOR "MHASignal::uint_↩
vector_t"
```

6.161.1.2 MHA_ID_MATRIX

```
#define MHA_ID_MATRIX "MHASignal::matrix_t"
```

6.161.1.3 ASSERT_EQUAL_DIM

```
#define ASSERT_EQUAL_DIM(
    a,
    b )
```

6.161.1.4 ASSERT_EQUAL_DIM_PTR

```
#define ASSERT_EQUAL_DIM_PTR(
    a,
    b )
```

6.161.2 Function Documentation

6.161.2.1 set_minabs()

```
void set_minabs (
    mha_spec_t & self,
  const  mha_real_t & m )
```

6.161.2.2 safe_div()

```
mha_spec_t& safe_div (
    mha_spec_t & self,
  const  mha_spec_t & v,
    mha_real_t eps )
```

In-Place division with lower limit on divisor.

© 2005-2021 HörTech gGmbH, Oldenburg

6.161.2.3 intensity()

```
static mha_real_t intensity (
    const mha_spec_t & s,
    unsigned int channel,
    unsigned int fftlen,
    mha_real_t * sqfreq_response = 0 ) [static]
```

6.162 mha_signal.hh File Reference

Header file for audio signal handling and processing classes.

Classes

- class **MHASignal::spectrum_t**
  *a signal processing class for spectral data (based on **mha_spec_t** (p. 794))*
- class **MHASignal::waveform_t**
  *signal processing class for waveform data (based on **mha_wave_t** (p. 840))*
- class **MHASignal::doublebuffer_t**
  Double-buffering class.
- class **MHASignal::ringbuffer_t**
  A ringbuffer class for time domain audio signal, which makes no assumptions with respect to fragment size.
- class **MHASignal::hilbert_t**
  Hilbert transformation of a waveform segment.
- class **MHASignal::minphase_t**
  Minimal phase function.
- class **MHASignal::stat_t**
- class **MHASignal::delay_wave_t**
  Delayline containing wave fragments.
- class **MHASignal::delay_spec_t**
- class **MHASignal::async_rmslevel_t**
  Class for asynchronous level metering.
- class **MHASignal::uint_vector_t**
  Vector of unsigned values, used for size and index description of n-dimensional matrixes.
- class **MHASignal::matrix_t**
  n-dimensional matrix with real or complex floating point values.
- class **MHASignal::schroeder_t**
  Schroeder tone complex class.
- class **MHASignal::quantizer_t**
  Simple simulation of fixpoint quantization.
- class **MHASignal::loop_wavefragment_t**
  Copy a fixed waveform fragment to a series of waveform fragments of other size.
- class **MHASignal::delay_t**
  Class to realize a simple delay of waveform streams.
- class **MHASignal::subsample_delay_t**
  implements subsample delay in spectral domain.

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **MHASignal**
  Namespace for audio signal handling and processing classes.

Macros

- #define **M_PI** 3.14159265358979323846
- #define **mha_round**(x) (int)((float)x+0.5)

Functions

- void **MHASignal::for_each** (**mha_wave_t** ∗s, **mha_real_t**(∗fun)(**mha_real_t**))
  *Apply a function to each element of a **mha_wave_t*** (p. 840).
- **mha_real_t MHASignal::lin2db** (**mha_real_t** x, **mha_real_t** eps)
  Conversion from linear scale to dB (no SPL reference)
- **mha_real_t MHASignal::lin2db** (**mha_real_t** x)
  Conversion from linear scale to dB (no SPL reference)
- **mha_real_t MHASignal::db2lin** (**mha_real_t** x)
  Conversion from dB scale to linear (no SPL reference)
- **mha_real_t MHASignal::sq2db** (**mha_real_t** x, **mha_real_t** eps=0.0f)
  conversion from squared values to dB (no SPL reference)
- **mha_real_t MHASignal::db2sq** (**mha_real_t** x)
  conversion from dB to squared values (no SPL reference)
- **mha_real_t MHASignal::pa2dbspl** (**mha_real_t** x, **mha_real_t** eps)
  Conversion from linear Pascal scale to dB SPL.
- **mha_real_t MHASignal::pa2dbspl** (**mha_real_t** x)
  Conversion from linear Pascal scale to dB SPL.
- **mha_real_t MHASignal::dbspl2pa** (**mha_real_t** x)
  Conversion from dB SPL to linear Pascal scale.
- **mha_real_t MHASignal::pa22dbspl** (**mha_real_t** x, **mha_real_t** eps=0.0f)
  Conversion from squared Pascal scale to dB SPL.
- **mha_real_t MHASignal::dbspl2pa2** (**mha_real_t** x)
  conversion from dB SPL to squared Pascal scale
- **mha_real_t MHASignal::smp2sec** (**mha_real_t** n, **mha_real_t** srate)
  conversion from samples to seconds
- **mha_real_t MHASignal::sec2smp** (**mha_real_t** sec, **mha_real_t** srate)
  conversion from seconds to samples
- **mha_real_t MHASignal::bin2freq** (**mha_real_t** bin, unsigned fftlen, **mha_real_t** srate)
  conversion from fft bin index to frequency
- **mha_real_t MHASignal::freq2bin** (**mha_real_t** freq, unsigned fftlen, **mha_real_t** srate)

© 2005-2021 HörTech gGmbH, Oldenburg

conversion from frequency to fft bin index

- **mha_real_t MHASignal::smp2rad** (**mha_real_t** samples, unsigned bin, unsigned fftlen)

  conversion from delay in samples to phase shift

- **mha_real_t MHASignal::rad2smp** (**mha_real_t** phase_shift, unsigned bin, unsigned fftlen)

  conversion from phase shift to delay in samples

- template<class elem_type >
  std::vector< elem_type > **MHASignal::dupvec** (std::vector< elem_type > vec, unsigned n)

  Duplicate last vector element to match desired size.

- template<class elem_type >
  std::vector< elem_type > **MHASignal::dupvec_chk** (std::vector< elem_type > vec, unsigned n)

  Duplicate last vector element to match desired size, check for dimension.

- bool **equal_dim** (const **mha_wave_t** &a, const **mha_wave_t** &b)

  Test for equal dimension of waveform structures.

- bool **equal_dim** (const **mha_wave_t** &a, const **mhaconfig_t** &b)

  Test for match of waveform dimension with mhaconfig structure.

- bool **equal_dim** (const **mha_spec_t** &a, const **mha_spec_t** &b)

  Test for equal dimension of spectrum structures.

- bool **equal_dim** (const **mha_spec_t** &a, const **mhaconfig_t** &b)

  Test for match of spectrum dimension with mhaconfig structure.

- bool **equal_dim** (const **mha_wave_t** &a, const **mha_spec_t** &b)

  Test for equal dimension of waveform/spectrum structures.

- bool **equal_dim** (const **mha_spec_t** &a, const **mha_wave_t** &b)

  Test for equal dimension of waveform/spectrum structures.

- void **integrate** (**mha_wave_t** &s)

  Numeric integration of a signal vector (real values)

- void **integrate** (**mha_spec_t** &s)

  Numeric integration of a signal vector (complex values)

- unsigned int **mha_min_1** (unsigned int a)
- unsigned int **size** (const **mha_wave_t** &s)

  Return size of a waveform structure.

- unsigned int **size** (const **mha_spec_t** &s)

  Return size of a spectrum structure.

- unsigned int **size** (const **mha_wave_t** *s)

  Return size of a waveform structure.

- unsigned int **size** (const **mha_spec_t** *s)

  Return size of a spectrum structure.

- void **clear** (**mha_wave_t** &s)

  Set all values of waveform to zero.

- void **clear** (**mha_wave_t** *s)

  Set all values of waveform to zero.

- void **clear** (**mha_spec_t** &s)

  Set all values of spectrum to zero.

© 2005-2021 HörTech gGmbH, Oldenburg

- void **clear** (**mha_spec_t** ∗s)

  Set all values of spectrum to zero.
- void **assign** (**mha_wave_t** self, **mha_real_t** val)

  Set all values of waveform 'self' to 'val'.
- void **assign** (**mha_wave_t** self, const **mha_wave_t** &val)

  Set all values of waveform 'self' to 'val'.
- void **assign** (**mha_spec_t** self, const **mha_spec_t** &val)

  Set all values of spectrum 'self' to 'val'.
- void **timeshift** (**mha_wave_t** &self, int shift)

  Time shift of waveform chunk.
- **mha_wave_t** **range** (**mha_wave_t** s, unsigned int k0, unsigned int len)

  Return a time interval from a waveform chunk.
- **mha_spec_t** **channels** (**mha_spec_t** s, unsigned int ch_start, unsigned int nch)

  Return a channel interval from a spectrum.
- **mha_real_t** & **value** (**mha_wave_t** ∗s, unsigned int fr, unsigned int ch)

  Access an element of a waveform structure.
- const **mha_real_t** & **value** (const **mha_wave_t** ∗s, unsigned int fr, unsigned int ch)

  Constant access to an element of a waveform structure.
- **mha_real_t** & **value** (**mha_wave_t** ∗s, unsigned int k)
- **mha_complex_t** & **value** (**mha_spec_t** ∗s, unsigned int k)
- **mha_complex_t** & **value** (**mha_spec_t** ∗s, unsigned int fr, unsigned int ch)

  Access to an element of a spectrum.
- const **mha_complex_t** & **value** (const **mha_spec_t** ∗s, unsigned int fr, unsigned int ch)

  Constant access to an element of a spectrum.
- **mha_real_t** & **value** (**mha_wave_t** &s, unsigned int fr, unsigned int ch)

  Access to an element of a waveform structure.
- const **mha_real_t** & **value** (const **mha_wave_t** &s, unsigned int fr, unsigned int ch)

  Constant access to an element of a waveform structure.
- **mha_complex_t** & **value** (**mha_spec_t** &s, unsigned int fr, unsigned int ch)

  Access to an element of a spectrum.
- const **mha_complex_t** & **value** (const **mha_spec_t** &s, unsigned int fr, unsigned int ch)

  Constant access to an element of a spectrum.
- std::vector< float > **std_vector_float** (const **mha_wave_t** &)

  *Converts a **mha_wave_t** (p. 840) structure into a std::vector<float> (interleaved order).*
- std::vector< std::vector< float > > **std_vector_vector_float** (const **mha_wave_t** &)

  *Converts a **mha_wave_t** (p. 840) structure into a std::vector< std::vector<float> > (outer vector represents channels).*
- std::vector< std::vector< **mha_complex_t** > > **std_vector_vector_complex** (const **mha_spec_t** &)

  *Converts a **mha_spec_t** (p. 794) structure into a std::vector< std::vector<mha_complex_t> > (outer vector represents channels).*
- **mha_wave_t** & **operator+=** (**mha_wave_t** &, const **mha_real_t** &)

  Addition operator.
- **mha_wave_t** & **operator+=** (**mha_wave_t** &, const **mha_wave_t** &)

  Addition operator.

© 2005-2021 HörTech gGmbH, Oldenburg

- **mha_wave_t** & **operator-=** (**mha_wave_t** &, const **mha_wave_t** &)
  Subtraction operator.
- **mha_spec_t** & **operator-=** (**mha_spec_t** &, const **mha_spec_t** &)
  Subtraction operator.
- **mha_wave_t** & **operator∗=** (**mha_wave_t** &, const **mha_real_t** &)
  Element-wise multiplication operator.
- **mha_wave_t** & **operator∗=** (**mha_wave_t** &, const **mha_wave_t** &)
  Element-wise multiplication operator.
- **mha_spec_t** & **operator∗=** (**mha_spec_t** &, const **mha_real_t** &)
  Element-wise multiplication operator.
- **mha_spec_t** & **operator∗=** (**mha_spec_t** &, const **mha_wave_t** &)
  Element-wise multiplication operator.
- **mha_spec_t** & **operator∗=** (**mha_spec_t** &, const **mha_spec_t** &)
  Element-wise multiplication operator.
- **mha_spec_t** & **operator/=** (**mha_spec_t** &, const **mha_spec_t** &)
  Element-wise division operator.
- **mha_wave_t** & **operator/=** (**mha_wave_t** &, const **mha_wave_t** &)
  Element-wise division operator.
- **mha_spec_t** & **operator+=** (**mha_spec_t** &, const **mha_spec_t** &)
  Addition operator.
- **mha_spec_t** & **operator+=** (**mha_spec_t** &, const **mha_real_t** &)
  Addition operator.
- void **set_minabs** (**mha_spec_t** &, const **mha_real_t** &)
- **mha_spec_t** & **safe_div** (**mha_spec_t** &self, const **mha_spec_t** &v, **mha_real_t** eps)
  In-Place division with lower limit on divisor.
- **mha_wave_t** & **operator^=** (**mha_wave_t** &self, const **mha_real_t** &arg)
  Exponent operator.
- void **MHASignal::copy_channel** (**mha_spec_t** &self, const **mha_spec_t** &src, unsigned sch, unsigned dch)
  Copy one channel of a source signal.
- void **MHASignal::copy_channel** (**mha_wave_t** &self, const **mha_wave_t** &src, unsigned src_channel, unsigned dest_channel)
  Copy one channel of a source signal.
- **mha_real_t** **MHASignal::rmslevel** (const **mha_spec_t** &s, unsigned int channel, unsigned int fftlen)
  Return RMS level of a spectrum channel.
- **mha_real_t** **MHASignal::colored_intensity** (const **mha_spec_t** &s, unsigned int channel, unsigned int fftlen, **mha_real_t** ∗sqfreq_response=nullptr)
  Colored spectrum intensity.
- **mha_real_t** **MHASignal::maxabs** (const **mha_spec_t** &s, unsigned int channel)
  Find maximal absolute value.
- **mha_real_t** **MHASignal::rmslevel** (const **mha_wave_t** &s, unsigned int channel)
  Return RMS level of a waveform channel.
- **mha_real_t** **MHASignal::maxabs** (const **mha_wave_t** &s, unsigned int channel)
  Find maximal absolute value.

© 2005-2021 HörTech gGmbH, Oldenburg

- **mha_real_t MHASignal::maxabs** (const **mha_wave_t** &s)

  Find maximal absolute value.
- **mha_real_t MHASignal::max** (const **mha_wave_t** &s)

  Find maximal value.
- **mha_real_t MHASignal::min** (const **mha_wave_t** &s)

  Find minimal value.
- **mha_real_t MHASignal::sumsqr_channel** (const **mha_wave_t** &s, unsigned int channel)

  Calculate sum of squared values in one channel.
- **mha_real_t MHASignal::sumsqr_frame** (const **mha_wave_t** &s, unsigned int frame)

  Calculate sum over all channels of squared values.
- void **MHASignal::scale** (**mha_spec_t** ∗dest, const **mha_wave_t** ∗src)
- void **MHASignal::limit** (**mha_wave_t** &s, const **mha_real_t** & **min**, const **mha_real_t** & **max**)

  Limit the singal in the waveform buffer to the range [min, max].
- **mha_complex_t** & **set** (**mha_complex_t** &self, **mha_real_t** real, **mha_real_t** imag=0)

  *Assign real and imaginary parts to a **mha_complex_t** (p. 745) variable.*
- **mha_complex_t mha_complex** (**mha_real_t** real, **mha_real_t** imag=0)

  *Create a new **mha_complex_t** (p. 745) with specified real and imaginary parts.*
- **mha_complex_t** & **set** (**mha_complex_t** &self, const std::complex< **mha_real_t** > & **stdcomplex**)

  *Assign a **mha_complex_t** (p. 745) variable from a std::complex.*
- std::complex< **mha_real_t** > **stdcomplex** (const **mha_complex_t** &self)

  *Create a std::complex from **mha_complex_t** (p. 745).*
- **mha_complex_t** & **expi** (**mha_complex_t** &self, **mha_real_t angle**)

  *replaces the value of the given **mha_complex_t** (p. 745) with exp(i∗b).*
- double **angle** (const **mha_complex_t** &self)

  Computes the angle of a complex number in the complex plane.
- **mha_complex_t** & **operator+=** (**mha_complex_t** &self, const **mha_complex_t** &other)

  Addition of two complex numbers, overwriting the first.
- **mha_complex_t operator+** (const **mha_complex_t** &self, const **mha_complex_t** &other)

  Addition of two complex numbers, result is a temporary object.
- **mha_complex_t** & **operator+=** (**mha_complex_t** &self, **mha_real_t** other_real)

  Addition of a complex and a real number, overwriting the complex.
- **mha_complex_t operator+** (const **mha_complex_t** &self, **mha_real_t** other_real)

  Addition of a complex and a real number, result is a temporary object.
- **mha_complex_t** & **operator-=** (**mha_complex_t** &self, const **mha_complex_t** &other)

  Subtraction of two complex numbers, overwriting the first.
- **mha_complex_t operator-** (const **mha_complex_t** &self, const **mha_complex_t** &other)

  Subtraction of two complex numbers, result is a temporary object.
- **mha_complex_t** & **operator-=** (**mha_complex_t** &self, **mha_real_t** other_real)

  Subtraction of a complex and a real number, overwriting the complex.
- **mha_complex_t operator-** (const **mha_complex_t** &self, **mha_real_t** other_real)

© 2005-2021 HörTech gGmbH, Oldenburg

Subtraction of a complex and a real number, result is a temporary object.

- **mha_complex_t** & **operator∗=** (**mha_complex_t** &self, const **mha_complex_t** &other)

  Multiplication of two complex numbers, overwriting the first.
- **mha_complex_t** **operator∗** (const **mha_complex_t** &self, const **mha_complex_t** &other)

  Multiplication of two complex numbers, result is a temporary object.
- **mha_complex_t** & **operator∗=** (**mha_complex_t** &self, **mha_real_t** other_real)

  Multiplication of a complex and a real number, overwriting the complex.
- **mha_complex_t** & **expi** (**mha_complex_t** &self, **mha_real_t** **angle**, **mha_real_t** factor)

  *replaces (!) the value of the given **mha_complex_t** (p. 745) with a ∗ exp(i∗b)*
- **mha_complex_t** **operator∗** (const **mha_complex_t** &self, **mha_real_t** other_real)

  Multiplication of a complex and a real number, result is a temporary object.
- **mha_real_t** **abs2** (const **mha_complex_t** &self)

  Compute the square of the absolute value of a complex value.
- **mha_real_t** **abs** (const **mha_complex_t** &self)

  Compute the absolute value of a complex value.
- **mha_complex_t** & **operator/=** (**mha_complex_t** &self, **mha_real_t** other_real)

  Division of a complex and a real number, overwriting the complex.
- **mha_complex_t** **operator/** (const **mha_complex_t** &self, **mha_real_t** other_real)

  Division of a complex and a real number, result is a temporary object.
- **mha_complex_t** & **safe_div** (**mha_complex_t** &self, const **mha_complex_t** &other, **mha_real_t** eps, **mha_real_t** eps2)
- **mha_complex_t** & **operator/=** (**mha_complex_t** &self, const **mha_complex_t** &other)

  Division of two complex numbers, overwriting the first.
- **mha_complex_t** **operator/** (const **mha_complex_t** &self, const **mha_complex_t** &other)

  Division of two complex numbers, result is a temporary object.
- **mha_complex_t** **operator-** (const **mha_complex_t** &self)

  Unary minus on a complex results in a negative temporary object.
- bool **operator==** (const **mha_complex_t** &x, const **mha_complex_t** &y)

  Compare two complex numbers for equality.
- bool **operator!=** (const **mha_complex_t** &x, const **mha_complex_t** &y)

  Compare two complex numbers for inequality.
- void **conjugate** (**mha_complex_t** &self)

  *Replace (!) the value of this **mha_complex_t** (p. 745) with its conjugate.*
- void **conjugate** (**mha_spec_t** &self)

  *Replace (!) the value of this **mha_spec_t** (p. 794) with its conjugate.*
- **mha_complex_t** **_conjugate** (const **mha_complex_t** &self)

  Compute the cojugate of this complex value.
- void **reciprocal** (**mha_complex_t** &self)

  Replace the value of this complex with its reciprocal.
- **mha_complex_t** **_reciprocal** (const **mha_complex_t** &self)

  compute the reciprocal of this complex value.
- void **normalize** (**mha_complex_t** &self)

© 2005-2021 HörTech gGmbH, Oldenburg

Divide a complex by its absolute value, thereby normalizing it (projecting onto the unit circle).

- void **normalize** (**mha_complex_t** &self, **mha_real_t** margin)

  Divide a complex by its absolute value, thereby normalizing it (projecting onto the unit circle), with a safety margin.

- bool **almost** (const **mha_complex_t** &self, const **mha_complex_t** &other, **mha_real_t** times_epsilon=1e2)

  Compare two complex numbers for equality except for a small relative error.

- bool **operator**< (const **mha_complex_t** &x, const **mha_complex_t** &y)

  Compares the absolute values of two complex numbers.

- std::ostream & **operator**<< (std::ostream &o, const **mha_complex_t** &c)

  *ostream operator for **mha_complex_t** (p. 745)*

- std::istream & **operator**>> (std::istream &i, **mha_complex_t** &c)

  *preliminary istream operator for **mha_complex_t** (p. 745) without error checking*

- **mha_fft_t** **mha_fft_new** (unsigned int n)

  Create a new FFT handle.

- void **mha_fft_free** (**mha_fft_t** h)

  Destroy an FFT handle.

- void **mha_fft_wave2spec** (**mha_fft_t** h, const **mha_wave_t** *in, **mha_spec_t** *out)

  Tranform waveform segment into spectrum.

- void **mha_fft_wave2spec** (**mha_fft_t** h, const **mha_wave_t** *in, **mha_spec_t** *out, bool swaps)

  Tranform waveform segment into spectrum.

- void **mha_fft_spec2wave** (**mha_fft_t** h, const **mha_spec_t** *in, **mha_wave_t** *out)

  Tranform spectrum into waveform segment.

- void **mha_fft_spec2wave** (**mha_fft_t** h, const **mha_spec_t** *in, **mha_wave_t** *out, unsigned int offset)

  Tranform spectrum into waveform segment. out may have fewer number of frames than needed for a complete iFFT. Only as many frames are written into out as fit, starting with offset offset of the complete iFFT.

- void **mha_fft_forward** (**mha_fft_t** h, **mha_spec_t** *sIn, **mha_spec_t** *sOut)

  Complex to complex FFT (forward).

- void **mha_fft_backward** (**mha_fft_t** h, **mha_spec_t** *sIn, **mha_spec_t** *sOut)

  Complex to complex FFT (backward).

- void **mha_fft_forward_scale** (**mha_fft_t** h, **mha_spec_t** *sIn, **mha_spec_t** *sOut)

  Complex to complex FFT (forward).

- void **mha_fft_backward_scale** (**mha_fft_t** h, **mha_spec_t** *sIn, **mha_spec_t** *sOut)

  Complex to complex FFT (backward).

- void **mha_fft_wave2spec_scale** (**mha_fft_t** h, const **mha_wave_t** *in, **mha_spec_t** *out)

  Tranform waveform segment into spectrum.

- void **mha_fft_spec2wave_scale** (**mha_fft_t** h, const **mha_spec_t** *in, **mha_wave_t** *out)

  Tranform spectrum into waveform segment.

- template<class elem_type >
  elem_type **MHASignal::kth_smallest** (elem_type array[ ], unsigned n, unsigned k)

  Fast search for the kth smallest element of an array.

© 2005-2021 HörTech gGmbH, Oldenburg

- template<class elem_type >
  elem_type **MHASignal::median** (elem_type array[ ], unsigned n)
  Fast median search.
- template<class elem_type >
  elem_type **MHASignal::mean** (const std::vector< elem_type > &data, elem_type start↩_val)
  Calculate average of elements in a vector.
- template<class elem_type >
  std::vector< elem_type > **MHASignal::quantile** (std::vector< elem_type > data, const std::vector< elem_type > &p)
  Calculate quantile of elements in a vector.
- void **MHASignal::saveas_mat4** (const **mha_spec_t** &data, const std::string &varname, FILE *fh)
  Save a openMHA spectrum as a variable in a Matlab4 file.
- void **MHASignal::saveas_mat4** (const **mha_wave_t** &data, const std::string &varname, FILE *fh)
  Save a openMHA waveform as a variable in a Matlab4 file.
- void **MHASignal::saveas_mat4** (const std::vector< **mha_real_t** > &data, const std↩::string &varname, FILE *fh)
  Save a float vector as a variable in a Matlab4 file.
- void **MHASignal::copy_permuted** (**mha_wave_t** *dest, const **mha_wave_t** *src)
  Copy contents of a waveform to a permuted waveform.

Variables

- unsigned long int **MHASignal::signal_counter** = 0
  Signal counter to produce signal ID strings.

6.162.1 Detailed Description

Header file for audio signal handling and processing classes.

The classes for waveform, spectrum and filterbank signals defined in this file are "intelligent" versions of the basic waveform, spectrum and filterbank structures used in the C function calls.

6.162.2 Macro Definition Documentation

6.162.2.1 M_PI `#define M_PI 3.14159265358979323846`

© 2005-2021 HörTech gGmbH, Oldenburg

6.162.2.2 mha_round

```
#define mha_round(
  x ) (int)((float)x+0.5)
```

6.162.3 Function Documentation

6.162.3.1 mha_min_1()

```
unsigned int mha_min_1 (
  unsigned int a ) [inline]
```

6.162.3.2 value() [1/2]

```
mha_real_t& value (
  mha_wave_t * s,
  unsigned int k ) [inline]
```

6.162.3.3 value() [2/2]

```
mha_complex_t& value (
  mha_spec_t * s,
  unsigned int k ) [inline]
```

6.162.3.4 set_minabs()

```
void set_minabs (
  mha_spec_t & ,
  const mha_real_t & )
```

6.162.3.5 safe_div()

```
mha_spec_t& safe_div (
  mha_spec_t & self,
  const mha_spec_t & v,
  mha_real_t eps )
```

In-Place division with lower limit on divisor.

© 2005-2021 HörTech gGmbH, Oldenburg

6.162.3.6 operator<<()

```
std::ostream& operator<< (
    std::ostream & o,
    const mha_complex_t & c ) [inline]
```

ostream operator for **mha_complex_t** (p. 745)

6.162.3.7 operator>>()

```
std::istream& operator>> (
    std::istream & i,
    mha_complex_t & c ) [inline]
```

preliminary istream operator for **mha_complex_t** (p. 745) without error checking

6.163 mha_signal_fft.h File Reference

Classes

- class **MHASignal::fft_t**

Namespaces

- **MHASignal**
  Namespace for audio signal handling and processing classes.

6.164 mha_tablelookup.cpp File Reference

6.165 mha_tablelookup.hh File Reference

Header file for table lookup classes.

Classes

- class **MHATableLookup::table_t**
- class **MHATableLookup::linear_table_t**
  Class for interpolation with equidistant x values.
- class **MHATableLookup::xy_table_t**
  Class for interpolation with non-equidistant x values.

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **MHATableLookup**
  Namespace for table lookup classes.

6.165.1 Detailed Description

Header file for table lookup classes.

6.166 mha_tcp.cpp File Reference

Classes

- class **MHA_TCP::sock_init_t**

Namespaces

- **MHA_TCP**
  A Namespace for TCP helper classes.

Macros

- #define **INVALID_SOCKET** (-1)
- #define **SOCKET_ERROR** (-1)
- #define **closesocket**(fd) (close((fd)))
- #define **ASYNC_CONNECT_STARTED** EINPROGRESS

Typedefs

- typedef int **SOCKET**

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- std::string **MHA_TCP::STRERROR** (int err)
  Portable conversion from error number to error string.
- std::string **MHA_TCP::HSTRERROR** (int err)
  Portable conversion from hostname error number to error string.
- int **MHA_TCP::N_ERRNO** ()
  Portable access to last network error number.
- int **MHA_TCP::H_ERRNO** ()
  Portable access to last hostname error number.
- int **MHA_TCP::G_ERRNO** ()
  Portable access to last non-network error number.
- static sockaddr_in **host_port_to_sock_addr** (const std::string &host, unsigned short port)
- static **SOCKET tcp_connect_to** (const std::string &host, unsigned short port)
- static **SOCKET tcp_connect_to_with_timeout** (const std::string &host, unsigned short port, **Timeout_Watcher** &timeout_watcher)
- static void ∗ **thread_start_func** (void ∗thread)

Variables

- class **MHA_TCP::sock_init_t MHA_TCP::sock_initializer**

6.166.1 Macro Definition Documentation

6.166.1.1 INVALID_SOCKET `#define INVALID_SOCKET (-1)`

6.166.1.2 SOCKET_ERROR `#define SOCKET_ERROR (-1)`

6.166.1.3 closesocket `#define closesocket(`
`fd ) (close((fd)))`

© 2005-2021 HörTech gGmbH, Oldenburg

6.166.1.4 ASYNC_CONNECT_STARTED `#define ASYNC_CONNECT_STARTED EINPROGRESS`

6.166.2 Typedef Documentation

6.166.2.1 SOCKET `typedef int` **SOCKET**

6.166.3 Function Documentation

6.166.3.1 host_port_to_sock_addr()

```
static sockaddr_in host_port_to_sock_addr (
    const std::string & host,
    unsigned short port ) [static]
```

6.166.3.2 tcp_connect_to()

```
static SOCKET tcp_connect_to (
    const std::string & host,
    unsigned short port ) [static]
```

6.166.3.3 tcp_connect_to_with_timeout()

```
static SOCKET tcp_connect_to_with_timeout
(
    const std::string & host,
    unsigned short port,
    Timeout_Watcher & timeout_watcher ) [static]
```

6.166.3.4 thread_start_func()

```
static void* thread_start_func (
    void * thread ) [static]
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.167 mha_tcp.hh File Reference

Classes

- struct **MHA_TCP::OS_EVENT_TYPE**
- class **MHA_TCP::Wakeup_Event**

  A base class for asynchronous wakeup events.
- class **MHA_TCP::Async_Notify**

  Portable Multiplexable cross-thread notification.
- class **MHA_TCP::Event_Watcher**

  OS-independent event watcher, uses select on Unix and WaitForMultipleObjects on Windows.
- class **MHA_TCP::Timeout_Event**
- class **MHA_TCP::Timeout_Watcher**

  OS-independent event watcher with internal fixed-end-time timeout.
- class **MHA_TCP::Sockread_Event**

  Watch socket for incoming data.
- class **MHA_TCP::Sockwrite_Event**
- class **MHA_TCP::Sockaccept_Event**
- class **MHA_TCP::Connection**

  Connection *(p. 803) handles Communication between client and server, is used on both sides.*
- class **MHA_TCP::Server**
- class **MHA_TCP::Client**

  A portable class for a tcp client connections.
- class **MHA_TCP::Thread**

  A very simple class for portable threads.

Namespaces

- **MHA_TCP**

  A Namespace for TCP helper classes.

Macros

- #define **Sleep**(x) usleep((x)∗1000);

Typedefs

- typedef int **MHA_TCP::SOCKET**

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- std::string **MHA_TCP::STRERROR** (int err)
  Portable conversion from error number to error string.
- std::string **MHA_TCP::HSTRERROR** (int err)
  Portable conversion from hostname error number to error string.
- int **MHA_TCP::N_ERRNO** ()
  Portable access to last network error number.
- int **MHA_TCP::H_ERRNO** ()
  Portable access to last hostname error number.
- int **MHA_TCP::G_ERRNO** ()
  Portable access to last non-network error number.
- double **MHA_TCP::dtime** ()
  Time access function for system's high resolution time, retrieve current time as double.
- double **MHA_TCP::dtime** (const struct timeval &tv)
  Time access function for unix' high resolution time, converts struct timeval to double.
- struct timeval **MHA_TCP::stime** (double d)
  Time access function for unix' high resolution time, converts time from double to struct timeval.

6.167.1 Macro Definition Documentation

6.167.1.1 Sleep

```
#define Sleep(
  x ) usleep((x)*1000);
```

6.168 mha_tcp_server.cpp File Reference

Namespaces

- **mha_tcp**
  namespace for network communication classes of MHA

6.169 mha_tcp_server.hh File Reference

Classes

- class **mha_tcp::buffered_socket_t**
  An asio TCP socket with an associated streambuf buffer for receiving lines of text, as well as string buffers for sending responses.
- class **mha_tcp::server_t**
  Class for accepting TCP connections from clients.

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **mha_tcp**
  namespace for network communication classes of MHA

6.170 mha_toolbox.h File Reference

6.171 mha_utils.cpp File Reference

6.172 mha_utils.hh File Reference

Namespaces

- **MHAUtils**

Functions

- bool **MHAUtils::is_multiple_of** (const unsigned big, const unsigned small)
- bool **MHAUtils::is_power_of_two** (const unsigned n)
- bool **MHAUtils::is_multiple_of_by_power_of_two** (const unsigned big, const unsigned small)
- std::string **MHAUtils::strip** (const std::string &line)
- std::string **MHAUtils::remove** (const std::string &str_, char c)
- bool **MHAUtils::is_denormal** (**mha_real_t** x)
  Get the normal-ness of a mha_real_t.
- bool **MHAUtils::is_denormal** (const **mha_complex_t** &x)
  Get the normal-ness of a complex number.
- bool **MHAUtils::is_denormal** (const std::complex< **mha_real_t** > &x)
  Get the normal-ness of a complex number.
- **mha_real_t MHAUtils::spl2hl** (**mha_real_t** f)
  Get the offset of between dB(SPL) and dB(HL) for a given frequency according to ISO 389-7↩ :2005 (freefield); e.g.

6.173 mha_windowparser.cpp File Reference

Variables

- float(∗ **wnd_funs** [ ])(float)

© 2005-2021 HörTech gGmbH, Oldenburg

6.173.1 Variable Documentation

6.173.1.1 wnd_funs `float(* wnd_funs[])(float)`

6.174 mha_windowparser.h File Reference

Classes

- class **MHAWindow::base_t**
  Common base for window types.
- class **MHAWindow::fun_t**
  Generic window based on a generator function.
- class **MHAWindow::rect_t**
  Rectangular window.
- class **MHAWindow::bartlett_t**
  Bartlett window.
- class **MHAWindow::hanning_t**
  von-Hann window
- class **MHAWindow::hamming_t**
  Hamming window.
- class **MHAWindow::blackman_t**
  Blackman window.
- class **MHAWindow::user_t**
  User defined window.
- class **MHAParser::window_t**
  MHA configuration interface for a window function generator.

Namespaces

- **MHAWindow**
  Collection of Window types.
- **MHAParser**
  Name space for the openMHA-Parser configuration language.

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- float **MHAWindow::rect** (float)
  Rectangular window function.
- float **MHAWindow::bartlett** (float)
  Bartlett window function.
- float **MHAWindow::hanning** (float)
  Hanning window function.
- float **MHAWindow::hamming** (float)
  Hamming window function.
- float **MHAWindow::blackman** (float)
  Blackman window function.

6.175 mhachain.cpp File Reference

Classes

- class **mhachain::mhachain_t**

Namespaces

- **mhachain**

6.176 mhafw_lib.cpp File Reference

6.177 mhafw_lib.h File Reference

Classes

- class **io_lib_t**
  Class for loading MHA sound IO module.
- class **fw_vars_t**
- class **fw_t**

6.178 MHAIOalsa.cpp File Reference

Classes

- class **alsa_base_t**
- class **alsa_dev_par_parser_t**
  Parser variables corresponding to one alsa device.
- class **alsa_t**$<$ **T** $>$
  Our representation of one alsa device.
- class **io_alsa_t**
  MHA IO interface class for ALSA IO.

© 2005-2021 HörTech gGmbH, Oldenburg

Macros

- #define **DBG**(x) fprintf(stderr,"%s:%d\n",__FILE__,__LINE__)
- #define **ERR_SUCCESS** 0
- #define **ERR_IHANDLE** -1
- #define **ERR_USER** -1000
- #define **MAX_USER_ERR** 0x500
- #define **IOInit** MHA_STATIC_MHAIOalsa_IOInit
- #define **IOPrepare** MHA_STATIC_MHAIOalsa_IOPrepare
- #define **IOStart** MHA_STATIC_MHAIOalsa_IOStart
- #define **IOStop** MHA_STATIC_MHAIOalsa_IOStop
- #define **IORelease** MHA_STATIC_MHAIOalsa_IORelease
- #define **IOSetVar** MHA_STATIC_MHAIOalsa_IOSetVar
- #define **IOStrError** MHA_STATIC_MHAIOalsa_IOStrError
- #define **IODestroy** MHA_STATIC_MHAIOalsa_IODestroy
- #define **dummy_interface_test** MHA_STATIC_MHAIOalsa_dummy_interface_test

Functions

- int **IOInit** (int fragsize, float samplerate, **IOProcessEvent_t** proc_event, void *proc_handle, **IOStartedEvent_t** start_event, void *start_handle, **IOStoppedEvent_t** stop_event, void *stop_handle, void **handle)

  IO library initialization function, called by framework after loading this IO library into the MHA process.
- int **IOPrepare** (void *handle, int nch_in, int nch_out)

  IO library prepare function, called after the MHA prepared the processing plugins.
- int **IOStart** (void *handle)
- int **IOStop** (void *handle)
- int **IORelease** (void *handle)
- int **IOSetVar** (void *handle, const char *command, char *retval, unsigned int maxretlen)
- const char * **IOStrError** (void *, int err)
- void **IODestroy** (void *handle)

Variables

- static char **user_err_msg** [**MAX_USER_ERR**]

6.178.1 Macro Definition Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.178.1.1 DBG

```
#define DBG(
    x ) fprintf(stderr,"%s:%d\n",__FILE__,__LINE__)
```

6.178.1.2 ERR_SUCCESS

```
#define ERR_SUCCESS 0
```

6.178.1.3 ERR_IHANDLE

```
#define ERR_IHANDLE -1
```

6.178.1.4 ERR_USER

```
#define ERR_USER -1000
```

6.178.1.5 MAX_USER_ERR

```
#define MAX_USER_ERR 0x500
```

6.178.1.6 IOInit

```
#define IOInit MHA_STATIC_MHAIOalsa_IOInit
```

6.178.1.7 IOPrepare

```
#define IOPrepare MHA_STATIC_MHAIOalsa_IOPrepare
```

6.178.1.8 IOStart

```
#define IOStart MHA_STATIC_MHAIOalsa_IOStart
```

6.178.1.9 IOStop

```
#define IOStop MHA_STATIC_MHAIOalsa_IOStop
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.178.1.10 IORelease `#define IORelease MHA_STATIC_MHAIOalsa_IORelease`

6.178.1.11 IOSetVar `#define IOSetVar MHA_STATIC_MHAIOalsa_IOSetVar`

6.178.1.12 IOStrError `#define IOStrError MHA_STATIC_MHAIOalsa_IOStrError`

6.178.1.13 IODestroy `#define IODestroy MHA_STATIC_MHAIOalsa_IODestroy`

6.178.1.14 dummy_interface_test `void dummy_interface_test MHA_STATIC_MHAIOalsa_↩ dummy_interface_test`

6.178.2 Function Documentation

6.178.2.1 IOInit()

```
int IOInit (
    int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle,
    void ** handle )
```

IO library initialization function, called by framework after loading this IO library into the MHA process.

Gives plugin callback functions and callback handles to interact with the MHA framework.

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

handle	output parameter. IO library returns pointer to void to the caller via this parameter. All other function calls from the MHA framework will use this handle.

6.178.2.2 IOPrepare()

```
int IOPrepare (
    void * handle,
    int nch_in,
    int nch_out )
```

IO library prepare function, called after the MHA prepared the processing plugins.

6.178.2.3 IOStart()

```
int IOStart (
    void * handle )
```

6.178.2.4 IOStop()

```
int IOStop (
    void * handle )
```

6.178.2.5 IORelease()

```
int IORelease (
    void * handle )
```

6.178.2.6 IOSetVar()

```
int IOSetVar (
    void * handle,
    const char * command,
    char * retval,
    unsigned int maxretlen )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.178.2.7 IOStrError()

```
const char* IOStrError (
    void * ,
    int err )
```

6.178.2.8 IODestroy()

```
void IODestroy (
    void * handle )
```

6.178.3 Variable Documentation

6.178.3.1 user_err_msg

```
char user_err_msg[ MAX_USER_ERR] [static]
```

6.179 MHAIOAsterisk.cpp File Reference

Classes

- class **io_asterisk_parser_t**
  The parser interface of the IOAsterisk library.
- class **io_asterisk_sound_t**
  Sound data handling of io tcp library.
- class **io_asterisk_fwcb_t**
  TCP sound-io library's interface to the framework callbacks.
- class **io_asterisk_t**
  The tcp sound io library.

Macros

- #define **ERR_SUCCESS** 0
- #define **ERR_IHANDLE** -1
- #define **ERR_USER** -1000
- #define **MAX_USER_ERR** 0x2000
- #define **MHA_ErrorMsg2**(x, y) **MHA_Error**(__FILE__,__LINE__,(x),(y))
- #define **MHA_ErrorMsg3**(x, y, z) **MHA_Error**(__FILE__,__LINE__,(x),(y),(z))
- #define **MIN_TCP_PORT** 0
- #define **MIN_TCP_PORT_STR** "0"
- #define **MAX_TCP_PORT** 65535
- #define **MAX_TCP_PORT_STR** "65535"

© 2005-2021 HörTech gGmbH, Oldenburg

- #define **IOInit** MHA_STATIC_MHAIOTCP_IOInit
- #define **IOPrepare** MHA_STATIC_MHAIOTCP_IOPrepare
- #define **IOStart** MHA_STATIC_MHAIOTCP_IOStart
- #define **IOStop** MHA_STATIC_MHAIOTCP_IOStop
- #define **IORelease** MHA_STATIC_MHAIOTCP_IORelease
- #define **IOSetVar** MHA_STATIC_MHAIOTCP_IOSetVar
- #define **IOStrError** MHA_STATIC_MHAIOTCP_IOStrError
- #define **IODestroy** MHA_STATIC_MHAIOTCP_IODestroy
- #define **dummy_interface_test** MHA_STATIC_MHAIOTCP_dummy_interface_test

Functions

- static int **copy_error** (**MHA_Error** &e)
- static void * **thread_startup_function** (void *parameter)
- int **IOInit** (int fragsize, float samplerate, **IOProcessEvent_t** proc_event, void *proc_handle, **IOStartedEvent_t** start_event, void *start_handle, **IOStoppedEvent_t** stop_event, void *stop_handle, void **handle)
- int **IOPrepare** (void *handle, int num_inchannels, int num_outchannels)
- int **IOStart** (void *handle)
- int **IOStop** (void *handle)
- int **IORelease** (void *handle)
- int **IOSetVar** (void *handle, const char *cmd, char *retval, unsigned int len)
- const char * **IOStrError** (void *handle, int err)
- void **IODestroy** (void *handle)

Variables

- static char **user_err_msg** [**MAX_USER_ERR**]

6.179.1 Macro Definition Documentation

6.179.1.1 ERR_SUCCESS `#define ERR_SUCCESS 0`

6.179.1.2 ERR_IHANDLE `#define ERR_IHANDLE -1`

© 2005-2021 HörTech gGmbH, Oldenburg

6.179.1.3 ERR_USER

```
#define ERR_USER -1000
```

6.179.1.4 MAX_USER_ERR

```
#define MAX_USER_ERR 0x2000
```

6.179.1.5 MHA_ErrorMsg2

```
#define MHA_ErrorMsg2(
    x,
    y )  MHA_Error(__FILE__,__LINE__,(x),(y))
```

6.179.1.6 MHA_ErrorMsg3

```
#define MHA_ErrorMsg3(
    x,
    y,
    z )  MHA_Error(__FILE__,__LINE__,(x),(y),(z))
```

6.179.1.7 MIN_TCP_PORT

```
#define MIN_TCP_PORT 0
```

6.179.1.8 MIN_TCP_PORT_STR

```
#define MIN_TCP_PORT_STR "0"
```

6.179.1.9 MAX_TCP_PORT

```
#define MAX_TCP_PORT 65535
```

6.179.1.10 MAX_TCP_PORT_STR

```
#define MAX_TCP_PORT_STR "65535"
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.179.1.11 IOInit

```
#define IOInit MHA_STATIC_MHAIOTCP_IOInit
```

6.179.1.12 IOPrepare

```
#define IOPrepare MHA_STATIC_MHAIOTCP_IOPrepare
```

6.179.1.13 IOStart

```
#define IOStart MHA_STATIC_MHAIOTCP_IOStart
```

6.179.1.14 IOStop

```
#define IOStop MHA_STATIC_MHAIOTCP_IOStop
```

6.179.1.15 IORelease

```
#define IORelease MHA_STATIC_MHAIOTCP_IORelease
```

6.179.1.16 IOSetVar

```
#define IOSetVar MHA_STATIC_MHAIOTCP_IOSetVar
```

6.179.1.17 IOStrError

```
#define IOStrError MHA_STATIC_MHAIOTCP_IOStrError
```

6.179.1.18 IODestroy

```
#define IODestroy MHA_STATIC_MHAIOTCP_IODestroy
```

6.179.1.19 dummy_interface_test

```
#define dummy_interface_test MHA_STATIC_MHAIOTC↩
P_dummy_interface_test
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.179.2 Function Documentation

6.179.2.1 copy_error()

```
static int copy_error (
    MHA_Error & e ) [static]
```

6.179.2.2 thread_startup_function()

```
static void* thread_startup_function (
    void * parameter ) [static]
```

6.179.2.3 IOInit()

```
int IOInit (
    int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle,
    void ** handle )
```

6.179.2.4 IOPrepare()

```
int IOPrepare (
    void * handle,
    int num_inchannels,
    int num_outchannels )
```

6.179.2.5 IOStart()

```
int IOStart (
    void * handle )
```

6.179.2.6 IOStop()

```
int IOStop (
    void * handle )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.179.2.7 IORelease()

```
int IORelease (
    void * handle )
```

6.179.2.8 IOSetVar()

```
int IOSetVar (
    void * handle,
    const char * cmd,
    char * retval,
    unsigned int len )
```

6.179.2.9 IOStrError()

```
const char* IOStrError (
    void * handle,
    int err )
```

6.179.2.10 IODestroy()

```
void IODestroy (
    void * handle )
```

6.179.3 Variable Documentation

6.179.3.1 user_err_msg

```
char user_err_msg[ MAX_USER_ERR] [static]
```

6.180 MHAIOFile.cpp File Reference

Classes

- class **io_file_t**

  File IO.

© 2005-2021 HörTech gGmbH, Oldenburg

Macros

- #define **DEBUG**(x) std::cerr << __FILE__ << ":" << __LINE__ << " " << #x " = " << x << std::endl
- #define **ERR_SUCCESS** 0
- #define **ERR_IHANDLE** -1
- #define **ERR_USER** -1000
- #define **MAX_USER_ERR** 0x500
- #define **IOInit** MHA_STATIC_MHAIOFile_IOInit
- #define **IOPrepare** MHA_STATIC_MHAIOFile_IOPrepare
- #define **IOStart** MHA_STATIC_MHAIOFile_IOStart
- #define **IOStop** MHA_STATIC_MHAIOFile_IOStop
- #define **IORelease** MHA_STATIC_MHAIOFile_IORelease
- #define **IOSetVar** MHA_STATIC_MHAIOFile_IOSetVar
- #define **IOStrError** MHA_STATIC_MHAIOFile_IOStrError
- #define **IODestroy** MHA_STATIC_MHAIOFile_IODestroy
- #define **dummy_interface_test** MHA_STATIC_MHAIOFile_dummy_interface_test

Functions

- int **IOInit** (int fragsize, float samplerate, **IOProcessEvent_t** proc_event, void *proc_handle, **IOStartedEvent_t** start_event, void *start_handle, **IOStoppedEvent_t** stop_event, void *stop_handle, void **handle)
- int **IOPrepare** (void *handle, int nch_in, int nch_out)
- int **IOStart** (void *handle)
- int **IOStop** (void *handle)
- int **IORelease** (void *handle)
- int **IOSetVar** (void *handle, const char *command, char *retval, unsigned int maxretlen)
- const char * **IOStrError** (void *, int err)
- void **IODestroy** (void *handle)

Variables

- static char **user_err_msg** [**MAX_USER_ERR**]

6.180.1 Macro Definition Documentation

6.180.1.1 DEBUG

```
#define DEBUG(
            x ) std::cerr << __FILE__ << ":" << __LINE__ << " " << #x " = " <<
x << std::endl
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.180.1.2 ERR_SUCCESS `#define ERR_SUCCESS 0`

6.180.1.3 ERR_IHANDLE `#define ERR_IHANDLE -1`

6.180.1.4 ERR_USER `#define ERR_USER -1000`

6.180.1.5 MAX_USER_ERR `#define MAX_USER_ERR 0x500`

6.180.1.6 IOInit `#define IOInit MHA_STATIC_MHAIOFile_IOInit`

6.180.1.7 IOPrepare `#define IOPrepare MHA_STATIC_MHAIOFile_IOPrepare`

6.180.1.8 IOStart `#define IOStart MHA_STATIC_MHAIOFile_IOStart`

6.180.1.9 IOStop `#define IOStop MHA_STATIC_MHAIOFile_IOStop`

6.180.1.10 IORelease `#define IORelease MHA_STATIC_MHAIOFile_IORelease`

© 2005-2021 HörTech gGmbH, Oldenburg

6.180.1.11 IOSetVar `#define IOSetVar MHA_STATIC_MHAIOFile_IOSetVar`

6.180.1.12 IOStrError `#define IOStrError MHA_STATIC_MHAIOFile_IOStrError`

6.180.1.13 IODestroy `#define IODestroy MHA_STATIC_MHAIOFile_IODestroy`

6.180.1.14 dummy_interface_test `#define dummy_interface_test MHA_STATIC_MHAIO↩ File_dummy_interface_test`

6.180.2 Function Documentation

6.180.2.1 IOInit()

```
int IOInit (
    int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle,
    void ** handle )
```

6.180.2.2 IOPrepare()

```
int IOPrepare (
    void * handle,
    int nch_in,
    int nch_out )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.180.2.3 IOStart()

```
int IOStart (
    void * handle )
```

6.180.2.4 IOStop()

```
int IOStop (
    void * handle )
```

6.180.2.5 IORelease()

```
int IORelease (
    void * handle )
```

6.180.2.6 IOSetVar()

```
int IOSetVar (
    void * handle,
    const char * command,
    char * retval,
    unsigned int maxretlen )
```

6.180.2.7 IOStrError()

```
const char* IOStrError (
    void * ,
    int err )
```

6.180.2.8 IODestroy()

```
void IODestroy (
    void * handle )
```

6.180.3 Variable Documentation

6.180.3.1 user_err_msg

```
char user_err_msg[ MAX_USER_ERR] [static]
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.181 MHAIOJack.cpp File Reference

Classes

- class **MHAIOJack::io_jack_t**
  Main class for JACK IO.

Namespaces

- **MHAIOJack**
  JACK IO.

Macros

- #define **ERR_SUCCESS** 0
- #define **ERR_IHANDLE** -1
- #define **ERR_USER** -1000
- #define **MAX_USER_ERR** 0x500
- #define **IOInit** MHA_STATIC_MHAIOJack_IOInit
- #define **IOPrepare** MHA_STATIC_MHAIOJack_IOPrepare
- #define **IOStart** MHA_STATIC_MHAIOJack_IOStart
- #define **IOStop** MHA_STATIC_MHAIOJack_IOStop
- #define **IORelease** MHA_STATIC_MHAIOJack_IORelease
- #define **IOSetVar** MHA_STATIC_MHAIOJack_IOSetVar
- #define **IOStrError** MHA_STATIC_MHAIOJack_IOStrError
- #define **IODestroy** MHA_STATIC_MHAIOJack_IODestroy
- #define **dummy_interface_test** MHA_STATIC_MHAIOJack_dummy_interface_test

Functions

- int **IOInit** (int fragsize, float samplerate, **IOProcessEvent_t** proc_event, void ∗proc_handle, **IOStartedEvent_t** start_event, void ∗start_handle, **IOStoppedEvent_t** stop_event, void ∗stop_handle, void ∗∗handle)
- int **IOPrepare** (void ∗handle, int nch_in, int nch_out)
- int **IOStart** (void ∗handle)
- int **IOStop** (void ∗handle)
- int **IORelease** (void ∗handle)
- int **IOSetVar** (void ∗handle, const char ∗command, char ∗retval, unsigned int maxretlen)
- const char ∗ **IOStrError** (void ∗, int err)
- void **IODestroy** (void ∗handle)

© 2005-2021 HörTech gGmbH, Oldenburg

Variables

- static char **user_err_msg** [**MAX_USER_ERR**] = ""

6.181.1 Macro Definition Documentation

6.181.1.1 ERR_SUCCESS `#define ERR_SUCCESS 0`

6.181.1.2 ERR_IHANDLE `#define ERR_IHANDLE -1`

6.181.1.3 ERR_USER `#define ERR_USER -1000`

6.181.1.4 MAX_USER_ERR `#define MAX_USER_ERR 0x500`

6.181.1.5 IOInit `#define IOInit MHA_STATIC_MHAIOJack_IOInit`

6.181.1.6 IOPrepare `#define IOPrepare MHA_STATIC_MHAIOJack_IOPrepare`

6.181.1.7 IOStart `#define IOStart MHA_STATIC_MHAIOJack_IOStart`

© 2005-2021 HörTech gGmbH, Oldenburg

6.181.1.8 IOStop

```
#define IOStop MHA_STATIC_MHAIOJack_IOStop
```

6.181.1.9 IORelease

```
#define IORelease MHA_STATIC_MHAIOJack_IORelease
```

6.181.1.10 IOSetVar

```
#define IOSetVar MHA_STATIC_MHAIOJack_IOSetVar
```

6.181.1.11 IOStrError

```
#define IOStrError MHA_STATIC_MHAIOJack_IOStrError
```

6.181.1.12 IODestroy

```
#define IODestroy MHA_STATIC_MHAIOJack_IODestroy
```

6.181.1.13 dummy_interface_test

```
#define dummy_interface_test MHA_STATIC_MHAIO↩
Jack_dummy_interface_test
```

6.181.2 Function Documentation

6.181.2.1 IOInit()

```
int IOInit (
    int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle,
    void ** handle )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.181.2.2 IOPrepare()

```
int IOPrepare (
    void * handle,
    int nch_in,
    int nch_out )
```

6.181.2.3 IOStart()

```
int IOStart (
    void * handle )
```

6.181.2.4 IOStop()

```
int IOStop (
    void * handle )
```

6.181.2.5 IORelease()

```
int IORelease (
    void * handle )
```

6.181.2.6 IOSetVar()

```
int IOSetVar (
    void * handle,
    const char * command,
    char * retval,
    unsigned int maxretlen )
```

6.181.2.7 IOStrError()

```
const char* IOStrError (
    void * ,
    int err )
```

6.181.2.8 IODestroy()

```
void IODestroy (
    void * handle )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.181.3 Variable Documentation

6.181.3.1 user_err_msg `char user_err_msg[ MAX_USER_ERR] = "" [static]`

6.182 MHAIOJackdb.cpp File Reference

Classes

- class **MHAIOJackdb::io_jack_t**
  Main class for JACK IO.

Namespaces

- **MHAIOJackdb**

Macros

- #define **ERR_SUCCESS** 0
- #define **ERR_IHANDLE** -1
- #define **ERR_USER** -1000
- #define **MAX_USER_ERR** 0x500
- #define **IOInit** MHA_STATIC_MHAIOJackdb_IOInit
- #define **IOPrepare** MHA_STATIC_MHAIOJackdb_IOPrepare
- #define **IOStart** MHA_STATIC_MHAIOJackdb_IOStart
- #define **IOStop** MHA_STATIC_MHAIOJackdb_IOStop
- #define **IORelease** MHA_STATIC_MHAIOJackdb_IORelease
- #define **IOSetVar** MHA_STATIC_MHAIOJackdb_IOSetVar
- #define **IOStrError** MHA_STATIC_MHAIOJackdb_IOStrError
- #define **IODestroy** MHA_STATIC_MHAIOJackdb_IODestroy
- #define **dummy_interface_test** MHA_STATIC_MHAIOJackdb_dummy_interface_test

Functions

- int **IOInit** (int fragsize, float samplerate, **IOProcessEvent_t** proc_event, void ∗proc_handle, **IOStartedEvent_t** start_event, void ∗start_handle, **IOStoppedEvent_t** stop_event, void ∗stop_handle, void ∗∗handle)
- int **IOPrepare** (void ∗handle, int nch_in, int nch_out)
- int **IOStart** (void ∗handle)
- int **IOStop** (void ∗handle)
- int **IORelease** (void ∗handle)
- int **IOSetVar** (void ∗handle, const char ∗command, char ∗retval, unsigned int maxretlen)
- const char ∗ **IOStrError** (void ∗, int err)
- void **IODestroy** (void ∗handle)

© 2005-2021 HörTech gGmbH, Oldenburg

Variables

- static char **user_err_msg** [**MAX_USER_ERR**] = ""

6.182.1 Macro Definition Documentation

6.182.1.1 ERR_SUCCESS `#define ERR_SUCCESS 0`

6.182.1.2 ERR_IHANDLE `#define ERR_IHANDLE -1`

6.182.1.3 ERR_USER `#define ERR_USER -1000`

6.182.1.4 MAX_USER_ERR `#define MAX_USER_ERR 0x500`

6.182.1.5 IOInit `#define IOInit MHA_STATIC_MHAIOJackdb_IOInit`

6.182.1.6 IOPrepare `#define IOPrepare MHA_STATIC_MHAIOJackdb_IOPrepare`

6.182.1.7 IOStart `#define IOStart MHA_STATIC_MHAIOJackdb_IOStart`

© 2005-2021 HörTech gGmbH, Oldenburg

6.182.1.8 IOStop `#define IOStop MHA_STATIC_MHAIOJackdb_IOStop`

6.182.1.9 IORelease `#define IORelease MHA_STATIC_MHAIOJackdb_IORelease`

6.182.1.10 IOSetVar `#define IOSetVar MHA_STATIC_MHAIOJackdb_IOSetVar`

6.182.1.11 IOStrError `#define IOStrError MHA_STATIC_MHAIOJackdb_IOStrError`

6.182.1.12 IODestroy `#define IODestroy MHA_STATIC_MHAIOJackdb_IODestroy`

6.182.1.13 dummy_interface_test `#define dummy_interface_test MHA_STATIC_MHAIO↩ Jackdb_dummy_interface_test`

6.182.2 Function Documentation

6.182.2.1 IOInit()

```
int IOInit (
    int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle,
    void ** handle )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.182.2.2 IOPrepare()

```
int IOPrepare (
    void * handle,
    int nch_in,
    int nch_out )
```

6.182.2.3 IOStart()

```
int IOStart (
    void * handle )
```

6.182.2.4 IOStop()

```
int IOStop (
    void * handle )
```

6.182.2.5 IORelease()

```
int IORelease (
    void * handle )
```

6.182.2.6 IOSetVar()

```
int IOSetVar (
    void * handle,
    const char * command,
    char * retval,
    unsigned int maxretlen )
```

6.182.2.7 IOStrError()

```
const char* IOStrError (
    void * ,
    int err )
```

6.182.2.8 IODestroy()

```
void IODestroy (
    void * handle )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.182.3 Variable Documentation

6.182.3.1 user_err_msg `char user_err_msg[ MAX_USER_ERR] = "" [static]`

6.183 MHAIOParser.cpp File Reference

Classes

- class **io_parser_t**
  Main class for Parser IO.

Macros

- #define **ERR_SUCCESS** 0
- #define **ERR_IHANDLE** -1
- #define **ERR_USER** -1000
- #define **MAX_USER_ERR** 0x500
- #define **IOInit** MHA_STATIC_MHAIOParser_IOInit
- #define **IOPrepare** MHA_STATIC_MHAIOParser_IOPrepare
- #define **IOStart** MHA_STATIC_MHAIOParser_IOStart
- #define **IOStop** MHA_STATIC_MHAIOParser_IOStop
- #define **IORelease** MHA_STATIC_MHAIOParser_IORelease
- #define **IOSetVar** MHA_STATIC_MHAIOParser_IOSetVar
- #define **IOStrError** MHA_STATIC_MHAIOParser_IOStrError
- #define **IODestroy** MHA_STATIC_MHAIOParser_IODestroy
- #define **dummy_interface_test** MHA_STATIC_MHAIOParser_dummy_interface_test

Functions

- int **IOInit** (int fragsize, float, **IOProcessEvent_t** proc_event, void ∗proc_handle, **IOStartedEvent_t** start_event, void ∗start_handle, **IOStoppedEvent_t** stop_event, void ∗stop_handle, void ∗∗handle)
- int **IOPrepare** (void ∗handle, int nch_in, int nch_out)
- int **IOStart** (void ∗handle)
- int **IOStop** (void ∗handle)
- int **IORelease** (void ∗handle)
- int **IOSetVar** (void ∗handle, const char ∗command, char ∗retval, unsigned int maxretlen)
- const char ∗ **IOStrError** (void ∗, int err)
- void **IODestroy** (void ∗handle)

© 2005-2021 HörTech gGmbH, Oldenburg

Variables

- static char **user_err_msg** [**MAX_USER_ERR**]

6.183.1 Macro Definition Documentation

6.183.1.1 ERR_SUCCESS `#define ERR_SUCCESS 0`

6.183.1.2 ERR_IHANDLE `#define ERR_IHANDLE -1`

6.183.1.3 ERR_USER `#define ERR_USER -1000`

6.183.1.4 MAX_USER_ERR `#define MAX_USER_ERR 0x500`

6.183.1.5 IOInit `#define IOInit MHA_STATIC_MHAIOParser_IOInit`

6.183.1.6 IOPrepare `#define IOPrepare MHA_STATIC_MHAIOParser_IOPrepare`

6.183.1.7 IOStart `#define IOStart MHA_STATIC_MHAIOParser_IOStart`

© 2005-2021 HörTech gGmbH, Oldenburg

6.183.1.8 IOStop

```
#define IOStop MHA_STATIC_MHAIOParser_IOStop
```

6.183.1.9 IORelease

```
#define IORelease MHA_STATIC_MHAIOParser_IORelease
```

6.183.1.10 IOSetVar

```
#define IOSetVar MHA_STATIC_MHAIOParser_IOSetVar
```

6.183.1.11 IOStrError

```
#define IOStrError MHA_STATIC_MHAIOParser_IOStrError
```

6.183.1.12 IODestroy

```
#define IODestroy MHA_STATIC_MHAIOParser_IODestroy
```

6.183.1.13 dummy_interface_test

```
#define dummy_interface_test MHA_STATIC_MHAIO↩
Parser_dummy_interface_test
```

6.183.2 Function Documentation

6.183.2.1 IOInit()

```
int IOInit (
    int fragsize,
    float ,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle,
    void ** handle )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.183.2.2 IOPrepare()

```
int IOPrepare (
    void * handle,
    int nch_in,
    int nch_out )
```

6.183.2.3 IOStart()

```
int IOStart (
    void * handle )
```

6.183.2.4 IOStop()

```
int IOStop (
    void * handle )
```

6.183.2.5 IORelease()

```
int IORelease (
    void * handle )
```

6.183.2.6 IOSetVar()

```
int IOSetVar (
    void * handle,
    const char * command,
    char * retval,
    unsigned int maxretlen )
```

6.183.2.7 IOStrError()

```
const char* IOStrError (
    void * ,
    int err )
```

6.183.2.8 IODestroy()

```
void IODestroy (
    void * handle )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.183.3 Variable Documentation

6.183.3.1 user_err_msg `char user_err_msg[ MAX_USER_ERR] [static]`

6.184 MHAIOPortAudio.cpp File Reference

Classes

- class **MHAIOPortAudio::stream_info_t**
- class **MHAIOPortAudio::device_info_t**
- class **MHAIOPortAudio::io_portaudio_t**

  Main class for Portaudio sound IO.

Namespaces

- **MHAIOPortAudio**

Macros

- #define **ERR_SUCCESS** 0
- #define **ERR_IHANDLE** -1
- #define **ERR_USER** -1000
- #define **MAX_USER_ERR** 0x500
- #define **IOInit** MHA_STATIC_MHAIOPortAudio_IOInit
- #define **IOPrepare** MHA_STATIC_MHAIOPortAudio_IOPrepare
- #define **IOStart** MHA_STATIC_MHAIOPortAudio_IOStart
- #define **IOStop** MHA_STATIC_MHAIOPortAudio_IOStop
- #define **IORelease** MHA_STATIC_MHAIOPortAudio_IORelease
- #define **IOSetVar** MHA_STATIC_MHAIOPortAudio_IOSetVar
- #define **IOStrError** MHA_STATIC_MHAIOPortAudio_IOStrError
- #define **IODestroy** MHA_STATIC_MHAIOPortAudio_IODestroy
- #define **dummy_interface_test** MHA_STATIC_MHAIOPortAudio_dummy_interface_↩ test

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- static std::string **MHAIOPortAudio::parserFriendlyName** (const std::string &in)
- int **portaudio_callback** (const void ∗input, void ∗output, unsigned long frameCount, const PaStreamCallbackTimeInfo ∗timeInfo, PaStreamCallbackFlags statusFlags, void ∗userData)
- int **IOInit** (int fragsize, float samplerate, **IOProcessEvent_t** proc_event, void ∗proc_handle, **IOStartedEvent_t** start_event, void ∗start_handle, **IOStoppedEvent_t** stop_event, void ∗stop_handle, void ∗∗handle)
- int **IOPrepare** (void ∗handle, int nch_in, int nch_out)
- int **IOStart** (void ∗handle)
- int **IOStop** (void ∗handle)
- int **IORelease** (void ∗handle)
- int **IOSetVar** (void ∗handle, const char ∗command, char ∗retval, unsigned int maxretlen)
- const char ∗ **IOStrError** (void ∗, int err)
- void **IODestroy** (void ∗handle)

Variables

- static char **user_err_msg** [**MAX_USER_ERR**] = ""
- PaStreamCallback **portaudio_callback**

6.184.1 Macro Definition Documentation

6.184.1.1 ERR_SUCCESS `#define ERR_SUCCESS 0`

6.184.1.2 ERR_IHANDLE `#define ERR_IHANDLE -1`

6.184.1.3 ERR_USER `#define ERR_USER -1000`

6.184.1.4 MAX_USER_ERR `#define MAX_USER_ERR 0x500`

© 2005-2021 HörTech gGmbH, Oldenburg

6.184.1.5 IOInit

```
#define IOInit MHA_STATIC_MHAIOPortAudio_IOInit
```

6.184.1.6 IOPrepare

```
#define IOPrepare MHA_STATIC_MHAIOPortAudio_IOPrepare
```

6.184.1.7 IOStart

```
#define IOStart MHA_STATIC_MHAIOPortAudio_IOStart
```

6.184.1.8 IOStop

```
#define IOStop MHA_STATIC_MHAIOPortAudio_IOStop
```

6.184.1.9 IORelease

```
#define IORelease MHA_STATIC_MHAIOPortAudio_IORelease
```

6.184.1.10 IOSetVar

```
#define IOSetVar MHA_STATIC_MHAIOPortAudio_IOSetVar
```

6.184.1.11 IOStrError

```
#define IOStrError MHA_STATIC_MHAIOPortAudio_IOStrError
```

6.184.1.12 IODestroy

```
#define IODestroy MHA_STATIC_MHAIOPortAudio_IODestroy
```

6.184.1.13 dummy_interface_test

```
#define dummy_interface_test MHA_STATIC_MHAIO↩
PortAudio_dummy_interface_test
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.184.2 Function Documentation

6.184.2.1 portaudio_callback()

```
int portaudio_callback (
    const void * input,
    void * output,
    unsigned long frameCount,
    const PaStreamCallbackTimeInfo * timeInfo,
    PaStreamCallbackFlags statusFlags,
    void * userData )
```

6.184.2.2 IOInit()

```
int IOInit (
    int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle,
    void ** handle )
```

6.184.2.3 IOPrepare()

```
int IOPrepare (
    void * handle,
    int nch_in,
    int nch_out )
```

6.184.2.4 IOStart()

```
int IOStart (
    void * handle )
```

6.184.2.5 IOStop()

```
int IOStop (
    void * handle )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.184.2.6 IORelease()

```
int IORelease (
    void * handle )
```

6.184.2.7 IOSetVar()

```
int IOSetVar (
    void * handle,
    const char * command,
    char * retval,
    unsigned int maxretlen )
```

6.184.2.8 IOStrError()

```
const char* IOStrError (
    void * ,
    int err )
```

6.184.2.9 IODestroy()

```
void IODestroy (
    void * handle )
```

6.184.3 Variable Documentation

6.184.3.1 user_err_msg

```
char user_err_msg[ MAX_USER_ERR] = "" [static]
```

6.184.3.2 portaudio_callback

```
PaStreamCallback portaudio_callback
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.185 MHAIOTCP.cpp File Reference

Classes

- class **io_tcp_parser_t**

  The parser interface of the IOTCP library.
- class **io_tcp_sound_t**

  Sound data handling of io tcp library.
- union **io_tcp_sound_t::float_union**

  This union helps in conversion of floats from host byte order to network byte order and back again.
- class **io_tcp_fwcb_t**

  TCP sound-io library's interface to the framework callbacks.
- class **io_tcp_t**

  The tcp sound io library.

Macros

- #define **ERR_SUCCESS** 0
- #define **ERR_IHANDLE** -1
- #define **ERR_USER** -1000
- #define **MAX_USER_ERR** 0x2000
- #define **MHA_ErrorMsg2**(x, y) **MHA_Error**(__FILE__,__LINE__,(x),(y))
- #define **MHA_ErrorMsg3**(x, y, z) **MHA_Error**(__FILE__,__LINE__,(x),(y),(z))
- #define **MIN_TCP_PORT** 0
- #define **MIN_TCP_PORT_STR** "0"
- #define **MAX_TCP_PORT** 65535
- #define **MAX_TCP_PORT_STR** "65535"
- #define **IOInit** MHA_STATIC_MHAIOTCP_IOInit
- #define **IOPrepare** MHA_STATIC_MHAIOTCP_IOPrepare
- #define **IOStart** MHA_STATIC_MHAIOTCP_IOStart
- #define **IOStop** MHA_STATIC_MHAIOTCP_IOStop
- #define **IORelease** MHA_STATIC_MHAIOTCP_IORelease
- #define **IOSetVar** MHA_STATIC_MHAIOTCP_IOSetVar
- #define **IOStrError** MHA_STATIC_MHAIOTCP_IOStrError
- #define **IODestroy** MHA_STATIC_MHAIOTCP_IODestroy
- #define **dummy_interface_test** MHA_STATIC_MHAIOTCP_dummy_interface_test

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- static int **copy_error** (**MHA_Error** &e)
- static void * **thread_startup_function** (void *parameter)
- int **IOInit** (int fragsize, float samplerate, **IOProcessEvent_t** proc_event, void *proc_↩handle, **IOStartedEvent_t** start_event, void *start_handle, **IOStoppedEvent_t** stop_↩event, void *stop_handle, void **handle)
- int **IOPrepare** (void *handle, int num_inchannels, int num_outchannels)
- int **IOStart** (void *handle)
- int **IOStop** (void *handle)
- int **IORelease** (void *handle)
- int **IOSetVar** (void *handle, const char *cmd, char *retval, unsigned int len)
- const char * **IOStrError** (void *handle, int err)
- void **IODestroy** (void *handle)

Variables

- static char **user_err_msg** [**MAX_USER_ERR**]

6.185.1 Macro Definition Documentation

6.185.1.1 ERR_SUCCESS `#define ERR_SUCCESS 0`

6.185.1.2 ERR_IHANDLE `#define ERR_IHANDLE -1`

6.185.1.3 ERR_USER `#define ERR_USER -1000`

6.185.1.4 MAX_USER_ERR `#define MAX_USER_ERR 0x2000`

© 2005-2021 HörTech gGmbH, Oldenburg

6.185.1.5 MHA_ErrorMsg2

```
#define MHA_ErrorMsg2(
    x,
    y ) MHA_Error(__FILE__,__LINE__,(x),(y))
```

6.185.1.6 MHA_ErrorMsg3

```
#define MHA_ErrorMsg3(
    x,
    y,
    z ) MHA_Error(__FILE__,__LINE__,(x),(y),(z))
```

6.185.1.7 MIN_TCP_PORT

```
#define MIN_TCP_PORT 0
```

6.185.1.8 MIN_TCP_PORT_STR

```
#define MIN_TCP_PORT_STR "0"
```

6.185.1.9 MAX_TCP_PORT

```
#define MAX_TCP_PORT 65535
```

6.185.1.10 MAX_TCP_PORT_STR

```
#define MAX_TCP_PORT_STR "65535"
```

6.185.1.11 IOInit

```
#define IOInit MHA_STATIC_MHAIOTCP_IOInit
```

6.185.1.12 IOPrepare

```
#define IOPrepare MHA_STATIC_MHAIOTCP_IOPrepare
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.185.1.13 IOStart `#define IOStart MHA_STATIC_MHAIOTCP_IOStart`

6.185.1.14 IOStop `#define IOStop MHA_STATIC_MHAIOTCP_IOStop`

6.185.1.15 IORelease `#define IORelease MHA_STATIC_MHAIOTCP_IORelease`

6.185.1.16 IOSetVar `#define IOSetVar MHA_STATIC_MHAIOTCP_IOSetVar`

6.185.1.17 IOStrError `#define IOStrError MHA_STATIC_MHAIOTCP_IOStrError`

6.185.1.18 IODestroy `#define IODestroy MHA_STATIC_MHAIOTCP_IODestroy`

6.185.1.19 dummy_interface_test `#define dummy_interface_test MHA_STATIC_MHAIOTC↩P_dummy_interface_test`

6.185.2 Function Documentation

6.185.2.1 copy_error() `static int copy_error (`
`MHA_Error & e ) [static]`

6.185.2.2 thread_startup_function()

```
static void* thread_startup_function (
    void * parameter ) [static]
```

6.185.2.3 IOInit()

```
int IOInit (
    int fragsize,
    float samplerate,
    IOProcessEvent_t proc_event,
    void * proc_handle,
    IOStartedEvent_t start_event,
    void * start_handle,
    IOStoppedEvent_t stop_event,
    void * stop_handle,
    void ** handle )
```

6.185.2.4 IOPrepare()

```
int IOPrepare (
    void * handle,
    int num_inchannels,
    int num_outchannels )
```

6.185.2.5 IOStart()

```
int IOStart (
    void * handle )
```

6.185.2.6 IOStop()

```
int IOStop (
    void * handle )
```

6.185.2.7 IORelease()

```
int IORelease (
    void * handle )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.185.2.8 IOSetVar()

```
int IOSetVar (
    void * handle,
    const char * cmd,
    char * retval,
    unsigned int len )
```

6.185.2.9 IOStrError()

```
const char* IOStrError (
    void * handle,
    int err )
```

6.185.2.10 IODestroy()

```
void IODestroy (
    void * handle )
```

6.185.3 Variable Documentation

6.185.3.1 user_err_msg

```
char user_err_msg[ MAX_USER_ERR] [static]
```

6.186 mhajack.cpp File Reference

Functions

- static void **jack_error_handler** (const char ∗msg)
- static int **dummy_jack_proc_cb** (jack_nframes_t, void ∗)
- void **make_friendly_number** (jack_default_audio_sample_t &x)

Variables

- char **last_jack_err_msg** [**MAX_USER_ERR**] = ""
- int **last_jack_err** = 0

6.186.1 Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.186.1.1 jack_error_handler()

```
static void jack_error_handler (
    const char * msg ) [static]
```

6.186.1.2 dummy_jack_proc_cb()

```
static int dummy_jack_proc_cb (
    jack_nframes_t ,
    void * ) [static]
```

6.186.1.3 make_friendly_number()

```
void make_friendly_number (
    jack_default_audio_sample_t & x ) [inline]
```

6.186.2 Variable Documentation

6.186.2.1 last_jack_err_msg

```
char last_jack_err_msg[ MAX_USER_ERR] = ""
```

6.186.2.2 last_jack_err

```
int last_jack_err = 0
```

6.187 mhajack.h File Reference

Classes

- class **MHAJack::port_t**
  Class for one channel/port.
- class **MHAJack::client_t**
  Generic asynchronous JACK client.
- class **MHAJack::client_noncont_t**
  Generic client for synchronous playback and recording of waveform fragments.
- class **MHAJack::client_avg_t**
  Generic JACK client for averaging a system response across time.

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **MHAJack**

  Classes and functions for openMHA and JACK interaction.

Macros

- #define **MHAJACK_FW_STARTED** 1
- #define **MHAJACK_STOPPED** 2
- #define **MHAJACK_STARTING** 8
- #define **IO_ERROR_JACK** 11
- #define **IO_ERROR_MHAJACKLIB** 12
- #define **MAX_USER_ERR** 0x500

Functions

- void **MHAJack::io** (**mha_wave_t** ∗s_out, **mha_wave_t** ∗s_in, const std::string &name, const std::vector< std::string > &p_out, const std::vector< std::string > &p_in, float ∗srate=NULL, unsigned int ∗fragsize=NULL, bool use_jack_transport=false)

  Functional form of generic client for synchronous playback and recording of waveform fragments.
- std::vector< unsigned int > **MHAJack::get_port_capture_latency** (const std::vector< std::string > &ports)

  Return the JACK port latency of ports.
- std::vector< int > **MHAJack::get_port_capture_latency_int** (const std::vector< std↩::string > &ports)

  Return the JACK port latency of ports.
- std::vector< unsigned int > **MHAJack::get_port_playback_latency** (const std::vector< std::string > &ports)

  Return the JACK port latency of ports.
- std::vector< int > **MHAJack::get_port_playback_latency_int** (const std::vector< std↩::string > &ports)

Variables

- char **last_jack_err_msg** [**MAX_USER_ERR**]

6.187.1 Macro Definition Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.187.1.1 MHAJACK_FW_STARTED `#define MHAJACK_FW_STARTED 1`

6.187.1.2 MHAJACK_STOPPED `#define MHAJACK_STOPPED 2`

6.187.1.3 MHAJACK_STARTING `#define MHAJACK_STARTING 8`

6.187.1.4 IO_ERROR_JACK `#define IO_ERROR_JACK 11`

6.187.1.5 IO_ERROR_MHAJACKLIB `#define IO_ERROR_MHAJACKLIB 12`

6.187.1.6 MAX_USER_ERR `#define MAX_USER_ERR 0x500`

6.187.2 Variable Documentation

6.187.2.1 last_jack_err_msg `char last_jack_err_msg[ MAX_USER_ERR]`

6.188 mhamain.cpp File Reference

Classes

- class **mhaserver_t**
  MHA Framework listening on TCP port for commands.
- class **mhaserver_t::tcp_server_t**

© 2005-2021 HörTech gGmbH, Oldenburg

Macros

- #define **HELP_TEXT**
- #define **NORELEASE_WARNING**
- #define **VERSION_EXTENSION** "+"
- #define **GREETING_TEXT**

Functions

- int **mhamain** (int argc, char *argv[ ])

6.188.1 Macro Definition Documentation

6.188.1.1 HELP_TEXT `#define HELP_TEXT`

6.188.1.2 NORELEASE_WARNING `#define NORELEASE_WARNING`

6.188.1.3 VERSION_EXTENSION `#define VERSION_EXTENSION "+"`

6.188.1.4 GREETING_TEXT `#define GREETING_TEXT`

6.188.2 Function Documentation

6.188.2.1 mhamain()

```
int mhamain (
    int argc,
    char * argv[] )
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.189 mhapluginloader.cpp File Reference

6.190 mhapluginloader.h File Reference

Classes

- class **PluginLoader::config_file_splitter_t**
- class **PluginLoader::fourway_processor_t**

  This abstract class defines the interface for classes that implement all types of signal domain processing supported by the MHA: wave2wave, spec2spec, wave2spec, and spec2wave.
- class **PluginLoader::mhapluginloader_t**
- class **MHAParser::mhapluginloader_t**

  Class to create a plugin loader in a parser, including the load logic.

Namespaces

- **PluginLoader**
- **MHAParser**

  Name space for the openMHA-Parser configuration language.

Functions

- const char ∗ **PluginLoader::mhastrdomain** (**mha_domain_t**)
- void **PluginLoader::mhaconfig_compare** (const **mhaconfig_t** &req, const **mhaconfig_t** &avail, const std::string &pref="")

  *Compare two **mhaconfig_t** (p. 851) structures, and report differences as an error.*

6.191 mhasndfile.cpp File Reference

Functions

- void **write_wave** (const **mha_wave_t** &sig, const char ∗fname, const float &srate, const int &format)
- unsigned int **validator_channels** (std::vector< int > channel_map, unsigned int **channels**)
- unsigned int **validator_length** (unsigned int maxlen, unsigned int frames, unsigned int startpos)

6.191.1 Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.191.1.1 write_wave()

```
void write_wave (
    const mha_wave_t & sig,
    const char * fname,
    const float & srate,
    const int & format )
```

6.191.1.2 validator_channels()

```
unsigned int validator_channels (
    std::vector< int > channel_map,
    unsigned int channels )
```

6.191.1.3 validator_length()

```
unsigned int validator_length (
    unsigned int maxlen,
    unsigned int frames,
    unsigned int startpos )
```

6.192 mhasndfile.h File Reference

Classes

- class **MHASndFile::sf_t**
- class **MHASndFile::sf_wave_t**

Namespaces

- **MHASndFile**

Functions

- void **write_wave** (const **mha_wave_t** &sig, const char *fname, const float &srate=44100, const int &format=SF_FORMAT_WAV|SF_FORMAT_FLOAT|SF_EN↩DIAN_FILE)

6.192.1 Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.192.1.1 write_wave()

```
void write_wave (
    const mha_wave_t & sig,
    const char * fname,
    const float & srate = 44100,
    const int & format = SF_FORMAT_WAV|SF_FORMAT_FLOAT|SF_ENDIAN_FILE )
```

6.193 multibandcompressor.cpp File Reference

Classes

- class **multibandcompressor::plugin_signals_t**
- class **multibandcompressor::fftfb_plug_t**
- class **multibandcompressor::interface_t**

Namespaces

- **multibandcompressor**

6.194 nlms_wave.cpp File Reference

Classes

- class **rt_nlms_t**
- class **nlms_t**

Macros

- #define **NORMALIZATION_TYPES** "[none default sum]"
- #define **NORM_NONE** 0
- #define **NORM_DEFAULT** 1
- #define **NORM_SUM** 2
- #define **ESTIMATION_TYPES** "[previous current]"
- #define **ESTIM_PREV** 0
- #define **ESTIM_CUR** 1

Functions

- void **make_friendly_number_by_limiting** (**mha_real_t** &x)

© 2005-2021 HörTech gGmbH, Oldenburg

6.194.1 Macro Definition Documentation

6.194.1.1 NORMALIZATION_TYPES `#define NORMALIZATION_TYPES "[none default sum]"`

6.194.1.2 NORM_NONE `#define NORM_NONE 0`

6.194.1.3 NORM_DEFAULT `#define NORM_DEFAULT 1`

6.194.1.4 NORM_SUM `#define NORM_SUM 2`

6.194.1.5 ESTIMATION_TYPES `#define ESTIMATION_TYPES "[previous current]"`

6.194.1.6 ESTIM_PREV `#define ESTIM_PREV 0`

6.194.1.7 ESTIM_CUR `#define ESTIM_CUR 1`

6.194.2 Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.194.2.1 make_friendly_number_by_limiting()

```
void make_friendly_number_by_limiting (
    mha_real_t & x ) [inline]
```

6.195 noise.cpp File Reference

Classes

- class **cfg_t**
- class **noise_t**

6.196 noise_psd_estimator.cpp File Reference

Classes

- class **noise_psd_estimator::noise_psd_estimator_t**
- class **noise_psd_estimator::noise_psd_estimator_if_t**

Namespaces

- **noise_psd_estimator**

Macros

- #define **POWSPEC_FACTOR** 0.0025

6.196.1 Macro Definition Documentation

6.196.1.1 POWSPEC_FACTOR

```
#define POWSPEC_FACTOR 0.0025
```

6.197 osc2ac.cpp File Reference

Classes

- class **osc_variable_t**

  Class for converting messages received at a single osc address to a single AC variable.
- class **osc_server_t**

  OSC receiver implemented using liblo.
- class **osc2ac_t**

© 2005-2021 HörTech gGmbH, Oldenburg

6.198 overlapadd.cpp File Reference

Namespaces

- **overlapadd**

6.199 overlapadd.hh File Reference

Classes

- class **overlapadd::overlapadd_t**
- class **overlapadd::overlapadd_if_t**

Namespaces

- **overlapadd**

6.200 plingploing.cpp File Reference

Classes

- class **plingploing::plingploing_t**
  Run-time configuration of the plingploing music generator.
- class **plingploing::if_t**
  Plugin class of the plingploing music generator.

Namespaces

- **plingploing**
  All classes for the plingploing music generator live in this namespace.

Functions

- double **plingploing::drand** (double a, double b)

© 2005-2021 HörTech gGmbH, Oldenburg

6.201 pluginbrowser.cpp File Reference

6.202 pluginbrowser.h File Reference

Classes

- class **plugindescription_t**
- class **pluginloader_t**
- class **pluginbrowser_t**

6.203 proc_counter.cpp File Reference

Classes

- class **proc_counter_t**

6.204 resampling.cpp File Reference

Classes

- class **MHAPlugin_Resampling::resampling_t**
- class **MHAPlugin_Resampling::resampling_if_t**

Namespaces

- **MHAPlugin_Resampling**

6.205 rmslevel.cpp File Reference

Classes

- class **rmslevel::mon_t**
- class **rmslevel::rmslevel_t**

  Run-time configuration class of the rmslevel plugin.
- class **rmslevel::rmslevel_if_t**

  Interface class of the rmslevel plugin.

© 2005-2021 HörTech gGmbH, Oldenburg

Namespaces

- **rmslevel**

Enumerations

- enum **rmslevel::UNIT** { **rmslevel::UNIT::SPL** =0, **rmslevel::UNIT::HL** =1 }

6.206 rohBeam.cpp File Reference

Namespaces

- **rohBeam**

Variables

- auto **rohBeam::scalarify** =[ ](auto t){return t(0);}

6.207 rohBeam.hh File Reference

Classes

- struct **rohBeam::configOptions**
- class **rohBeam::rohConfig**
- class **rohBeam::rohBeam**

Namespaces

- **rohBeam**

Macros

- #define **NDEBUG**

Functions

- double **rohBeam::j0** (double x)

  Cylindrical bessel function of the first kind of order 0.

© 2005-2021 HörTech gGmbH, Oldenburg

Variables

- constexpr float **rohBeam::CONST_C** = 343.0115f
- constexpr int **rohBeam::refL** = 0
- constexpr int **rohBeam::refR** = 3

6.207.1 Macro Definition Documentation

6.207.1.1 NDEBUG `#define NDEBUG`

6.208 route.cpp File Reference

Classes

- class **route::process_t**
- class **route::interface_t**

Namespaces

- **route**

6.209 save_spec.cpp File Reference

Classes

- class **save_spec_t**

6.210 save_wave.cpp File Reference

Classes

- class **save_wave_t**

© 2005-2021 HörTech gGmbH, Oldenburg

6.211 shadowfilter_begin.cpp File Reference

Classes

- class **shadowfilter_begin::cfg_t**
- class **shadowfilter_begin::shadowfilter_begin_t**

Namespaces

- **shadowfilter_begin**

6.212 shadowfilter_end.cpp File Reference

Classes

- class **shadowfilter_end::cfg_t**
- class **shadowfilter_end::shadowfilter_end_t**

Namespaces

- **shadowfilter_end**

6.213 sine.cpp File Reference

Classes

- struct **sine_cfg_t**

  Runtime configuration of the sine plugin.
- class **sine_t**

  Interface class of plugin `sine`*, a sinusoid generator plugin.*

6.214 smooth_cepstrum.cpp File Reference

Macros

- #define **INSERT_VAR**(var) insert_item(#var, &var)
- #define **PATCH_VAR**(var)
- #define **INSERT_PATCH**(var) **INSERT_VAR**(var); **PATCH_VAR**(var)

© 2005-2021 HörTech gGmbH, Oldenburg

6.214.1 Macro Definition Documentation

6.214.1.1 INSERT_VAR

```
#define INSERT_VAR(
    var ) insert_item(#var, &var)
```

6.214.1.2 PATCH_VAR

```
#define PATCH_VAR(
    var )
```

6.214.1.3 INSERT_PATCH

```
#define INSERT_PATCH(
    var ) INSERT_VAR(var); PATCH_VAR(var)
```

6.215 smooth_cepstrum.hh File Reference

Classes

- class **smooth_cepstrum::smooth_params**
- class **smooth_cepstrum::smooth_cepstrum_t**
- class **smooth_cepstrum::smooth_cepstrum_if_t**

Namespaces

- **smooth_cepstrum**

6.216 smoothgains_bridge.cpp File Reference

Classes

- class **smoothgains_bridge::smoothspec_wrap_t**
- class **smoothgains_bridge::overlapadd_if_t**

Namespaces

- **smoothgains_bridge**

© 2005-2021 HörTech gGmbH, Oldenburg

6.217 softclip.cpp File Reference

Classes

- class **cfg_t**
- class **softclip_t**

6.218 spec2wave.cpp File Reference

Classes

- class **hanning_ramps_t**
- class **spec2wave_t**
- class **spec2wave_if_t**

Functions

- unsigned int **max** (unsigned int a, unsigned int b)
- unsigned int **min** (unsigned int a, unsigned int b)

6.218.1 Function Documentation

6.218.1.1 max()

```
unsigned int max (
    unsigned int a,
    unsigned int b ) [inline]
```

6.218.1.2 min()

```
unsigned int min (
    unsigned int a,
    unsigned int b ) [inline]
```

6.219 speechnoise.cpp File Reference

Macros

- #define **NUM_ENTR_MHAORIG** 76
- #define **NUM_ENTR_LTASS** 25
- #define **NUM_ENTR_OLNOISE** 49

© 2005-2021 HörTech gGmbH, Oldenburg

Functions

- float **fhz2bandno** (float x)
- float **erb_hz_f_hz** (float f_hz)
- float **hz2hz** (float x)

  Dummy scale transformation Hz to Hz.
- float **bandw_correction** (float f, float ldb)

Variables

- float **vMHAOrigSpec** [**NUM_ENTR_MHAORIG**] = {-1.473, 0, -4.939, -10.14, -13.94, -14.83, -14.27, -15.66, -16.16, -18.22, -20.5, -21.23, -22.13, -22.58, -23.98, -26.58, -26.4, -25.15, -23.89, -25.54, -27, -30.15, -31.68, -30.14, -27.55, -25.79, -25.89, -26.11, -27.↩ 48, -30.37, -33.13, -36.23, -36.64, -36.35, -35.03, -35.48, -36.35, -37.95, -40.53, -42.37, -41.29, -38.49, -36.32, -34.85, -34.05, -33.81, -33.48, -34.1, -35.19, -36.29, -36.94, -37.↩ 53, -38.71, -38.7, -38.92, -40.36, -41.26, -42.19, -43.65, -44.37, -43.95, -43.15, -42.57, -41.57, -41.86, -42.34, -42.87, -42.35, -42.71, -42.85, -43.47, -47.43, -67.54, -76.3, -77.↩ 43, -77.43}
- float **vMHAOrigFreq** [**NUM_ENTR_MHAORIG**] = {172.266,344.532,516.797,689.↩ 063,861.329,1033.59,1205.86,1378.13,1550.39,1722.66,1894.92,2067.19,2239.↩ 46,2411.72,2583.99,2756.25,2928.52,3100.78,3273.05,3445.32,3617.58,3789.85,3962.↩ 11,4134.38,4306.64,4478.91,4651.18,4823.44,4995.71,5167.97,5340.24,5512.51,5684.↩ 77,5857.04,6029.3,6201.57,6373.83,6546.1,6718.37,6890.63,7062.9,7235.16,7407.↩ 43,7579.69,7751.96,7924.23,8096.49,8268.76,8441.02,8613.29,8785.56,8957.82,9130.↩ 09,9302.35,9474.62,9646.88,9819.15,9991.42,10163.7,10335.9,10508.2,10680.↩ 5,10852.7,11025,11197.3,11369.5,11541.8,11714.1,11886.3,12058.6,12230.9,12403.↩ 1,12575.4,12747.7,12919.9,13092.2}
- float **vLTASS_freq** [**NUM_ENTR_LTASS**] = {63, 80, 100, 125, 160, 200, 250, 315, 400, 500, 630, 800, 1000, 1250, 1600, 2000, 2500, 3150, 4000, 5000, 6300, 8000, 10000, 12500, 16000}
- float **vLTASS_combined_lev** [**NUM_ENTR_LTASS**] = {38.6, 43.5, 54.4, 57.7, 56.8, 60.2, 60.3, 59.0, 62.1, 62.1, 60.5, 56.8, 53.7, 53.0, 52.0, 48.7, 48.1, 46.8, 45.6, 44.5, 44.3, 43.7, 43.4, 41.3, 40.7}
- float **vLTASS_female_lev** [**NUM_ENTR_LTASS**] = {37.0,36.0,37.5,40.1,53.4,62.↩ 2,60.9,58.1,61.7,61.7,60.4,58,54.3,52.3,51.7,48.8,47.3,46.7,45.3,44.6,45.2,44.9,45.↩ 0,42.8,41.1}
- float **vLTASS_male_lev** [**NUM_ENTR_LTASS**] = {38.6,43.5,54.4,57.7,56.8,58.2,59.↩ 7,60.0,62.4,62.6,60.6,55.7,53.1,53.7,52.3,48.7,48.9,47.0,46.0,44.4,43.3,42.4,41.9,39.↩ 8,40.4}
- float **vOlnoiseFreq** [**NUM_ENTR_OLNOISE**] = {62.5,70.1539,78.7451,88.3884,99.↩ 2126,111.362,125,140.308,157.49,176.777,198.425,222.725,250,280.616,314.98,353.↩ 553,396.85,445.449,500,561.231,629.961,707.107,793.701,890.899,1000,1122.↩ 46,1259.92,1414.21,1587.4,1781.8,2000,2244.92,2519.84,2828.43,3174.8,3563.↩ 59,4000,4489.85,5039.68,5656.85,6349.6,7127.19,8000,8979.7,10079.4,11313.↩ 7,12699.2,14254.4,16000}
- float **vOlnoiseLev** [**NUM_ENTR_OLNOISE**] = {45.9042,38.044,48.9444,61.3697,67.↩ 6953,69.7451,71.6201,71.2431,65.2754,63.2547,70.2264,72.1434,73.4433,73.2659,69.↩ 8424,71.0132,70.9577,70.3492,68.691,64.8436,64.0435,64.2879,60.5889,60.6596,60.↩ 3727,61.2003,61.8477,61.1478,61.2312,58.6584,57.2892,56.8299,56.0191,53.3018,56.↩ 0525,54.3592,50.8823,55.992,54.6768,47.2616,46.9914,45.209,50.413,47.5848,43.↩ 3215,43.754,38.5773,-0.39427,5.74224}

© 2005-2021 HörTech gGmbH, Oldenburg

6.219.1 Macro Definition Documentation

6.219.1.1 NUM_ENTR_MHAORIG `#define NUM_ENTR_MHAORIG 76`

6.219.1.2 NUM_ENTR_LTASS `#define NUM_ENTR_LTASS 25`

6.219.1.3 NUM_ENTR_OLNOISE `#define NUM_ENTR_OLNOISE 49`

6.219.2 Function Documentation

6.219.2.1 fhz2bandno() `float fhz2bandno ( float x )`

6.219.2.2 erb_hz_f_hz() `float erb_hz_f_hz ( float f_hz )`

6.219.2.3 hz2hz() `float hz2hz ( float x )`

Dummy scale transformation Hz to Hz.

This function implements a dummy scale transformation (linear frequency scale).

© 2005-2021 HörTech gGmbH, Oldenburg

Parameters

x	Input frequency in Hz

Returns

Frequency in Hz

6.219.2.4 bandw_correction()

```
float bandw_correction (
    float f,
    float ldb )
```

6.219.3 Variable Documentation

6.219.3.1 vMHAOrigSpec

```
float vMHAOrigSpec[ NUM_ENTR_MHAORIG] = {-1.473, 0, -4.↩
939, -10.14, -13.94, -14.83, -14.27, -15.66, -16.16, -18.22, -20.5, -21.23, -22.13,
-22.58, -23.98, -26.58, -26.4, -25.15, -23.89, -25.54, -27, -30.15, -31.68, -30.↩
14, -27.55, -25.79, -25.89, -26.11, -27.48, -30.37, -33.13, -36.23, -36.64, -36.↩
35, -35.03, -35.48, -36.35, -37.95, -40.53, -42.37, -41.29, -38.49, -36.32, -34.85,
-34.05, -33.81, -33.48, -34.1, -35.19, -36.29, -36.94, -37.53, -38.71, -38.7, -38.↩
92, -40.36, -41.26, -42.19, -43.65, -44.37, -43.95, -43.15, -42.57, -41.57, -41.↩
86, -42.34, -42.87, -42.35, -42.71, -42.85, -43.47, -47.43, -67.54, -76.3, -77.43,
-77.43}
```

6.219.3.2 vMHAOrigFreq

```
float vMHAOrigFreq[ NUM_ENTR_MHAORIG] = {172.266,344.↩
532,516.797,689.063,861.329,1033.59,1205.86,1378.13,1550.39,1722.66,1894.92,2067.↩
19,2239.46,2411.72,2583.99,2756.25,2928.52,3100.78,3273.05,3445.32,3617.58,3789.↩
85,3962.11,4134.38,4306.64,4478.91,4651.18,4823.44,4995.71,5167.97,5340.24,5512.↩
51,5684.77,5857.04,6029.3,6201.57,6373.83,6546.1,6718.37,6890.63,7062.9,7235.16,7407.↩
43,7579.69,7751.96,7924.23,8096.49,8268.76,8441.02,8613.29,8785.56,8957.82,9130.↩
09,9302.35,9474.62,9646.88,9819.15,9991.42,10163.7,10335.9,10508.2,10680.5,10852.↩
7,11025,11197.3,11369.5,11541.8,11714.1,11886.3,12058.6,12230.9,12403.1,12575.↩
4,12747.7,12919.9,13092.2}
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.219.3.3 vLTASS_freq

```
float vLTASS_freq[ NUM_ENTR_LTASS] = {63, 80, 100, 125,
160, 200, 250, 315, 400, 500, 630, 800, 1000, 1250, 1600, 2000, 2500, 3150, 4000,
5000, 6300, 8000, 10000, 12500, 16000}
```

6.219.3.4 vLTASS_combined_lev

```
float vLTASS_combined_lev[ NUM_ENTR_LTASS] = {38.↩
6, 43.5, 54.4, 57.7, 56.8, 60.2, 60.3, 59.0, 62.1, 62.1, 60.5, 56.8, 53.7, 53.0,
52.0, 48.7, 48.1, 46.8, 45.6, 44.5, 44.3, 43.7, 43.4, 41.3, 40.7}
```

6.219.3.5 vLTASS_female_lev

```
float vLTASS_female_lev[ NUM_ENTR_LTASS] = {37.↩
0,36.0,37.5,40.1,53.4,62.2,60.9,58.1,61.7,61.7,60.4,58,54.3,52.3,51.7,48.8,47.↩
3,46.7,45.3,44.6,45.2,44.9,45.0,42.8,41.1}
```

6.219.3.6 vLTASS_male_lev

```
float vLTASS_male_lev[ NUM_ENTR_LTASS] = {38.6,43.↩
5,54.4,57.7,56.8,58.2,59.7,60.0,62.4,62.6,60.6,55.7,53.1,53.7,52.3,48.7,48.9,47.↩
0,46.0,44.4,43.3,42.4,41.9,39.8,40.4}
```

6.219.3.7 vOlnoiseFreq

```
float vOlnoiseFreq[ NUM_ENTR_OLNOISE] = {62.5,70.1539,78.↩
7451,88.3884,99.2126,111.362,125,140.308,157.49,176.777,198.425,222.725,250,280.↩
616,314.98,353.553,396.85,445.449,500,561.231,629.961,707.107,793.701,890.899,1000,1122.↩
46,1259.92,1414.21,1587.4,1781.8,2000,2244.92,2519.84,2828.43,3174.8,3563.59,4000,4489.↩
85,5039.68,5656.85,6349.6,7127.19,8000,8979.7,10079.4,11313.7,12699.2,14254.4,16000}
```

6.219.3.8 vOlnoiseLev

```
float vOlnoiseLev[ NUM_ENTR_OLNOISE] = {45.9042,38.044,48.↩
9444,61.3697,67.6953,69.7451,71.6201,71.2431,65.2754,63.2547,70.2264,72.1434,73.↩
4433,73.2659,69.8424,71.0132,70.9577,70.3492,68.691,64.8436,64.0435,64.2879,60.↩
5889,60.6596,60.3727,61.2003,61.8477,61.1478,61.2312,58.6584,57.2892,56.8299,56.↩
0191,53.3018,56.0525,54.3592,50.8823,55.992,54.6768,47.2616,46.9914,45.209,50.↩
413,47.5848,43.3215,43.754,38.5773,-0.39427,5.74224}
```

6.220 speechnoise.h File Reference

Classes

- class **speechnoise_t**

© 2005-2021 HörTech gGmbH, Oldenburg

6.221 split.cpp File Reference

Classes

- class **MHAPlugin_Split::uni_processor_t**

  An interface to a class that sports a process method with no parameters and no return value.
- class **MHAPlugin_Split::thread_platform_t**

  Basic interface for encapsulating thread creation, thread priority setting, and synchronization on any threading platform (i.e., pthreads or win32threads).
- class **MHAPlugin_Split::dummy_threads_t**

  Dummy specification of a thread platform: This class implements everything in a single thread.
- class **MHAPlugin_Split::posix_threads_t**

  Posix threads specification of thread platform.
- class **MHAPlugin_Split::domain_handler_t**

  Handles domain-specific partial input and output signal.
- class **MHAPlugin_Split::splitted_part_t**

  *The **splitted_part_t** (p. 1183) instance manages the plugin that performs processing on the reduced set of channels.*
- class **MHAPlugin_Split::split_t**

  Implements split plugin.

Namespaces

- **MHAPlugin_Split**

Macros

- #define **MHAPLUGIN_OVERLOAD_OUTDOMAIN**

  This define modifies the definition of MHAPLUGIN_CALLBACKS and friends.
- #define **posixthreads** 1
- #define **native_thread_platform_type** posix_threads_t

Enumerations

- enum { **MHAPlugin_Split::INVALID_THREAD_PRIORITY** = 999999999 }

  Invalid thread priority.

6.221.1 Detailed Description

Source code for the split plugin. The split plugin splits the audio signal by channel. The splitted paths execute in parallel.

© 2005-2021 HörTech gGmbH, Oldenburg

6.221.2 Macro Definition Documentation

6.221.2.1 MHAPLUGIN_OVERLOAD_OUTDOMAIN `#define MHAPLUGIN_OVERLOAD_OUTDO↩ MAIN`

This define modifies the definition of MHAPLUGIN_CALLBACKS and friends.

The output signal is transferred through a second parameter to the process method, enabling all four domain transformations in a single plugin.

6.221.2.2 posixthreads `#define posixthreads 1`

6.221.2.3 native_thread_platform_type `#define native_thread_platform_type posix_↩ threads_t`

6.222 steerbf.cpp File Reference

Macros

- #define **PATCH_VAR**(var) patchbay.connect(&var.valuechanged, this, & **steerbf↩ ::update_cfg**)
- #define **INSERT_PATCH**(var) **insert_member**(var); **PATCH_VAR**(var)

6.222.1 Macro Definition Documentation

6.222.1.1 PATCH_VAR `#define PATCH_VAR(`
`var ) patchbay.connect(&var.valuechanged, this, & steerbf::update_cfg)`

6.222.1.2 INSERT_PATCH `#define INSERT_PATCH(`
`var ) insert_member(var); PATCH_VAR(var)`

© 2005-2021 HörTech gGmbH, Oldenburg

6.223 steerbf.h File Reference

Classes

- class **parser_int_dyn**
- class **steerbf_config**
- class **steerbf**

6.224 testalsadevice.c File Reference

Functions

- int **main** (int argc, char ∗∗argv)

6.224.1 Function Documentation

6.224.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

6.225 testplugin.cpp File Reference

Classes

- class **testplugin::config_parser_t**
- class **testplugin::ac_parser_t**
- class **testplugin::signal_parser_t**
- class **testplugin::if_t**

Namespaces

- **testplugin**

© 2005-2021 HörTech gGmbH, Oldenburg

6.226 transducers.cpp File Reference

Classes

- class **softclipper_variables_t**
  Parser aggregate of all configuration variables for the output soft clipper.
- class **softclipper_t**
  Soft clipper signal processing implementation.
- class **calibrator_variables_t**
- class **calibrator_runtime_layer_t**
- class **calibrator_t**
- class **bbcalib_interface_t**

Typedefs

- typedef **MHAPlugin::config_t**< **MHASignal::async_rmslevel_t** > **rmslevelmeter**
- typedef **MHAPlugin::plugin_t**< **calibrator_runtime_layer_t** > **rtcalibrator**

Functions

- **speechnoise_t::noise_type_t kw_index2type** (unsigned int idx)
- std::vector< int > **vint_0123n1** (unsigned int n)

6.226.1 Typedef Documentation

6.226.1.1 rmslevelmeter `typedef` **`MHAPlugin::config_t< MHASignal::async_rmslevel_t> rmslevelmeter`**

6.226.1.2 rtcalibrator `typedef` **`MHAPlugin::plugin_t< calibrator_runtime_layer_t> rtcalibrator`**

6.226.2 Function Documentation

© 2005-2021 HörTech gGmbH, Oldenburg

6.226.2.1 kw_index2type()

```
speechnoise_t::noise_type_t kw_index2type (
    unsigned int idx )
```

6.226.2.2 vint_0123n1()

```
std::vector<int> vint_0123n1 (
    unsigned int n )
```

6.227 upsample.cpp File Reference

Classes

- class **us_t**

6.228 wave2spec.cpp File Reference

6.229 wave2spec.hh File Reference

Classes

- class **wave2spec_t**
  Runtime configuration class for plugin wave2spec.
- class **wave2spec_if_t**
  *Plugin wave2spec interface class, uses **wave2spec_t** (p. 1497) as runtime configuration.*

Macros

- #define **MHAPLUGIN_OVERLOAD_OUTDOMAIN**

6.229.1 Macro Definition Documentation

6.229.1.1 MHAPLUGIN_OVERLOAD_OUTDOMAIN

```
#define MHAPLUGIN_OVERLOAD_OUTDO↩
MAIN
```

© 2005-2021 HörTech gGmbH, Oldenburg

6.230 wavrec.cpp File Reference

Classes

- class **wavwriter_t**
- class **wavrec_t**

6.231 windnoise.cpp File Reference

Namespaces

- **windnoise**
  namespace for plugin windnoise which detects and cancels wind noise

Macros

- #define **register_configuration_variable**(v)

6.231.1 Macro Definition Documentation

6.231.1.1 register_configuration_variable `#define register_configuration_variable( v )`

6.232 windnoise.hh File Reference

Classes

- class **windnoise::cfg_t**
  Runtime config class for windnoise plugin.
- class **windnoise::if_t**
  interface class for windnoise plugin

Namespaces

- **windnoise**
  namespace for plugin windnoise which detects and cancels wind noise

6.233 windowselector.cpp File Reference

6.234 windowselector.h File Reference

Classes

- class **windowselector_t**
  A combination of mha parser variables to describe an overalapadd analysis window.

© 2005-2021 HörTech gGmbH, Oldenburg

Index

_MHA_AC_CHAR
testplugin::ac_parser_t, 1481
_MHA_AC_DOUBLE
testplugin::ac_parser_t, 1481
_MHA_AC_FLOAT
testplugin::ac_parser_t, 1481
_MHA_AC_INT
testplugin::ac_parser_t, 1481
_MHA_AC_MHACOMPLEX
testplugin::ac_parser_t, 1481
_MHA_AC_MHAREAL
testplugin::ac_parser_t, 1481
__MHA_FUN__
mha_defs.h, 1588
__declspec
mha_defs.h, 1589
mha_plugin.hh, 1623
_ac
gtfb_simple_rt_t, 569
_cf
DynComp::dc_afterburn_t, 455
_channels
DynComp::dc_afterburn_t, 455
_conjugate
Complex arithmetics in the openMHA, 67
_fmax
audiometerbackend::lnn3rdoct_t, 322
_fmin
audiometerbackend::lnn3rdoct_t, 321
_linphase_asym
MHAFilter::smoothspec_t, 934
_order
gtfb_simple_rt_t, 567
_pre_stages
gtfb_simple_rt_t, 568
_prepare
testplugin::if_t, 1488
_reciprocal
Complex arithmetics in the openMHA, 67
_srate
DynComp::dc_afterburn_t, 455
_steerbf
steerbf_config, 1479
_unknown
testplugin::ac_parser_t, 1481
~Async_Notify
MHA_TCP::Async_Notify, 799
~Connection
MHA_TCP::Connection, 805
~Delay
ADM::Delay< F >, 267
~Event_Watcher
MHA_TCP::Event_Watcher, 813
~Linearphase_FIR
ADM::Linearphase_FIR< F >, 269
~MHA_Error
MHA_Error, 765
~Server
MHA_TCP::Server, 816
~Thread
MHA_TCP::Thread, 831
~Timeout_Watcher
MHA_TCP::Timeout_Watcher, 835
~Wakeup_Event
MHA_TCP::Wakeup_Event, 837
~acConcat_wave
acConcat_wave, 200
~acConcat_wave_config
acConcat_wave_config, 202
~acPooling_wave
acPooling_wave, 212
~acPooling_wave_config
acPooling_wave_config, 216
~acSteer
acSteer, 229
~acSteer_config
acSteer_config, 231
~acTransform_wave
acTransform_wave, 234
~acTransform_wave_config
acTransform_wave_config, 238
~acmon_t
acmon::acmon_t, 208
~acspace2matrix_t
MHA_AC::acspace2matrix_t, 727
~acwriter_t
plugins::hoertech::acrec::acwriter_t, 1384
~adaptive_feedback_canceller
adaptive_feedback_canceller, 241
~adaptive_feedback_canceller_config
adaptive_feedback_canceller_config, 245
~adm_rtconfig_t
adm_rtconfig_t, 277
~algo_comm_class_t
MHAKernel::algo_comm_class_t, 990
~alsa_base_t

alsa_base_t, 287
~alsa_t
alsa_t< T >, 292
~analysepath_t
analysepath_t, 307
~analysispath_if_t
analysispath_if_t, 311
~bark2hz_t
MHAOvlFilter::barkscale::bark2hz_t, 1003
~base_t
MHAParser::base_t, 1035
~bbcalib_interface_t
bbcalib_interface_t, 334
~blockprocessing_polyphase_resampling_t
MHAFilter::blockprocessing_polyphase_resampling_t, 870
~c_ifc_parser_t
MHAParser::c_ifc_parser_t, 1051
~cfg_node_t
MHAPlugin::cfg_node_t< runtime_cfg_t >, 1144
~cfg_t
acsave::cfg_t, 223
equalize::cfg_t, 463
~config_t
MHAPlugin::config_t< runtime_cfg_t >, 1148
~connector_base_t
MHAEvents::connector_base_t, 854
~connector_t
MHAEvents::connector_t< receiver_t >, 857
~db_if_t
db_if_t, 370
dbasync_native::db_if_t, 374
~dbasync_t
dbasync_native::dbasync_t, 377
~delay_spec_t
MHASignal::delay_spec_t, 1202
~delay_t
MHASignal::delay_t, 1204
~delay_wave_t
MHASignal::delay_wave_t, 1206
~doasvm_classification
doasvm_classification, 422
~doasvm_classification_config
doasvm_classification_config, 425
~doasvm_feature_extraction
doasvm_feature_extraction, 428
~doasvm_feature_extraction_config
doasvm_feature_extraction_config, 431
~domain_handler_t
MHAPlugin_Split::domain_handler_t, 1165
~double2acvar_t
double2acvar::double2acvar_t, 435
~double_t
MHA_AC::double_t, 730
~doublebuffer_t
MHASignal::doublebuffer_t, 1209
~dynamiclib_t
dynamiclib_t, 448
~emitter_t
MHAEvents::emitter_t, 860
~fft_t
MHASignal::fft_t, 1212
~fftfb_t
MHAOvlFilter::fftfb_t, 1008
~fftfilter_t
MHAFilter::fftfilter_t, 878
~fftfilterbank_t
MHAFilter::fftfilterbank_t, 884
~filter_t
MHAFilter::filter_t, 890
~float_t
MHA_AC::float_t, 732
~fourway_processor_t
PluginLoader::fourway_processor_t, 1366
~fshift_config_t
fshift::fshift_config_t, 510
~fshift_t
fshift::fshift_t, 513
~fw_t
fw_t, 525
~gaintable_t
DynComp::gaintable_t, 459
~gamma_flt_t
MHAFilter::gamma_flt_t, 894
~gsc_adaptive_stage
gsc_adaptive_stage::gsc_adaptive_stage, 539
~gsc_adaptive_stage_if
gsc_adaptive_stage::gsc_adaptive_stage_if, 546
~gtfb_analyzer_cfg_t
gtfb_analyzer::gtfb_analyzer_cfg_t, 551
~gtfb_simd_cfg_t
gtfb_simd_cfg_t, 558
~hanning_ramps_t
hanning_ramps_t, 575
~hilbert_fftw_t
MHASignal::hilbert_fftw_t, 1216

© 2005-2021 HörTech gGmbH, Oldenburg

~hilbert_shifter_t
fshift_hilbert::hilbert_shifter_t, 520
~hilbert_t
MHASignal::hilbert_t, 1219
~hz2bark_t
MHAOvlFilter::barkscale::hz2bark_t, 1004
~int_t
MHA_AC::int_t, 734
~io_asterisk_fwcb_t
io_asterisk_fwcb_t, 586
~io_asterisk_parser_t
io_asterisk_parser_t, 591
~io_asterisk_sound_t
io_asterisk_sound_t, 598
~io_asterisk_t
io_asterisk_t, 602
~io_file_t
io_file_t, 607
~io_lib_t
io_lib_t, 613
~io_parser_t
io_parser_t, 618
~io_portaudio_t
MHAIOPortAudio::io_portaudio_t, 960
~io_tcp_fwcb_t
io_tcp_fwcb_t, 622
~io_tcp_parser_t
io_tcp_parser_t, 627
~io_tcp_sound_t
io_tcp_sound_t, 634
~io_tcp_t
io_tcp_t, 640
~io_wrapper
io_wrapper, 643
~level_matching_config_t
level_matching::level_matching_config_t, 652
~level_matching_t
level_matching::level_matching_t, 655
~linear_table_t
MHATableLookup::linear_table_t, 1280
~lpc
lpc, 662
~lpc_bl_predictor
lpc_bl_predictor, 665
~lpc_bl_predictor_config
lpc_bl_predictor_config, 668
~lpc_burglattice
lpc_burglattice, 671
~lpc_burglattice_config
lpc_burglattice_config, 674
~lpc_config
lpc_config, 677
~matlab_wrapper_rt_cfg_t
matlab_wrapper::matlab_wrapper_rt_cfg_t, 696
~matrix_t
MHASignal::matrix_t, 1230
~mha_dblbuf_t
mha_dblbuf_t< FIFO >, 750
~mha_fifo_lw_t
mha_fifo_lw_t< T >, 771
~mha_fifo_posix_threads_t
mha_fifo_posix_threads_t, 774
~mha_fifo_t
mha_fifo_t< T >, 779
~mha_fifo_thread_guard_t
mha_fifo_thread_guard_t, 784
~mha_fifo_thread_platform_t
mha_fifo_thread_platform_t, 785
~mha_rt_fifo_element_t
mha_rt_fifo_element_t< T >, 789
~mha_rt_fifo_t
mha_rt_fifo_t< T >, 791
~mha_stash_environment_variable_t
mha_stash_environment_variable_t, 797
~mhaplug_cfg_t
mhaplug_cfg_t, 1143
~mhapluginloader_t
MHAParser::mhapluginloader_t, 1092
PluginLoader::mhapluginloader_t, 1371
~mhaserver_t
mhaserver_t, 1195
~osc_server_t
osc_server_t, 1324
~overlapadd_if_t
overlapadd::overlapadd_if_t, 1332
smoothgains_bridge::overlapadd_if_t, 1454
~overlapadd_t
overlapadd::overlapadd_t, 1335
~parser_t
MHAParser::parser_t, 1104
~partitioned_convolution_t
MHAFilter::partitioned_convolution_t, 918
~patchbay_t
MHAEvents::patchbay_t< receiver_t >, 862
~plug_t
plug_t, 1349
~plug_wrapper
plug_wrapper, 1351

© 2005-2021 HörTech gGmbH, Oldenburg

~plug_wrapperI
plug_wrapperI, 1353
~plugin_t
MHAPlugin::plugin_t< runtime_cfg_t >, 1153
~pluginlib_t
pluginlib_t, 1361
~pluginloader_t
pluginloader_t, 1377
~plugs_t
mhachain::plugs_t, 848
~port_t
MHAJack::port_t, 986
~posix_threads_t
MHAPlugin_Split::posix_threads_t, 1173
~proc_counter_t
proc_counter_t, 1389
~rmslevel_t
rmslevel::rmslevel_t, 1395
~rohBeam
rohBeam::rohBeam, 1400
~rohConfig
rohBeam::rohConfig, 1408
~rt_nlms_t
rt_nlms_t, 1418
~save_var_base_t
ac2lsl::save_var_base_t, 170
~save_var_t
ac2lsl::save_var_t< mha_complex_t >, 177
ac2lsl::save_var_t< T >, 173
acsave::save_var_t, 226
lsl2ac::save_var_t, 688
~server_t
mha_tcp::server_t, 821
~sf_t
MHASndFile::sf_t, 1276
~smooth_cepstrum_t
smooth_cepstrum::smooth_cepstrum_t, 1444
~smoothspec_t
MHAFilter::smoothspec_t, 932
~spec2wave_t
spec2wave_t, 1469
~spec_fader_t
spec_fader_t, 1471
~spectrum_t
MHA_AC::spectrum_t, 736
MHASignal::spectrum_t, 1250
~split_t
MHAPlugin_Split::split_t, 1178
~splitted_part_t
MHAPlugin_Split::splitted_part_t, 1185
~steerbf
steerbf, 1476
~steerbf_config
steerbf_config, 1478
~table_t
MHATableLookup::table_t, 1284
~thread_platform_t
MHAPlugin_Split::thread_platform_t, 1190
~uint_vector_t
MHASignal::uint_vector_t, 1260
~uni_processor_t
MHAPlugin_Split::uni_processor_t, 1192
~wave2spec_t
wave2spec_t, 1499
~waveform_t
MHA_AC::waveform_t, 740
MHASignal::waveform_t, 1266
~wavwriter_t
wavwriter_t, 1507
~windowselector_t
windowselector_t, 1520
~wrapped_plugin_t
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 705

A
lpc_config, 678
MHAFilter::filter_t, 891
MHAFilter::gamma_flt_t, 896
MHAFilter::iir_filter_t, 902
a
MHAParser::base_t::replace_t, 1045
A_
MHAFilter::complex_bandpass_t, 875
MHAFilter::iir_ord1_real_t, 905
abandonned
mha_rt_fifo_element_t< T >, 790
abs
Complex arithmetics in the openMHA, 65
abs2
Complex arithmetics in the openMHA, 65
ac
ac2lsl::cfg_t, 169
ac2wave_t, 190
ac_mul_t, 194
acConcat_wave_config, 203
acmon::acmon_t, 209
acPooling_wave_config, 216
acsave::cfg_t, 223

© 2005-2021 HörTech gGmbH, Oldenburg

acsave::save_var_t, 227
acTransform_wave_config, 238
adaptive_feedback_canceller_config, 246
doasvm_classification_config, 426
fw_t, 530
gsc_adaptive_stage::gsc_adaptive_stage, 540
latex_doc_t, 647
lpc_bl_predictor_config, 669
lpc_burglattice_config, 674
lsl2ac::save_var_t, 691
MHA_AC::ac2matrix_helper_t, 722
MHA_AC::double_t, 731
MHA_AC::float_t, 732
MHA_AC::int_t, 734
MHA_AC::spectrum_t, 737
MHA_AC::waveform_t, 741
mhachain::plugs_t, 849
MHAKernel::algo_comm_class_t, 993
MHAMultiSrc::base_t, 996
MHAPlugin::plugin_t< runtime_cfg_t >, 1156
PluginLoader::mhapluginloader_t, 1374
plugins::hoertech::acrec::acrec_t, 1382
proc_counter_t, 1390
rt_nlms_t, 1419
shadowfilter_end::cfg_t, 1430
smooth_cepstrum::smooth_cepstrum_t, 1445
steerbf_config, 1479
testplugin::if_t, 1488
AC variable, 4
ac2lsl, 78
types, 78
ac2lsl.cpp, 1522
ac2lsl::ac2lsl_t, 162
ac2lsl_t, 163
activate, 165
get_all_names_from_ac_space, 164
is_first_run, 166
patchbay, 165
prepare, 163
process, 164
release, 164
rt_strict, 165
skip, 165
source_id, 165
update, 165
vars, 165
ac2lsl::cfg_t, 166
ac, 169
cfg_t, 167
check_vars, 167
create_or_replace_var, 167
process, 168
skip, 168
skipcnt, 168
source_id, 168
srate, 168
update_varlist, 168
varlist, 168
ac2lsl::save_var_base_t, 169
~save_var_base_t, 170
data_type, 171
get_buf_address, 170
info, 170
num_entries, 171
send_frame, 170
set_buf_address, 170
ac2lsl::save_var_t< mha_complex_t >, 175
~save_var_t, 177
buf, 178
data_type, 178
get_buf_address, 177
info, 177
num_entries, 177
save_var_t, 176
send_frame, 178
set_buf_address, 177
stream, 178
ac2lsl::save_var_t< T >, 171
~save_var_t, 173
buf, 175
data_type, 174
data_type_, 175
get_buf_address, 173
info, 174
num_entries, 174
save_var_t, 172
send_frame, 174
set_buf_address, 173
stream, 175
ac2lsl::type_info, 179
format, 179
name, 179
ac2lsl_t
ac2lsl::ac2lsl_t, 163
ac2matrix_helper_t
MHA_AC::ac2matrix_helper_t, 722
ac2matrix_t
MHA_AC::ac2matrix_t, 724
ac2osc.cpp, 1523

© 2005-2021 HörTech gGmbH, Oldenburg

ac2osc_t, 180
ac2osc_t, 181
acspace, 184
b_record, 184
framerate, 184
host, 183
is_first_run, 185
lo_addr, 185
mode, 183
patchbay, 184
port, 183
prepare, 182
process, 182
release, 182
rt_strict, 184
rtmem, 184
send_osc_float, 182
skip, 184
skipcnt, 184
ttl, 183
update_mode, 183
vars, 183
ac2wave.cpp, 1523
ac2wave_if_t, 185
ac2wave_if_t, 186
delay_ac, 188
delay_in, 188
gain_ac, 187
gain_in, 187
name, 187
patchbay, 188
prepare, 187
prepared, 188
process, 186, 187
release, 187
update, 187
zeros, 188
ac2wave_t, 188
ac, 190
ac2wave_t, 189
channels, 189
delay_ac, 190
delay_in, 190
frames, 189
gain_ac, 190
gain_in, 190
name, 190
process, 189
w, 190
ac_
combc_t, 358
MHAParser::mhapluginloader_t, 1094
ac_data
osc_variable_t, 1329
AC_DIM_MISMATCH
mha_algo_comm.cpp, 1585
ac_double
double2acvar::double2acvar_t, 436
ac_fifo
analysepath_t, 308
ac_insert
osc_server_t, 1325
osc_variable_t, 1328
AC_INVALID_HANDLE
mha_algo_comm.cpp, 1584
AC_INVALID_NAME
mha_algo_comm.cpp, 1585
AC_INVALID_OUTPTR
mha_algo_comm.cpp, 1585
ac_monitor_t
acmon::ac_monitor_t, 204
ac_monitor_type.cpp, 1523
ac_monitor_type.hh, 1523
ac_mul.cpp, 1524
ac_mul.hh, 1524
ARG_CC, 1524
ARG_CR, 1524
ARG_RC, 1524
ARG_RR, 1524
arg_type_t, 1524
VAL_COMPLEX, 1525
VAL_REAL, 1525
val_type_t, 1524
ac_mul_t, 191
ac, 194
ac_mul_t, 192
algo, 194
argt, 195
get_arg_type_and_dimension, 193, 194
num_channels, 195
num_frames, 195
prepare_, 193
process, 193, 194
process_cc, 194
process_cr, 194
process_rc, 194
process_rr, 194
release_, 193
res_c, 195
res_r, 195
scan_syntax, 193
str_a, 195

© 2005-2021 HörTech gGmbH, Oldenburg

str_b, 195
ac_parser_t
testplugin::ac_parser_t, 1481
ac_proc, 78
ac_proc.cpp, 1525
ac_proc::interface_t, 196
algo, 198
b_permute, 198
input, 198
interface_t, 197
permute, 198
plug, 198
prepare, 197
process, 197, 198
release, 197
s_in, 199
s_in_perm, 198
s_out, 198
ac_resynthesis_gain
gtfb_simple_rt_t, 569
AC_STRING_TRUNCATED
mha_algo_comm.cpp, 1585
AC_SUCCESS
mha_algo_comm.cpp, 1584
AC_TYPE_MISMATCH
mha_algo_comm.cpp, 1585
ac_wndshape_name
wave2spec_t, 1502
acbw
gtfb_simple_rt_t, 568
accept
MHA_TCP::Server, 817
accept_event
MHA_TCP::Server, 818
accept_loop
io_asterisk_t, 603
io_tcp_t, 641
acceptor
mha_tcp::server_t, 824
acceptor_started
mhaserver_t, 1195
accf
gtfb_simple_rt_t, 568
acConcat_wave, 199
∼acConcat_wave, 200
acConcat_wave, 200
name_con_AC, 201
num_AC, 201
numchannels, 201
patchbay, 202
prefix_names_AC, 201
prepare, 200
process, 200
release, 201
samples_AC, 201
update_cfg, 201
acConcat_wave.cpp, 1525
INSERT_PATCH, 1525
PATCH_VAR, 1525
acConcat_wave.h, 1526
acConcat_wave_config, 202
∼acConcat_wave_config, 202
ac, 203
acConcat_wave_config, 202
numSamples_AC, 203
process, 203
strNames_AC, 203
vGCC, 203
vGCC_con, 203
ack_fail
mhaserver_t, 1196
ack_ok
mhaserver_t, 1196
acmon, 79
acmon.cpp, 1526
acmon::ac_monitor_t, 203
ac_monitor_t, 204
dimstr, 205
getvar, 205
mon, 205
mon_complex, 206
mon_mat, 206
mon_mat_complex, 206
name, 205
p_parser, 206
use_mat, 206
acmon::acmon_t, 207
∼acmon_t, 208
ac, 209
acmon_t, 208
algo, 210
b_cont, 210
b_snapshot, 210
dimensions, 209
dispmode, 209
patchbay, 210
prepare, 208
process, 209
recmode, 210
release, 208
save_vars, 209
update_recmode, 209

© 2005-2021 HörTech gGmbH, Oldenburg

varlist, 209
vars, 210
acmon_t
acmon::acmon_t, 208
acname
osc_variable_t, 1329
acPooling_wave, 211
~acPooling_wave, 212
acPooling_wave, 212
alpha, 214
like_ratio_name, 214
lower_threshold, 213
max_pool_ind_name, 214
neighbourhood, 214
numsamples, 213
p_biased_name, 214
p_name, 214
patchbay, 214
pool_name, 214
pooling_type, 213
pooling_wndlen, 213
prepare, 212
prob_bias, 214
process, 212
release, 213
update_cfg, 213
upper_threshold, 213
acPooling_wave.cpp, 1526
INSERT_PATCH, 1526
PATCH_VAR, 1526
acPooling_wave.h, 1527
acPooling_wave_config, 215
~acPooling_wave_config, 216
ac, 216
acPooling_wave_config, 215
alpha, 218
c, 217
insert, 216
like_ratio, 217
low_thresh, 217
neigh, 217
p, 216
p_biased, 216
p_max, 217
pool, 218
pooling_ind, 217
pooling_option, 217
pooling_size, 217
prob_bias_func, 218
process, 216
raw_p_name, 216
up_thresh, 217
acrec.cpp, 1527
acrec.hh, 1527
acrec_t
plugins::hoertech::acrec::acrec_t, 1379
acsave, 79
acsave.cpp, 1527
ACSAVE_FMT_M, 1528
ACSAVE_FMT_MAT4, 1528
ACSAVE_FMT_TXT, 1528
ACSAVE_SFMT_M, 1528
ACSAVE_SFMT_MAT4, 1528
ACSAVE_SFMT_TXT, 1528
acsave::acsave_t, 218
acsave_t, 220
algo, 221
b_flushed, 222
b_prepared, 221
bflush, 221
event_start_recording, 220
event_stop_and_flush, 220
fileformat, 221
fname, 221
patchbay, 222
prepare, 220
process, 220, 221
reclen, 221
release, 220
variables, 221
varlist, 221
varlist_t, 219
acsave::cfg_t, 222
~cfg_t, 223
ac, 223
cfg_t, 222
flush_data, 223
max_frames, 224
nvars, 223
rec_frames, 224
store_frame, 223
varlist, 223
acsave::mat4head_t, 224
cols, 224
imag, 225
namelen, 225
rows, 224
t, 224
acsave::save_var_t, 225
~save_var_t, 226
ac, 227
b_complex, 227

© 2005-2021 HörTech gGmbH, Oldenburg

data, 227
framecnt, 227
maxframe, 227
name, 227
ndim, 227
nframes, 227
save_m, 226
save_mat4, 226
save_txt, 226
save_var_t, 226
store_frame, 226
ACSAVE_FMT_M
acsave.cpp, 1528
ACSAVE_FMT_MAT4
acsave.cpp, 1528
ACSAVE_FMT_TXT
acsave.cpp, 1528
ACSAVE_SFMT_M
acsave.cpp, 1528
ACSAVE_SFMT_MAT4
acsave.cpp, 1528
ACSAVE_SFMT_TXT
acsave.cpp, 1528
acsave_t
acsave::acsave_t, 220
acspace
ac2osc_t, 184
acspace2matrix_t
MHA_AC::acspace2matrix_t, 726
acspace_template
analysispath_if_t, 312
acSteer, 228
~acSteer, 229
acSteer, 229
acSteerName1, 230
acSteerName2, 230
nrefmic, 230
nsteerchan, 230
patchbay, 231
prepare, 229
process, 229
release, 230
steerFile, 230
update_cfg, 230
acSteer.cpp, 1529
INSERT_PATCH, 1529
PATCH_VAR, 1529
acSteer.h, 1529
acSteer_config, 231
~acSteer_config, 231
acSteer_config, 231
insert, 232
nangle, 232
nchan, 232
nfreq, 232
nrefmic, 232
nsteerchan, 232
specSteer1, 232
specSteer2, 232
acSteerName1
acSteer, 230
acSteerName2
acSteer, 230
act_
wavwriter_t, 1508
actgains
fader_if_t, 489
activate
ac2lsl::ac2lsl_t, 165
lsl2ac::lsl2ac_t, 684
activate_query
MHAParser::base_t, 1041
active
addsndfile::addsndfile_if_t, 254
plugins::hoertech::acrec::acwriter_t, 1386
acTransform_wave, 233
~acTransform_wave, 234
acTransform_wave, 234
ang_name, 236
numsamples, 237
patchbay, 237
prepare, 235
process, 234
raw_p_max_name, 236
raw_p_name, 236
release, 236
rotated_p_max_name, 236
rotated_p_name, 236
to_from, 237
update_cfg, 236
acTransform_wave.cpp, 1529
INSERT_PATCH, 1530
PATCH_VAR, 1529
acTransform_wave.h, 1530
acTransform_wave_config, 237
~acTransform_wave_config, 238
ac, 238
acTransform_wave_config, 238
ang_name, 238
offset, 239
process, 238
raw_p_max_name, 238

© 2005-2021 HörTech gGmbH, Oldenburg

raw_p_name, 238
resolution, 239
rotated_i, 239
rotated_p, 239
to_from, 239
acvar
MHA_AC::ac2matrix_helper_t, 723
acwriter_t
plugins::hoertech::acrec::acwriter_t, 1384
adapt_filter_param_t
MHAFilter::adapt_filter_param_t, 864
adapt_filter_state_t
MHAFilter::adapt_filter_state_t, 865
adapt_filter_t
MHAFilter::adapt_filter_t, 867
adaptation_ratio
adm_if_t, 274
adm_rtconfig_t, 278
adaptive_feedback_canceller, 240
~adaptive_feedback_canceller, 241
adaptive_feedback_canceller, 241
afc_delay, 243
c, 242
delay_d, 243
delay_w, 243
gains, 243
lpc_order, 243
n_no_update, 244
name_e, 243
name_f, 243
name_lpc, 243
ntaps, 243
patchbay, 244
prepare, 242
process, 241
release, 242
rho, 242
update_cfg, 242
adaptive_feedback_canceller.cpp, 1530
INSERT_PATCH, 1530
make_friendly_number_by_limiting, 1531
PATCH_VAR, 1530
adaptive_feedback_canceller.h, 1531
adaptive_feedback_canceller_config, 244
~adaptive_feedback_canceller_config, 245
ac, 246
adaptive_feedback_canceller_config, 245
channels, 246
EPrew, 249
F, 246
F_Uflt, 248
frames, 246
insert, 246
iter, 247
n_no_update_, 247
name_d_, 247
name_lpc_, 247
no_iter, 247
ntaps, 246
process, 245
PSD_val, 247
Pu, 246
s_E, 246
s_E_afc_delay, 247
s_LPC, 249
s_U, 247
s_U_delay, 248
s_U_delayflt, 248
s_Usmpl, 249
s_W, 248
s_Wflt, 248
s_Y_delay, 248
s_Y_delayflt, 248
smpl, 249
UbufferPrew, 248
UPrew, 249
UPrewW, 249
v_G, 247
YPrew, 249
add
MHASignal::loop_wavefragment_t, 1222
add4f
gtfb_simd.cpp, 1562
add_connection
mha_tcp::server_t, 824
add_entry
MHAParser::keyword_list_t, 1074
MHATableLookup::linear_table_t, 1281
MHATableLookup::xy_table_t, 1287
add_fun
MHAOvlFilter::scale_var_t, 1029
add_parent_on_insert
MHAParser::base_t, 1041
add_plug
altplugs_t, 304
add_plugin
pluginbrowser_t, 1357
add_plugins
pluginbrowser_t, 1357
add_replace_pair
MHAParser::base_t, 1041

© 2005-2021 HörTech gGmbH, Oldenburg

added_via_plugs
altplugs_t, 305
addsndfile, 79
addsndfile_resampling_mode_t, 80
DO_RESAMPLE, 80
DONT_RESAMPLE_PERMISSIVE, 80
DONT_RESAMPLE_STRICT, 80
level_adaptor, 80
resampled_num_frames, 81
wave_reader, 80
addsndfile.cpp, 1531
DEBUG, 1532
addsndfile::addsndfile_if_t, 250
active, 254
addsndfile_if_t, 251
change_mode, 252
channels, 253
filename, 252
level, 253
levelmode, 253
loop, 252
mapping, 253
mhachannels, 254
mode, 253
numchannels, 253
patchbay, 254
path, 252
prepare, 251
process, 251
ramplen, 253
release, 252
resamplingmode, 253
scan_dir, 252
search_pattern, 254
search_result, 254
set_level, 252
startpos, 253
uint_mode, 254
update, 252
addsndfile::level_adapt_t, 255
can_update, 256
get_level, 256
ilen, 256
l_new, 256
l_old, 257
level_adapt_t, 255
pos, 256
update_frame, 256
wnd, 256
addsndfile::resampled_soundfile_t, 257
resampled_soundfile_t, 258
addsndfile::sndfile_t, 259
sndfile_t, 259
addsndfile::waveform_proxy_t, 260
waveform_proxy_t, 261
addsndfile_if_t
addsndfile::addsndfile_if_t, 251
addsndfile_resampling_mode_t
addsndfile, 80
ADM, 81
ADM::ADM< F >, 262
C, 82
DELAY_FREQ, 82
PI, 82
START_BETA, 82
subsampledelay_coeff, 81
adm
adm_rtconfig_t, 277
adm.cpp, 1532
adm_fir_decomb, 1533
adm_fir_lp, 1533
adm.hh, 1533
ADM::ADM< F >, 261
ADM, 262
beta, 263
m_beta, 264
m_decomb, 264
m_delay_back, 264
m_delay_front, 264
m_lp_bf, 264
m_lp_result, 264
m_mu_beta, 264
m_powerfilter_coeff, 265
m_powerfilter_norm, 265
m_powerfilter_state, 265
process, 263
ADM::Delay< F >, 265
~Delay, 267
Delay, 266
m_coeff, 267
m_fullsamples, 267
m_norm, 268
m_now_in, 268
m_state, 268
process, 267
ADM::Linearphase_FIR< F >, 268
~Linearphase_FIR, 269
Linearphase_FIR, 269
m_alphas, 270
m_now, 270
m_order, 270
m_output, 270

© 2005-2021 HörTech gGmbH, Oldenburg

process, 270
adm_fir_decomb
adm.cpp, 1533
adm_fir_lp
adm.cpp, 1533
adm_if_t, 271
adaptation_ratio, 274
adm_if_t, 272
beta, 274
bypass, 274
coeff_decomb, 274
coeff_lp, 274
decomb_order, 274
distances, 273
framecnt, 275
front_channels, 273
input_channels, 274
is_prepared, 273
lp_order, 273
mu_beta, 274
out, 273
patchbay, 275
prepare, 272
process, 272
rear_channels, 273
release, 273
srate, 275
tau_beta, 274
update, 273
adm_rtconfig_t, 275
~adm_rtconfig_t, 277
adaptation_ratio, 278
adm, 277
adm_rtconfig_t, 276
adm_t, 276
adms, 279
check_index, 277
decomb_coeffs, 279
front_channel, 278
front_channels, 278
get_adaptation_ratio, 278
lp_coeffs, 279
num_adms, 277
rear_channel, 278
rear_channels, 278
adm_t
adm_rtconfig_t, 276
adms
adm_rtconfig_t, 279
afc_delay
adaptive_feedback_canceller, 243
algo
ac_mul_t, 194
ac_proc::interface_t, 198
acmon::acmon_t, 210
acsave::acsave_t, 221
analysispath_if_t, 312
coherence::cohflt_if_t, 349
db_if_t, 371
dbasync_native::db_if_t, 376
dc::dc_if_t, 382
fftfilterbank::fftfb_interface_t, 506
MHAPlugin_Resampling::resampling_if_t, 1160
multibandcompressor::interface_t, 1306
nlms_t, 1312
overlapadd::overlapadd_if_t, 1334
route::interface_t, 1415
smoothgains_bridge::overlapadd_if_t, 1456
wave2spec_if_t, 1497
algo_comm_class_t
MHAKernel::algo_comm_class_t, 989
algo_comm_default
mha_algo_comm.cpp, 1585
mha_algo_comm.hh, 1587
ALGO_COMM_ID_STR
mha_algo_comm.hh, 1587
algo_comm_id_string
MHAKernel::algo_comm_class_t, 993
algo_comm_id_string_len
MHAKernel::algo_comm_class_t, 993
algo_comm_safe_cast
MHAKernel, 115
algo_comm_t, 279
get_entries, 285
get_error, 285
get_var, 283
get_var_double, 285
get_var_float, 284
get_var_int, 284
handle, 280
insert_var, 281
insert_var_double, 282
insert_var_float, 281
insert_var_int, 281
is_var, 283
mha.hh, 1582
remove_ref, 282
remove_var, 282
algo_name
lpc, 663

© 2005-2021 HörTech gGmbH, Oldenburg

algos
mhachain::chain_base_t, 844
mhachain::plugs_t, 849
MHAPlugin_Split::split_t, 1181
all_dump
MHAParser, 127
all_ids
MHAParser, 127
alloc_plugs
mhachain::plugs_t, 848
almost
Complex arithmetics in the openMHA, 68
alp
gsc_adaptive_stage::gsc_adaptive_stage, 542
gsc_adaptive_stage::gsc_adaptive_stage_if_t, 549
alph
plingploing::plingploing_t, 1348
alpha
acPooling_wave, 214
acPooling_wave_config, 218
cfg_t, 346
coherence::cohflt_t, 351
coherence::vars_t, 354
alpha_blocking_XkXi
rohBeam::configOptions, 1398
rohBeam::rohConfig, 1410
alpha_blocking_XkY
rohBeam::configOptions, 1398
rohBeam::rohConfig, 1410
alpha_const
smooth_cepstrum::smooth_cepstrum_t, 1446
alpha_const_limits_hz
smooth_cepstrum::smooth_cepstrum_if_t, 1442
smooth_cepstrum::smooth_params, 1452
alpha_const_vals
smooth_cepstrum::smooth_cepstrum_if_t, 1442
smooth_cepstrum::smooth_params, 1452
alpha_frame
smooth_cepstrum::smooth_cepstrum_t, 1448
alpha_hat
smooth_cepstrum::smooth_cepstrum_t, 1447
alpha_Lowpass
windnoise::cfg_t, 1512
alpha_pitch
smooth_cepstrum::smooth_cepstrum_if_t, 1441
smooth_cepstrum::smooth_params, 1452
alpha_postfilter
rohBeam::configOptions, 1398
rohBeam::rohConfig, 1410
alpha_prev
smooth_cepstrum::smooth_cepstrum_t, 1446
alphaPH1mean
noise_psd_estimator::noise_psd_estimator_if_t, 1314
alphaPH1mean_
noise_psd_estimator::noise_psd_estimator_t, 1317
alphaPSD
noise_psd_estimator::noise_psd_estimator_if_t, 1314
alphaPSD_
noise_psd_estimator::noise_psd_estimator_t, 1317
alsa_base_t, 286
~alsa_base_t, 287
alsa_base_t, 287
pcm, 288
read, 287
start, 287
stop, 287
write, 288
alsa_dev_par_parser_t, 289
alsa_dev_par_parser_t, 290
device, 290
nperiods, 290
stream_dir, 290
alsa_start_counter
io_alsa_t, 584
alsa_t
alsa_t< T >, 292
alsa_t< T >, 291
~alsa_t, 292
alsa_t, 292
buffer, 294
channels, 294
fragsize, 294
frame_data, 294
gain, 294
invgain, 294
pcm_format, 294
read, 293
start, 293
stop, 293

© 2005-2021 HörTech gGmbH, Oldenburg

wave, 294
write, 293
altconfig.cpp, 1534
altconfig.hh, 1534
MHAPLUGIN_OVERLOAD_OUTDOMAIN, 1534
altconfig_t, 295
altconfig_t, 296
configs, 299
event_select_all, 298
on_set_algos, 297
on_set_select, 297
parser_algos, 299
patchbay, 299
prepare, 297
release, 297
restore_state, 298
save_state, 298
select_plug, 299
selectall, 299
altplugs.cpp, 1534
MHAPLUGIN_OVERLOAD_OUTDOMAIN, 1535
altplugs_t, 300
add_plug, 304
added_via_plugs, 305
altplugs_t, 301
cfin, 305
cfout, 305
current, 304
delete_plug, 304
event_add_plug, 303
event_delete_plug, 303
event_select_plug, 303
event_set_plugs, 302
fallback_spec, 305
fallback_wave, 305
nondefault_labels, 304
parse, 302
parser_plugs, 303
patchbay, 304
plugs, 304
prepare, 301
prepared, 305
proc_ramp, 303
process, 302
ramp_counter, 305
ramp_len, 305
ramplen, 304
release, 301
select_plug, 304
selected_plug, 304
update_ramplen, 303
update_selector_list, 303
use_own_ac, 303
amplitude
sine_cfg_t, 1434
analysemhaplugin.cpp, 1535
document_io_plugin, 1535
document_plugin, 1535
main, 1536
print_ac, 1535
strdom, 1535
analysepath_t, 306
~analysepath_t, 307
ac_fifo, 308
analysepath_t, 306
attr, 309
cond_to_process, 309
flag_terminate_inner_thread, 309
has_inner_error, 308
inner_ac_copy, 308
inner_error, 308
inner_input, 307
inner_out_domain, 308
inner_process_wave2spec, 307
inner_process_wave2wave, 307
input_to_process, 309
libdata, 308
outer_ac, 308
outer_ac_copy, 308
priority, 309
ProcessMutex, 309
rt_process, 307
scheduler, 309
svc, 307
thread, 309
wave_fifo, 308
analysispath.cpp, 1536
thread_start, 1536
analysispath_if_t, 310
~analysispath_if_t, 311
acspace_template, 312
algo, 312
analysispath_if_t, 311
fifolen, 312
fragsize, 312
libname, 312
loadlib, 311
patchbay, 312
plug, 312
prepare, 311

© 2005-2021 HörTech gGmbH, Oldenburg

priority, 312
process, 311
release, 311
vars, 312
analytic
fshift_hilbert::hilbert_shifter_t, 520
ang_name
acTransform_wave, 236
acTransform_wave_config, 238
angle
Complex arithmetics in the openMHA, 62
angle_ind
steerbf, 1477
angle_src
steerbf, 1477
angles
doasvm_classification, 424
announce_port
mhaserver_t, 1197
antialias
ds_t, 445
us_t, 1492
apply_gains
MHAOvlFilter::fftfb_t, 1008
multibandcompressor::plugin_signals_t, 1307
aquire_mutex
mha_fifo_posix_threads_t, 775
mha_fifo_thread_platform_t, 786
arg
MHA_TCP::Thread, 831
ARG_CC
ac_mul.hh, 1524
ARG_CR
ac_mul.hh, 1524
ARG_RC
ac_mul.hh, 1524
ARG_RR
ac_mul.hh, 1524
arg_type_t
ac_mul.hh, 1524
argt
ac_mul_t, 195
array
matlab_wrapper::types< MHA_WAVEFORM >, 712
array_type
matlab_wrapper::types< MHA_SPECTRUM >, 711
ASSERT_EQUAL_DIM
mha_signal.cpp, 1632
ASSERT_EQUAL_DIM_PTR
mha_signal.cpp, 1632
assign
MHASignal::waveform_t, 1270, 1271
Vector and matrix processing toolbox, 44, 45
assign_channel
MHASignal::waveform_t, 1271
assign_frame
MHASignal::waveform_t, 1271
async_accept_has_been_triggered
mha_tcp::server_t, 824
ASYNC_CONNECT_STARTED
mha_tcp.cpp, 1645
Async_Notify
MHA_TCP::Async_Notify, 798
async_poll_msg
fw_t, 528
async_read
fw_t, 528
async_rmslevel_t
MHASignal::async_rmslevel_t, 1200
atomic_read_ptr
mha_fifo_lf_t< T >, 770
atomic_write_ptr
mha_fifo_lf_t< T >, 770
attack
cfg_t, 346
dc::dc_t, 385
dc_simple::level_smoother_t, 409
softclip_t, 1460
softclipper_t, 1462
attenuate20.cpp, 1536
attenuate20_t, 313
attenuate20_t, 313
prepare, 314
process, 314
release, 314
attr
analysepath_t, 309
dbasync_native::dbasync_t, 378
MHAPlugin_Split::posix_threads_t, 1175
audiometer_if_t
audiometerbackend::audiometer_if_t, 316
audiometerbackend, 82
gcd, 83
generator, 83
level_adaptor, 83
return_sig, 83
audiometerbackend.cpp, 1537
DEBUG, 1537

© 2005-2021 HörTech gGmbH, Oldenburg

audiometerbackend::audiometer_if_t, 315
audiometer_if_t, 316
change_mode, 316
freq, 317
level, 317
mode, 317
outchannel, 317
patchbay, 317
pmode, 317
prepare, 316
process, 316
ramplen, 317
set_level, 316
sigtype, 317
update, 316
audiometerbackend::level_adapt_t, 318
can_update, 319
get_level, 319
ilen, 319
l_new, 320
l_old, 320
level_adapt_t, 319
pos, 319
update_frame, 319
wnd, 320
audiometerbackend::lnn3rdoct_t, 320
_fmax, 322
_fmin, 321
bandpass, 321
dev, 322
iterate_lnn, 321
lnn3rdoct_t, 321
random, 322
rng, 322
audiometerbackend::signal_gen_t, 322
signal_gen_t, 323
audiometerbackend::sine_t, 323
sine_t, 324
auditory_profile.cpp, 1538
auditory_profile.h, 1538
AuditoryProfile, 84
AuditoryProfile::fmap_t, 324
get_frequencies, 325
get_values, 325
isempty, 325
AuditoryProfile::parser_t, 325
get_current_profile, 326
L, 327
parser_t, 326
R, 327
AuditoryProfile::parser_t::ear_t, 327
ear_t, 328
get_ear, 328
HTL, 328
UCL, 328
AuditoryProfile::parser_t::fmap_t, 329
f, 330
fmap_t, 329
get_fmap, 330
name_, 330
patchbay, 330
validate, 330
value, 330
AuditoryProfile::profile_t, 331
get_ear, 331
L, 332
R, 332
AuditoryProfile::profile_t::ear_t, 332
convert_empty2normal, 333
HTL, 333
UCL, 333
available_streams
lsl2ac::lsl2ac_t, 686
average
coherence::vars_t, 354
avg_ipd
coherence::cohflt_t, 351
azimuth
mha_direction_t, 755

B
MHAFilter::filter_t, 892
MHAFilter::iir_filter_t, 902
b
doasvm_classification, 424
MHAParser::base_t::replace_t, 1045
B_
MHAFilter::complex_bandpass_t, 876
MHAFilter::iir_ord1_real_t, 905
b_check_version
PluginLoader::mhapluginloader_t, 1376
b_complex
acsave::save_var_t, 227
b_cont
acmon::acmon_t, 210
b_est
lpc_bl_predictor_config, 669
b_exit_request
fw_t, 531
b_flushed
acsave::acsave_t, 222
b_fw_started
io_parser_t, 620

© 2005-2021 HörTech gGmbH, Oldenburg

b_interactive
mhaserver_t, 1197
b_is_input
calibrator_runtime_layer_t, 337
calibrator_t, 340
b_is_prepared
PluginLoader::mhapluginloader_t, 1376
b_loop
MHASignal::loop_wavefragment_t, 1225
b_ltg
coherence::cohflt_t, 352
b_permute
ac_proc::interface_t, 198
b_prepared
acsave::acsave_t, 221
io_file_t, 610
io_parser_t, 621
mhachain::chain_base_t, 845
mhachain::plugs_t, 849
MHAJack::client_t, 983
b_process
io_alsa_t, 582
b_ready
MHAJack::client_avg_t, 971
b_record
ac2osc_t, 184
b_snapshot
acmon::acmon_t, 210
b_starting
io_parser_t, 621
b_stopped
io_parser_t, 621
MHAJack::client_avg_t, 970
MHAJack::client_noncont_t, 973
b_use_clipping
calibrator_runtime_layer_t, 337
b_use_fir
calibrator_runtime_layer_t, 337
b_use_profiling
mhachain::plugs_t, 851
backward
lpc_bl_predictor_config, 669
lpc_burglattice_config, 675
MHASignal::fft_t, 1213
backward_scale
MHASignal::fft_t, 1214
band_weights
dc::dc_vars_t, 390
bandpass
audiometerbackend::lnn3rdoct_t, 321
bands
gtfb_analyzer::gtfb_analyzer_cfg_t, 551
gtfb_simd_cfg_t, 559
MHAOvlFilter::fspacing_t, 1021
bandsXchannels
gtfb_simd_cfg_t, 559
bandw_correction
speechnoise.cpp, 1707
bark2hz_t
MHAOvlFilter::barkscale::bark2hz_t, 1003
BARKSCALE_ENTRIES
mha_fftfb.cpp, 1596
bartlett
MHAWindow, 154
bartlett_t
MHAWindow::bartlett_t, 1291
base_t
MHAMultiSrc::base_t, 996
MHAParser::base_t, 1034, 1035
MHAWindow::base_t, 1292
basename
save_spec_t, 1423
save_wave_t, 1425
shadowfilter_begin::shadowfilter_begin_t, 1428
shadowfilter_end::shadowfilter_end_t, 1432
bass
plingploing::plingploing_t, 1346
bassmod
plingploing::if_t, 1343
bassmod_
plingploing::plingploing_t, 1348
bassperiod
plingploing::if_t, 1344
bassperiod_
plingploing::plingploing_t, 1348
bbcalib_interface_t, 333
~bbcalib_interface_t, 334
bbcalib_interface_t, 334
calib_in, 335
calib_out, 335
plugloader, 335
prepare, 334
process, 334
release, 335
beam1
rohBeam::rohConfig, 1410
beamA
rohBeam::rohConfig, 1410
beamExport
rohBeam::rohBeam, 1405

© 2005-2021 HörTech gGmbH, Oldenburg

beamW
rohBeam::rohConfig, 1410
beta
ADM::ADM< F >, 263
adm_if_t, 274
beta_const
smooth_cepstrum::smooth_cepstrum_if_t, 1441
smooth_cepstrum::smooth_params, 1452
bf_src
steerbf, 1477
bf_src_copy
steerbf_config, 1479
bf_vec
steerbf_config, 1479
bflush
acsave::acsave_t, 221
bin1
MHAOvlFilter::fftfb_t, 1008
bin2
MHAOvlFilter::fftfb_t, 1009
bin2freq
Vector and matrix processing toolbox, 39
binaural_type
rohBeam::rohBeam, 1404
binaural_type_index
rohBeam::configOptions, 1398
rohBeam::rohConfig, 1409
bInvert
coherence::cohflt_t, 352
blackman
MHAWindow, 155
blackman_t
MHAWindow::blackman_t, 1294
blockprocessing_polyphase_resampling_t
MHAFilter::blockprocessing_polyphase_resampling_t, 869
blocks
droptect_t, 442
blockSpec
rohBeam::rohConfig, 1410
blockXp
rohBeam::rohConfig, 1411
bookkeeping
MHAFilter::partitioned_convolution_t, 920
MHAParser::mhapluginloader_t, 1094
bool_mon_t
MHAParser::bool_mon_t, 1046
bool_t
MHAParser::bool_t, 1048
bpm
plingploing::if_t, 1343
bprofiling
mhachain::chain_base_t, 844
bracket_balance
MHAParser::StrCnv, 130
brown
speechnoise_t, 1473
browsemhaplugins.cpp, 1538
DEBUG, 1538
main, 1539
bt
plingploing::plingploing_t, 1346
buf
ac2lsl::save_var_t< mha_complex_t >, 178
ac2lsl::save_var_t< T >, 175
lsl2ac::save_var_t, 690
mha_fifo_t< T >, 782
mha_spec_t, 795
mha_wave_t, 841
buf_c_in
MHASignal::hilbert_fftw_t, 1217
buf_c_out
MHASignal::hilbert_fftw_t, 1217
buf_in
MHASignal::fft_t, 1215
buf_out
MHASignal::fft_t, 1215
buf_r_in
MHASignal::hilbert_fftw_t, 1217
buf_r_out
MHASignal::hilbert_fftw_t, 1217
buffer
alsa_t< T >, 294
MHASignal::delay_spec_t, 1203
MHASignal::delay_t, 1205
MHASignal::delay_wave_t, 1207
buffered_incoming_bytes
MHA_TCP::Connection, 810
buffered_outgoing_bytes
MHA_TCP::Connection, 810
buffersize
lsl2ac::lsl2ac_t, 685
bufSize
gsc_adaptive_stage::gsc_adaptive_stage, 540
bufsize
lsl2ac::save_var_t, 693
burn
DynComp::dc_afterburn_rt_t, 451
DynComp::dc_afterburn_t, 454

© 2005-2021 HörTech gGmbH, Oldenburg

multibandcompressor::interface_t, 1306
butter_stop_ord1
MHAFilter, 107
bw
MHAOvlFilter::fscale_bw_t, 1016
bw_
MHAFilter::gamma_flt_t, 896
bw_generator
MHAFilter::thirdoctave_analyzer_t, 936
bw_hz
MHAOvlFilter::fscale_bw_t, 1016
bw_name
dc::dc_vars_t, 389
bwv
MHAOvlFilter::fftfb_ac_info_t, 1006
multibandcompressor::fftfb_plug_t, 1303
bypass
adm_if_t, 274
db_if_t, 371
dc::dc_t, 386
dc::dc_vars_t, 389
dc_simple::dc_vars_t, 405
DynComp::dc_afterburn_vars_t, 458

C
ADM, 82
c
acPooling_wave_config, 217
adaptive_feedback_canceller, 242
doasvm_classification_config, 426
io_tcp_sound_t::float_union, 638
mha_complex_test_array_t, 747
nlms_t, 1311
c1_a
MHAFilter::o1_ar_filter_t, 909
c1_r
MHAFilter::o1_ar_filter_t, 909
c2_a
MHAFilter::o1_ar_filter_t, 909
c2_r
MHAFilter::o1_ar_filter_t, 909
c_ifc_parser_t
MHAParser::c_ifc_parser_t, 1051
c_min
coherence::cohflt_t, 351
c_parse_cmd
MHAParser::c_ifc_parser_t, 1052
c_parse_cmd_t
MHAParser, 126
c_parse_err
MHAParser::c_ifc_parser_t, 1052
c_parse_err_t
MHAParser, 128
c_scale
coherence::cohflt_t, 351
calc_in
wave2spec_t, 1502
calc_out
overlapadd::overlapadd_t, 1337
spec2wave_t, 1470
calc_pre_wnd
wave2spec_t, 1501
calc_sine
cpuload::cpuload_cfg_t, 365
calib_in
bbcalib_interface_t, 335
calib_out
bbcalib_interface_t, 335
calibrator_runtime_layer_t, 335
b_is_input, 337
b_use_clipping, 337
b_use_fir, 337
calibrator_runtime_layer_t, 336
fir, 337
firfir2fftlen, 336
firfirlen, 336
gain, 337
pmode, 338
process, 336
quant, 337
softclip, 337
speechnoise, 337
calibrator_t, 338
b_is_input, 340
calibrator_t, 339
patchbay, 340
prepare, 339
prepared, 340
process, 339
read_levels, 340
release, 339
update, 340
update_tau_level, 340
vars, 340
calibrator_variables_t, 341
calibrator_variables_t, 341
config_parser, 343
do_clipping, 343
fir, 341
fragsize, 342
nbits, 342
num_channels, 343
peaklevel, 341

© 2005-2021 HörTech gGmbH, Oldenburg

rmslevel, 342
softclip, 343
spnoise_channels, 342
spnoise_level, 342
spnoise_mode, 342
spnoise_parser, 342
srate, 342
tau_level, 342
callback
matlab_wrapper::callback, 694
matlab_wrapper::matlab_wrapper_t, 702
callbacks
matlab_wrapper::matlab_wrapper_t, 702
can_read
MHAFilter::blockprocessing_polyphase_resampling_t, 871
can_read_bytes
MHA_TCP::Connection, 808
can_read_line
MHA_TCP::Connection, 807
can_sysread
MHA_TCP::Connection, 806
can_syswrite
MHA_TCP::Connection, 806
can_update
addsndfile::level_adapt_t, 256
audiometerbackend::level_adapt_t, 319
fader_wave::level_adapt_t, 493
catch_condition
MHAPlugin_Split::posix_threads_t, 1175
catch_thread
MHAPlugin_Split::dummy_threads_t, 1171
MHAPlugin_Split::posix_threads_t, 1173
MHAPlugin_Split::thread_platform_t, 1190
categories
plugindescription_t, 1359
cb_patchbay
matlab_wrapper::matlab_wrapper_t, 702
cdata
mha_audio_t, 744
MHASignal::matrix_t, 1236
center_frequencies
dc::dc_vars_t, 390
dc_simple::dc_if_t, 396
cf
mha_audio_descriptor_t, 742
MHAFilter::thirdoctave_analyzer_t, 936
MHAOvlFilter::band_descriptor_t, 1002
MHAOvlFilter::fftfb_vars_t, 1013
plingploing::plingploing_t, 1346
cf2bands
MHAOvlFilter::fspacing_t, 1020
cf_
MHAFilter::gamma_flt_t, 896
wavwriter_t, 1508
cf_generator
MHAFilter::thirdoctave_analyzer_t, 936
cf_h
MHAOvlFilter::band_descriptor_t, 1002
cf_in
overlapadd::overlapadd_if_t, 1334
smoothgains_bridge::overlapadd_if_t, 1456
cf_in_
MHAParser::mhapluginloader_t, 1094
cf_input
PluginLoader::mhapluginloader_t, 1375
cf_l
MHAOvlFilter::band_descriptor_t, 1001
cf_name
dc::dc_vars_t, 389
cf_out
overlapadd::overlapadd_if_t, 1334
smoothgains_bridge::overlapadd_if_t, 1456
cf_out_
MHAParser::mhapluginloader_t, 1094
cf_output
PluginLoader::mhapluginloader_t, 1376
cfac
route::interface_t, 1415
cfg
MHAPlugin::config_t< runtime_cfg_t >, 1150
cfg_
MHAFilter::thirdoctave_analyzer_t, 936
cfg_dump
MHAParser, 126
cfg_dump_short
MHAParser, 126
cfg_node_current
MHAPlugin::config_t< runtime_cfg_t >, 1151
cfg_node_t
MHAPlugin::cfg_node_t< runtime_cfg_t >, 1144
cfg_root
MHAPlugin::config_t< runtime_cfg_t >, 1151
cfg_t, 343

© 2005-2021 HörTech gGmbH, Oldenburg

ac2lsl::cfg_t, 167
acsave::cfg_t, 222
alpha, 346
attack, 346
cfg_t, 344
channel, 345
decay, 346
equalize::cfg_t, 463
frozen_noise_, 346
gain_spec_, 345
gain_wave_, 345
lsl2ac::cfg_t, 680
matrixmixer::cfg_t, 713
pos, 346
process, 345
rand_dist, 346
replace_, 345
rng, 346
scale, 345
shadowfilter_begin::cfg_t, 1426
shadowfilter_end::cfg_t, 1429
start_lin, 346
use_frozen_, 345
windnoise::cfg_t, 1510
cfin
altplugs_t, 305
fw_t, 530
mhachain::chain_base_t, 845
route::interface_t, 1415
cfout
altplugs_t, 305
fw_t, 530
mhachain::chain_base_t, 845
route::interface_t, 1415
cfv
MHAOvlFilter::fftfb_ac_info_t, 1005
multibandcompressor::fftfb_plug_t, 1303
cg
coherence::cohflt_t, 351
ch
MHASignal::doublebuffer_t, 1211
chain_base_t
mhachain::chain_base_t, 843
chains
MHAPlugin_Split::split_t, 1182
chance
dropgen_t, 439
change_mode
addsndfile::addsndfile_if_t, 252
audiometerbackend::audiometer_if_t, 316
channel
cfg_t, 345
example5_t, 482
MHAMultiSrc::channel_t, 997
channel_gain_name
combc_if_t, 357
channel_gains_
combc_t, 359
channel_info
mha_spec_t, 795
mha_wave_t, 841
channel_no
example6_t, 484
channel_pair
level_matching::channel_pair, 648, 649
channelconfig_out_
MHAOvlFilter::overlap_save_filterbank_t, 1025
channels
ac2wave_t, 189
adaptive_feedback_canceller_config, 246
addsndfile::addsndfile_if_t, 253
alsa_t< T >, 294
fftfilter::fftfilter_t, 500
gtfb_analyzer::gtfb_analyzer_cfg_t, 551
gtfb_simd_cfg_t, 559
level_matching::level_matching_t, 657
mhaconfig_t, 852
MHAFilter::fftfilter_t, 881
MHAFilter::filter_t, 892
MHAParser::mhaconfig_mon_t, 1089
MHAPlugin_Split::split_t, 1181
MHASignal::delay_t, 1205
rt_nlms_t, 1419
sine_cfg_t, 1434
sine_t, 1437
testplugin::config_parser_t, 1484
Vector and matrix processing toolbox, 38
channels_t
MHAMultiSrc::channels_t, 997
char_data
testplugin::ac_parser_t, 1482
chdir
mha_audio_descriptor_t, 743
check_alignment
gtfb_simd.cpp, 1563
CHECK_EXPR
mha_defs.h, 1588
check_index
adm_rtconfig_t, 277
check_low
MHAParser::range_var_t, 1112

© 2005-2021 HörTech gGmbH, Oldenburg

check_parenthesis_complex
mha_parser.cpp, 1616
check_range
MHAParser::range_var_t, 1112
check_sign_complex
mha_parser.cpp, 1616
check_sound_data_type
io_tcp_sound_t, 635
check_up
MHAParser::range_var_t, 1112
CHECK_VAR
mha_defs.h, 1588
check_vars
ac2lsl::cfg_t, 167
chname
dc::dc_vars_t, 389
chunkbytes_in
io_asterisk_sound_t, 598
io_tcp_sound_t, 635
chunksize
lsl2ac::lsl2ac_t, 685
lsl2ac::save_var_t, 692
ci
matrixmixer::matmix_t, 716
class_signal_type
matlab_wrapper::types< MHA_SPECTRUM >, 712
matlab_wrapper::types< MHA_WAVEFORM >, 712
cleanup_plugs
mhachain::plugs_t, 849
cleanup_unused_cfg
MHAPlugin::config_t< runtime_cfg_t >, 1150
clear
mha_fifo_t< T >, 781
MHATableLookup::linear_table_t, 1282
MHATableLookup::table_t, 1284
MHATableLookup::xy_table_t, 1288
Vector and matrix processing toolbox, 44
clear_chains
MHAPlugin_Split::split_t, 1179
clear_plugins
pluginbrowser_t, 1357
Client
MHA_TCP::Client, 802
client_avg_t
MHAJack::client_avg_t, 968
client_noncont_t
MHAJack::client_noncont_t, 972
client_t
MHAJack::client_t, 976
clientid
dc::dc_vars_t, 390
dc_simple::dc_if_t, 395
clientname
MHAIOJack::io_jack_t, 945
MHAIOJackdb::io_jack_t, 953
clipmeter
softclipper_t, 1462
clipped
softclipper_variables_t, 1465
close_session
plugins::hoertech::acrec::acwriter_t, 1386
wavwriter_t, 1508
closed
MHA_TCP::Connection, 811
closesocket
mha_tcp.cpp, 1645
cLTASS
gtfb_simple_rt_t, 569
gtfb_simple_t, 574
MHAOvlFilter::fftfb_ac_info_t, 1006
MHAOvlFilter::fftfb_vars_t, 1014
cmd_prepare
MHAIOPortAudio::io_portaudio_t, 961
cmd_release
MHAIOPortAudio::io_portaudio_t, 962
cmd_start
MHAIOPortAudio::io_portaudio_t, 961
cmd_stop
MHAIOPortAudio::io_portaudio_t, 961
co
matrixmixer::matmix_t, 716
coeff
gtfb_analyzer::gtfb_analyzer_cfg_t, 552
gtfb_analyzer::gtfb_analyzer_t, 556
gtfb_simd_t, 564
coeff_decomb
adm_if_t, 274
coeff_lp
adm_if_t, 274
coh_c
coherence::cohflt_t, 352
coh_rlp
coherence::cohflt_t, 352
coherence, 84
getcipd, 84
coherence.cpp, 1539
coherence::cohflt_if_t, 347
algo, 349
cohflt_if_t, 348

© 2005-2021 HörTech gGmbH, Oldenburg

patchbay, 348
prepare, 348
process, 348
release, 348
update, 348
vars, 349
coherence::cohflt_t, 349
alpha, 351
avg_ipd, 351
b_ltg, 352
bInvert, 352
c_min, 351
c_scale, 351
cg, 351
coh_c, 352
coh_rlp, 352
cohflt_t, 350
g, 351
gain, 352
gain_delay, 352
insert, 350
limit, 351
lp1i, 352
lp1ltg, 352
lp1r, 351
nbands, 351
process, 350
s_out, 352
staticgain, 353
coherence::vars_t, 353
alpha, 354
average, 354
delay, 355
invert, 354
limit, 354
ltgcomp, 355
ltgtau, 355
mapping, 354
staticgain, 355
tau, 354
tau_unit, 354
vars_t, 354
cohflt_if_t
coherence::cohflt_if_t, 348
cohflt_t
coherence::cohflt_t, 350
collect_result
MHAPlugin_Split::split_t, 1180
MHAPlugin_Split::splitted_part_t, 1187
colored_intensity
Vector and matrix processing toolbox, 54
cols
acsave::mat4head_t, 224
combc_if_t, 355
channel_gain_name, 357
combc_if_t, 356
element_gain_name, 357
interleaved, 357
outchannels, 357
prepare, 356
process, 356
combc_t, 357
ac_, 358
channel_gains_, 359
combc_t, 358
element_gain_name_, 359
interleaved_, 359
nbands, 359
process, 358
s_out, 359
w_out, 359
combinechannels.cpp, 1539
comm_var_t, 359
data, 361
data_type, 360
num_entries, 360
stride, 361
commentate
MHAParser, 126
commit
DynComp::dc_afterburn_vars_t, 458
commit_pending
DynComp::dc_afterburn_t, 455
commit_t
MHAParser::commit_t< receiver_t >, 1054
Communication between algorithms, 23
get_var_float, 26
get_var_int, 26
get_var_spectrum, 25
get_var_vfloat, 27
get_var_waveform, 25
comp_each_iter
lpc, 663
lpc_config, 678
comp_iter
lpc_config, 678
compensation
windnoise::cfg_t, 1512
COMPILER_ID
compiler_id.hh, 1541
compiler_id.cpp, 1540

© 2005-2021 HörTech gGmbH, Oldenburg

compiler_id.hh, 1540
COMPILER_ID, 1541
COMPILER_ID_MAJOR, 1540
COMPILER_ID_MINOR, 1540
COMPILER_ID_PATCH, 1540
COMPILER_ID_VENDOR, 1540
COMPILER_ID_VERSION, 1541
COMPILER_ID_VERSION_HELPER1, 1541
COMPILER_ID_VERSION_HELPER2, 1540
COMPILER_ID_MAJOR
compiler_id.hh, 1540
COMPILER_ID_MINOR
compiler_id.hh, 1540
COMPILER_ID_PATCH
compiler_id.hh, 1540
COMPILER_ID_VENDOR
compiler_id.hh, 1540
COMPILER_ID_VERSION
compiler_id.hh, 1541
COMPILER_ID_VERSION_HELPER1
compiler_id.hh, 1541
COMPILER_ID_VERSION_HELPER2
compiler_id.hh, 1540
Complex arithmetics in the openMHA, 58
_conjugate, 67
_reciprocal, 67
abs, 65
abs2, 65
almost, 68
angle, 62
conjugate, 67
expi, 61, 64
mha_complex, 60
normalize, 68
operator!=, 67
operator<, 69
operator*, 64, 65
operator*=, 64
operator+, 62, 63
operator+=, 62
operator-, 63, 66
operator-=, 63
operator/, 65, 66
operator/=, 65, 66
operator==, 66
reciprocal, 67
safe_div, 66
set, 60, 61
stdcomplex, 61
complex_bandpass_t
MHAFilter::complex_bandpass_t, 873
complex_data
testplugin::ac_parser_t, 1482
complex_filter.cpp, 1541
complex_filter.h, 1541
complex_mon_t
MHAParser::complex_mon_t, 1056
complex_ofs
MHASignal::matrix_t, 1236
complex_scale_channel.cpp, 1542
complex_scale_channel_t, 361
complex_scale_channel_t, 362
factor, 363
patchbay, 363
prepare, 362
process, 362
scale_ch, 363
update_cfg, 363
complex_t
MHAParser::complex_t, 1058
compression
dc_simple::dc_t, 400
compute_beamW
rohBeam::rohBeam, 1402
compute_delaycomp_vec
rohBeam::rohBeam, 1402
compute_diff2D
rohBeam::rohBeam, 1402
compute_diff3D
rohBeam::rohBeam, 1402
compute_head_model_alpha
rohBeam::rohBeam, 1401
compute_head_model_mat
rohBeam::rohBeam, 1401
compute_head_model_T
rohBeam::rohBeam, 1401
compute_uncorr
rohBeam::rohBeam, 1402
compute_wng
rohBeam::rohBeam, 1403
Concept of Variables and Data Exchange in the openMHA, 4
cond_to_process
analysepath_t, 309
config_file_splitter_t
PluginLoader::config_file_splitter_t, 1363
config_in
testplugin::if_t, 1488
config_out
testplugin::if_t, 1488

© 2005-2021 HörTech gGmbH, Oldenburg

config_parser
calibrator_variables_t, 343
config_parser_t
testplugin::config_parser_t, 1484
config_t
MHAPlugin::config_t< runtime_cfg_t >, 1148
configfile
PluginLoader::config_file_splitter_t, 1364
configname
PluginLoader::config_file_splitter_t, 1364
configs
altconfig_t, 299
configuration, 4
configuration variable, 4
conflux
DynComp::dc_afterburn_rt_t, 452
DynComp::dc_afterburn_vars_t, 457
conjugate
Complex arithmetics in the openMHA, 67
Vector and matrix processing toolbox, 57
connect
MHAEvents::emitter_t, 860
MHAEvents::patchbay_t< receiver_t >, 862, 863
connect_input
MHAJack::client_t, 978
connect_output
MHAJack::client_t, 978
connect_to
MHAJack::port_t, 987
connected
io_asterisk_parser_t, 595
io_tcp_parser_t, 632
Connection
MHA_TCP::Connection, 805
connection_loop
io_asterisk_t, 603
io_tcp_t, 641
connections
mha_tcp::server_t, 824
MHAEvents::emitter_t, 861
connections_in
MHAIOJack::io_jack_t, 945
MHAIOJackdb::io_jack_t, 953
connections_out
MHAIOJack::io_jack_t, 946
MHAIOJackdb::io_jack_t, 953
connector
MHAFilter::adapt_filter_t, 868
MHAFilter::iir_filter_t, 902
MHAParser::mhapluginloader_t, 1094
connector_base_t
MHAEvents::connector_base_t, 854
connector_t
MHAEvents::connector_t< receiver_t >, 857
cons
MHAEvents::patchbay_t< receiver_t >, 863
consecutive_dropouts
droptect_t, 442
CONST_C
rohBeam, 160
contained_frames
MHASignal::ringbuffer_t, 1242
conv2latex
generatemhaplugindoc.cpp, 1557
convert_empty2normal
AuditoryProfile::profile_t::ear_t, 333
convert_f2logf
gaintable.cpp, 1555
copy
MHASignal::spectrum_t, 1251
MHASignal::waveform_t, 1271, 1272
copy_channel
MHASignal::spectrum_t, 1252
MHASignal::waveform_t, 1272
Vector and matrix processing toolbox, 52, 53
copy_error
MHAIOAsterisk.cpp, 1660
MHAIOTCP.cpp, 1686
copy_from_at
MHASignal::waveform_t, 1272
copy_output_spec
MHAPlugin_Split::split_t, 1179
copy_output_wave
MHAPlugin_Split::split_t, 1179
copy_permuted
MHASignal, 149
copyfixedbfoutput
rohBeam::rohConfig, 1409
corr_out
lpc_config, 679
corrLL
rohBeam::rohConfig, 1411
corrRR
rohBeam::rohConfig, 1411
corrXpXp
rohBeam::rohConfig, 1411
corrXpYf

© 2005-2021 HörTech gGmbH, Oldenburg

rohBeam::rohConfig, 1411
corrZZ
rohBeam::rohConfig, 1411
cpuload, 85
cpuload.cpp, 1542
cpuload::cpuload_cfg_t, 363
calc_sine, 365
cpuload_cfg_t, 364
factor, 365
phase, 365
process, 364
result, 365
table, 365
use_sine, 365
write_to_table, 365
cpuload::cpuload_if_t, 366
cpuload_if_t, 367
factor, 368
patchbay, 368
prepare, 367
process, 367
table_size, 368
update, 367
use_sine, 368
cpuload_cfg_t
cpuload::cpuload_cfg_t, 364
cpuload_if_t
cpuload::cpuload_if_t, 367
create_datafile
plugins::hoertech::acrec::acwriter_t, 1385
create_latex_doc
generatemhaplugindoc.cpp, 1558
create_or_replace_var
ac2lsl::cfg_t, 167
create_soundfile
wavwriter_t, 1507
creator
speechnoise_t, 1474
creator_A
MHAFilter::complex_bandpass_t, 873
creator_B
MHAFilter::complex_bandpass_t, 874
cstr_strerror
mha_errno.c, 1591
current
altplugs_t, 304
mha_rt_fifo_t< T >, 793
current_input_signal_buffer_half_index
MHAFilter::partitioned_convolution_t, 920
current_message
mha_tcp::buffered_socket_t, 801
current_output_partition_index
MHAFilter::partitioned_convolution_t, 921
current_powspec
droptect_t, 443
current_thread_priority
MHAPlugin_Split::posix_threads_t, 1174
current_thread_scheduler
MHAPlugin_Split::posix_threads_t, 1174
cv
lsl2ac::save_var_t, 691
plugins::hoertech::acrec::acrec_t, 1381
cvalue
gtfb_analyzer::gtfb_analyzer_cfg_t, 552

d
gsc_adaptive_stage::gsc_adaptive_stage, 543
data
acsave::save_var_t, 227
comm_var_t, 361
DynComp::gaintable_t, 462
MHA_AC::acspace2matrix_t, 729
MHA_AC::double_t, 730
MHA_AC::float_t, 732
MHA_AC::int_t, 734
mha_rt_fifo_element_t< T >, 790
MHAParser::bool_mon_t, 1047
MHAParser::bool_t, 1049
MHAParser::complex_mon_t, 1056
MHAParser::complex_t, 1059
MHAParser::float_mon_t, 1063
MHAParser::float_t, 1066
MHAParser::int_mon_t, 1068
MHAParser::int_t, 1071
MHAParser::kw_t, 1078
MHAParser::mcomplex_mon_t, 1080
MHAParser::mcomplex_t, 1083
MHAParser::mfloat_mon_t, 1085
MHAParser::mfloat_t, 1087
MHAParser::mint_mon_t, 1097
MHAParser::mint_t, 1099
MHAParser::string_mon_t, 1115
MHAParser::string_t, 1117
MHAParser::vcomplex_mon_t, 1122
MHAParser::vcomplex_t, 1124
MHAParser::vfloat_mon_t, 1126
MHAParser::vfloat_t, 1129
MHAParser::vint_mon_t, 1131
MHAParser::vint_t, 1134
MHAParser::vstring_mon_t, 1136
MHAParser::vstring_t, 1138

© 2005-2021 HörTech gGmbH, Oldenburg

MHAPlugin::cfg_node_t< runtime_cfg_t >, 1145
MHASignal::uint_vector_t, 1262
wavwriter_t, 1509
data_is_initialized
MHAParser::base_t, 1043
data_type
ac2lsl::save_var_base_t, 171
ac2lsl::save_var_t< mha_complex_t >, 178
ac2lsl::save_var_t< T >, 174
comm_var_t, 360
testplugin::ac_parser_t, 1482
data_type_
ac2lsl::save_var_t< T >, 175
data_type_t
testplugin::ac_parser_t, 1481
db.cpp, 1542
db2lin
MHASignal, 140
db2sq
MHASignal, 141
db_if_t, 369
~db_if_t, 370
algo, 371
bypass, 371
db_if_t, 370
dbasync_native::db_if_t, 374
fragsize, 371
patchbay, 370
plugloader, 371
prepare, 370
process, 370
release, 370
db_t, 371
db_t, 372
inner_process, 372
plugloader, 372
dbasync.cpp, 1542
dbasync_native, 85
gcd, 86
INVALID_THREAD_PRIORITY, 86
thread_start, 86
dbasync_native::db_if_t, 373
~db_if_t, 374
algo, 376
db_if_t, 374
delay, 375
fragsize, 375
framework_thread_priority, 376
framework_thread_scheduler, 376
plugloader, 375
prepare, 374
process, 374
release, 375
sub_ac, 375
worker_thread_priority, 375
worker_thread_scheduler, 375
dbasync_native::dbasync_t, 376
~dbasync_t, 377
attr, 378
dbasync_t, 377
inner_input, 378
outer_output, 378
outer_process, 377
plugloader, 378
priority, 378
scheduler, 378
svc, 378
thread, 378
dbasync_native::delay_check_t, 379
delay_check_t, 379
dbasync_t
dbasync_native::dbasync_t, 377
DBG
MHAIOalsa.cpp, 1652
dbspl2pa
MHASignal, 142
dbspl2pa2
MHASignal, 143
DC
dc_simple, 87
dc, 86
dc.cpp, 1543
DUPVEC, 1543
get_audiochannels, 1543
dc.hh, 1544
dc::dc_if_t, 380
algo, 382
dc_if_t, 381
patchbay, 382
prepare, 381
process, 381
update, 381
update_monitors, 381
dc::dc_t, 382
attack, 385
bypass, 386
dc_t, 383
decay, 386
explicit_insert, 384
fftlen, 386

© 2005-2021 HörTech gGmbH, Oldenburg

get_attack_filter_state, 385
get_decay_filter_state, 385
get_level_in_db, 384
get_level_in_db_adjusted, 385
get_nbands, 384
get_nch, 384
get_rmslevel_filter_state, 385
gt, 385
level_in_db, 386
level_in_db_adjusted, 386
log_interp, 386
naudiochannels, 386
nbands, 386
nch, 386
offset, 385
process, 384
rmslevel, 385
dc::dc_vars_t, 387
band_weights, 390
bw_name, 389
bypass, 389
center_frequencies, 390
cf_name, 389
chname, 389
clientid, 390
dc_vars_t, 388
edge_frequencies, 390
ef_name, 389
filterbank, 389
filtered_level, 390
gainrule, 390
gtdata, 388
gtmin, 388
gtstep, 388
input_level, 390
log_interp, 389
modified, 390
offset, 389
preset, 390
tauattack, 388
taudecay, 389
taurmslevel, 388
dc::dc_vars_validator_t, 391
dc_vars_validator_t, 391
dc_afterburn.cpp, 1544
mylogf, 1544
dc_afterburn.h, 1544
dc_afterburn_rt_t
DynComp::dc_afterburn_rt_t, 451
dc_afterburn_t
DynComp::dc_afterburn_t, 454
dc_afterburn_vars_t
DynComp::dc_afterburn_vars_t, 457
dc_if_t
dc::dc_if_t, 381
dc_simple::dc_if_t, 393
dc_simple, 86
DC, 87
force_resize, 88
LEVEL, 87
not_zero, 88
test_fail, 87
dc_simple.cpp, 1545
dc_simple.hh, 1545
dc_simple::dc_if_t, 392
center_frequencies, 396
clientid, 395
dc_if_t, 393
edge_frequencies, 396
filterbank, 396
gainrule, 396
has_been_modified, 395
modified, 396
mon_g, 396
mon_l, 396
patchbay, 396
prepare, 393
prepared, 397
preset, 396
process, 394
read_modified, 395
release, 394
update_dc, 395
update_gain_mon, 395
update_level, 395
update_level_mon, 395
dc_simple::dc_t, 397
compression, 400
dc_t, 398
expansion, 400
expansion_threshold, 400
limiter, 400
limiter_threshold, 400
maxgain, 400
mon_g, 401
mon_l, 401
nbands, 400
process, 399
dc_simple::dc_t::line_t, 401
line_t, 402
m, 403
operator(), 402

© 2005-2021 HörTech gGmbH, Oldenburg

y0, 403
dc_simple::dc_vars_t, 403
bypass, 405
dc_vars_t, 404
expansion_slope, 405
expansion_threshold, 405
g50, 404
g80, 404
limiter_threshold, 405
maxgain, 404
tauattack, 405
taudecay, 405
dc_simple::dc_vars_validator_t, 406
dc_vars_validator_t, 406
dc_simple::level_smoother_t, 407
attack, 409
decay, 409
fftlen, 410
level_smoother_t, 408
level_spec, 410
level_wave, 410
nbands, 409
process, 408, 409
dc_t
dc::dc_t, 383
dc_simple::dc_t, 398
dc_vars_t
dc::dc_vars_t, 388
dc_simple::dc_vars_t, 404
dc_vars_validator_t
dc::dc_vars_validator_t, 391
dc_simple::dc_vars_validator_t, 406
deallocate_domains
MHAPlugin_Split::domain_handler_t, 1166
DEBUG
addsndfile.cpp, 1532
audiometerbackend.cpp, 1537
browsemhaplugins.cpp, 1538
fader_wave.cpp, 1552
MHAIOFile.cpp, 1662
debug
io_asterisk_parser_t, 595
io_tcp_parser_t, 631
debug_file
io_asterisk_parser_t, 596
io_tcp_parser_t, 632
debug_filename
io_asterisk_parser_t, 596
io_tcp_parser_t, 632
decay
cfg_t, 346
dc::dc_t, 386
dc_simple::level_smoother_t, 409
softclip_t, 1460
softclipper_t, 1462
decomb_coeffs
adm_rtconfig_t, 279
decomb_order
adm_if_t, 274
decrease_condition
mha_fifo_posix_threads_t, 776
decrement
mha_fifo_posix_threads_t, 775
mha_fifo_thread_platform_t, 787
DEFAULT_RETSIZE
mha_parser.hh, 1622
defaultHighInputLatency
MHAIOPortAudio::device_info_t, 958
defaultHighOutputLatency
MHAIOPortAudio::device_info_t, 958
defaultLowInputLatency
MHAIOPortAudio::device_info_t, 958
defaultLowOutputLatency
MHAIOPortAudio::device_info_t, 958
defaultSampleRate
MHAIOPortAudio::device_info_t, 958
Delay
ADM::Delay< F >, 266
delay, 89
coherence::vars_t, 355
dbasync_native::db_if_t, 375
delaysum::delaysum_wave_if_t, 415
mha_dblbuf_t< FIFO >, 753
MHAFilter::gamma_flt_t, 896
MHAFilter::partitioned_convolution_t::index_t, 923
MHAPlugin_Split::split_t, 1182
MHASignal::delay_spec_t, 1203
MHASignal::delay_wave_t, 1207
delay.cpp, 1546
delay.hh, 1546
delay::interface_t, 410
delays, 412
interface_t, 411
patchbay, 412
prepare, 411
process, 411
update, 412
delay_ac
ac2wave_if_t, 188
ac2wave_t, 190

© 2005-2021 HörTech gGmbH, Oldenburg

delay_check_t
dbasync_native::delay_check_t, 379
delay_d
adaptive_feedback_canceller, 243
DELAY_FREQ
ADM, 82
delay_in
ac2wave_if_t, 188
ac2wave_t, 190
delay_spec_t
MHASignal::delay_spec_t, 1202
delay_t
MHASignal::delay_t, 1204
delay_w
adaptive_feedback_canceller, 243
delay_wave_t
MHASignal::delay_wave_t, 1206
delayComp
rohBeam::rohConfig, 1410
delays
delay::interface_t, 412
MHASignal::delay_t, 1205
delays_in
MHAIOJack::io_jack_t, 945
delays_out
MHAIOJack::io_jack_t, 946
delaysum, 89
delaysum::delaysum_wave_if_t, 412
delay, 415
delaysum_wave_if_t, 414
patchbay, 415
prepare, 414
process, 414
release, 414
update_cfg, 414
weights, 415
delaysum::delaysum_wave_t, 415
delaysum_wave_t, 416
out, 417
process, 417
weights, 417
delaysum_spec, 89
delaysum_spec.cpp, 1547
delaysum_spec::delaysum_spec_if_t, 418
delaysum_spec_if_t, 419
gain, 419
groupdelay, 419
patchbay, 420
prepare, 419
process, 419
update_cfg, 419
delaysum_spec::delaysum_t, 420
delaysum_t, 420
output, 421
process, 420
scale, 421
delaysum_spec_if_t
delaysum_spec::delaysum_spec_if_t, 419
delaysum_t
delaysum_spec::delaysum_t, 420
delaysum_wave.cpp, 1547
delaysum_wave_if_t
delaysum::delaysum_wave_if_t, 414
delaysum_wave_t
delaysum::delaysum_wave_t, 416
delete_plug
altplugs_t, 304
DELT
gsc_adaptive_stage, 96
delta_phi
fshift::fshift_config_t, 511
fshift_hilbert::hilbert_shifter_t, 522
delta_phi_total
fshift::fshift_config_t, 511
fshift_hilbert::hilbert_shifter_t, 522
delta_pitch
smooth_cepstrum::smooth_cepstrum_if_t, 1441
smooth_cepstrum::smooth_params, 1451
descriptor
mha_audio_t, 743
desired_chan
gsc_adaptive_stage::gsc_adaptive_stage, 541
desired_delay
gtfb_simple_t, 573
desired_fill_count
mha_drifter_fifo_t< T >, 762
detected
windnoise::if_t, 1517
detected_acname
windnoise::if_t, 1517
dev
audiometerbackend::lnn3rdoct_t, 322
dev_in
io_alsa_t, 583
dev_out
io_alsa_t, 583
device
alsa_dev_par_parser_t, 290
device_index_in
MHAIOPortAudio::io_portaudio_t, 964

© 2005-2021 HörTech gGmbH, Oldenburg

device_index_in_updated
MHAIOPortAudio::io_portaudio_t, 961
device_index_out
MHAIOPortAudio::io_portaudio_t, 964
device_index_out_updated
MHAIOPortAudio::io_portaudio_t, 961
device_info
MHAIOPortAudio::io_portaudio_t, 962
device_info_t
MHAIOPortAudio::device_info_t, 956
device_name_in
MHAIOPortAudio::io_portaudio_t, 964
device_name_in_updated
MHAIOPortAudio::io_portaudio_t, 961
device_name_out
MHAIOPortAudio::io_portaudio_t, 964
device_name_out_updated
MHAIOPortAudio::io_portaudio_t, 961
df
fshift::fshift_config_t, 511
fshift::fshift_t, 514
fshift_hilbert::frequency_translator_t, 517
fshift_hilbert::hilbert_shifter_t, 521
diag_loading_mu
rohBeam::rohBeam, 1404
diff_coeffs
mha_filter.cpp, 1599
diff_t
MHAFilter::diff_t, 877
digits
mha_error_helpers, 99
dimension
MHASignal::matrix_t, 1230
dimensions
acmon::acmon_t, 209
dimstr
acmon::ac_monitor_t, 205
dir
mha_channel_info_t, 745
dir_t
MHAJack::port_t, 985
dir_type
MHAJack::port_t, 987
dis
dropgen_t, 439
Discard
lsl2ac, 97
discard
MHASignal::ringbuffer_t, 1243
disconnect
MHAEvents::emitter_t, 860
disk_write_threshold_min_num_samples
plugins::hoertech::acrec::acwriter_t, 1386
diskbuffer
plugins::hoertech::acrec::acwriter_t, 1387
dispmode
acmon::acmon_t, 209
dist
plingploing::plingploing_t, 1347
dist1
plingploing::plingploing_t, 1347
distance
mha_direction_t, 756
distances
adm_if_t, 273
dm
lpc_burglattice_config, 675
do_clipping
calibrator_variables_t, 343
do_get_var
testplugin::ac_parser_t, 1481
do_insert_var
testplugin::ac_parser_t, 1481
DO_RESAMPLE
addsndfile, 80
doagcc
doasvm_feature_extraction_config, 431
doasvm
doasvm_classification_config, 426
doasvm_classification, 421
~doasvm_classification, 422
angles, 424
b, 424
doasvm_classification, 422
max_p_ind_name, 424
p_name, 424
patchbay, 424
prepare, 423
process, 423
release, 423
update_cfg, 423
vGCC_name, 424
w, 424
x, 424
y, 424
doasvm_classification.cpp, 1547
INSERT_PATCH, 1548
PATCH_VAR, 1547
doasvm_classification.h, 1548
doasvm_classification_config, 425
~doasvm_classification_config, 425
ac, 426

© 2005-2021 HörTech gGmbH, Oldenburg

c, 426
doasvm, 426
doasvm_classification_config, 425
p, 426
p_max, 426
process, 426
doasvm_feature_extraction, 427
~doasvm_feature_extraction, 428
doasvm_feature_extraction, 428
fftlen, 429
max_lag, 429
nupsample, 429
patchbay, 429
prepare, 428
process, 428
release, 429
update_cfg, 429
vGCC_name, 429
doasvm_feature_extraction.cpp, 1548
INSERT_PATCH, 1548
PATCH_VAR, 1548
doasvm_feature_extraction.h, 1549
doasvm_feature_extraction_config, 430
~doasvm_feature_extraction_config, 431
doagcc, 431
doasvm_feature_extraction_config, 430
fft, 432
fftlen, 431
G, 433
G_length, 431
GCC_end, 432
GCC_start, 431
hifftwin, 432
hifftwin_sum, 432
hwin, 432
ifft, 432
in_spec, 433
proc_wave, 432
process, 431
vGCC, 432
vGCC_ac, 432
wndlen, 431
doc_appendix.h, 1549
doc_examples.h, 1549
doc_frameworks.h, 1549
doc_general.h, 1549
doc_kernel.h, 1549
doc_matlab.h, 1549
doc_mhamain.h, 1549
doc_parser.h, 1549
doc_plugins.h, 1549
doc_system.h, 1549
doc_toolbox.h, 1549
doCircularComp
gsc_adaptive_stage::gsc_adaptive_stage, 541
gsc_adaptive_stage::gsc_adaptive_stage_if, 548
document_io_plugin
analysemhaplugin.cpp, 1535
document_plugin
analysemhaplugin.cpp, 1535
documentation
plugindescription_t, 1359
domain
mhaconfig_t, 852
MHAParser::mhaconfig_mon_t, 1089
MHAPlugin_Split::splitted_part_t, 1187
rmslevel::rmslevel_t, 1397
testplugin::config_parser_t, 1485
domain_handler_t
MHAPlugin_Split::domain_handler_t, 1164
DONT_RESAMPLE_PERMISSIVE
addsndfile, 80
DONT_RESAMPLE_STRICT
addsndfile, 80
double2acvar, 90
double2acvar.cpp, 1549
double2acvar::double2acvar_t, 433
~double2acvar_t, 435
ac_double, 436
double2acvar_t, 434
is_prepared, 436
on_configuration_update, 436
patchbay, 436
poll_latest_value_and_reinsert, 435
prepare_, 435
process, 435
release_, 436
double2acvar_t
double2acvar::double2acvar_t, 434
double_t
MHA_AC::double_t, 730
doublebuffer_t
MHASignal::doublebuffer_t, 1208
down
MHASignal::schroeder_t, 1246
downsample.cpp, 1550
downsampling_factor
MHAFilter::polyphase_resampling_t, 927
downscale

© 2005-2021 HörTech gGmbH, Oldenburg

MHASignal::quantizer_t, 1240
drain
DynComp::dc_afterburn_vars_t, 457
drain_inv
DynComp::dc_afterburn_rt_t, 452
drand
plingploing, 156
dropgen.cpp, 1550
dropgen_t, 437
chance, 439
dis, 439
dropgen_t, 438
max_sleep_time, 439
min_sleep_time, 439
patchbay, 439
prepare, 438
process, 438
r, 439
random_engine, 439
release, 438
dropouts
droptect_t, 442
droptect.cpp, 1550
droptect_t, 440
blocks, 442
consecutive_dropouts, 442
current_powspec, 443
dropouts, 442
droptect_t, 441
filter_activated, 443
filtered_powspec, 443
filtered_powspec_mon, 443
level_mon, 443
period, 443
prepare, 441
process, 442
release, 442
reset, 442
tau, 443
threshold, 443
ds_t, 444
antialias, 445
ds_t, 445
prepare, 445
process, 445
ratio, 445
release, 445
dt
mha_audio_descriptor_t, 742
dtime
MHA_TCP, 103
dummy_interface_test
MHAIOalsa.cpp, 1654
MHAIOAsterisk.cpp, 1659
MHAIOFile.cpp, 1664
MHAIOJack.cpp, 1668
MHAIOJackdb.cpp, 1672
MHAIOParser.cpp, 1676
MHAIOPortAudio.cpp, 1680
MHAIOTCP.cpp, 1686
dummy_jack_proc_cb
mhajack.cpp, 1689
dummy_threads_t
MHAPlugin_Split::dummy_threads_t, 1170
dump_mha
fw_t, 530
dup
MHAFilter::thirdoctave_analyzer_t, 936
duplicate_vector
gtfb_simple_rt_t, 567
DUPVEC
dc.cpp, 1543
dupvec
Vector and matrix processing toolbox, 41
dupvec_chk
Vector and matrix processing toolbox, 41
dur_
plingploing::plingploing_t, 1346
dynamiclib_t, 446
~dynamiclib_t, 448
dynamiclib_t, 447, 448
fullname, 450
getmodulename, 449
getname, 449
h, 450
load_lib, 449
modulename, 450
resolve, 448
resolve_checked, 448
DynComp, 90
interp1, 90
interp2, 91
DynComp::dc_afterburn_rt_t, 450
burn, 451
conflux, 452
dc_afterburn_rt_t, 451
drain_inv, 452
lp, 452
maxgain, 452
mpo_inv, 452
DynComp::dc_afterburn_t, 453

© 2005-2021 HörTech gGmbH, Oldenburg

_cf, 455
_channels, 455
_srate, 455
burn, 454
commit_pending, 455
dc_afterburn_t, 454
fb_pars_configured, 455
patchbay, 455
set_fb_pars, 454
unset_fb_pars, 454
update, 454
update_burner, 454
DynComp::dc_afterburn_vars_t, 456
bypass, 458
commit, 458
conflux, 457
dc_afterburn_vars_t, 457
drain, 457
f, 457
maxgain, 457
mpo, 457
taugain, 457
DynComp::gaintable_t, 458
~gaintable_t, 459
data, 462
gaintable_t, 459
get_gain, 460, 461
get_iofun, 461
get_vF, 461
get_vL, 461
nbands, 461
nchannels, 461
num_channels, 462
num_F, 462
num_L, 462
update, 460
vF, 462
vFlog, 462
vL, 462

E
gsc_adaptive_stage::gsc_adaptive_stage, 543
e
gsc_adaptive_stage::gsc_adaptive_stage, 543
E2
gsc_adaptive_stage::gsc_adaptive_stage, 543
e_out
gsc_adaptive_stage::gsc_adaptive_stage, 544
ear_t
AuditoryProfile::parser_t::ear_t, 328
edge_frequencies
dc::dc_vars_t, 390
dc_simple::dc_if_t, 396
ef
MHAOvlFilter::fftfb_vars_t, 1014
ef2bands
MHAOvlFilter::fspacing_t, 1020
ef_h
MHAOvlFilter::band_descriptor_t, 1002
ef_l
MHAOvlFilter::band_descriptor_t, 1002
ef_name
dc::dc_vars_t, 389
efv
MHAOvlFilter::fftfb_ac_info_t, 1006
multibandcompressor::fftfb_plug_t, 1303
element_gain_name
combc_if_t, 357
gtfb_simple_t, 573
element_gain_name_
combc_t, 359
gtfb_simple_rt_t, 569
elevation
mha_direction_t, 756
emit_event
MHAEvents::connector_base_t, 854, 855
MHAEvents::connector_t< receiver_t >, 857, 858
emitter
MHAEvents::connector_t< receiver_t >, 858
emitter_die
MHAEvents::connector_base_t, 855
emitter_is_alive
MHAEvents::connector_base_t, 855
empty_string
MHAParser::keyword_list_t, 1075
enable_adaptive_beam
rohBeam::configOptions, 1398
rohBeam::rohBeam, 1404
rohBeam::rohConfig, 1409
enable_export
rohBeam::rohBeam, 1405
enable_wng_optimization
rohBeam::rohBeam, 1405
end_time
MHA_TCP::Timeout_Event, 834
entries
MHAParser::keyword_list_t, 1075

© 2005-2021 HörTech gGmbH, Oldenburg

MHAParser::parser_t, 1108
entry
MHAParser::entry_t, 1060
entry_map_t
MHAParser, 126
entry_t
MHAParser::entry_t, 1059
envelope_delay
MHAFilter::gamma_flt_t, 896
envreplace
MHAParser, 127
eof
MHA_TCP::Connection, 807
EPrew
adaptive_feedback_canceller_config, 249
EPSILON
lpc_bl_predictor.h, 1573
lpc_burg-lattice.h, 1574
epsilon
smoothgains_bridge::overlapadd_if_t, 1455
equal_dim
Vector and matrix processing toolbox, 42
equalize, 91
equalize.cpp, 1550
equalize::cfg_t, 463
~cfg_t, 463
cfg_t, 463
fftgains, 464
nchannels, 464
num_bins, 464
operator=, 463
equalize::freqgains_t, 464
fftgains, 466
freqgains_t, 465
id, 466
patchbay, 466
prepare, 465
process, 465
update_gains, 466
update_id, 466
equidist2bands
MHAOvlFilter::fspacing_t, 1020
erb_hz_f_hz
speechnoise.cpp, 1706
ERR_IHANDLE
MHAIOalsa.cpp, 1653
MHAIOAsterisk.cpp, 1657
MHAIOFile.cpp, 1663
MHAIOJack.cpp, 1667
MHAIOJackdb.cpp, 1671
MHAIOParser.cpp, 1675
MHAIOPortAudio.cpp, 1679
MHAIOTCP.cpp, 1684
err_in
MHAFilter::adapt_filter_param_t, 864
MHAFilter::adapt_filter_t, 868
ERR_SUCCESS
MHAIOalsa.cpp, 1653
MHAIOAsterisk.cpp, 1657
MHAIOFile.cpp, 1662
MHAIOJack.cpp, 1667
MHAIOJackdb.cpp, 1671
MHAIOParser.cpp, 1675
MHAIOPortAudio.cpp, 1679
MHAIOTCP.cpp, 1684
ERR_USER
MHAIOalsa.cpp, 1653
MHAIOAsterisk.cpp, 1657
MHAIOFile.cpp, 1663
MHAIOJack.cpp, 1667
MHAIOJackdb.cpp, 1671
MHAIOParser.cpp, 1675
MHAIOPortAudio.cpp, 1679
MHAIOTCP.cpp, 1684
error
mha_fifo_lw_t< T >, 773
MHA_TCP::Thread, 832
Error handling in the openMHA, 28
MHA_assert, 29
MHA_assert_equal, 29
mha_debug, 29
MHA_ErrorMsg, 28
error_h
osc_server_t, 1325
errorlog
fw_t, 529
ESTIM_CUR
nlms_wave.cpp, 1696
ESTIM_PREV
nlms_wave.cpp, 1696
estimateDebug
noise_psd_estimator::noise_psd_estimator_t, 1317
ESTIMATION_TYPES
nlms_wave.cpp, 1696
estimtype
nlms_t, 1311
event_add_plug
altplugs_t, 303
event_delete_plug
altplugs_t, 303

© 2005-2021 HörTech gGmbH, Oldenburg

event_select_all
altconfig_t, 298
event_select_plug
altplugs_t, 303
event_set_plugs
altplugs_t, 302
event_start_recording
acsave::acsave_t, 220
event_stop_and_flush
acsave::acsave_t, 220
eventhandler
MHAEvents::connector_t< receiver_t >, 858
eventhandler_s
MHAEvents::connector_t< receiver_t >, 858
eventhandler_suu
MHAEvents::connector_t< receiver_t >, 859
Events
MHA_TCP::Event_Watcher, 812
events
MHA_TCP::Event_Watcher, 813
example1.cpp, 1550
example1_t, 467
example1_t, 468
prepare, 468
process, 468
release, 468
example2.cpp, 1551
example2_t, 469
example2_t, 470
factor, 472
prepare, 471
process, 472
release, 472
scale_ch, 472
example3.cpp, 1551
example3_t, 473
example3_t, 474
factor, 476
on_prereadaccess, 475
on_scale_ch_readaccess, 475
on_scale_ch_valuechanged, 475
on_scale_ch_writeaccess, 475
patchbay, 477
prepare, 475
prepared, 476
process, 476
release, 475
scale_ch, 476
example4.cpp, 1551
example4_t, 477
example4_t, 478
factor, 480
on_prereadaccess, 479
on_scale_ch_readaccess, 479
on_scale_ch_valuechanged, 479
on_scale_ch_writeaccess, 479
patchbay, 481
prepare, 479
prepared, 481
process, 480
release, 480
scale_ch, 480
example5.cpp, 1551
example5_t, 481
channel, 482
example5_t, 481
process, 482
scale, 482
example6.cpp, 1551
example6_t, 482
channel_no, 484
example6_t, 483
patchbay, 484
prepare, 483
process, 483
rmsdb, 484
update_cfg, 484
example7.cpp, 1552
example7.hh, 1552
example7_t, 485
example7_t, 485
prepare, 486
process, 486
release, 486
exec_fw_command
fw_t, 527
existed_before
mha_stash_environment_variable_t, 797
exit_on_stop
fw_t, 529
exit_request
fw_t, 526
plugins::hoertech::acrec::acwriter_t, 1385
wavwriter_t, 1507
expansion
dc_simple::dc_t, 400
expansion_slope
dc_simple::dc_vars_t, 405
expansion_threshold

© 2005-2021 HörTech gGmbH, Oldenburg

dc_simple::dc_t, 400
dc_simple::dc_vars_t, 405
expflt
MHAOvlFilter::ShapeFun, 122
expi
Complex arithmetics in the openMHA, 61, 64
explicit_insert
dc::dc_t, 384
export_beam_design
rohBeam::rohBeam, 1403
export_to
MHASignal::spectrum_t, 1252
MHASignal::waveform_t, 1273
expression_t, 486
MHAParser::expression_t, 1060, 1061
extern_connector
MHAParser::commit_t< receiver_t >, 1054

F
adaptive_feedback_canceller_config, 246
rt_nlms_t, 1419
f
AuditoryProfile::parser_t::fmap_t, 330
DynComp::dc_afterburn_vars_t, 457
io_tcp_sound_t::float_union, 638
MHAOvlFilter::fftfb_vars_t, 1013
MHAOvlFilter::fscale_t, 1018
f0_high
smooth_cepstrum::smooth_cepstrum_if_t, 1441
smooth_cepstrum::smooth_params, 1451
f0_low
smooth_cepstrum::smooth_cepstrum_if_t, 1441
smooth_cepstrum::smooth_params, 1451
f_est
lpc_bl_predictor_config, 669
f_hz
MHAOvlFilter::fscale_t, 1018
F_Uflt
adaptive_feedback_canceller_config, 248
factor
complex_scale_channel_t, 363
cpuload::cpuload_cfg_t, 365
cpuload::cpuload_if_t, 368
example2_t, 472
example3_t, 476
example4_t, 480
plugin_interface_t, 1356
fader_if_t, 487
actgains, 489
fader_if_t, 488
newgains, 489
patchbay, 488
prepare, 488
process, 488
tau, 488
update_cfg, 488
fader_spec.cpp, 1552
fader_wave, 92
level_adaptor, 92
fader_wave.cpp, 1552
DEBUG, 1552
fader_wave::fader_wave_if_t, 489
fader_wave_if_t, 490
gain, 491
patchbay, 491
prepare, 490
prepared, 491
process, 490
ramplen, 491
release, 490
set_level, 491
fader_wave::level_adapt_t, 492
can_update, 493
get_level, 493
ilen, 493
l_new, 493
l_old, 494
level_adapt_t, 492
pos, 493
update_frame, 493
wnd, 493
fader_wave_if_t
fader_wave::fader_wave_if_t, 490
fail_on_async_jackerr
MHAIOJackdb::io_jack_t, 953
fail_on_async_jackerror
MHAIOJackdb::io_jack_t, 950
MHAJack::client_t, 984
fail_on_nonmonotonic
MHAOvlFilter::fftfb_vars_t, 1013
fail_on_nonmonotonic_cf
MHAOvlFilter::fspacing_t, 1020
fail_on_unique_bins
MHAOvlFilter::fftfb_vars_t, 1013
fail_on_unique_fftbins
MHAOvlFilter::fspacing_t, 1020
fallback_spec
altplugs_t, 305
fallback_wave

© 2005-2021 HörTech gGmbH, Oldenburg

altplugs_t, 305
Fast Fourier Transform functions, 70
mha_fft_backward, 75
mha_fft_backward_scale, 76
mha_fft_forward, 75
mha_fft_forward_scale, 76
mha_fft_free, 72
mha_fft_new, 71
mha_fft_spec2wave, 73, 74
mha_fft_spec2wave_scale, 77
mha_fft_t, 71
mha_fft_wave2spec, 72, 73
mha_fft_wave2spec_scale, 76
fatallog
fw_t, 530
fb
MHAFilter::thirdoctave_analyzer_t, 936
fb_acinfo
fftfilterbank::fftfb_plug_t, 509
fb_pars_configured
DynComp::dc_afterburn_t, 455
fbpow
fftfbpow::fftfbpow_t, 498
fcn_init
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 708
fcn_prepare
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 709
fcn_process_ss
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 709
fcn_process_sw
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 709
fcn_process_ws
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 709
fcn_process_ww
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 708
fcn_release
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 709
fcn_terminate
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 708
fd
MHA_TCP::Connection, 811
MHA_TCP::OS_EVENT_TYPE, 815
fft
doasvm_feature_extraction_config, 432
MHAFilter::fftfilter_t, 882
MHAFilter::fftfilterbank_t, 887
MHAFilter::partitioned_convolution_t, 921
MHAFilter::smoothspec_t, 934
overlapadd::overlapadd_t, 1336
fft_find_bin
fshift, 94
fft_t
MHASignal::fft_t, 1212
fftfb_ac_info_t
MHAOvlFilter::fftfb_ac_info_t, 1005
fftfb_interface_t
fftfilterbank::fftfb_interface_t, 504
fftfb_plug_t
fftfilterbank::fftfb_plug_t, 508
multibandcompressor::fftfb_plug_t, 1302
fftfb_t
MHAOvlFilter::fftfb_t, 1007
fftfb_vars_t
MHAOvlFilter::fftfb_vars_t, 1012
fftfbpow, 92
fftfbpow.cpp, 1553
fftfbpow::fftfbpow_interface_t, 494
fftfbpow_interface_t, 495
name, 496
patchbay, 497
prepare, 495
process, 496
update_cfg, 496
fftfbpow::fftfbpow_t, 497
fbpow, 498
fftfbpow_t, 498
fftfbpow_interface_t
fftfbpow::fftfbpow_interface_t, 495
fftfbpow_t
fftfbpow::fftfbpow_t, 498
fftfilt
fftfilter::fftfilter_t, 501
fftfilter, 92
irs_length, 93
irs_validator, 93
fftfilter.cpp, 1553
fftfilter::fftfilter_t, 499
channels, 500
fftfilt, 501
fftfilter_t, 499
fftlen, 500
fragsize, 500
irslen, 500
process, 500

fftfilter::interface_t, 501
fftlen, 503
fftlen_final, 503
interface_t, 502
irs, 503
patchbay, 503
prepare, 502
process, 502
update, 503
fftfilter_t
fftfilter::fftfilter_t, 499
MHAFilter::fftfilter_t, 878
fftfilterbank, 94
fftfilterbank.cpp, 1553
fftfilterbank::fftfb_interface_t, 504
algo, 506
fftfb_interface_t, 504
nbands, 506
nchannels, 506
patchbay, 506
prepare, 505
prepared, 506
process, 505, 506
release, 505
return_imag, 506
update_cfg, 506
fftfilterbank::fftfb_plug_t, 507
fb_acinfo, 509
fftfb_plug_t, 508
imag, 509
insert, 508
process, 508
return_imag_, 509
s_out, 509
fftfilterbank_t
MHAFilter::fftfilterbank_t, 883
fftgains
equalize::cfg_t, 464
equalize::freqgains_t, 466
fftlen
dc::dc_t, 386
dc_simple::level_smoother_t, 410
doasvm_feature_extraction, 429
doasvm_feature_extraction_config, 431
fftfilter::fftfilter_t, 500
fftfilter::interface_t, 503
level_matching::level_matching_config_t, 653
mhaconfig_t, 853
MHAFilter::fftfilter_t, 881
MHAFilter::fftfilterbank_t, 886
MHAFilter::smoothspec_t, 933
MHAOvlFilter::fftfb_t, 1010
MHAOvlFilter::overlap_save_filterbank_t::vars_t, 1027
MHAParser::mhaconfig_mon_t, 1090
rmslevel::rmslevel_t, 1397
smooth_cepstrum::smooth_cepstrum_t, 1445
testplugin::config_parser_t, 1485
fftlen_final
fftfilter::interface_t, 503
fftw_plan_fft
MHASignal::fft_t, 1215
fftw_plan_ifft
MHASignal::fft_t, 1215
fftw_plan_spec2wave
MHASignal::fft_t, 1215
fftw_plan_wave2spec
MHASignal::fft_t, 1215
fhz2bandno
speechnoise.cpp, 1706
fifo
plugins::hoertech::acrec::acwriter_t, 1386
wavwriter_t, 1508
fifo_size
mha_dblbuf_t< FIFO >, 754
fifolen
analysispath_if_t, 312
plugins::hoertech::acrec::acrec_t, 1381
wavrec_t, 1505
fileformat
acsave::acsave_t, 221
filename
addsndfile::addsndfile_if_t, 252
filename_input
io_file_t, 609
filename_output
io_file_t, 609
fill
rmslevel::rmslevel_t, 1396
fill_info
MHAIOPortAudio::device_info_t, 957
MHAIOPortAudio::stream_info_t, 966
filled
MHASignal::async_rmslevel_t, 1202
filter
MHAFilter::adapt_filter_state_t, 865
MHAFilter::adapt_filter_t, 867
MHAFilter::complex_bandpass_t, 874, 875
MHAFilter::fftfilter_t, 879, 880

© 2005-2021 HörTech gGmbH, Oldenburg

MHAFilter::fftfilterbank_t, 884, 885
MHAFilter::filter_t, 890, 891
MHAFilter::iir_filter_t, 900
filter_activated
droptect_t, 443
filter_analytic
MHAOvlFilter::overlap_save_filterbank_analytic_t, 1023
filter_complex
gtfb_analyzer.cpp, 1560
filter_partitions
MHAFilter::partitioned_convolution_t, 919
filter_real
gtfb_analyzer.cpp, 1560
filter_simd
gtfb_simd.cpp, 1566
filter_sisd_complex
gtfb_simd.cpp, 1563
filter_sisd_real
gtfb_simd.cpp, 1564
filter_t
MHAFilter::filter_t, 889
filterbank
dc::dc_vars_t, 389
dc_simple::dc_if_t, 396
filtered_level
dc::dc_vars_t, 390
filtered_powspec
droptect_t, 443
filtered_powspec_mon
droptect_t, 443
filtershapefun
mha_fftfb.cpp, 1596
FINISHED
MHA_TCP::Thread, 830
fir
calibrator_runtime_layer_t, 337
calibrator_variables_t, 341
fir_lp
MHAFilter, 107
firchannels
MHAFilter::fftfilterbank_t, 886
firfir2fftlen
calibrator_runtime_layer_t, 336
firfirlen
calibrator_runtime_layer_t, 336
flag_allow_empty_bands
MHAOvlFilter::fftfb_vars_t, 1013
flag_terminate_inner_thread
analysepath_t, 309
flags
MHAJack::client_t, 983
flatten
MHASignal::waveform_t, 1266
float_data
testplugin::ac_parser_t, 1482
float_mon_t
MHAParser::float_mon_t, 1062
float_t
MHA_AC::float_t, 732
MHAParser::float_t, 1065
flush_data
acsave::cfg_t, 223
fmap_t
AuditoryProfile::parser_t::fmap_t, 329
fmax
fshift::fshift_t, 514
fshift_hilbert::frequency_translator_t, 517
fmin
fshift::fshift_t, 514
fshift_hilbert::frequency_translator_t, 517
FMTsz
mha_os.h, 1611
fname
acsave::acsave_t, 221
for_each
MHASignal, 139
force_remove_item
MHAParser::parser_t, 1105
force_resize
dc_simple, 88
format
ac2lsl::type_info, 179
io_alsa_t, 584
format_name
wavwriter_t, 1509
forward
lpc_bl_predictor_config, 669
lpc_burglattice_config, 675
MHASignal::fft_t, 1213
forward_scale
MHASignal::fft_t, 1213
fr
spec_fader_t, 1471
frac_old
gsc_adaptive_stage::gsc_adaptive_stage, 540
frag_out
MHAJack::client_avg_t, 970
MHAJack::client_noncont_t, 974
fragsize
alsa_t< T >, 294

© 2005-2021 HörTech gGmbH, Oldenburg

analysispath_if_t, 312
calibrator_variables_t, 342
db_if_t, 371
dbasync_native::db_if_t, 375
fftfilter::fftfilter_t, 500
io_asterisk_sound_t, 599
io_file_t, 608
io_parser_t, 619
io_tcp_sound_t, 637
mconv::MConv, 720
mhaconfig_t, 852
MHAFilter::fftfilter_t, 880
MHAFilter::fftfilterbank_t, 886
MHAFilter::partitioned_convolution_t, 919
MHAFilter::resampling_filter_t, 930
MHAIOPortAudio::io_portaudio_t, 963
MHAJack::client_t, 981
MHAParser::mhaconfig_mon_t, 1089
MHAPlugin_Resampling::resampling_if_t, 1159
testplugin::config_parser_t, 1485
fragsize_in
MHAFilter::blockprocessing_polyphase_resampling_t, 871
fragsize_out
MHAFilter::blockprocessing_polyphase_resampling_t, 871
fragsize_ratio
MHAIOJackdb::io_jack_t, 952
fragsize_validator
MHAFilter::resampling_filter_t, 930
frame
MHA_AC::acspace2matrix_t, 728
frame_data
alsa_t< T >, 294
framecnt
acsave::save_var_t, 227
adm_if_t, 275
frameno
MHA_AC::acspace2matrix_t, 729
noise_psd_estimator::noise_psd_estimator_t, 1318
framerate
ac2osc_t, 184
frames
ac2wave_t, 189
adaptive_feedback_canceller_config, 246
gtfb_analyzer::gtfb_analyzer_cfg_t, 551
rt_nlms_t, 1419
frameshift
fshift_hilbert::hilbert_shifter_t, 522
framework_thread_priority
dbasync_native::db_if_t, 376
MHAPlugin_Split::split_t, 1182
framework_thread_scheduler
dbasync_native::db_if_t, 376
MHAPlugin_Split::split_t, 1181
freq
audiometerbackend::audiometer_if_t, 317
plingploing::plingploing_t, 1347
freq2bin
Vector and matrix processing toolbox, 39
freq_offsets
rmslevel::rmslevel_t, 1397
freqgains_t
equalize::freqgains_t, 465
freqResp
rohBeam::rohConfig, 1411
frequency
sine_t, 1437
frequency_response
MHAFilter::partitioned_convolution_t, 920
frequency_translator_t
fshift_hilbert::frequency_translator_t, 516
FrequencyBinLowPass
windnoise::cfg_t, 1512
front_channel
adm_rtconfig_t, 278
front_channels
adm_if_t, 273
adm_rtconfig_t, 278
frozen_noise_
cfg_t, 346
frozennoise_length
noise_t, 1320
fs
MHAFilter::o1_ar_filter_t, 909
fs_
MHAOvlFilter::fspacing_t, 1021
fscale
gtfb_simple_t, 573
MHAOvlFilter::fftfb_vars_t, 1012
fscale_bw_t
MHAOvlFilter::fscale_bw_t, 1015
fscale_t
MHAOvlFilter::fscale_t, 1017
fshift, 94
fft_find_bin, 94
fshift.cpp, 1554
fshift.hh, 1554
fshift::fshift_config_t, 509
~fshift_config_t, 510

© 2005-2021 HörTech gGmbH, Oldenburg

delta_phi, 511
delta_phi_total, 511
df, 511
fshift_config_t, 510
kmax, 511
kmin, 511
process, 510
fshift::fshift_t, 512
~fshift_t, 513
df, 514
fmax, 514
fmin, 514
fshift_t, 513
m_df, 515
m_fmax, 515
m_fmin, 514
patchbay, 514
prepare, 513
process, 513
release, 514
update_cfg, 514
fshift_config_t
fshift::fshift_config_t, 510
fshift_hilbert, 95
fshift_hilbert.cpp, 1554
fshift_hilbert::frequency_translator_t, 515
df, 517
fmax, 517
fmin, 517
frequency_translator_t, 516
irslen, 518
patchbay, 517
phasemode, 518
prepare, 517
process, 516
release, 517
update, 517
fshift_hilbert::hilbert_shifter_t, 518
~hilbert_shifter_t, 520
analytic, 520
delta_phi, 522
delta_phi_total, 522
df, 521
frameshift, 522
fullspec, 520
hilbert_shifter_t, 519
kmax, 522
kmin, 521
mhafft, 521
mixw_ref, 521
mixw_shift, 521
plan_spec2analytic, 521
process, 520
shifted, 520
fshift_t
fshift::fshift_t, 513
fspacing_t
MHAOvlFilter::fspacing_t, 1019
ft
spec2wave_t, 1469
wave2spec_t, 1501
ftype
MHAOvlFilter::fftfb_vars_t, 1012
fu
rt_nlms_t, 1420
fu_previous
rt_nlms_t, 1420
fuflt
rt_nlms_t, 1420
fullname
dynamiclib_t, 450
MHAParser::base_t, 1041
plugindescription_t, 1359
fullspec
fshift_hilbert::hilbert_shifter_t, 520
fun1
plingploing::plingploing_t, 1347
fun1_key
plingploing::if_t, 1342
plingploing::plingploing_t, 1347
fun1_range
plingploing::if_t, 1343
plingploing::plingploing_t, 1347
fun2
plingploing::plingploing_t, 1347
fun2_key
plingploing::if_t, 1343
plingploing::plingploing_t, 1347
fun2_range
plingploing::if_t, 1343
plingploing::plingploing_t, 1347
fun_t
MHAWindow::fun_t, 1295
funs
MHAOvlFilter::scale_var_t, 1030
fw_cmd
fw_t, 529
fw_exiting
fw_t, 525
fw_fragsize
io_alsa_t, 582
MHAIOJack::io_jack_t, 945

© 2005-2021 HörTech gGmbH, Oldenburg

fw_running
fw_t, 525
fw_samplerate
io_alsa_t, 582
MHAIOJack::io_jack_t, 945
fw_sleep
fw_t, 529
fw_sleep_cmd
fw_t, 528
fw_starting
fw_t, 525
fw_stopped
fw_t, 525
fw_stopping
fw_t, 525
fw_t, 523
~fw_t, 525
ac, 530
async_poll_msg, 528
async_read, 528
b_exit_request, 531
cfin, 530
cfout, 530
dump_mha, 530
errorlog, 529
exec_fw_command, 527
exit_on_stop, 529
exit_request, 526
fatallog, 530
fw_cmd, 529
fw_exiting, 525
fw_running, 525
fw_sleep, 529
fw_sleep_cmd, 528
fw_starting, 525
fw_stopped, 525
fw_stopping, 525
fw_t, 525
fw_unprepared, 525
fw_until, 529
fw_until_cmd, 528
get_input_signal_dimension, 528
get_parserstate, 528
inst_name, 530
io_error, 531
io_lib, 531
io_name, 529
load_io_lib, 528
load_proc_lib, 527
nchannels_out, 529
parserstate, 529
patchbay, 531
plugin_paths, 530
plugins, 530
prepare, 526
prepare_vars, 528
proc_error, 531
proc_error_string, 531
proc_lib, 530
proc_name, 529
process, 527
quit, 526
release, 526
start, 526
started, 527
state, 531
state_t, 525
stop, 526
stopped, 526, 527
fw_unprepared
fw_t, 525
fw_until
fw_t, 529
fw_until_cmd
fw_t, 528
fw_vars_t, 532
fw_vars_t, 532
lock_channels, 532
lock_srate_fragsize, 532
pfragmentsize, 533
pinchannels, 533
psrate, 533
unlock_channels, 532
unlock_srate_fragsize, 532
fwcb
io_asterisk_t, 604
io_tcp_t, 642

G
doasvm_feature_extraction_config, 433
g
coherence::cohflt_t, 351
g50
dc_simple::dc_vars_t, 404
g80
dc_simple::dc_vars_t, 404
G_ERRNO
MHA_TCP, 102
G_length
doasvm_feature_extraction_config, 431
gain, 95
alsa_t< T >, 294
calibrator_runtime_layer_t, 337

© 2005-2021 HörTech gGmbH, Oldenburg

coherence::cohflt_t, 352
delaysum_spec::delaysum_spec_if_t, 419
fader_wave::fader_wave_if_t, 491
multibandcompressor::plugin_signals_t, 1308
gain.cpp, 1555
gain::gain_if_t, 533
gain_if_t, 534
gains, 535
patchbay, 535
prepare, 535
process, 534
release, 535
update_gain, 535
update_minmax, 535
vmax, 536
vmin, 535
gain::scaler_t, 536
scaler_t, 536
gain_ac
ac2wave_if_t, 187
ac2wave_t, 190
gain_delay
coherence::cohflt_t, 352
gain_if_t
gain::gain_if_t, 534
gain_in
ac2wave_if_t, 187
ac2wave_t, 190
gain_min
smooth_cepstrum::smooth_cepstrum_t, 1446
gain_min_db
smooth_cepstrum::smooth_cepstrum_if_t, 1442
smooth_cepstrum::smooth_params, 1452
gain_spec_
cfg_t, 345
gain_wave_
cfg_t, 345
gain_wiener
smooth_cepstrum::smooth_cepstrum_t, 1448
gainrule
dc::dc_vars_t, 390
dc_simple::dc_if_t, 396
gains
adaptive_feedback_canceller, 243
gain::gain_if_t, 535
shadowfilter_end::cfg_t, 1431
spec_fader_t, 1471
gaintable.cpp, 1555
convert_f2logf, 1555
isempty, 1555
gaintable.h, 1555
gaintable_t
DynComp::gaintable_t, 459
gamma_flt_t
MHAFilter::gamma_flt_t, 893
gamma_post
smooth_cepstrum::smooth_cepstrum_t, 1447
gauss
MHAOvlFilter::ShapeFun, 122
GCC_end
doasvm_feature_extraction_config, 432
GCC_start
doasvm_feature_extraction_config, 431
gcd
audiometerbackend, 83
dbasync_native, 86
MHAFilter, 108
generatemhaplugindoc.cpp, 1556
conv2latex, 1557
create_latex_doc, 1558
main, 1558
print_plugin_references, 1557
generator
audiometerbackend, 83
get
testplugin::config_parser_t, 1484
get_A
MHAFilter::gamma_flt_t, 895
get_a
MHAParser::base_t::replace_t, 1044
get_ac
latex_doc_t, 647
plug_t, 1350
get_accept_event
MHA_TCP::Server, 817
get_adaptation_ratio
adm_rtconfig_t, 278
get_address
mha_tcp::server_t, 821
get_all_input_ports
MHAIOJack::io_jack_t, 944
MHAIOJackdb::io_jack_t, 951
get_all_names_from_ac_space
ac2lsl::ac2lsl_t, 164
get_all_output_ports
MHAIOJack::io_jack_t, 944
MHAIOJackdb::io_jack_t, 951

© 2005-2021 HörTech gGmbH, Oldenburg

get_all_stream_names
lsl2ac::lsl2ac_t, 684
get_arg_type_and_dimension
ac_mul_t, 193, 194
get_attack_filter_state
dc::dc_t, 385
get_audiochannels
dc.cpp, 1543
get_available_space
mha_drifter_fifo_t< T >, 760
mha_fifo_lf_t< T >, 769
mha_fifo_t< T >, 780
get_b
MHAParser::base_t::replace_t, 1044
get_bands
gtfb_simd_cfg_t, 558
get_buf_address
ac2lsl::save_var_base_t, 170
ac2lsl::save_var_t< mha_complex_t >, 177
ac2lsl::save_var_t< T >, 173
get_buffer
mha_tcp::buffered_socket_t, 800
get_bw_hz
MHAOvlFilter::fscale_bw_t, 1015
get_c1
MHAFilter::o1flt_lowpass_t, 912
get_c_handle
MHAKernel::algo_comm_class_t, 990
get_categories
io_lib_t, 614
io_wrapper, 644
latex_doc_t, 646
plug_wrapper, 1351
plug_wrapperI, 1353
PluginLoader::mhapluginloader_t, 1373
get_cdata
MHASignal::matrix_t, 1236
get_cf_fftbin
MHAOvlFilter::fspacing_t, 1019
get_cf_hz
MHAFilter::thirdoctave_analyzer_t, 935
MHAOvlFilter::fspacing_t, 1020
get_cfin
MHAParser::mhapluginloader_t, 1093
get_cfout
MHAParser::mhapluginloader_t, 1093
get_channelconfig
MHAOvlFilter::overlap_save_filterbank_t, 1025
get_channels
gtfb_simd_cfg_t, 558
get_comm_var
MHASignal::matrix_t, 1230
get_configfile
PluginLoader::config_file_splitter_t, 1364
get_configname
PluginLoader::config_file_splitter_t, 1363
get_connected
io_asterisk_parser_t, 593
io_tcp_parser_t, 629
get_context
mha_tcp::server_t, 823
get_cpu_load
MHAJack::client_t, 980
get_current_profile
AuditoryProfile::parser_t, 326
get_decay_filter_state
dc::dc_t, 385
get_delay
mha_dblbuf_t< FIFO >, 750
get_delays_in
MHAIOJack::io_jack_t, 944
get_delays_out
MHAIOJack::io_jack_t, 944
get_des_fill_count
mha_drifter_fifo_t< T >, 760
get_documentation
io_lib_t, 614
io_wrapper, 644
plug_wrapper, 1351
plug_wrapperI, 1353
PluginLoader::mhapluginloader_t, 1373
get_ear
AuditoryProfile::parser_t::ear_t, 328
AuditoryProfile::profile_t, 331
get_ef_hz
MHAOvlFilter::fspacing_t, 1020
get_endpoint
mha_tcp::server_t, 821
get_entries
algo_comm_t, 285
MHAKernel::algo_comm_class_t, 992
MHAParser::keyword_list_t, 1074
get_error
algo_comm_t, 285
MHAKernel::algo_comm_class_t, 992
get_f_hz
MHAOvlFilter::fscale_t, 1017
get_fbpower
MHAOvlFilter::fftfb_t, 1008
get_fbpower_db

© 2005-2021 HörTech gGmbH, Oldenburg

MHAOvlFilter::fftfb_t, 1008
get_fd
MHA_TCP::Connection, 807
get_fftlen
MHAOvlFilter::fftfb_t, 1009
get_fifo_size
mha_dblbuf_t< FIFO >, 750
get_fill_count
mha_drifter_fifo_t< T >, 760
mha_fifo_lf_t< T >, 769
mha_fifo_t< T >, 780, 782
get_fmap
AuditoryProfile::parser_t::fmap_t, 330
get_fragsize
MHAJack::client_t, 978
get_frames
gtfb_simd_cfg_t, 558
get_frequencies
AuditoryProfile::fmap_t, 325
get_fun
MHAOvlFilter::scale_var_t, 1029
get_gain
DynComp::gaintable_t, 460, 461
get_gf
gtfb_simple_rt_t, 567
get_handle
plug_t, 1350
get_idx
level_matching::channel_pair, 650
get_index
MHAParser::keyword_list_t, 1074
MHASignal::matrix_t, 1235
get_inner_error
mha_dblbuf_t< FIFO >, 751
get_inner_size
mha_dblbuf_t< FIFO >, 750
get_input_channels
mha_dblbuf_t< FIFO >, 750
get_input_fifo_fill_count
mha_dblbuf_t< FIFO >, 751
get_input_fifo_space
mha_dblbuf_t< FIFO >, 751
get_input_signal_dimension
fw_t, 528
get_interface
MHA_TCP::Server, 817
get_iofun
DynComp::gaintable_t, 461
get_irs
MHAFilter::fftfilterbank_t, 885
get_last_name
MHAParser::mhapluginloader_t, 1093
get_last_output
MHAFilter::o1flt_lowpass_t, 912
get_latex_doc
latex_doc_t, 646
get_len_A
MHAFilter::filter_t, 891
get_len_B
MHAFilter::filter_t, 891
get_length
MHASignal::uint_vector_t, 1261
get_level
addsndfile::level_adapt_t, 256
audiometerbackend::level_adapt_t, 319
fader_wave::level_adapt_t, 493
get_level_in_db
dc::dc_t, 384
get_level_in_db_adjusted
dc::dc_t, 385
get_libname
PluginLoader::config_file_splitter_t, 1363
get_local_address
io_asterisk_parser_t, 591
io_tcp_parser_t, 628
get_local_port
io_asterisk_parser_t, 591
io_tcp_parser_t, 628
get_longmsg
MHA_Error, 766
get_ltass_gain_db
MHAOvlFilter::fftfb_t, 1008
get_main_category
latex_doc_t, 646
get_mapping
MHASignal::loop_wavefragment_t, 1223
get_max_fill_count
mha_fifo_t< T >, 781
get_min_fill_count
mha_drifter_fifo_t< T >, 761
get_mismatch
level_matching::channel_pair, 650
get_msg
MHA_Error, 766
get_my_input_ports
MHAJack::client_t, 979
get_my_output_ports
MHAJack::client_t, 979
get_name
MHAOvlFilter::scale_var_t, 1029
get_nbands
dc::dc_t, 384

© 2005-2021 HörTech gGmbH, Oldenburg

get_nch
dc::dc_t, 384
get_nelements
MHASignal::matrix_t, 1231
get_noise_model_func
rohBeam::rohBeam, 1403
get_nreals
MHASignal::matrix_t, 1235
get_num_accepted_connections
mha_tcp::server_t, 822
get_origname
PluginLoader::config_file_splitter_t, 1363
get_os_event
MHA_TCP::Timeout_Event, 833
MHA_TCP::Wakeup_Event, 838
get_outer_size
mha_dblbuf_t< FIFO >, 750
get_output_channels
mha_dblbuf_t< FIFO >, 751
get_output_fifo_fill_count
mha_dblbuf_t< FIFO >, 751
get_output_fifo_space
mha_dblbuf_t< FIFO >, 751
get_parser_tab
latex_doc_t, 647
get_parser_var
latex_doc_t, 647
get_parserstate
fw_t, 528
get_paths
pluginbrowser_t, 1357
get_peer_address
MHA_TCP::Connection, 807
get_peer_port
MHA_TCP::Connection, 807
get_physical_input_ports
MHAIOJack::io_jack_t, 944
MHAIOJackdb::io_jack_t, 951
get_physical_output_ports
MHAIOJack::io_jack_t, 944
MHAIOJackdb::io_jack_t, 951
get_plugins
pluginbrowser_t, 1358
get_port
MHA_TCP::Server, 817
mha_tcp::server_t, 821
get_port_capture_latency
MHAJack, 112
get_port_capture_latency_int
MHAJack, 112
get_port_playback_latency
MHAJack, 114
get_port_playback_latency_int
MHAJack, 114
get_ports
MHAJack::client_t, 979
get_precision
MHAParser, 126
get_process_spec
plug_t, 1350
get_process_wave
plug_t, 1349
get_rdata
MHASignal::matrix_t, 1235
get_read_event
MHA_TCP::Connection, 807
get_read_ptr
mha_fifo_t< T >, 782
get_resynthesis_gain
MHAFilter::gamma_flt_t, 895
get_rmslevel_filter_state
dc::dc_t, 385
get_server_port_open
io_asterisk_parser_t, 592
io_tcp_parser_t, 629
get_short_name
MHAJack::port_t, 987
get_signal
MHAPlugin_Split::domain_handler_t, 1167
get_size
MHASignal::waveform_t, 1275
get_srate
MHAJack::client_t, 978
get_time_correction
lsl2ac::save_var_t, 690
get_type
MHAParser::window_t, 1141
get_value
MHAParser::keyword_list_t, 1073
get_values
AuditoryProfile::fmap_t, 325
get_var
algo_comm_t, 283
MHAKernel::algo_comm_class_t, 991
testplugin::ac_parser_t, 1482
get_var_double
algo_comm_t, 285
MHAKernel::algo_comm_class_t, 992
get_var_float
algo_comm_t, 284
Communication between algorithms, 26

© 2005-2021 HörTech gGmbH, Oldenburg

MHAKernel::algo_comm_class_t, 992
get_var_int
algo_comm_t, 284
Communication between algorithms, 26
MHAKernel::algo_comm_class_t, 991
get_var_spectrum
Communication between algorithms, 25
get_var_vfloat
Communication between algorithms, 27
get_var_waveform
Communication between algorithms, 25
get_varname
plugins::hoertech::acrec::acwriter_t, 1385
get_vF
DynComp::gaintable_t, 461
get_vL
DynComp::gaintable_t, 461
get_weights
MHAFilter::complex_bandpass_t, 874
MHAFilter::gamma_flt_t, 895
get_window
MHAParser::window_t, 1140, 1141
get_window_data
windowselector_t, 1520
get_write_event
MHA_TCP::Connection, 807
get_write_ptr
mha_fifo_t< T >, 781
get_xlimits
MHATableLookup::xy_table_t, 1289
get_xruns
MHAJack::client_t, 978
get_xruns_reset
MHAJack::client_t, 979
get_zeropadding
wave2spec_t, 1500
getcipd
coherence, 84
getdata
MHASignal::uint_vector_t, 1262
getfullname
PluginLoader::mhapluginloader_t, 1372
getmodulename
dynamiclib_t, 449
Getmsg
mha_error.hh, 1594
getname
dynamiclib_t, 449
MHA_AC::ac2matrix_t, 724
getusername
MHA_AC::ac2matrix_t, 724
getvar
acmon::ac_monitor_t, 205
MHA_AC::ac2matrix_helper_t, 722
GF
MHAFilter::gamma_flt_t, 896
gf
gtfb_simple_rt_t, 569
gf_internals
gtfb_simple_t, 574
GITCOMMITHASH
mha_git_commit_hash.cpp, 1602
GLR
smooth_cepstrum::smooth_cepstrum_t, 1449
GLRDebug
noise_psd_estimator::noise_psd_estimator_t, 1317
GLRexp
noise_psd_estimator::noise_psd_estimator_t, 1318
smooth_cepstrum::smooth_cepstrum_t, 1449
Grad
gsc_adaptive_stage::gsc_adaptive_stage, 543
grad
gsc_adaptive_stage::gsc_adaptive_stage, 543
GREETING_TEXT
mhamain.cpp, 1692
groupdelay
delaysum_spec::delaysum_spec_if_t, 419
groupdelay_t
MHASignal::schroeder_t, 1245
gsc_adaptive_stage, 95
DELT, 96
gsc_adaptive_stage::gsc_adaptive_stage, 538
gsc_adaptive_stage.cpp, 1558
gsc_adaptive_stage.hh, 1558
gsc_adaptive_stage::gsc_adaptive_stage, 537
~gsc_adaptive_stage, 539
ac, 540
alp, 542
bufSize, 540
d, 543
desired_chan, 541
doCircularComp, 541
E, 543
e, 543

© 2005-2021 HörTech gGmbH, Oldenburg

E2, 543
e_out, 544
frac_old, 540
Grad, 543
grad, 543
gsc_adaptive_stage, 538
insert, 540
lenNewSamps, 540
lenOldSamps, 540
mha_fft, 541
mu, 541
nchan, 541
nfreq, 541
P, 544
process, 539
Psum, 544
useVAD, 542
vadName, 542
W, 542
X, 542
x, 542
Y, 542
y, 543
gsc_adaptive_stage::gsc_adaptive_stage_if, 545
~gsc_adaptive_stage_if, 546
alp, 549
doCircularComp, 548
gsc_adaptive_stage_if, 546
lenOldSamps, 548
mu, 548
on_model_param_valuechanged, 548
patchbay, 548
prepare, 547
process, 547
release, 547
update_cfg, 548
useVAD, 549
vadName, 549
gsc_adaptive_stage_if
gsc_adaptive_stage::gsc_adaptive_stage_if, 546
gsc_adaptive_stage_if.cpp, 1559
gsc_adaptive_stage_if.hh, 1559
gt
dc::dc_t, 385
gtdata
dc::dc_vars_t, 388
gtfb_analyzer, 96
gtfb_analyzer.cpp, 1559
filter_complex, 1560
filter_real, 1560
gtfb_analyzer::gtfb_analyzer_cfg_t, 549
~gtfb_analyzer_cfg_t, 551
bands, 551
channels, 551
coeff, 552
cvalue, 552
frames, 551
gtfb_analyzer_cfg_t, 550
norm_phase, 552
order, 552
s_out, 552
state, 553
states, 551
gtfb_analyzer::gtfb_analyzer_t, 553
coeff, 556
gtfb_analyzer_t, 555
norm_phase, 556
order, 556
patchbay, 556
prepare, 555
prepared, 556
process, 555
release, 555
update_cfg, 555
gtfb_analyzer_cfg_t
gtfb_analyzer::gtfb_analyzer_cfg_t, 550
gtfb_analyzer_t
gtfb_analyzer::gtfb_analyzer_t, 555
gtfb_simd.cpp, 1561
add4f, 1562
check_alignment, 1563
filter_simd, 1566
filter_sisd_complex, 1563
filter_sisd_real, 1564
mul4f, 1563
MXCSR_DAZ, 1563
MXCSR_FTZ, 1563
sub4f, 1562
gtfb_simd_cfg_t, 556
~gtfb_simd_cfg_t, 558
bands, 559
bandsXchannels, 559
channels, 559
get_bands, 558
get_channels, 558
get_frames, 558
gtfb_simd_cfg_t, 557, 558
icoefficients, 560
iinputs, 560
istates, 560

© 2005-2021 HörTech gGmbH, Oldenburg

large_array, 561
norm_phase, 560
operator=, 559
order, 559
process, 559
rcoefficients, 560
rinputs, 560
rstates, 560
s_out, 561
sout_buf, 561
gtfb_simd_t, 562
coeff, 564
gtfb_simd_t, 563
norm_phase, 564
order, 564
patchbay, 563
prepare, 563
prepared, 563
process, 563
update_cfg, 563
gtfb_simple_bridge.cpp, 1567
gtfb_simple_rt_t, 564
_ac, 569
_order, 567
_pre_stages, 568
ac_resynthesis_gain, 569
acbw, 568
accf, 568
cLTASS, 569
duplicate_vector, 567
element_gain_name_, 569
get_gf, 567
gf, 569
gtfb_simple_rt_t, 565
imag, 568
input, 568
nbands, 568
output, 568
post_plugin, 566
pre_plugin, 566
gtfb_simple_t, 570
cLTASS, 574
desired_delay, 573
element_gain_name, 573
fscale, 573
gf_internals, 574
gtfb_simple_t, 571
name_, 574
order, 573
plug, 573
prepare, 572
prestages, 573
process, 572
release, 572
resynthesis_gain, 574
setlock, 572
gtmin
dc::dc_vars_t, 388
gtstep
dc::dc_vars_t, 388

h
dynamiclib_t, 450
MHASignal::hilbert_t, 1219
H_ERRNO
MHA_TCP, 102
hamming
MHAWindow, 155
hamming_t
MHAWindow::hamming_t, 1297
handle
algo_comm_t, 280
handler
osc_variable_t, 1328, 1329
hann
hann.cpp, 1568
hann.h, 1568
MHAOvlFilter::ShapeFun, 121
hann.cpp, 1567
hann, 1568
hannf, 1568
PI, 1567
hann.h, 1568
hann, 1568
hannf, 1568
hann1
plingploing::plingploing_t, 1348
hann2
plingploing::plingploing_t, 1348
hannf
hann.cpp, 1568
hann.h, 1568
hanning
MHAWindow, 155
hanning_ramps_t, 574
~hanning_ramps_t, 575
hanning_ramps_t, 575
len_a, 575
len_b, 575
operator(), 575
ramp_a, 575
ramp_b, 576
hanning_t

© 2005-2021 HörTech gGmbH, Oldenburg

MHAWindow::hanning_t, 1298
hardlimit
softclipper_t, 1463
softclipper_variables_t, 1465
has_been_modified
dc_simple::dc_if_t, 395
has_entry
MHAParser::parser_t, 1108
has_inner_error
analysepath_t, 308
has_key
MHAKernel::comm_var_map_t, 994
has_parser
io_wrapper, 644
plug_wrapper, 1351
plug_wrapperI, 1353
PluginLoader::mhapluginloader_t, 1371
has_process
io_wrapper, 644
plug_wrapper, 1352
plug_wrapperI, 1354
PluginLoader::mhapluginloader_t, 1371
head_model_sphere_radius_cm
rohBeam::rohBeam, 1404
header
io_asterisk_sound_t, 599
io_tcp_sound_t, 636
headModel
rohBeam::rohConfig, 1409
help
MHAParser::base_t, 1043
HELP_TEXT
mhamain.cpp, 1692
hhCorrXpXp
rohBeam::rohConfig, 1411
hifftwin
doasvm_feature_extraction_config, 432
hifftwin_sum
doasvm_feature_extraction_config, 432
high_side_flat
MHAOvlFilter::band_descriptor_t, 1002
hilbert
MHASignal::hilbert_fftw_t, 1217
hilbert_fftw_t
MHASignal::hilbert_fftw_t, 1216
hilbert_shifter_t
fshift_hilbert::hilbert_shifter_t, 519
hilbert_t
MHASignal::hilbert_t, 1219
HINSTANCE
mha_plugin.hh, 1624
HL
rmslevel, 159
host
ac2osc_t, 183
osc2ac_t, 1323
host_port_to_sock_addr
mha_tcp.cpp, 1646
hostApi
MHAIOPortAudio::device_info_t, 957
Hs
MHAFilter::fftfilterbank_t, 886
HSTRERROR
MHA_TCP, 102
HTL
AuditoryProfile::parser_t::ear_t, 328
AuditoryProfile::profile_t::ear_t, 333
hton
io_asterisk_sound_t, 599
io_tcp_sound_t, 636
hw
MHAFilter::fftfilterbank_t, 886
hwin
doasvm_feature_extraction_config, 432
hz2bark
MHAOvlFilter::FreqScaleFun, 118
hz2bark_analytic
MHAOvlFilter::FreqScaleFun, 119
hz2bark_t
MHAOvlFilter::barkscale::hz2bark_t, 1004
hz2erb
MHAOvlFilter::FreqScaleFun, 119
hz2erb_glasberg1990
MHAOvlFilter::FreqScaleFun, 119
hz2hz
MHAOvlFilter::FreqScaleFun, 118
speechnoise.cpp, 1706
hz2khz
MHAOvlFilter::FreqScaleFun, 118
hz2log
MHAOvlFilter::FreqScaleFun, 119
hz2octave
MHAOvlFilter::FreqScaleFun, 118
hz2third_octave
MHAOvlFilter::FreqScaleFun, 118
hz2unit
MHAOvlFilter::scale_var_t, 1029

i
io_tcp_sound_t::float_union, 638
icoefficients
gtfb_simd_cfg_t, 560
id

© 2005-2021 HörTech gGmbH, Oldenburg

equalize::freqgains_t, 466
mha_channel_info_t, 745
id_str
MHAParser::base_t, 1043
id_string
MHAParser::parser_t, 1108
identity
MHASignal::schroeder_t, 1247
identity.cpp, 1568
identity_t, 576
identity_t, 577
prepare, 577
process, 577
release, 577
idstr
mha_channel_info_t, 745
idx
level_matching::channel_pair, 650
if_t
plingploing::if_t, 1341
testplugin::if_t, 1487
windnoise::if_t, 1515
iface
MHA_TCP::Server, 818
ifft
doasvm_feature_extraction_config, 432
ifftshift
ifftshift.cpp, 1569
ifftshift.h, 1569
ifftshift.cpp, 1569
ifftshift, 1569
ifftshift.h, 1569
ifftshift, 1569
Ignore
lsl2ac, 97
ignore
MHA_TCP::Event_Watcher, 813
ignored_by
MHA_TCP::Wakeup_Event, 837
iinputs
gtfb_simd_cfg_t, 560
iir_filter_state_t
MHAFilter::iir_filter_state_t, 897
iir_filter_t
MHAFilter::iir_filter_t, 899
iir_ord1_real_t
MHAFilter::iir_ord1_real_t, 903
iirfilter.cpp, 1569
iirfilter_t, 578
iirfilter_t, 578
prepare_, 579
process, 579
release_, 579
ilen
addsndfile::level_adapt_t, 256
audiometerbackend::level_adapt_t, 319
fader_wave::level_adapt_t, 493
im
mha_complex_t, 746
imag
acsave::mat4head_t, 225
fftfilterbank::fftfb_plug_t, 509
gtfb_simple_rt_t, 568
MHASignal::matrix_t, 1232–1234
imagfb
MHAOvlFilter::overlap_save_filterbank_analytic_t, 1023
impulse_response
MHAFilter::polyphase_resampling_t, 927
MHAFilter::transfer_function_t, 940
in_buf
wave2spec_t, 1502
in_cfg
rohBeam::rohConfig, 1409
smooth_cepstrum::smooth_params, 1451
in_spec
doasvm_feature_extraction_config, 433
shadowfilter_end::cfg_t, 1430
in_spec_copy
shadowfilter_begin::cfg_t, 1426
inbuf
MHA_TCP::Connection, 810
inch
mconv::MConv, 720
MHAJack::client_t, 983
increase_condition
mha_fifo_posix_threads_t, 776
increment
mha_fifo_posix_threads_t, 775
mha_fifo_thread_platform_t, 787
index
MHAParser::keyword_list_t, 1074
index_t
MHAFilter::partitioned_convolution_t::index_t, 922
info
ac2lsl::save_var_base_t, 170
ac2lsl::save_var_t< mha_complex_t >, 177
ac2lsl::save_var_t< T >, 174
lsl2ac::save_var_t, 689
init_dynamic

© 2005-2021 HörTech gGmbH, Oldenburg

rohBeam::rohConfig, 1408
Init_mha_ruby
mha_ruby.cpp, 1629
init_peer_data
MHA_TCP::Connection, 805
initialize
MHA_TCP::Server, 817
inner2outer_resampling
MHAPlugin_Resampling::resampling_t, 1162
inner_ac_copy
analysepath_t, 308
inner_error
analysepath_t, 308
mha_dblbuf_t< FIFO >, 754
inner_fragsize
MHAPlugin_Resampling::resampling_t, 1161
inner_in
MHASignal::doublebuffer_t, 1210
inner_input
analysepath_t, 307
dbasync_native::dbasync_t, 378
inner_out
MHASignal::doublebuffer_t, 1210
inner_out_domain
analysepath_t, 308
inner_process
db_t, 372
MHASignal::doublebuffer_t, 1209
inner_process_wave2spec
analysepath_t, 307
inner_process_wave2wave
analysepath_t, 307
inner_signal
MHAPlugin_Resampling::resampling_t, 1162
inner_size
mha_dblbuf_t< FIFO >, 753
inner_srate
MHAPlugin_Resampling::resampling_t, 1161
input
ac_proc::interface_t, 198
gtfb_simple_rt_t, 568
io_parser_t, 620
mha_dblbuf_t< FIFO >, 752
MHAJack::port_t, 985
MHASignal::loop_wavefragment_t, 1222
input_cfg
MHAPlugin::plugin_t< runtime_cfg_t >, 1155
input_cfg_
MHAPlugin::plugin_t< runtime_cfg_t >, 1156
input_channels
adm_if_t, 274
mha_dblbuf_t< FIFO >, 754
input_domain
PluginLoader::mhapluginloader_t, 1371
input_fifo
mha_dblbuf_t< FIFO >, 754
input_level
dc::dc_vars_t, 390
input_portnames
MHAJack::client_t, 984
input_signal_spec
MHAFilter::partitioned_convolution_t, 920
input_signal_wave
MHAFilter::partitioned_convolution_t, 920
input_spec
testplugin::signal_parser_t, 1490
input_to_process
analysepath_t, 309
input_wave
testplugin::signal_parser_t, 1490
inputchannels
MHAFilter::fftfilterbank_t, 886
inputPow
noise_psd_estimator::noise_psd_estimator_t, 1316
inputSpec
noise_psd_estimator::noise_psd_estimator_t, 1317
insert
acPooling_wave_config, 216
acSteer_config, 232
adaptive_feedback_canceller_config, 246
coherence::cohflt_t, 350
fftfilterbank::fftfb_plug_t, 508
gsc_adaptive_stage::gsc_adaptive_stage, 540
lpc_config, 677
MHA_AC::ac2matrix_t, 725
MHA_AC::acspace2matrix_t, 728
MHA_AC::double_t, 730
MHA_AC::float_t, 732
MHA_AC::int_t, 734
MHA_AC::spectrum_t, 737
MHA_AC::stat_t, 738
MHA_AC::waveform_t, 741
MHAOvlFilter::fftfb_ac_info_t, 1005

© 2005-2021 HörTech gGmbH, Oldenburg

multibandcompressor::fftfb_plug_t, 1303
noise_psd_estimator::noise_psd_estimator_t, 1316
rmslevel::rmslevel_t, 1396
rt_nlms_t, 1419
windnoise::if_t, 1516
insert_config_vars
matlab_wrapper::matlab_wrapper_t, 701
insert_item
MHAParser::parser_t, 1104
insert_items
windowselector_t, 1520
insert_member
mha_parser.hh, 1622
insert_monitors
matlab_wrapper::matlab_wrapper_t, 701
INSERT_PATCH
acConcat_wave.cpp, 1525
acPooling_wave.cpp, 1526
acSteer.cpp, 1529
acTransform_wave.cpp, 1530
adaptive_feedback_canceller.cpp, 1530
doasvm_classification.cpp, 1548
doasvm_feature_extraction.cpp, 1548
level_matching.cpp, 1570
lpc.cpp, 1571
lpc_bl_predictor.cpp, 1572
lpc_burg-lattice.cpp, 1573
smooth_cepstrum.cpp, 1703
steerbf.cpp, 1710
INSERT_VAR
smooth_cepstrum.cpp, 1703
insert_var
algo_comm_t, 281
MHAKernel::algo_comm_class_t, 990
testplugin::ac_parser_t, 1482
insert_var_double
algo_comm_t, 282
MHAKernel::algo_comm_class_t, 991
insert_var_float
algo_comm_t, 281
MHAKernel::algo_comm_class_t, 990
insert_var_int
algo_comm_t, 281
MHAKernel::algo_comm_class_t, 990
insert_var_vfloat
MHAKernel::algo_comm_class_t, 990
insert_variable
osc_server_t, 1325
insert_vars
lsl2ac::save_var_t, 690
inspect
MHAFilter::complex_bandpass_t, 875
MHAFilter::gamma_flt_t, 895
MHASignal::delay_t, 1205
inst_name
fw_t, 530
int_data
testplugin::ac_parser_t, 1482
int_mon_t
MHAParser::int_mon_t, 1067
int_t
MHA_AC::int_t, 734
MHAParser::int_t, 1070
integrate
Vector and matrix processing toolbox, 43
intensity
mha_signal.cpp, 1632
interface_t
ac_proc::interface_t, 197
delay::interface_t, 411
fftfilter::interface_t, 502
multibandcompressor::interface_t, 1305
route::interface_t, 1413
interleaved
combc_if_t, 357
interleaved_
combc_t, 359
intermic_distance_cm
rohBeam::rohBeam, 1404
intern_level
MHASignal::loop_wavefragment_t, 1225
internal_fir
MHAFilter::smoothspec_t, 933
internal_start
MHAJack::client_t, 980
internal_stop
MHAJack::client_t, 980
interp
MHATableLookup::linear_table_t, 1280
MHATableLookup::table_t, 1284
MHATableLookup::xy_table_t, 1287
interp1
DynComp, 90
interp2
DynComp, 91
inv_scale
MHAOvlFilter::FreqScaleFun, 120
INVALID_SOCKET
mha_tcp.cpp, 1645
INVALID_THREAD_PRIORITY
dbasync_native, 86

© 2005-2021 HörTech gGmbH, Oldenburg

MHAPlugin_Split, 136
invalidate_window_data
windowselector_t, 1521
invert
coherence::vars_t, 354
invgain
alsa_t< T >, 294
inwave
lpc_config, 678
io
MHAJack, 112
MHAJack::client_avg_t, 968
MHAJack::client_noncont_t, 972
io_alsa_t, 579
alsa_start_counter, 584
b_process, 582
dev_in, 583
dev_out, 583
format, 584
fw_fragsize, 582
fw_samplerate, 582
io_alsa_t, 581
p_in, 584
p_out, 584
patchbay, 584
pcmlink, 584
prepare, 581, 582
priority, 584
proc_event, 583
proc_handle, 583
proc_thread, 583
process, 582
release, 581
start, 581
start_event, 583
start_handle, 583
stop, 582
stop_event, 583
stop_handle, 583
thread_start, 582
io_asterisk_fwcb_t, 584
~io_asterisk_fwcb_t, 586
io_asterisk_fwcb_t, 585
io_err, 589
proc_err, 588
proc_event, 588
proc_handle, 588
process, 586
set_errnos, 587
start, 586
start_event, 588
start_handle, 588
stop, 587
stop_event, 588
stop_handle, 588
io_asterisk_parser_t, 589
~io_asterisk_parser_t, 591
connected, 595
debug, 595
debug_file, 596
debug_filename, 596
get_connected, 593
get_local_address, 591
get_local_port, 591
get_server_port_open, 592
io_asterisk_parser_t, 591
local_address, 595
local_port, 595
peer_address, 596
peer_port, 596
server_port_open, 595
set_connected, 593
set_local_port, 592
set_new_peer, 594
set_server_port_open, 592
io_asterisk_sound_t, 596
~io_asterisk_sound_t, 598
chunkbytes_in, 598
fragsize, 599
header, 599
hton, 599
io_asterisk_sound_t, 597
ntoh, 599
num_inchannels, 600
num_outchannels, 600
output_data, 600
prepare, 598
release, 598
s_in, 600
samplerate, 599
io_asterisk_t, 600
~io_asterisk_t, 602
accept_loop, 603
connection_loop, 603
fwcb, 604
io_asterisk_t, 601
notify_release, 604
notify_start, 604
notify_stop, 604
parse, 603
parser, 603
prepare, 602

© 2005-2021 HörTech gGmbH, Oldenburg

release, 602
server, 604
sound, 603
start, 602
stop, 602
thread, 604
io_context
mha_tcp::server_t, 824
io_err
io_asterisk_fwcb_t, 589
io_tcp_fwcb_t, 625
io_error
fw_t, 531
IO_ERROR_JACK
mhajack.h, 1691
IO_ERROR_MHAJACKLIB
mhajack.h, 1691
io_file_t, 605
~io_file_t, 607
b_prepared, 610
filename_input, 609
filename_output, 609
fragsize, 608
io_file_t, 606
length, 610
nchannels_file_in, 608
nchannels_in, 608
nchannels_out, 608
output_sample_format, 609
prepare, 607
proc_event, 609
proc_handle, 609
release, 607
s_file_in, 610
s_in, 610
s_out, 610
samplerate, 608
setlock, 608
sf_in, 610
sf_out, 611
sfinf_in, 611
sfinf_out, 611
start, 607
start_event, 609
start_handle, 609
startsample, 610
stop, 607
stop_event, 609
stop_handle, 609
stopped, 607
strict_channel_match, 610
strict_srate_match, 610
total_read, 611
io_jack_t
MHAIOJack::io_jack_t, 943
MHAIOJackdb::io_jack_t, 950
io_lib
fw_t, 531
io_lib_t, 611
~io_lib_t, 613
get_categories, 614
get_documentation, 614
io_lib_t, 613
IODestroy_cb, 616
IOInit_cb, 615
IOPrepare_cb, 615
IORelease_cb, 615
IOSetVar_cb, 615
IOStart_cb, 615
IOStop_cb, 615
IOStrError_cb, 615
lib_data, 615
lib_err, 614
lib_handle, 615
lib_str_error, 614
plugin_categories, 616
plugin_documentation, 616
prepare, 613
release, 614
start, 614
stop, 614
test_error, 614
io_name
fw_t, 529
io_parser_t, 616
~io_parser_t, 618
b_fw_started, 620
b_prepared, 621
b_starting, 621
b_stopped, 621
fragsize, 619
input, 620
io_parser_t, 618
nchannels_in, 619
nchannels_out, 619
output, 620
patchbay, 621
prepare, 618
proc_event, 619
proc_handle, 619
process_frame, 619
release, 618

© 2005-2021 HörTech gGmbH, Oldenburg

s_in, 620
s_out, 620
start, 618
start_event, 620
start_handle, 620
started, 619
stop, 618
stop_event, 620
stop_handle, 620
stopped, 619
io_portaudio_t
MHAIOPortAudio::io_portaudio_t, 960
io_tcp_fwcb_t, 621
~io_tcp_fwcb_t, 622
io_err, 625
io_tcp_fwcb_t, 622
proc_err, 625
proc_event, 624
proc_handle, 625
process, 623
set_errnos, 623
start, 623
start_event, 624
start_handle, 625
stop, 624
stop_event, 624
stop_handle, 625
io_tcp_parser_t, 626
~io_tcp_parser_t, 627
connected, 632
debug, 631
debug_file, 632
debug_filename, 632
get_connected, 629
get_local_address, 628
get_local_port, 628
get_server_port_open, 629
io_tcp_parser_t, 627
local_address, 631
local_port, 631
peer_address, 632
peer_port, 632
server_port_open, 632
set_connected, 630
set_local_port, 628
set_new_peer, 630
set_server_port_open, 629
io_tcp_sound_t, 633
~io_tcp_sound_t, 634
check_sound_data_type, 635
chunkbytes_in, 635
fragsize, 637
header, 636
hton, 636
io_tcp_sound_t, 634
ntoh, 636
num_inchannels, 637
num_outchannels, 637
prepare, 635
release, 635
s_in, 637
samplerate, 637
io_tcp_sound_t::float_union, 638
c, 638
f, 638
i, 638
io_tcp_t, 638
~io_tcp_t, 640
accept_loop, 641
connection_loop, 641
fwcb, 642
io_tcp_t, 639
notify_release, 642
notify_start, 642
notify_stop, 642
parse, 641
parser, 641
prepare, 640
release, 640
server, 642
sound, 641
start, 640
stop, 640
thread, 642
io_wrapper, 643
~io_wrapper, 643
get_categories, 644
get_documentation, 644
has_parser, 644
has_process, 644
io_wrapper, 643
parse, 644
iob
MHAJack::port_t, 988
IODestroy
MHAIOalsa.cpp, 1654, 1656
MHAIOAsterisk.cpp, 1659, 1661
MHAIOFile.cpp, 1664, 1665
MHAIOJack.cpp, 1668, 1669
MHAIOJackdb.cpp, 1672, 1673
MHAIOParser.cpp, 1676, 1677
MHAIOPortAudio.cpp, 1680, 1682

© 2005-2021 HörTech gGmbH, Oldenburg

MHAIOTCP.cpp, 1686, 1688
IODestroy_cb
io_lib_t, 616
IODestroy_t
mha_io_ifc.h, 1605
IOInit
MHAIOalsa.cpp, 1653, 1654
MHAIOAsterisk.cpp, 1658, 1660
MHAIOFile.cpp, 1663, 1664
MHAIOJack.cpp, 1667, 1668
MHAIOJackdb.cpp, 1671, 1672
MHAIOParser.cpp, 1675, 1676
MHAIOPortAudio.cpp, 1679, 1681
MHAIOTCP.cpp, 1685, 1687
IOInit_cb
io_lib_t, 615
IOInit_t
mha_io_ifc.h, 1605
IOPrepare
MHAIOalsa.cpp, 1653, 1655
MHAIOAsterisk.cpp, 1659, 1660
MHAIOFile.cpp, 1663, 1664
MHAIOJack.cpp, 1667, 1668
MHAIOJackdb.cpp, 1671, 1672
MHAIOParser.cpp, 1675, 1676
MHAIOPortAudio.cpp, 1680, 1681
MHAIOTCP.cpp, 1685, 1687
IOPrepare_cb
io_lib_t, 615
IOPrepare_t
mha_io_ifc.h, 1605
IOProcessEvent_inner
MHAIOJackdb::io_jack_t, 950
IOProcessEvent_t
mha_io_ifc.h, 1604
IORelease
MHAIOalsa.cpp, 1653, 1655
MHAIOAsterisk.cpp, 1659, 1660
MHAIOFile.cpp, 1663, 1665
MHAIOJack.cpp, 1668, 1669
MHAIOJackdb.cpp, 1672, 1673
MHAIOParser.cpp, 1676, 1677
MHAIOPortAudio.cpp, 1680, 1681
MHAIOTCP.cpp, 1686, 1687
IORelease_cb
io_lib_t, 615
IORelease_t
mha_io_ifc.h, 1605
IOSetVar
MHAIOalsa.cpp, 1654, 1655
MHAIOAsterisk.cpp, 1659, 1661
MHAIOFile.cpp, 1663, 1665
MHAIOJack.cpp, 1668, 1669
MHAIOJackdb.cpp, 1672, 1673
MHAIOParser.cpp, 1676, 1677
MHAIOPortAudio.cpp, 1680, 1682
MHAIOTCP.cpp, 1686, 1687
IOSetVar_cb
io_lib_t, 615
IOSetVar_t
mha_io_ifc.h, 1605
IOStart
MHAIOalsa.cpp, 1653, 1655
MHAIOAsterisk.cpp, 1659, 1660
MHAIOFile.cpp, 1663, 1664
MHAIOJack.cpp, 1667, 1669
MHAIOJackdb.cpp, 1671, 1673
MHAIOParser.cpp, 1675, 1677
MHAIOPortAudio.cpp, 1680, 1681
MHAIOTCP.cpp, 1685, 1687
IOStart_cb
io_lib_t, 615
IOStart_t
mha_io_ifc.h, 1605
IOStartedEvent_t
mha_io_ifc.h, 1604
IOStop
MHAIOalsa.cpp, 1653, 1655
MHAIOAsterisk.cpp, 1659, 1660
MHAIOFile.cpp, 1663, 1665
MHAIOJack.cpp, 1667, 1669
MHAIOJackdb.cpp, 1671, 1673
MHAIOParser.cpp, 1675, 1677
MHAIOPortAudio.cpp, 1680, 1681
MHAIOTCP.cpp, 1686, 1687
IOStop_cb
io_lib_t, 615
IOStop_t
mha_io_ifc.h, 1605
IOStoppedEvent
MHAJack::client_avg_t, 969
MHAJack::client_noncont_t, 973
IOStoppedEvent_t
mha_io_ifc.h, 1604
IOStrError
MHAIOalsa.cpp, 1654, 1655
MHAIOAsterisk.cpp, 1659, 1661
MHAIOFile.cpp, 1664, 1665
MHAIOJack.cpp, 1668, 1669
MHAIOJackdb.cpp, 1672, 1673
MHAIOParser.cpp, 1676, 1677
MHAIOPortAudio.cpp, 1680, 1682

© 2005-2021 HörTech gGmbH, Oldenburg

MHAIOTCP.cpp, 1686, 1688
IOStrError_cb
io_lib_t, 615
IOStrError_t
mha_io_ifc.h, 1605
irs
fftfilter::interface_t, 503
mconv::MConv, 720
irs_length
fftfilter, 93
irs_validator
fftfilter, 93
irslen
fftfilter::fftfilter_t, 500
fshift_hilbert::frequency_translator_t, 518
irslen_inner2outer
MHAPlugin_Resampling::resampling_if_t, 1159
irslen_outer2inner
MHAPlugin_Resampling::resampling_if_t, 1159
irswnd
MHAOvlFilter::overlap_save_filterbank_t::vars_t, 1027
smoothgains_bridge::overlapadd_if_t, 1455
is_complex
MHA_AC::ac2matrix_helper_t, 722
mha_audio_descriptor_t, 742
plugins::hoertech::acrec::acwriter_t, 1387
is_denormal
MHAUtils, 151, 152
is_first_run
ac2lsl::ac2lsl_t, 166
ac2osc_t, 185
is_multiple_of
MHAUtils, 151
is_multiple_of_by_power_of_two
MHAUtils, 151
is_num_channels_known
plugins::hoertech::acrec::acwriter_t, 1387
is_playback_active
MHASignal::loop_wavefragment_t, 1224
is_power_of_two
MHAUtils, 151
is_prepared
adm_if_t, 273
double2acvar::double2acvar_t, 436
MHAJack::client_t, 980
MHAPlugin::plugin_t< runtime_cfg_t >, 1155
PluginLoader::mhapluginloader_t, 1373
is_prepared_
MHAPlugin::plugin_t< runtime_cfg_t >, 1156
is_running
osc_server_t, 1326
is_same_size
MHASignal::matrix_t, 1231
is_var
algo_comm_t, 283
MHAKernel::algo_comm_class_t, 991
iscomplex
MHASignal::matrix_t, 1231
isempty
AuditoryProfile::fmap_t, 325
gaintable.cpp, 1555
MHAFilter::transfer_function_t, 939
istates
gtfb_simd_cfg_t, 560
isval
MHAParser::kw_t, 1077
iter
adaptive_feedback_canceller_config, 247
iterate_lnn
audiometerbackend::lnn3rdoct_t, 321
iterator
MHA_TCP::Event_Watcher, 812

j0
rohBeam, 159
jack_error_handler
mhajack.cpp, 1688
jack_proc_cb
MHAJack::client_t, 981
jack_xrun_cb
MHAJack::client_t, 981
jc
MHAJack::client_t, 983
MHAJack::port_t, 988
jstate_prev
MHAJack::client_t, 983

k_inner
MHASignal::doublebuffer_t, 1210
k_outer
MHASignal::doublebuffer_t, 1211
kappa
lpc_burglattice_config, 675
kappa_block
lpc_burglattice_config, 675
kappa_const

© 2005-2021 HörTech gGmbH, Oldenburg

smooth_cepstrum::smooth_cepstrum_if_t, 1441
smooth_cepstrum::smooth_params, 1452
keyword_list_t
MHAParser::keyword_list_t, 1073
kick_condition
MHAPlugin_Split::posix_threads_t, 1174
kick_thread
MHAPlugin_Split::dummy_threads_t, 1170
MHAPlugin_Split::posix_threads_t, 1173
MHAPlugin_Split::thread_platform_t, 1190
kicked
MHAPlugin_Split::posix_threads_t, 1175
km
lpc_bl_predictor_config, 670
kmax
fshift::fshift_config_t, 511
fshift_hilbert::hilbert_shifter_t, 522
kmin
fshift::fshift_config_t, 511
fshift_hilbert::hilbert_shifter_t, 521
kth_smallest
MHASignal, 144
kw_index2type
transducers.cpp, 1712
kw_t
MHAParser::kw_t, 1076, 1077

L
AuditoryProfile::parser_t, 327
AuditoryProfile::profile_t, 332
l_new
addsndfile::level_adapt_t, 256
audiometerbackend::level_adapt_t, 320
fader_wave::level_adapt_t, 493
l_old
addsndfile::level_adapt_t, 257
audiometerbackend::level_adapt_t, 320
fader_wave::level_adapt_t, 494
lambda
lpc_burglattice, 673
lpc_burglattice_config, 675
lambda_ceps
smooth_cepstrum::smooth_cepstrum_t, 1448
lambda_ceps_prev
smooth_cepstrum::smooth_cepstrum_t, 1448
lambda_ml_ceps
smooth_cepstrum::smooth_cepstrum_t, 1447
lambda_ml_full
smooth_cepstrum::smooth_cepstrum_t, 1447
lambda_ml_smooth
smooth_cepstrum::smooth_cepstrum_t, 1447
lambda_smoothing_power
nlms_t, 1311
lambda_spec
smooth_cepstrum::smooth_cepstrum_t, 1448
lambda_thresh
smooth_cepstrum::smooth_cepstrum_if_t, 1441
smooth_cepstrum::smooth_params, 1451
large_array
gtfb_simd_cfg_t, 561
last_complex_bin
MHASignal::subsample_delay_t, 1258
last_errormsg
MHAParser::parser_t, 1108
last_jack_err
mhajack.cpp, 1689
last_jack_err_msg
mhajack.cpp, 1689
mhajack.h, 1691
last_name
MHAParser::mhapluginloader_t, 1094
latex_doc_t, 645
ac, 647
get_ac, 647
get_categories, 646
get_latex_doc, 646
get_main_category, 646
get_parser_tab, 647
get_parser_var, 647
latex_doc_t, 645
latex_plugname, 647
loader, 648
parsername, 647
plugin_macro, 648
plugname, 647
strdom, 646
latex_plugname
latex_doc_t, 647
len
MHA_AC::acspace2matrix_t, 729
MHAFilter::filter_t, 892
MHATableLookup::linear_table_t, 1282

© 2005-2021 HörTech gGmbH, Oldenburg

plingploing::plingploing_t, 1346
len_A
MHAFilter::filter_t, 892
len_a
hanning_ramps_t, 575
len_B
MHAFilter::filter_t, 892
len_b
hanning_ramps_t, 575
length
io_file_t, 610
MHASignal::uint_vector_t, 1262
lenNewSamps
gsc_adaptive_stage::gsc_adaptive_stage, 540
lenOldSamps
gsc_adaptive_stage::gsc_adaptive_stage, 540
gsc_adaptive_stage::gsc_adaptive_stage_if, 548
lev
noise_t, 1320
sine_t, 1437
LEVEL
dc_simple, 87
level
addsndfile::addsndfile_if_t, 253
audiometerbackend::audiometer_if_t, 317
plingploing::if_t, 1342
plingploing::plingploing_t, 1348
level_adapt_t
addsndfile::level_adapt_t, 255
audiometerbackend::level_adapt_t, 319
fader_wave::level_adapt_t, 492
level_adaptor
addsndfile, 80
audiometerbackend, 83
fader_wave, 92
level_in_db
dc::dc_t, 386
level_in_db_adjusted
dc::dc_t, 386
level_matching, 96
level_matching.cpp, 1569
INSERT_PATCH, 1570
PATCH_VAR, 1570
level_matching.hh, 1570
level_matching::channel_pair, 648
channel_pair, 648, 649
get_idx, 650
get_mismatch, 650
idx, 650
mismatch, 651
update_mismatch, 649, 650
level_matching::level_matching_config_t, 651
~level_matching_config_t, 652
fftlen, 653
level_matching_config_t, 652
lp, 653
pairings, 653
process, 652, 653
range, 653
tmp, 653
level_matching::level_matching_t, 654
~level_matching_t, 655
channels, 657
level_matching_t, 655
lp_level_tau, 657
lp_signal_fpass, 657
lp_signal_fstop, 657
lp_signal_order, 657
patchbay, 657
prepare, 656
process, 655, 656
range, 657
release, 656
update_cfg, 656
level_matching_config_t
level_matching::level_matching_config_t, 652
level_matching_t
level_matching::level_matching_t, 655
level_mode_t
MHASignal::loop_wavefragment_t, 1221
level_mon
droptect_t, 443
level_smoother_t
dc_simple::level_smoother_t, 408
level_spec
dc_simple::level_smoother_t, 410
level_wave
dc_simple::level_smoother_t, 410
levelmeter.cpp, 1570
PASCALE, 1570
levelmeter_t, 658
levelmeter_t, 659
mode, 660
patchbay, 660
peak, 660
prepare, 659
process, 659
query_peak, 660

© 2005-2021 HörTech gGmbH, Oldenburg

query_rms, 659
rms, 660
tau, 660
update_tau, 659
levelmode
addsndfile::addsndfile_if_t, 253
Levinson2
lpc.cpp, 1571
lib_data
io_lib_t, 615
PluginLoader::mhapluginloader_t, 1374
lib_err
io_lib_t, 614
PluginLoader::mhapluginloader_t, 1374
lib_handle
io_lib_t, 615
PluginLoader::mhapluginloader_t, 1374
lib_str_error
io_lib_t, 614
libdata
analysepath_t, 308
MHAParser::c_ifc_parser_t, 1052
liberr
MHAParser::c_ifc_parser_t, 1052
libname
analysispath_if_t, 312
PluginLoader::config_file_splitter_t, 1364
library_handle
matlab_wrapper::matlab_wrapper_t::wrapped_plugin, 708
library_name
matlab_wrapper::matlab_wrapper_t, 702
library_paths
pluginbrowser_t, 1358
like_ratio
acPooling_wave_config, 217
like_ratio_name
acPooling_wave, 214
limit
coherence::cohflt_t, 351
coherence::vars_t, 354
MHASignal, 144
MHASignal::quantizer_t, 1240
MHASignal::waveform_t, 1273
limiter
dc_simple::dc_t, 400
limiter_threshold
dc_simple::dc_t, 400
dc_simple::dc_vars_t, 405
lin2db
MHASignal, 139, 140
line_t
dc_simple::dc_t::line_t, 402
linear
MHAOvlFilter::ShapeFun, 121
softclipper_t, 1463
softclipper_variables_t, 1465
linear_table_t
MHATableLookup::linear_table_t, 1280
Linearphase_FIR
ADM::Linearphase_FIR< F >, 269
list_dir
mha_os.cpp, 1608
mha_os.h, 1613
lnn3rdoct_t
audiometerbackend::lnn3rdoct_t, 321
lo_addr
ac2osc_t, 185
load_io_lib
fw_t, 528
load_lib
dynamiclib_t, 449
matlab_wrapper::matlab_wrapper_t, 701
load_plug
MHAParser::mhapluginloader_t, 1093
load_proc_lib
fw_t, 527
loader
latex_doc_t, 648
loadplugin_t,
analysispath_if_t, 311
local_address
io_asterisk_parser_t, 595
io_tcp_parser_t, 631
local_get_entries
MHAKernel::algo_comm_class_t, 993
local_get_var
MHAKernel::algo_comm_class_t, 993
local_insert_var
MHAKernel::algo_comm_class_t, 992
local_is_var
MHAKernel::algo_comm_class_t, 993
local_port
io_asterisk_parser_t, 595
io_tcp_parser_t, 631
local_remove_ref
MHAKernel::algo_comm_class_t, 992
local_remove_var
MHAKernel::algo_comm_class_t, 992
locate
MHAIOJackdb::io_jack_t, 954
locate_end

© 2005-2021 HörTech gGmbH, Oldenburg

MHASignal::loop_wavefragment_t, 1224
lock_channels
fw_vars_t, 532
lock_srate_fragsize
fw_vars_t, 532
locked
MHAParser::variable_t, 1120
log_down
MHASignal::schroeder_t, 1248
log_interp
dc::dc_t, 386
dc::dc_vars_t, 389
log_lambda_spec
smooth_cepstrum::smooth_cepstrum_t, 1448
log_up
MHASignal::schroeder_t, 1247
logfile
mhaserver_t, 1197
logGLRFact
noise_psd_estimator::noise_psd_estimator_t, 1318
smooth_cepstrum::smooth_cepstrum_t, 1449
logstring
mhaserver_t, 1196
longmsg
MHA_Error, 766
lookup
MHATableLookup::linear_table_t, 1280
MHATableLookup::table_t, 1284
MHATableLookup::xy_table_t, 1286
loop
addsndfile::addsndfile_if_t, 252
loop_wavefragment_t
MHASignal::loop_wavefragment_t, 1222
lost
osc_server_t, 1326
low_incl
MHAParser::range_var_t, 1112
low_limit
MHAParser::range_var_t, 1112
low_side_flat
MHAOvlFilter::band_descriptor_t, 1002
low_thresh
acPooling_wave_config, 217
lower_threshold
acPooling_wave, 213
lowpass_quotient
windnoise::if_t, 1517
lowpass_quotient_acname
windnoise::if_t, 1518
LowPassCutOffFrequency
windnoise::if_t, 1517
LowPassFraction
windnoise::cfg_t, 1513
windnoise::if_t, 1517
LowPassWindGain
windnoise::cfg_t, 1513
windnoise::if_t, 1517
lp
DynComp::dc_afterburn_rt_t, 452
level_matching::level_matching_config_t, 653
lp1i
coherence::cohflt_t, 352
lp1ltg
coherence::cohflt_t, 352
lp1r
coherence::cohflt_t, 351
lp_coeffs
adm_rtconfig_t, 279
lp_level_tau
level_matching::level_matching_t, 657
lp_order
adm_if_t, 273
lp_signal_fpass
level_matching::level_matching_t, 657
lp_signal_fstop
level_matching::level_matching_t, 657
lp_signal_order
level_matching::level_matching_t, 657
lpc, 661
~lpc, 662
algo_name, 663
comp_each_iter, 663
lpc, 662
lpc_buffer_size, 663
lpc_order, 663
norm, 664
patchbay, 664
prepare, 662
process, 662
release, 663
shift, 663
update_cfg, 663
lpc.cpp, 1571
INSERT_PATCH, 1571
Levinson2, 1571
PATCH_VAR, 1571
lpc.h, 1572
lpc_bl_predictor, 664

© 2005-2021 HörTech gGmbH, Oldenburg

~lpc_bl_predictor, 665
lpc_bl_predictor, 665
lpc_order, 667
name_b, 667
name_f, 667
name_kappa, 667
name_lpc_b, 667
name_lpc_f, 667
patchbay, 667
prepare, 666
process, 666
release, 666
update_cfg, 666
lpc_bl_predictor.cpp, 1572
INSERT_PATCH, 1572
PATCH_VAR, 1572
lpc_bl_predictor.h, 1572
EPSILON, 1573
lpc_bl_predictor_config, 668
~lpc_bl_predictor_config, 668
ac, 669
b_est, 669
backward, 669
f_est, 669
forward, 669
km, 670
lpc_bl_predictor_config, 668
lpc_order, 669
name_b, 670
name_f, 669
name_km, 669
process, 668
s_b, 670
s_f, 670
lpc_buffer_size
lpc, 663
lpc_config, 678
lpc_burg-lattice.cpp, 1573
INSERT_PATCH, 1573
PATCH_VAR, 1573
lpc_burg-lattice.h, 1573
EPSILON, 1574
lpc_burglattice, 670
~lpc_burglattice, 671
lambda, 673
lpc_burglattice, 671
lpc_order, 673
name_b, 673
name_f, 673
name_kappa, 673
patchbay, 673
prepare, 672
process, 672
release, 672
update_cfg, 672
lpc_burglattice_config, 673
~lpc_burglattice_config, 674
ac, 674
backward, 675
dm, 675
forward, 675
kappa, 675
kappa_block, 675
lambda, 675
lpc_burglattice_config, 674
lpc_order, 675
name_b, 676
name_f, 675
nm, 675
process, 674
s_b, 676
s_f, 676
lpc_config, 676
~lpc_config, 677
A, 678
comp_each_iter, 678
comp_iter, 678
corr_out, 679
insert, 677
inwave, 678
lpc_buffer_size, 678
lpc_config, 677
lpc_out, 679
N, 678
norm, 677
order, 678
process, 677
R, 678
sample, 678
shift, 677
lpc_order
adaptive_feedback_canceller, 243
lpc, 663
lpc_bl_predictor, 667
lpc_bl_predictor_config, 669
lpc_burglattice, 673
lpc_burglattice_config, 675
lpc_out
lpc_config, 679
lsl2ac, 96
Discard, 97
Ignore, 97

© 2005-2021 HörTech gGmbH, Oldenburg

overrun_behavior, 97
lsl2ac.cpp, 1574
lsl2ac.hh, 1574
lsl2ac::cfg_t, 679
cfg_t, 680
process, 680
varlist, 680
lsl2ac::lsl2ac_t, 681
activate, 684
available_streams, 686
buffersize, 685
chunksize, 685
get_all_stream_names, 684
lsl2ac_t, 683
nchannels, 685
nsamples, 685
overrun_behavior, 685
patchbay, 685
prepare, 683
process, 683
release, 683
setlock, 684
streams, 684
update, 684
lsl2ac::save_var_t, 686
~save_var_t, 688
ac, 691
buf, 690
bufsize, 693
chunksize, 692
cv, 691
get_time_correction, 690
info, 689
insert_vars, 690
n_new_samples, 693
name, 692
nchannels, 692
new_name, 691
nsamples, 692
ob, 692
operator=, 689
pull_samples_discard, 689
pull_samples_ignore, 689
receive_frame, 689
save_var_t, 688
skip, 692
stream, 690
tc, 691
tc_name, 691
tic, 692
ts, 691
ts_buf, 690
ts_name, 691
lsl2ac_t
lsl2ac::lsl2ac_t, 683
LTASS_combined
speechnoise_t, 1473
LTASS_female
speechnoise_t, 1473
LTASS_male
speechnoise_t, 1473
ltgcomp
coherence::vars_t, 355
ltgtau
coherence::vars_t, 355
lval
MHAParser::expression_t, 1061

m
dc_simple::dc_t::line_t, 403
matrixmixer::cfg_t, 714
m_alphas
ADM::Linearphase_FIR< F >, 270
m_beta
ADM::ADM< F >, 264
m_coeff
ADM::Delay< F >, 267
m_decomb
ADM::ADM< F >, 264
m_delay_back
ADM::ADM< F >, 264
m_delay_front
ADM::ADM< F >, 264
m_df
fshift::fshift_t, 515
m_fmax
fshift::fshift_t, 515
m_fmin
fshift::fshift_t, 514
m_fullsamples
ADM::Delay< F >, 267
m_lp_bf
ADM::ADM< F >, 264
m_lp_result
ADM::ADM< F >, 264
m_mu_beta
ADM::ADM< F >, 264
m_norm
ADM::Delay< F >, 268
m_now
ADM::Linearphase_FIR< F >, 270
m_now_in
ADM::Delay< F >, 268

© 2005-2021 HörTech gGmbH, Oldenburg

m_order
ADM::Linearphase_FIR< F >, 270
m_output
ADM::Linearphase_FIR< F >, 270
M_PI
mha_defs.h, 1589
mha_signal.hh, 1641
m_powerfilter_coeff
ADM::ADM< F >, 265
m_powerfilter_norm
ADM::ADM< F >, 265
m_powerfilter_state
ADM::ADM< F >, 265
m_state
ADM::Delay< F >, 268
magResp
rohBeam::rohConfig, 1412
main
analysemhaplugin.cpp, 1536
browsemhaplugins.cpp, 1539
generatemhaplugindoc.cpp, 1558
mha.cpp, 1576
MHAPlugin_Split::posix_threads_t, 1174
testalsadevice.c, 1711
make_friendly_number
MHAFilter, 106
mhajack.cpp, 1689
make_friendly_number_by_limiting
adaptive_feedback_canceller.cpp, 1531
nlms_wave.cpp, 1696
mapping
addsndfile::addsndfile_if_t, 253
coherence::vars_t, 354
matlab_wrapper, 97
matlab_wrapper.cpp, 1574
matlab_wrapper.hh, 1574
MHAPLUGIN_OVERLOAD_OUTDOMAIN, 1575
matlab_wrapper::callback, 693
callback, 694
on_writeaccess, 694
parent, 695
user_config, 694
var, 695
matlab_wrapper::matlab_wrapper_rt_cfg_t, 695
~matlab_wrapper_rt_cfg_t, 696
matlab_wrapper_rt_cfg_t, 696
user_config, 696
matlab_wrapper::matlab_wrapper_t, 697
callback, 702
callbacks, 702
cb_patchbay, 702
insert_config_vars, 701
insert_monitors, 701
library_name, 702
load_lib, 701
matlab_wrapper_t, 699
monitors, 703
patchbay, 702
plug, 702
prepare, 700
process, 699, 700
release, 701
update, 701
update_monitors, 701
vars, 702
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 703
~wrapped_plugin_t, 705
fcn_init, 708
fcn_prepare, 709
fcn_process_ss, 709
fcn_process_sw, 709
fcn_process_ws, 709
fcn_process_ww, 708
fcn_release, 709
fcn_terminate, 708
library_handle, 708
mha_spec_out, 711
mha_wave_out, 710
prepare, 707
process_ss, 706
process_sw, 707
process_ws, 706
process_ww, 705
release, 707
signal_dimensions, 710
spec_in, 710
spec_out, 710
state, 708
user_config, 708
wave_in, 710
wave_out, 710
wrapped_plugin_t, 705
matlab_wrapper::types< MHA_SPECTRUM >, 711
array_type, 711
class_signal_type, 712
signal_type, 711
matlab_wrapper::types< MHA_WAVEFORM >, 712

© 2005-2021 HörTech gGmbH, Oldenburg

array, 712
class_signal_type, 712
signal_type, 712
matlab_wrapper::types< T >, 711
matlab_wrapper_rt_cfg_t
matlab_wrapper::matlab_wrapper_rt_cfg_t, 696
matlab_wrapper_t
matlab_wrapper::matlab_wrapper_t, 699
matmix_t
matrixmixer::matmix_t, 715
matrix_t
MHASignal::matrix_t, 1227, 1229
matrixmixer, 98
matrixmixer.cpp, 1575
matrixmixer::cfg_t, 713
cfg_t, 713
m, 714
process, 713
sout, 714
wout, 714
matrixmixer::matmix_t, 714
ci, 716
co, 716
matmix_t, 715
mixer, 716
patchbay, 716
prepare, 715
process, 715, 716
update_m, 716
MAX
mha_defs.h, 1589
max
spec2wave.cpp, 1704
Vector and matrix processing toolbox, 56
max_clipped
softclipper_variables_t, 1465
max_frames
acsave::cfg_t, 224
max_lag
doasvm_feature_extraction, 429
max_p_ind_name
doasvm_classification, 424
max_pool_ind_name
acPooling_wave, 214
max_q
smooth_cepstrum::smooth_cepstrum_t, 1449
max_sleep_time
dropgen_t, 439
MAX_TCP_PORT
MHAIOAsterisk.cpp, 1658
MHAIOTCP.cpp, 1685
MAX_TCP_PORT_STR
MHAIOAsterisk.cpp, 1658
MHAIOTCP.cpp, 1685
MAX_USER_ERR
MHAIOalsa.cpp, 1653
MHAIOAsterisk.cpp, 1658
MHAIOFile.cpp, 1663
MHAIOJack.cpp, 1667
MHAIOJackdb.cpp, 1671
MHAIOParser.cpp, 1675
MHAIOPortAudio.cpp, 1679
MHAIOTCP.cpp, 1684
mhajack.h, 1691
max_val
smooth_cepstrum::smooth_cepstrum_t, 1449
maxabs
Vector and matrix processing toolbox, 54, 55
maxframe
acsave::save_var_t, 227
maxgain
dc_simple::dc_t, 400
dc_simple::dc_vars_t, 404
DynComp::dc_afterburn_rt_t, 452
DynComp::dc_afterburn_vars_t, 457
maximum_reader_xruns_in_succession_before_stop
mha_drifter_fifo_t< T >, 763
maximum_writer_xruns_in_succession_before_stop
mha_drifter_fifo_t< T >, 763
maxInputChannels
MHAIOPortAudio::device_info_t, 957
maxlen
plingploing::if_t, 1343
maxlen_
plingploing::plingploing_t, 1346
maxLim
rohBeam::rohConfig, 1412
maxOutputChannels
MHAIOPortAudio::device_info_t, 957
mcomplex_mon_t
MHAParser::mcomplex_mon_t, 1080
mcomplex_t
MHAParser::mcomplex_t, 1082
MConv
mconv::MConv, 718
mconv, 98
mconv.cpp, 1575
mconv::MConv, 717

© 2005-2021 HörTech gGmbH, Oldenburg

fragsize, 720
inch, 720
irs, 720
MConv, 718
nchannels_in, 720
nchannels_out, 720
outch, 720
patchbay, 720
prepare, 719
process, 719
release, 719
update, 719
update_irs, 719
mean
MHA_AC::stat_t, 738
MHASignal, 147
MHASignal::stat_t, 1255
mean_std
MHASignal::stat_t, 1255
median
MHASignal, 145
mfloat_mon_t
MHAParser::mfloat_mon_t, 1084
mfloat_t
MHAParser::mfloat_t, 1086
mha
mhaserver_t::tcp_server_t, 1199
speechnoise_t, 1473
mha.cpp, 1576
main, 1576
mhamain, 1576
mha.hh, 1576
algo_comm_t, 1582
MHA_AC_CHAR, 1581
MHA_AC_DOUBLE, 1581
MHA_AC_FLOAT, 1581
MHA_AC_INT, 1581
MHA_AC_MHACOMPLEX, 1582
MHA_AC_MHAREAL, 1581
MHA_AC_UNKNOWN, 1581
MHA_AC_USER, 1582
MHA_AC_VEC_FLOAT, 1582
MHA_CALLBACK_TEST, 1579
MHA_CALLBACK_TEST_PREFIX, 1579
MHA_DOMAIN_MAX, 1581
mha_domain_t, 1582
MHA_DOMAIN_UNKNOWN, 1581
MHA_RELEASE_VERSION_STRING, 1580
MHA_SPECTRUM, 1581
MHA_STRF, 1579
MHA_STRUCT_SIZEMATCH, 1580
MHA_VERSION, 1580
MHA_VERSION_BUILD, 1580
MHA_VERSION_MAJOR, 1579
MHA_VERSION_MINOR, 1579
MHA_VERSION_RELEASE, 1579
MHA_VERSION_STRING, 1580
MHA_WAVEFORM, 1580
MHA_XSTRF, 1579
MHADestroy_t, 1583
MHAGetVersion_t, 1582
MHAInit_t, 1582
MHAPluginCategory_t, 1584
MHAPluginDocumentation_t, 1584
MHAPrepare_t, 1582
MHAProc_spec2spec_t, 1583
MHAProc_spec2wave_t, 1583
MHAProc_wave2spec_t, 1583
MHAProc_wave2wave_t, 1583
MHARelease_t, 1583
MHASet_t, 1583
MHASetcpp_t, 1583
MHAStrError_t, 1584
MHA_AC, 98
MHA_AC::ac2matrix_helper_t, 721
ac, 722
ac2matrix_helper_t, 722
acvar, 723
getvar, 722
is_complex, 722
name, 722
size, 722
username, 722
MHA_AC::ac2matrix_t, 723
ac2matrix_t, 724
getname, 724
getusername, 724
insert, 725
update, 724
MHA_AC::acspace2matrix_t, 725
~acspace2matrix_t, 727
acspace2matrix_t, 726
data, 729
frame, 728
frameno, 729
insert, 728
len, 729
operator=, 727
operator[], 727, 728
size, 728
update, 728

© 2005-2021 HörTech gGmbH, Oldenburg

MHA_AC::double_t, 729
~double_t, 730
ac, 731
data, 730
double_t, 730
insert, 730
name, 731
MHA_AC::float_t, 731
~float_t, 732
ac, 732
data, 732
float_t, 732
insert, 732
name, 733
MHA_AC::int_t, 733
~int_t, 734
ac, 734
data, 734
insert, 734
int_t, 734
name, 734
MHA_AC::spectrum_t, 735
~spectrum_t, 736
ac, 737
insert, 737
name, 737
spectrum_t, 736
MHA_AC::stat_t, 737
insert, 738
mean, 738
stat_t, 738
std, 739
update, 738
MHA_AC::waveform_t, 739
~waveform_t, 740
ac, 741
insert, 741
name, 741
waveform_t, 740
MHA_AC_CHAR
mha.hh, 1581
MHA_AC_DOUBLE
mha.hh, 1581
MHA_AC_FLOAT
mha.hh, 1581
MHA_AC_INT
mha.hh, 1581
MHA_AC_MHACOMPLEX
mha.hh, 1582
MHA_AC_MHAREAL
mha.hh, 1581
MHA_AC_UNKNOWN
mha.hh, 1581
MHA_AC_USER
mha.hh, 1582
MHA_AC_VEC_FLOAT
mha.hh, 1582
mha_algo_comm.cpp, 1584
AC_DIM_MISMATCH, 1585
AC_INVALID_HANDLE, 1584
AC_INVALID_NAME, 1585
AC_INVALID_OUTPTR, 1585
AC_STRING_TRUNCATED, 1585
AC_SUCCESS, 1584
AC_TYPE_MISMATCH, 1585
algo_comm_default, 1585
mha_algo_comm.h, 1585
mha_algo_comm.hh, 1587
algo_comm_default, 1587
ALGO_COMM_ID_STR, 1587
mha_alloc
mha_ruby.cpp, 1629
MHA_assert
Error handling in the openMHA, 29
MHA_assert_equal
Error handling in the openMHA, 29
mha_audio_descriptor_t, 741
cf, 742
chdir, 743
dt, 742
is_complex, 742
n_channels, 742
n_freqs, 742
n_samples, 742
mha_audio_t, 743
cdata, 744
descriptor, 743
rdata, 744
MHA_CALLBACK_TEST
mha.hh, 1579
MHA_CALLBACK_TEST_PREFIX
mha.hh, 1579
mha_channel_info_t, 744
dir, 745
id, 745
idstr, 745
peaklevel, 745
side, 745
mha_complex
Complex arithmetics in the openMHA, 60
mha_complex_t, 745
im, 746

© 2005-2021 HörTech gGmbH, Oldenburg

re, 746
mha_complex_test_array_t, 746
c, 747
mha_dblbuf_t
mha_dblbuf_t< FIFO >, 749
mha_dblbuf_t< FIFO >, 747
~mha_dblbuf_t, 750
delay, 753
fifo_size, 754
get_delay, 750
get_fifo_size, 750
get_inner_error, 751
get_inner_size, 750
get_input_channels, 750
get_input_fifo_fill_count, 751
get_input_fifo_space, 751
get_outer_size, 750
get_output_channels, 751
get_output_fifo_fill_count, 751
get_output_fifo_space, 751
inner_error, 754
inner_size, 753
input, 752
input_channels, 754
input_fifo, 754
mha_dblbuf_t, 749
outer_error, 755
outer_size, 753
output, 753
output_channels, 754
output_fifo, 754
process, 752
provoke_inner_error, 751
provoke_outer_error, 752
value_type, 749
mha_debug
Error handling in the openMHA, 29
mha_error.cpp, 1593
mha_defs.h, 1588
__MHA_FUN__, 1588
__declspec, 1589
CHECK_EXPR, 1588
CHECK_VAR, 1588
M_PI, 1589
MAX, 1589
MHA_EAR_LEFT, 1589
MHA_EAR_MAX, 1589
MHA_EAR_RIGHT, 1589
MIN, 1589
mha_delenv
mha_os.cpp, 1608
mha_os.h, 1613
mha_direction_t, 755
azimuth, 755
distance, 756
elevation, 756
MHA_DOMAIN_MAX
mha.hh, 1581
mha_domain_t
mha.hh, 1582
MHA_DOMAIN_UNKNOWN
mha.hh, 1581
mha_drifter_fifo_t
mha_drifter_fifo_t< T >, 758
mha_drifter_fifo_t< T >, 756
desired_fill_count, 762
get_available_space, 760
get_des_fill_count, 760
get_fill_count, 760
get_min_fill_count, 761
maximum_reader_xruns_in_succession_before_stop, 763
maximum_writer_xruns_in_succession_before_stop, 763
mha_drifter_fifo_t, 758
minimum_fill_count, 761
null_data, 763
read, 759
reader_started, 762
reader_xruns_in_succession, 763
reader_xruns_since_start, 763
reader_xruns_total, 762
starting, 761
startup_zeros, 764
stop, 761
write, 759
writer_started, 762
writer_xruns_in_succession, 763
writer_xruns_since_start, 762
writer_xruns_total, 762
MHA_EAR_LEFT
mha_defs.h, 1589
MHA_EAR_MAX
mha_defs.h, 1589
MHA_EAR_RIGHT
mha_defs.h, 1589
MHA_ERR_INVALID_HANDLE
mha_errno.h, 1591
MHA_ERR_NULL
mha_errno.h, 1592
MHA_ERR_SUCCESS
mha_errno.h, 1591

© 2005-2021 HörTech gGmbH, Oldenburg

MHA_ERR_UNKNOWN
mha_errno.h, 1591
MHA_ERR_USER
mha_errno.h, 1592
MHA_ERR_VARFMT
mha_errno.h, 1592
MHA_ERR_VARRANGE
mha_errno.h, 1592
mha_errno.c, 1590
cstr_strerror, 1591
mha_set_user_error, 1590
mha_strerror, 1590
next_except_str, 1591
STRLEN, 1590
mha_errno.h, 1591
MHA_ERR_INVALID_HANDLE, 1591
MHA_ERR_NULL, 1592
MHA_ERR_SUCCESS, 1591
MHA_ERR_UNKNOWN, 1591
MHA_ERR_USER, 1592
MHA_ERR_VARFMT, 1592
MHA_ERR_VARRANGE, 1592
mha_set_user_error, 1592
mha_strerror, 1592
MHA_Error, 764
~MHA_Error, 765
get_longmsg, 766
get_msg, 766
longmsg, 766
MHA_Error, 765
msg, 766
operator=, 766
what, 766
mha_error.cpp, 1592
mha_debug, 1593
mha_error.hh, 1593
Getmsg, 1594
mha_error_helpers, 99
digits, 99
snprintf_required_length, 100
MHA_ErrorMsg
Error handling in the openMHA, 28
MHA_ErrorMsg2
MHAIOAsterisk.cpp, 1658
MHAIOTCP.cpp, 1684
MHA_ErrorMsg3
MHAIOAsterisk.cpp, 1658
MHAIOTCP.cpp, 1685
mha_event_emitter.h, 1594
mha_events.cpp, 1595
mha_events.h, 1595
mha_exit_request
mha_ruby.cpp, 1629
mha_fft
gsc_adaptive_stage::gsc_adaptive_stage, 541
smooth_cepstrum::smooth_cepstrum_t, 1445
mha_fft_backward
Fast Fourier Transform functions, 75
mha_fft_backward_scale
Fast Fourier Transform functions, 76
mha_fft_forward
Fast Fourier Transform functions, 75
mha_fft_forward_scale
Fast Fourier Transform functions, 76
mha_fft_free
Fast Fourier Transform functions, 72
mha_fft_new
Fast Fourier Transform functions, 71
mha_fft_spec2wave
Fast Fourier Transform functions, 73, 74
mha_fft_spec2wave_scale
Fast Fourier Transform functions, 77
mha_fft_t
Fast Fourier Transform functions, 71
mha_fft_wave2spec
Fast Fourier Transform functions, 72, 73
mha_fft_wave2spec_scale
Fast Fourier Transform functions, 76
mha_fftfb.cpp, 1595
BARKSCALE_ENTRIES, 1596
filtershapefun, 1596
mha_fftfb.hh, 1597
mha_fifo.cpp, 1598
mha_fifo.h, 1598
mha_fifo_thread_platform_implementation_t, 1599
mha_fifo_lf_t
mha_fifo_lf_t< T >, 768
mha_fifo_lf_t< T >, 767
atomic_read_ptr, 770
atomic_write_ptr, 770
get_available_space, 769
get_fill_count, 769
mha_fifo_lf_t, 768
read, 769
write, 768
mha_fifo_lw_t
mha_fifo_lw_t< T >, 771
mha_fifo_lw_t< T >, 770
~mha_fifo_lw_t, 771

© 2005-2021 HörTech gGmbH, Oldenburg

error, 773
mha_fifo_lw_t, 771
read, 772
set_error, 773
sync, 773
write, 772
mha_fifo_posix_threads_t, 774
~mha_fifo_posix_threads_t, 774
aquire_mutex, 775
decrease_condition, 776
decrement, 775
increase_condition, 776
increment, 775
mha_fifo_posix_threads_t, 774
mutex, 776
release_mutex, 775
wait_for_decrease, 775
wait_for_increase, 775
mha_fifo_t
mha_fifo_t< T >, 778, 779
mha_fifo_t< T >, 776
~mha_fifo_t, 779
buf, 782
clear, 781
get_available_space, 780
get_fill_count, 780, 782
get_max_fill_count, 781
get_read_ptr, 782
get_write_ptr, 781
mha_fifo_t, 778, 779
operator=, 781
read, 780
read_ptr, 783
value_type, 778
write, 779
write_ptr, 782
mha_fifo_thread_guard_t, 783
~mha_fifo_thread_guard_t, 784
mha_fifo_thread_guard_t, 783
sync, 784
mha_fifo_thread_platform_implementation_t
mha_fifo.h, 1599
mha_fifo_thread_platform_t, 784
~mha_fifo_thread_platform_t, 785
aquire_mutex, 786
decrement, 787
increment, 787
mha_fifo_thread_platform_t, 785, 786
operator=, 787
release_mutex, 786
wait_for_decrease, 786
wait_for_increase, 787
mha_filter.cpp, 1599
diff_coeffs, 1599
mha_filter.hh, 1599
mha_fragsize
MHAIOJackdb::io_jack_t, 952
mha_free
mha_ruby.cpp, 1628
mha_freelib
mha_os.h, 1610
mha_freelib_success
mha_os.h, 1610
mha_generic_chain.cpp, 1601
mhaconfig_compare, 1601
mha_generic_chain.h, 1601
MHAPLUGIN_OVERLOAD_OUTDOMAIN, 1602
mha_getenv
mha_os.cpp, 1607
mha_os.h, 1612
mha_getlibfun
mha_os.h, 1610
mha_getlibfun_checked
mha_os.h, 1610
mha_git_commit_hash
mha_git_commit_hash.cpp, 1602
mha_git_commit_hash.hh, 1603
mha_git_commit_hash.cpp, 1602
GITCOMMITHASH, 1602
mha_git_commit_hash, 1602
mha_git_commit_hash.hh, 1603
mha_git_commit_hash, 1603
mha_hasenv
mha_os.cpp, 1607
mha_os.h, 1612
mha_hton
mha_os.h, 1613, 1614
MHA_ID_MATRIX
mha_signal.cpp, 1632
MHA_ID_UINT_VECTOR
mha_signal.cpp, 1631
mha_io_ifc.h, 1603
IODestroy_t, 1605
IOInit_t, 1605
IOPrepare_t, 1605
IOProcessEvent_t, 1604
IORelease_t, 1605
IOSetVar_t, 1605
IOStart_t, 1605
IOStartedEvent_t, 1604
IOStop_t, 1605

© 2005-2021 HörTech gGmbH, Oldenburg

IOStoppedEvent_t, 1604
IOStrError_t, 1605
MHAIO_DOCUMENTATION, 1604
MHAIO_DOCUMENTATION_PREFIX, 1604
mha_io_utils.cpp, 1606
mha_io_utils.hh, 1606
mha_lib_extension
mha_os.h, 1611
mha_libhandle_t
mha_os.h, 1611
mha_library_paths
mha_os.cpp, 1608
mha_os.h, 1613
mha_loadlib
mha_os.h, 1610
mha_loadlib_error
mha_os.h, 1611
mha_min_1
mha_signal.hh, 1642
mha_msleep
mha_os.h, 1611
mha_multisrc.cpp, 1606
mha_multisrc.h, 1606
mha_ntoh
mha_os.h, 1613, 1614
mha_os.cpp, 1607
list_dir, 1608
mha_delenv, 1608
mha_getenv, 1607
mha_hasenv, 1607
mha_library_paths, 1608
mha_setenv, 1608
mha_os.h, 1609
FMTsz, 1611
list_dir, 1613
mha_delenv, 1613
mha_freelib, 1610
mha_freelib_success, 1610
mha_getenv, 1612
mha_getlibfun, 1610
mha_getlibfun_checked, 1610
mha_hasenv, 1612
mha_hton, 1613, 1614
mha_lib_extension, 1611
mha_libhandle_t, 1611
mha_library_paths, 1613
mha_loadlib, 1610
mha_loadlib_error, 1611
mha_msleep, 1611
mha_ntoh, 1613, 1614
MHA_RESOLVE, 1611
MHA_RESOLVE_CHECKED, 1611
mha_setenv, 1612
mha_parse
mha_ruby.cpp, 1629
mha_parser.cpp, 1614
check_parenthesis_complex, 1616
check_sign_complex, 1616
MHAPLATFORM, 1615
parse_1_complex, 1617
parse_1_float, 1615
write_float, 1615
mha_parser.hh, 1618
DEFAULT_RETSIZE, 1622
insert_member, 1622
mha_platform_tic
mha_profiling.c, 1627
mha_profiling.h, 1628
mha_platform_tictoc_t
mha_profiling.h, 1627
mha_platform_toc
mha_profiling.c, 1627
mha_profiling.h, 1628
mha_plugin.cpp, 1622
mha_plugin.hh, 1622
__declspec, 1623
HINSTANCE, 1624
MHAPLUGIN_CALLBACKS, 1625
MHAPLUGIN_CALLBACKS_PREFIX, 1624
MHAPLUGIN_DOCUMENTATION, 1626
MHAPLUGIN_DOCUMENTATION_PREFIX, 1625
MHAPLUGIN_INIT_CALLBACKS, 1625
MHAPLUGIN_INIT_CALLBACKS_PREFIX, 1624
MHAPLUGIN_PROC_CALLBACK, 1625
MHAPLUGIN_PROC_CALLBACK_PREFIX, 1624
MHAPLUGIN_SETCPP_CALLBACK_PREFIX, 1624
WINAPI, 1623
mha_profiling.c, 1626
mha_platform_tic, 1627
mha_platform_toc, 1627
mha_tic, 1626
mha_toc, 1627
mha_profiling.h, 1627
mha_platform_tic, 1628
mha_platform_tictoc_t, 1627
mha_platform_toc, 1628

© 2005-2021 HörTech gGmbH, Oldenburg

mha_real_t
Vector and matrix processing toolbox, 37
mha_real_test_array_t, 788
r, 788
MHA_RELEASE_VERSION_STRING
mha.hh, 1580
MHA_RESOLVE
mha_os.h, 1611
MHA_RESOLVE_CHECKED
mha_os.h, 1611
mha_round
mha_signal.hh, 1641
mha_rt_fifo_element_t
mha_rt_fifo_element_t< T >, 789
mha_rt_fifo_element_t< T >, 788
~mha_rt_fifo_element_t, 789
abandonned, 790
data, 790
mha_rt_fifo_element_t, 789
next, 789
mha_rt_fifo_t
mha_rt_fifo_t< T >, 791
mha_rt_fifo_t< T >, 790
~mha_rt_fifo_t, 791
current, 793
mha_rt_fifo_t, 791
poll, 792
poll_1, 792
push, 792
remove_abandonned, 793
remove_all, 793
root, 793
mha_ruby.cpp, 1628
Init_mha_ruby, 1629
mha_alloc, 1629
mha_exit_request, 1629
mha_free, 1628
mha_parse, 1629
rb_f_t, 1628
mha_samplerate
MHAIOJackdb::io_jack_t, 952
mha_set_user_error
mha_errno.c, 1590
mha_errno.h, 1592
mha_setenv
mha_os.cpp, 1608
mha_os.h, 1612
mha_signal.cpp, 1629
ASSERT_EQUAL_DIM, 1632
ASSERT_EQUAL_DIM_PTR, 1632
intensity, 1632
MHA_ID_MATRIX, 1632
MHA_ID_UINT_VECTOR, 1631
safe_div, 1632
set_minabs, 1632
mha_signal.hh, 1633
M_PI, 1641
mha_min_1, 1642
mha_round, 1641
operator<<, 1642
operator>>, 1643
safe_div, 1642
set_minabs, 1642
value, 1642
mha_signal_fft.h, 1643
mha_spec_out
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 711
mha_spec_t, 794
buf, 795
channel_info, 795
num_channels, 795
num_frames, 795
MHA_SPECTRUM
mha.hh, 1581
mha_stash_environment_variable_t, 796
~mha_stash_environment_variable_t, 797
existed_before, 797
mha_stash_environment_variable_t, 796
original_content, 797
variable_name, 797
mha_strerror
mha_errno.c, 1590
mha_errno.h, 1592
MHA_STRF
mha.hh, 1579
MHA_STRUCT_SIZEMATCH
mha.hh, 1580
mha_tablelookup.cpp, 1643
mha_tablelookup.hh, 1643
MHA_TCP, 100
dtime, 103
G_ERRNO, 102
H_ERRNO, 102
HSTRERROR, 102
N_ERRNO, 102
sock_initializer, 103
SOCKET, 102
stime, 103
STRERROR, 102
mha_tcp, 103

© 2005-2021 HörTech gGmbH, Oldenburg

mha_tcp.cpp, 1644
ASYNC_CONNECT_STARTED, 1645
closesocket, 1645
host_port_to_sock_addr, 1646
INVALID_SOCKET, 1645
SOCKET, 1646
SOCKET_ERROR, 1645
tcp_connect_to, 1646
tcp_connect_to_with_timeout, 1646
thread_start_func, 1646
mha_tcp.hh, 1647
Sleep, 1648
MHA_TCP::Async_Notify, 798
~Async_Notify, 799
Async_Notify, 798
pipe, 799
reset, 799
set, 799
mha_tcp::buffered_socket_t, 799
current_message, 801
get_buffer, 800
next_message, 801
queue_write, 800
streambuf, 801
MHA_TCP::Client, 801
Client, 802
MHA_TCP::Connection, 803
~Connection, 805
buffered_incoming_bytes, 810
buffered_outgoing_bytes, 810
can_read_bytes, 808
can_read_line, 807
can_sysread, 806
can_syswrite, 806
closed, 811
Connection, 805
eof, 807
fd, 811
get_fd, 807
get_peer_address, 807
get_peer_port, 807
get_read_event, 807
get_write_event, 807
inbuf, 810
init_peer_data, 805
needs_write, 810
outbuf, 810
peer_addr, 811
read_bytes, 809
read_event, 810
read_line, 808
sysread, 806
syswrite, 806
try_write, 809
write, 809
write_event, 810
MHA_TCP::Event_Watcher, 811
~Event_Watcher, 813
Events, 812
events, 813
ignore, 813
iterator, 812
observe, 813
wait, 813
MHA_TCP::OS_EVENT_TYPE, 814
fd, 815
mode, 815
R, 814
T, 814
timeout, 815
W, 814
X, 814
MHA_TCP::Server, 815
~Server, 816
accept, 817
accept_event, 818
get_accept_event, 817
get_interface, 817
get_port, 817
iface, 818
initialize, 817
port, 818
Server, 816
serversocket, 818
sock_addr, 818
try_accept, 817
mha_tcp::server_t, 819
~server_t, 821
acceptor, 824
add_connection, 824
async_accept_has_been_triggered, 824
connections, 824
get_address, 821
get_context, 823
get_endpoint, 821
get_num_accepted_connections, 822
get_port, 821
io_context, 824
num_accepted_connections, 824
on_received_line, 822
post_trigger_read_line, 823
run, 822

© 2005-2021 HörTech gGmbH, Oldenburg

server_t, 820
shutdown, 823
trigger_accept, 823
trigger_read_line, 823
MHA_TCP::sock_init_t, 825
sock_init_t, 825
MHA_TCP::Sockaccept_Event, 825
Sockaccept_Event, 826
MHA_TCP::Sockread_Event, 826
Sockread_Event, 827
MHA_TCP::Sockwrite_Event, 827
Sockwrite_Event, 828
MHA_TCP::Thread, 828
~Thread, 831
arg, 831
error, 832
FINISHED, 830
PREPARED, 830
return_value, 832
run, 831
RUNNING, 830
state, 832
thr_f, 829
Thread, 830
thread_arg, 832
thread_attr, 831
thread_finish_event, 832
thread_func, 832
thread_handle, 831
MHA_TCP::Timeout_Event, 833
end_time, 834
get_os_event, 833
Timeout_Event, 833
MHA_TCP::Timeout_Watcher, 834
~Timeout_Watcher, 835
timeout, 835
Timeout_Watcher, 835
MHA_TCP::Wakeup_Event, 836
~Wakeup_Event, 837
get_os_event, 838
ignored_by, 837
observed_by, 837
observers, 838
os_event, 838
os_event_valid, 839
reset, 838
status, 838
Wakeup_Event, 837
mha_tcp_server.cpp, 1648
mha_tcp_server.hh, 1648
mha_test_struct_size
PluginLoader::mhapluginloader_t, 1373
mha_tic
mha_profiling.c, 1626
mha_tictoc_t, 839
t, 839
tv1, 839
tv2, 839
tz, 839
mha_toc
mha_profiling.c, 1627
mha_toolbox.h, 1649
mha_utils.cpp, 1649
mha_utils.hh, 1649
MHA_VERSION
mha.hh, 1580
MHA_VERSION_BUILD
mha.hh, 1580
MHA_VERSION_MAJOR
mha.hh, 1579
MHA_VERSION_MINOR
mha.hh, 1579
MHA_VERSION_RELEASE
mha.hh, 1579
MHA_VERSION_STRING
mha.hh, 1580
mha_wave_out
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 710
mha_wave_t, 840
buf, 841
channel_info, 841
num_channels, 841
num_frames, 841
MHA_WAVEFORM
mha.hh, 1580
mha_windowparser.cpp, 1649
wnd_funs, 1650
mha_windowparser.h, 1650
MHA_XSTRF
mha.hh, 1579
mhachain, 104
mhachain.cpp, 1651
mhachain::chain_base_t, 842
algos, 844
b_prepared, 845
bprofiling, 844
cfin, 845
cfout, 845
chain_base_t, 843
old_algos, 844
patchbay, 845

© 2005-2021 HörTech gGmbH, Oldenburg

prepare, 844
process, 843, 844
release, 844
update, 844
mhachain::mhachain_t, 846
mhachain_t, 846
mhachain::plugs_t, 847
~plugs_t, 848
ac, 849
algos, 849
alloc_plugs, 848
b_prepared, 849
b_use_profiling, 851
cleanup_plugs, 849
parser, 849
plugs_t, 847
prepare, 848
prepared, 848
proc_cnt, 850
process, 848
prof_algos, 849
prof_cfg, 850
prof_init, 850
prof_load_con, 851
prof_prepare, 850
prof_process, 850
prof_process_load, 850
prof_process_tt, 850
prof_release, 850
prof_tt_con, 851
profiling, 849
release, 848
tictoc, 851
update_proc_load, 849
mhachain_t
mhachain::mhachain_t, 846
mhachannels
addsndfile::addsndfile_if_t, 254
mhaconfig_compare
mha_generic_chain.cpp, 1601
PluginLoader, 157
mhaconfig_in
MHAPlugin::plugin_t< runtime_cfg_t >, 1157
mhaconfig_mon_t
MHAParser::mhaconfig_mon_t, 1089
mhaconfig_out
MHAPlugin::plugin_t< runtime_cfg_t >, 1157
mhaconfig_t, 851
channels, 852
domain, 852
fftlen, 853
fragsize, 852
srate, 853
wndlen, 853
MHADestroy_cb
PluginLoader::mhapluginloader_t, 1374
MHADestroy_t
mha.hh, 1583
MHAEvents, 104
MHAEvents::connector_base_t, 853
~connector_base_t, 854
connector_base_t, 854
emit_event, 854, 855
emitter_die, 855
emitter_is_alive, 855
MHAEvents::connector_t< receiver_t >, 856
~connector_t, 857
connector_t, 857
emit_event, 857, 858
emitter, 858
eventhandler, 858
eventhandler_s, 858
eventhandler_suu, 859
receiver, 858
MHAEvents::emitter_t, 859
~emitter_t, 860
connect, 860
connections, 861
disconnect, 860
operator(), 860
MHAEvents::patchbay_t< receiver_t >, 861
~patchbay_t, 862
connect, 862, 863
cons, 863
mhafft
fshift_hilbert::hilbert_shifter_t, 521
MHAFilter, 104
butter_stop_ord1, 107
fir_lp, 107
gcd, 108
make_friendly_number, 106
o1_lp_coeffs, 106
resampling_factors, 109
sinc, 109
spec2fir, 108
MHAFilter::adapt_filter_param_t, 863
adapt_filter_param_t, 864
err_in, 864
mu, 864
MHAFilter::adapt_filter_state_t, 864

© 2005-2021 HörTech gGmbH, Oldenburg

adapt_filter_state_t, 865
filter, 865
nchannels, 865
ntaps, 865
od, 866
oy, 866
W, 865
X, 865
MHAFilter::adapt_filter_t, 866
adapt_filter_t, 867
connector, 868
err_in, 868
filter, 867
mu, 868
nchannels, 868
ntaps, 868
set_channelcnt, 867
update_mu, 867
update_ntaps, 868
MHAFilter::blockprocessing_polyphase_resampling_t, 868
~blockprocessing_polyphase_resampling_t, 870
blockprocessing_polyphase_resampling_t, 869
can_read, 871
fragsize_in, 871
fragsize_out, 871
num_channels, 872
read, 871
resampling, 871
write, 870
MHAFilter::complex_bandpass_t, 872
A_, 875
B_, 876
complex_bandpass_t, 873
creator_A, 873
creator_B, 874
filter, 874, 875
get_weights, 874
inspect, 875
set_state, 874
set_weights, 874
Yn, 876
MHAFilter::diff_t, 876
diff_t, 877
MHAFilter::fftfilter_t, 877
~fftfilter_t, 878
channels, 881
fft, 882
fftfilter_t, 878
fftlen, 881
filter, 879, 880
fragsize, 880
sInput, 881
sWeights, 881
update_coeffs, 879
wInput, 881
wInput_fft, 881
wIRS_fft, 881
wOutput, 881
wOutput_fft, 881
MHAFilter::fftfilterbank_t, 882
~fftfilterbank_t, 884
fft, 887
fftfilterbank_t, 883
fftlen, 886
filter, 884, 885
firchannels, 886
fragsize, 886
get_irs, 885
Hs, 886
hw, 886
inputchannels, 886
outputchannels, 886
tail, 887
update_coeffs, 884
Xs, 886
xw, 886
Ys, 887
yw, 887
yw_temp, 887
MHAFilter::filter_t, 887
~filter_t, 890
A, 891
B, 892
channels, 892
filter, 890, 891
filter_t, 889
get_len_A, 891
get_len_B, 891
len, 892
len_A, 892
len_B, 892
operator=, 890
state, 892
MHAFilter::gamma_flt_t, 892
~gamma_flt_t, 894
A, 896
bw_, 896
cf_, 896
delay, 896

© 2005-2021 HörTech gGmbH, Oldenburg

envelope_delay, 896
gamma_flt_t, 893
get_A, 895
get_resynthesis_gain, 895
get_weights, 895
GF, 896
inspect, 895
operator(), 894
phase_correction, 894
reset_state, 895
resynthesis_gain, 896
set_weights, 895
srate_, 896
MHAFilter::iir_filter_state_t, 897
iir_filter_state_t, 897
MHAFilter::iir_filter_t, 898
A, 902
B, 902
connector, 902
filter, 900
iir_filter_t, 899
nchannels, 902
resize, 900
update_filter, 902
MHAFilter::iir_ord1_real_t, 902
A_, 905
B_, 905
iir_ord1_real_t, 903
operator(), 904, 905
set_state, 904
Yn, 905
MHAFilter::o1_ar_filter_t, 906
c1_a, 909
c1_r, 909
c2_a, 909
c2_r, 909
fs, 909
o1_ar_filter_t, 907
operator(), 908, 909
set_tau_attack, 908
set_tau_release, 908
MHAFilter::o1flt_lowpass_t, 910
get_c1, 912
get_last_output, 912
o1flt_lowpass_t, 911
set_tau, 912
MHAFilter::o1flt_maxtrack_t, 913
o1flt_maxtrack_t, 914
set_tau, 914
MHAFilter::o1flt_mintrack_t, 915
o1flt_mintrack_t, 916
set_tau, 916
MHAFilter::partitioned_convolution_t, 917
~partitioned_convolution_t, 918
bookkeeping, 920
current_input_signal_buffer_half_index, 920
current_output_partition_index, 921
fft, 921
filter_partitions, 919
fragsize, 919
frequency_response, 920
input_signal_spec, 920
input_signal_wave, 920
nchannels_in, 919
nchannels_out, 919
output_partitions, 919
output_signal_spec, 920
output_signal_wave, 921
partitioned_convolution_t, 918
process, 919
MHAFilter::partitioned_convolution_t::index_t, 921
delay, 923
index_t, 922
source_channel_index, 923
target_channel_index, 923
MHAFilter::polyphase_resampling_t, 923
downsampling_factor, 927
impulse_response, 927
now_index, 927
polyphase_resampling_t, 925
read, 926
readable_frames, 926
ringbuffer, 928
underflow, 927
upsampling_factor, 927
write, 926
MHAFilter::resampling_filter_t, 928
fragsize, 930
fragsize_validator, 930
resampling_filter_t, 929
MHAFilter::smoothspec_t, 930
_linphase_asym, 934
~smoothspec_t, 932
fft, 934
fftlen, 933
internal_fir, 933
minphase, 934
nchannels, 933
smoothspec, 932
smoothspec_t, 931

© 2005-2021 HörTech gGmbH, Oldenburg

spec2fir, 933
tmp_spec, 934
tmp_wave, 934
window, 933
MHAFilter::thirdoctave_analyzer_t, 934
bw_generator, 936
cf, 936
cf_generator, 936
cfg_, 936
dup, 936
fb, 936
get_cf_hz, 935
nbands, 935
nchannels, 935
out_chunk, 936
out_chunk_im, 937
process, 935
thirdoctave_analyzer_t, 935
MHAFilter::transfer_function_t, 937
impulse_response, 940
isempty, 939
non_empty_partitions, 939
partitions, 938
source_channel_index, 940
target_channel_index, 940
transfer_function_t, 938
MHAFilter::transfer_matrix_t, 940
non_empty_partitions, 941
partitions, 941
mhafw_lib.cpp, 1651
mhafw_lib.h, 1651
MHAGetVersion_cb
PluginLoader::mhapluginloader_t, 1374
MHAGetVersion_t
mha.hh, 1582
MHAInit_cb
PluginLoader::mhapluginloader_t, 1374
MHAInit_t
mha.hh, 1582
MHAIO_DOCUMENTATION
mha_io_ifc.h, 1604
MHAIO_DOCUMENTATION_PREFIX
mha_io_ifc.h, 1604
MHAIOalsa.cpp, 1651
DBG, 1652
dummy_interface_test, 1654
ERR_IHANDLE, 1653
ERR_SUCCESS, 1653
ERR_USER, 1653
IODestroy, 1654, 1656
IOInit, 1653, 1654
IOPrepare, 1653, 1655
IORelease, 1653, 1655
IOSetVar, 1654, 1655
IOStart, 1653, 1655
IOStop, 1653, 1655
IOStrError, 1654, 1655
MAX_USER_ERR, 1653
user_err_msg, 1656
MHAIOAsterisk.cpp, 1656
copy_error, 1660
dummy_interface_test, 1659
ERR_IHANDLE, 1657
ERR_SUCCESS, 1657
ERR_USER, 1657
IODestroy, 1659, 1661
IOInit, 1658, 1660
IOPrepare, 1659, 1660
IORelease, 1659, 1660
IOSetVar, 1659, 1661
IOStart, 1659, 1660
IOStop, 1659, 1660
IOStrError, 1659, 1661
MAX_TCP_PORT, 1658
MAX_TCP_PORT_STR, 1658
MAX_USER_ERR, 1658
MHA_ErrorMsg2, 1658
MHA_ErrorMsg3, 1658
MIN_TCP_PORT, 1658
MIN_TCP_PORT_STR, 1658
thread_startup_function, 1660
user_err_msg, 1661
MHAIOFile.cpp, 1661
DEBUG, 1662
dummy_interface_test, 1664
ERR_IHANDLE, 1663
ERR_SUCCESS, 1662
ERR_USER, 1663
IODestroy, 1664, 1665
IOInit, 1663, 1664
IOPrepare, 1663, 1664
IORelease, 1663, 1665
IOSetVar, 1663, 1665
IOStart, 1663, 1664
IOStop, 1663, 1665
IOStrError, 1664, 1665
MAX_USER_ERR, 1663
user_err_msg, 1665
MHAIOJack, 109
MHAIOJack.cpp, 1666
dummy_interface_test, 1668
ERR_IHANDLE, 1667

© 2005-2021 HörTech gGmbH, Oldenburg

ERR_SUCCESS, 1667
ERR_USER, 1667
IODestroy, 1668, 1669
IOInit, 1667, 1668
IOPrepare, 1667, 1668
IORelease, 1668, 1669
IOSetVar, 1668, 1669
IOStart, 1667, 1669
IOStop, 1667, 1669
IOStrError, 1668, 1669
MAX_USER_ERR, 1667
user_err_msg, 1670
MHAIOJack::io_jack_t, 941
clientname, 945
connections_in, 945
connections_out, 946
delays_in, 945
delays_out, 946
fw_fragsize, 945
fw_samplerate, 945
get_all_input_ports, 944
get_all_output_ports, 944
get_delays_in, 944
get_delays_out, 944
get_physical_input_ports, 944
get_physical_output_ports, 944
io_jack_t, 943
patchbay, 947
portnames_in, 946
portnames_out, 946
ports_in_all, 946
ports_in_physical, 946
ports_out_all, 946
ports_out_physical, 946
ports_parser, 947
prepare, 943
read_get_cpu_load, 944
read_get_scheduler, 945
read_get_xruns, 945
reconnect_inports, 943
reconnect_outports, 944
release, 943
servername, 945
state_cpuload, 947
state_parser, 947
state_priority, 947
state_scheduler, 947
state_xruns, 947
MHAIOJackdb, 110
MHAIOJackdb.cpp, 1670
dummy_interface_test, 1672
ERR_IHANDLE, 1671
ERR_SUCCESS, 1671
ERR_USER, 1671
IODestroy, 1672, 1673
IOInit, 1671, 1672
IOPrepare, 1671, 1672
IORelease, 1672, 1673
IOSetVar, 1672, 1673
IOStart, 1671, 1673
IOStop, 1671, 1673
IOStrError, 1672, 1673
MAX_USER_ERR, 1671
user_err_msg, 1674
MHAIOJackdb::io_jack_t, 948
clientname, 953
connections_in, 953
connections_out, 953
fail_on_async_jackerr, 953
fail_on_async_jackerror, 950
fragsize_ratio, 952
get_all_input_ports, 951
get_all_output_ports, 951
get_physical_input_ports, 951
get_physical_output_ports, 951
io_jack_t, 950
IOProcessEvent_inner, 950
locate, 954
mha_fragsize, 952
mha_samplerate, 952
patchbay, 955
portnames_in, 953
portnames_out, 953
ports_in_all, 954
ports_in_physical, 954
ports_out_all, 954
ports_out_physical, 954
ports_parser, 954
prepare, 950
proc_event, 952
proc_handle, 952
pwinner_out, 955
read_get_cpu_load, 951
read_get_scheduler, 952
read_get_xruns, 951
reconnect_inports, 951
reconnect_outports, 951
release, 950
server_fragsize, 954
server_srate, 954
servername, 953
set_locate, 952

© 2005-2021 HörTech gGmbH, Oldenburg

set_use_jack_transport, 952
state_cpuload, 955
state_parser, 955
state_priority, 955
state_scheduler, 955
state_xruns, 955
use_jack_transport, 953
MHAIOParser.cpp, 1674
dummy_interface_test, 1676
ERR_IHANDLE, 1675
ERR_SUCCESS, 1675
ERR_USER, 1675
IODestroy, 1676, 1677
IOInit, 1675, 1676
IOPrepare, 1675, 1676
IORelease, 1676, 1677
IOSetVar, 1676, 1677
IOStart, 1675, 1677
IOStop, 1675, 1677
IOStrError, 1676, 1677
MAX_USER_ERR, 1675
user_err_msg, 1678
MHAIOPortAudio, 110
parserFriendlyName, 110
MHAIOPortAudio.cpp, 1678
dummy_interface_test, 1680
ERR_IHANDLE, 1679
ERR_SUCCESS, 1679
ERR_USER, 1679
IODestroy, 1680, 1682
IOInit, 1679, 1681
IOPrepare, 1680, 1681
IORelease, 1680, 1681
IOSetVar, 1680, 1682
IOStart, 1680, 1681
IOStop, 1680, 1681
IOStrError, 1680, 1682
MAX_USER_ERR, 1679
portaudio_callback, 1681, 1682
user_err_msg, 1682
MHAIOPortAudio::device_info_t, 956
defaultHighInputLatency, 958
defaultHighOutputLatency, 958
defaultLowInputLatency, 958
defaultLowOutputLatency, 958
defaultSampleRate, 958
device_info_t, 956
fill_info, 957
hostApi, 957
maxInputChannels, 957
maxOutputChannels, 957
name, 957
numDevices, 957
structVersion, 957
MHAIOPortAudio::io_portaudio_t, 959
~io_portaudio_t, 960
cmd_prepare, 961
cmd_release, 962
cmd_start, 961
cmd_stop, 961
device_index_in, 964
device_index_in_updated, 961
device_index_out, 964
device_index_out_updated, 961
device_info, 962
device_name_in, 964
device_name_in_updated, 961
device_name_out, 964
device_name_out_updated, 961
fragsize, 963
io_portaudio_t, 960
nchannels_in, 963
nchannels_out, 963
patchbay, 964
portaudio_callback, 962
portaudio_stream, 964
proc_event, 963
proc_handle, 963
s_in, 962
s_out, 962
samplerate, 962
start_event, 963
start_handle, 963
stop_event, 963
stop_handle, 963
stream_info, 962
suggestedInputLatency, 964
suggestedOutputLatency, 964
MHAIOPortAudio::stream_info_t, 965
fill_info, 966
paInputLatency, 966
paOutputLatency, 966
paSampleRate, 966
stream_info_t, 965
MHAIOTCP.cpp, 1683
copy_error, 1686
dummy_interface_test, 1686
ERR_IHANDLE, 1684
ERR_SUCCESS, 1684
ERR_USER, 1684
IODestroy, 1686, 1688
IOInit, 1685, 1687

© 2005-2021 HörTech gGmbH, Oldenburg

IOPrepare, 1685, 1687
IORelease, 1686, 1687
IOSetVar, 1686, 1687
IOStart, 1685, 1687
IOStop, 1686, 1687
IOStrError, 1686, 1688
MAX_TCP_PORT, 1685
MAX_TCP_PORT_STR, 1685
MAX_USER_ERR, 1684
MHA_ErrorMsg2, 1684
MHA_ErrorMsg3, 1685
MIN_TCP_PORT, 1685
MIN_TCP_PORT_STR, 1685
thread_startup_function, 1686
user_err_msg, 1688
mhaioutils, 111
to_int_clamped, 111
MHAJack, 111
get_port_capture_latency, 112
get_port_capture_latency_int, 112
get_port_playback_latency, 114
get_port_playback_latency_int, 114
io, 112
mhajack.cpp, 1688
dummy_jack_proc_cb, 1689
jack_error_handler, 1688
last_jack_err, 1689
last_jack_err_msg, 1689
make_friendly_number, 1689
mhajack.h, 1689
IO_ERROR_JACK, 1691
IO_ERROR_MHAJACKLIB, 1691
last_jack_err_msg, 1691
MAX_USER_ERR, 1691
MHAJACK_FW_STARTED, 1690
MHAJACK_STARTING, 1691
MHAJACK_STOPPED, 1691
MHAJack::client_avg_t, 967
b_ready, 971
b_stopped, 970
client_avg_t, 968
frag_out, 970
io, 968
IOStoppedEvent, 969
n, 970
name, 970
nrep, 970
pos, 970
proc, 969
sn_in, 970
sn_out, 970
MHAJack::client_noncont_t, 971
b_stopped, 973
client_noncont_t, 972
frag_out, 974
io, 972
IOStoppedEvent, 973
name, 974
pos, 973
proc, 972, 973
sn_in, 973
sn_out, 973
MHAJack::client_t, 974
b_prepared, 983
client_t, 976
connect_input, 978
connect_output, 978
fail_on_async_jackerror, 984
flags, 983
fragsize, 981
get_cpu_load, 980
get_fragsize, 978
get_my_input_ports, 979
get_my_output_ports, 979
get_ports, 979
get_srate, 978
get_xruns, 978
get_xruns_reset, 979
inch, 983
input_portnames, 984
internal_start, 980
internal_stop, 980
is_prepared, 980
jack_proc_cb, 981
jack_xrun_cb, 981
jc, 983
jstate_prev, 983
nchannels_in, 982
nchannels_out, 982
num_xruns, 981
outch, 983
output_portnames, 984
prepare, 977
prepare_impl, 980
proc_event, 982
proc_handle, 982
release, 978
s_in, 983
s_out, 983
samplerate, 982
set_input_portnames, 979
set_output_portnames, 979

© 2005-2021 HörTech gGmbH, Oldenburg

set_use_jack_transport, 980
start, 978
start_event, 982
start_handle, 982
stop, 978
stop_event, 982
stop_handle, 982
stopped, 981
str_error, 979
use_jack_transport, 983
MHAJack::port_t, 984
~port_t, 986
connect_to, 987
dir_t, 985
dir_type, 987
get_short_name, 987
input, 985
iob, 988
jc, 988
mute, 987
output, 985
port, 987
port_t, 985, 986
read, 986
write, 986
MHAJACK_FW_STARTED
mhajack.h, 1690
MHAJACK_STARTING
mhajack.h, 1691
MHAJACK_STOPPED
mhajack.h, 1691
MHAKernel, 114
algo_comm_safe_cast, 115
MHAKernel::algo_comm_class_t, 988
~algo_comm_class_t, 990
ac, 993
algo_comm_class_t, 989
algo_comm_id_string, 993
algo_comm_id_string_len, 993
get_c_handle, 990
get_entries, 992
get_error, 992
get_var, 991
get_var_double, 992
get_var_float, 992
get_var_int, 991
insert_var, 990
insert_var_double, 991
insert_var_float, 990
insert_var_int, 990
insert_var_vfloat, 990
is_var, 991
local_get_entries, 993
local_get_var, 993
local_insert_var, 992
local_is_var, 993
local_remove_ref, 992
local_remove_var, 992
remove_ref, 991
remove_var, 991
size, 993
vars, 994
MHAKernel::comm_var_map_t, 994
has_key, 994
mhamain
mha.cpp, 1576
mhamain.cpp, 1692
mhamain.cpp, 1691
GREETING_TEXT, 1692
HELP_TEXT, 1692
mhamain, 1692
NORELEASE_WARNING, 1692
VERSION_EXTENSION, 1692
MHAMultiSrc, 115
MHAMultiSrc::base_t, 995
ac, 996
base_t, 996
select_source, 996
MHAMultiSrc::channel_t, 997
channel, 997
name, 997
MHAMultiSrc::channels_t, 997
channels_t, 997
MHAMultiSrc::spectrum_t, 998
spectrum_t, 999
update, 999
MHAMultiSrc::waveform_t, 1000
update, 1001
waveform_t, 1000
MHAOvlFilter, 115
scale_fun_t, 116
MHAOvlFilter::band_descriptor_t, 1001
cf, 1002
cf_h, 1002
cf_l, 1001
ef_h, 1002
ef_l, 1002
high_side_flat, 1002
low_side_flat, 1002
MHAOvlFilter::barkscale, 116
vbark, 117
vfreq, 117

© 2005-2021 HörTech gGmbH, Oldenburg

MHAOvlFilter::barkscale::bark2hz_t, 1003
~bark2hz_t, 1003
bark2hz_t, 1003
MHAOvlFilter::barkscale::hz2bark_t, 1004
~hz2bark_t, 1004
hz2bark_t, 1004
MHAOvlFilter::fftfb_ac_info_t, 1005
bwv, 1006
cfv, 1005
cLTASS, 1006
efv, 1006
fftfb_ac_info_t, 1005
insert, 1005
MHAOvlFilter::fftfb_t, 1006
~fftfb_t, 1008
apply_gains, 1008
bin1, 1008
bin2, 1009
fftfb_t, 1007
fftlen, 1010
get_fbpower, 1008
get_fbpower_db, 1008
get_fftlen, 1009
get_ltass_gain_db, 1008
samplingrate, 1010
shape, 1010
vbin1, 1009
vbin2, 1009
w, 1009
MHAOvlFilter::fftfb_vars_t, 1010
cf, 1013
cLTASS, 1014
ef, 1014
f, 1013
fail_on_nonmonotonic, 1013
fail_on_unique_bins, 1013
fftfb_vars_t, 1012
flag_allow_empty_bands, 1013
fscale, 1012
ftype, 1012
normalize, 1013
ovltype, 1012
plateau, 1012
shapes, 1014
MHAOvlFilter::FreqScaleFun, 117
hz2bark, 118
hz2bark_analytic, 119
hz2erb, 119
hz2erb_glasberg1990, 119
hz2hz, 118
hz2khz, 118
hz2log, 119
hz2octave, 118
hz2third_octave, 118
inv_scale, 120
MHAOvlFilter::fscale_bw_t, 1014
bw, 1016
bw_hz, 1016
fscale_bw_t, 1015
get_bw_hz, 1015
update_hz, 1015
updater, 1016
MHAOvlFilter::fscale_t, 1016
f, 1018
f_hz, 1018
fscale_t, 1017
get_f_hz, 1017
unit, 1017
update_hz, 1017
updater, 1018
MHAOvlFilter::fspacing_t, 1018
bands, 1021
cf2bands, 1020
ef2bands, 1020
equidist2bands, 1020
fail_on_nonmonotonic_cf, 1020
fail_on_unique_fftbins, 1020
fs_, 1021
fspacing_t, 1019
get_cf_fftbin, 1019
get_cf_hz, 1020
get_ef_hz, 1020
nbands, 1020
nfft_, 1021
symmetry_scale, 1021
MHAOvlFilter::overlap_save_filterbank_analytic_t, 1022
filter_analytic, 1023
imagfb, 1023
overlap_save_filterbank_analytic_t, 1022
MHAOvlFilter::overlap_save_filterbank_t, 1023
channelconfig_out_, 1025
get_channelconfig, 1025
overlap_save_filterbank_t, 1025
MHAOvlFilter::overlap_save_filterbank_t::vars_t, 1026
fftlen, 1027
irswnd, 1027
phasemodel, 1027
vars_t, 1026
MHAOvlFilter::scale_var_t, 1028

© 2005-2021 HörTech gGmbH, Oldenburg

add_fun, 1029
funs, 1030
get_fun, 1029
get_name, 1029
hz2unit, 1029
names, 1030
scale_var_t, 1029
unit2hz, 1029
MHAOvlFilter::ShapeFun, 120
expflt, 122
gauss, 122
hann, 121
linear, 121
rect, 120
MHAParser, 122
all_dump, 127
all_ids, 127
c_parse_cmd_t, 126
c_parse_err_t, 128
cfg_dump, 126
cfg_dump_short, 126
commentate, 126
entry_map_t, 126
envreplace, 127
get_precision, 126
mon_dump, 127
opact_map_t, 125
opact_t, 125
query_map_t, 126
query_t, 125
strreplace, 127
trim, 126
MHAParser::base_t, 1030
~base_t, 1035
activate_query, 1041
add_parent_on_insert, 1041
add_replace_pair, 1041
base_t, 1034, 1035
data_is_initialized, 1043
fullname, 1041
help, 1043
id_str, 1043
nested_lock, 1043
notify, 1041
op_query, 1037
op_setval, 1037
op_subparse, 1036
operator=, 1035
operators, 1043
oplist, 1041
parent, 1043
parse, 1035, 1036
prereadaccess, 1042
queries, 1042
query_addsubst, 1040
query_cmds, 1040
query_dump, 1037
query_entries, 1037
query_help, 1040
query_id, 1039
query_listids, 1039
query_perm, 1037
query_range, 1038
query_readfile, 1038
query_savefile, 1039
query_savefile_compact, 1039
query_savemons, 1039
query_subst, 1040
query_type, 1038
query_val, 1038
query_version, 1039
readaccess, 1042
repl_list, 1043
repl_list_t, 1034
rm_parent_on_remove, 1041
set_help, 1040
set_node_id, 1040
thefullname, 1043
valuechanged, 1042
writeaccess, 1042
MHAParser::base_t::replace_t, 1044
a, 1045
b, 1045
get_a, 1044
get_b, 1044
replace, 1044
replace_t, 1044
MHAParser::bool_mon_t, 1045
bool_mon_t, 1046
data, 1047
query_type, 1046
query_val, 1046
MHAParser::bool_t, 1047
bool_t, 1048
data, 1049
op_setval, 1049
query_type, 1049
query_val, 1049
MHAParser::c_ifc_parser_t, 1050
~c_ifc_parser_t, 1051
c_ifc_parser_t, 1051
c_parse_cmd, 1052

© 2005-2021 HörTech gGmbH, Oldenburg

c_parse_err, 1052
libdata, 1052
liberr, 1052
modulename, 1052
op_query, 1051
op_setval, 1051
op_subparse, 1051
ret_size, 1052
retv, 1052
set_parse_cb, 1051
test_error, 1052
MHAParser::commit_t< receiver_t >, 1053
commit_t, 1054
extern_connector, 1054
MHAParser::complex_mon_t, 1055
complex_mon_t, 1056
data, 1056
query_type, 1056
query_val, 1056
MHAParser::complex_t, 1057
complex_t, 1058
data, 1059
op_setval, 1058
query_type, 1058
query_val, 1058
MHAParser::entry_t, 1059
entry, 1060
entry_t, 1059
name, 1059
MHAParser::expression_t, 1060
expression_t, 1060, 1061
lval, 1061
op, 1061
rval, 1061
MHAParser::float_mon_t, 1061
data, 1063
float_mon_t, 1062
query_type, 1063
query_val, 1062
MHAParser::float_t, 1063
data, 1066
float_t, 1065
op_setval, 1065
query_type, 1065
query_val, 1066
MHAParser::int_mon_t, 1066
data, 1068
int_mon_t, 1067
query_type, 1068
query_val, 1068
MHAParser::int_t, 1069
data, 1071
int_t, 1070
op_setval, 1070
query_type, 1071
query_val, 1071
MHAParser::keyword_list_t, 1071
add_entry, 1074
empty_string, 1075
entries, 1075
get_entries, 1074
get_index, 1074
get_value, 1073
index, 1074
keyword_list_t, 1073
set_entries, 1073
set_index, 1074
set_value, 1073
size_t, 1072
validate, 1074
MHAParser::kw_t, 1075
data, 1078
isval, 1077
kw_t, 1076, 1077
op_setval, 1077
query_range, 1078
query_type, 1078
query_val, 1078
set_range, 1077
validate, 1077
MHAParser::mcomplex_mon_t, 1079
data, 1080
mcomplex_mon_t, 1080
query_type, 1080
query_val, 1080
MHAParser::mcomplex_t, 1081
data, 1083
mcomplex_t, 1082
op_setval, 1082
query_type, 1082
query_val, 1082
MHAParser::mfloat_mon_t, 1083
data, 1085
mfloat_mon_t, 1084
query_type, 1084
query_val, 1084
MHAParser::mfloat_t, 1085
data, 1087
mfloat_t, 1086
op_setval, 1087
query_type, 1087
query_val, 1087

© 2005-2021 HörTech gGmbH, Oldenburg

MHAParser::mhaconfig_mon_t, 1088
channels, 1089
domain, 1089
fftlen, 1090
fragsize, 1089
mhaconfig_mon_t, 1089
srate, 1090
update, 1089
wndlen, 1089
MHAParser::mhapluginloader_t, 1090
~mhapluginloader_t, 1092
ac_, 1094
bookkeeping, 1094
cf_in_, 1094
cf_out_, 1094
connector, 1094
get_cfin, 1093
get_cfout, 1093
get_last_name, 1093
last_name, 1094
load_plug, 1093
mhapluginloader_t, 1092
parent_, 1093
plug, 1093
plugname, 1094
plugname_name_, 1094
prefix_, 1094
prepare, 1092
process, 1092, 1093
release, 1092
MHAParser::mint_mon_t, 1095
data, 1097
mint_mon_t, 1096
query_type, 1096
query_val, 1096
MHAParser::mint_t, 1097
data, 1099
mint_t, 1098
op_setval, 1099
query_type, 1099
query_val, 1099
MHAParser::monitor_t, 1100
monitor_t, 1100, 1101
op_query, 1101
operator=, 1101
query_dump, 1101
query_perm, 1101
MHAParser::parser_t, 1102
~parser_t, 1104
entries, 1108
force_remove_item, 1105
has_entry, 1108
id_string, 1108
insert_item, 1104
last_errormsg, 1108
op_query, 1106
op_setval, 1106
op_subparse, 1105
parser_t, 1104
query_dump, 1106
query_entries, 1106
query_listids, 1107
query_readfile, 1106
query_savefile, 1107
query_savefile_compact, 1107
query_savemons, 1107
query_type, 1106
query_val, 1107
remove_item, 1104, 1105
set_id_string, 1107
MHAParser::range_var_t, 1108
check_low, 1112
check_range, 1112
check_up, 1112
low_incl, 1112
low_limit, 1112
query_range, 1110
range_var_t, 1110
set_range, 1110
up_incl, 1112
up_limit, 1112
validate, 1110, 1111
MHAParser::StrCnv, 128
bracket_balance, 130
num_brackets, 129
str2val, 130, 131
str2val< mha_real_t >, 131
val2str, 132–134
MHAParser::string_mon_t, 1113
data, 1115
query_type, 1114
query_val, 1114
string_mon_t, 1114
MHAParser::string_t, 1115
data, 1117
op_setval, 1117
query_type, 1117
query_val, 1117
string_t, 1116
MHAParser::variable_t, 1118
locked, 1120
op_setval, 1119

© 2005-2021 HörTech gGmbH, Oldenburg

query_perm, 1119
setlock, 1119
variable_t, 1119
MHAParser::vcomplex_mon_t, 1120
data, 1122
query_type, 1121
query_val, 1121
vcomplex_mon_t, 1121
MHAParser::vcomplex_t, 1122
data, 1124
op_setval, 1123
query_type, 1124
query_val, 1124
vcomplex_t, 1123
MHAParser::vfloat_mon_t, 1125
data, 1126
query_type, 1126
query_val, 1126
vfloat_mon_t, 1126
MHAParser::vfloat_t, 1127
data, 1129
op_setval, 1129
query_type, 1129
query_val, 1129
vfloat_t, 1128
MHAParser::vint_mon_t, 1130
data, 1131
query_type, 1131
query_val, 1131
vint_mon_t, 1131
MHAParser::vint_t, 1132
data, 1134
op_setval, 1133
query_type, 1133
query_val, 1133
vint_t, 1133
MHAParser::vstring_mon_t, 1134
data, 1136
query_type, 1136
query_val, 1135
vstring_mon_t, 1135
MHAParser::vstring_t, 1136
data, 1138
op_setval, 1137
query_type, 1137
query_val, 1138
vstring_t, 1137
MHAParser::window_t, 1138
get_type, 1141
get_window, 1140, 1141
setlock, 1141
user, 1142
window_t, 1140
wnd_bartlett, 1140
wnd_blackman, 1140
wnd_hamming, 1140
wnd_hann, 1140
wnd_rect, 1140
wnd_user, 1140
wtype, 1142
wtype_t, 1140
MHAPLATFORM
mha_parser.cpp, 1615
mhaplug_cfg_t, 1142
~mhaplug_cfg_t, 1143
mhaplug_cfg_t, 1142
MHAPlugin, 134
MHAPlugin::cfg_node_t< runtime_cfg_t >, 1143
~cfg_node_t, 1144
cfg_node_t, 1144
data, 1145
next, 1144
not_in_use, 1145
MHAPlugin::config_t< runtime_cfg_t >, 1146
~config_t, 1148
cfg, 1150
cfg_node_current, 1151
cfg_root, 1151
cleanup_unused_cfg, 1150
config_t, 1148
peek_config, 1149
poll_config, 1149
push_config, 1150
remove_all_cfg, 1150
MHAPlugin::plugin_t< runtime_cfg_t >, 1151
~plugin_t, 1153
ac, 1156
input_cfg, 1155
input_cfg_, 1156
is_prepared, 1155
is_prepared_, 1156
mhaconfig_in, 1157
mhaconfig_out, 1157
output_cfg, 1155
output_cfg_, 1156
plugin_t, 1153
prepare, 1153
prepare_, 1155
release, 1154
release_, 1155
tftype, 1156

© 2005-2021 HörTech gGmbH, Oldenburg

MHAPLUGIN_CALLBACKS
mha_plugin.hh, 1625
MHAPLUGIN_CALLBACKS_PREFIX
mha_plugin.hh, 1624
MHAPLUGIN_DOCUMENTATION
mha_plugin.hh, 1626
MHAPLUGIN_DOCUMENTATION_PREFIX
mha_plugin.hh, 1625
MHAPLUGIN_INIT_CALLBACKS
mha_plugin.hh, 1625
MHAPLUGIN_INIT_CALLBACKS_PREFIX
mha_plugin.hh, 1624
MHAPLUGIN_OVERLOAD_OUTDOMAIN
altconfig.hh, 1534
altplugs.cpp, 1535
matlab_wrapper.hh, 1575
mha_generic_chain.h, 1602
split.cpp, 1710
wave2spec.hh, 1713
MHAPLUGIN_PROC_CALLBACK
mha_plugin.hh, 1625
MHAPLUGIN_PROC_CALLBACK_PREFIX
mha_plugin.hh, 1624
MHAPlugin_Resampling, 134
MHAPlugin_Resampling::resampling_if_t, 1157
algo, 1160
fragsize, 1159
irslen_inner2outer, 1159
irslen_outer2inner, 1159
nyquist_ratio, 1159
plugloader, 1159
prepare, 1158
process, 1158
release, 1159
resampling_if_t, 1158
srate, 1159
MHAPlugin_Resampling::resampling_t, 1160
inner2outer_resampling, 1162
inner_fragsize, 1161
inner_signal, 1162
inner_srate, 1161
nchannels_in, 1161
nchannels_out, 1162
outer2inner_resampling, 1162
outer_fragsize, 1161
outer_srate, 1161
output_signal, 1162
plugloader, 1162
process, 1161
resampling_t, 1160
MHAPLUGIN_SETCPP_CALLBACK_PREFIX
mha_plugin.hh, 1624
MHAPlugin_Split, 135
INVALID_THREAD_PRIORITY, 136
MHAPlugin_Split::domain_handler_t, 1163
~domain_handler_t, 1165
deallocate_domains, 1166
domain_handler_t, 1164
get_signal, 1167
operator=, 1165
process, 1168
processor, 1169
put_signal, 1166
set_input_domain, 1165
set_output_domain, 1165
spec_in, 1169
spec_out, 1169
wave_in, 1168
wave_out, 1168
MHAPlugin_Split::dummy_threads_t, 1169
catch_thread, 1171
dummy_threads_t, 1170
kick_thread, 1170
MHAPlugin_Split::posix_threads_t, 1171
~posix_threads_t, 1173
attr, 1175
catch_condition, 1175
catch_thread, 1173
current_thread_priority, 1174
current_thread_scheduler, 1174
kick_condition, 1174
kick_thread, 1173
kicked, 1175
main, 1174
mutex, 1174
posix_threads_t, 1173
priority, 1175
processing_done, 1175
scheduler, 1175
termination_request, 1176
thread, 1175
thread_start, 1174
MHAPlugin_Split::split_t, 1176
~split_t, 1178
algos, 1181
chains, 1182
channels, 1181
clear_chains, 1179
collect_result, 1180
copy_output_spec, 1179
copy_output_wave, 1179

© 2005-2021 HörTech gGmbH, Oldenburg

delay, 1182
framework_thread_priority, 1182
framework_thread_scheduler, 1181
patchbay, 1180
prepare_, 1179
process, 1179
release_, 1179
signal_out, 1180
spec_out, 1182
split_t, 1178
thread_platform, 1181
trigger_processing, 1180
update, 1179
wave_out, 1182
worker_thread_priority, 1181
worker_thread_scheduler, 1181
MHAPlugin_Split::splitted_part_t, 1183
~splitted_part_t, 1185
collect_result, 1187
domain, 1187
operator=, 1185
parse, 1186
plug, 1187
prepare, 1185
release, 1186
splitted_part_t, 1184, 1185
thread, 1188
trigger_processing, 1186
MHAPlugin_Split::thread_platform_t, 1188
~thread_platform_t, 1190
catch_thread, 1190
kick_thread, 1190
operator=, 1190
processor, 1191
thread_platform_t, 1189
MHAPlugin_Split::uni_processor_t, 1191
~uni_processor_t, 1192
process, 1192
MHAPluginCategory_t
mha.hh, 1584
MHAPluginDocumentation_t
mha.hh, 1584
mhapluginloader.cpp, 1693
mhapluginloader.h, 1693
mhapluginloader_t
MHAParser::mhapluginloader_t, 1092
PluginLoader::mhapluginloader_t, 1370
MHAPrepare_cb
PluginLoader::mhapluginloader_t, 1374
MHAPrepare_t
mha.hh, 1582
MHAProc_spec2spec_cb
PluginLoader::mhapluginloader_t, 1375
MHAProc_spec2spec_t
mha.hh, 1583
MHAProc_spec2wave_cb
PluginLoader::mhapluginloader_t, 1375
MHAProc_spec2wave_t
mha.hh, 1583
MHAProc_wave2spec_cb
PluginLoader::mhapluginloader_t, 1375
MHAProc_wave2spec_t
mha.hh, 1583
MHAProc_wave2wave_cb
PluginLoader::mhapluginloader_t, 1375
MHAProc_wave2wave_t
mha.hh, 1583
MHARelease_cb
PluginLoader::mhapluginloader_t, 1374
MHARelease_t
mha.hh, 1583
mhaserver_t, 1193
~mhaserver_t, 1195
acceptor_started, 1195
ack_fail, 1196
ack_ok, 1196
announce_port, 1197
b_interactive, 1197
logfile, 1197
logstring, 1196
mhaserver_t, 1194
on_received_line, 1195
pid_mon, 1197
port, 1197
run, 1196
send_port_announcement, 1195
set_announce_port, 1195
start_stdin_thread, 1195
tcpserver, 1196
mhaserver_t::tcp_server_t, 1197
mha, 1199
on_received_line, 1198
tcp_server_t, 1198
MHASet_cb
PluginLoader::mhapluginloader_t, 1375
MHASet_t
mha.hh, 1583
MHASetcpp_cb
PluginLoader::mhapluginloader_t, 1375
MHASetcpp_t
mha.hh, 1583
MHASignal, 136

© 2005-2021 HörTech gGmbH, Oldenburg

copy_permuted, 149
db2lin, 140
db2sq, 141
dbspl2pa, 142
dbspl2pa2, 143
for_each, 139
kth_smallest, 144
limit, 144
lin2db, 139, 140
mean, 147
median, 145
pa22dbspl, 142
pa2dbspl, 141, 142
quantile, 147
saveas_mat4, 148, 149
scale, 144
sec2smp, 144
signal_counter, 149
smp2sec, 143
sq2db, 140
MHASignal::async_rmslevel_t, 1199
async_rmslevel_t, 1200
filled, 1202
peaklevel, 1201
pos, 1201
process, 1201
rmslevel, 1201
MHASignal::delay_spec_t, 1202
~delay_spec_t, 1202
buffer, 1203
delay, 1203
delay_spec_t, 1202
pos, 1203
process, 1203
MHASignal::delay_t, 1203
~delay_t, 1204
buffer, 1205
channels, 1205
delay_t, 1204
delays, 1205
inspect, 1205
pos, 1205
process, 1205
MHASignal::delay_wave_t, 1206
~delay_wave_t, 1206
buffer, 1207
delay, 1207
delay_wave_t, 1206
pos, 1207
process, 1207
MHASignal::doublebuffer_t, 1207
~doublebuffer_t, 1209
ch, 1211
doublebuffer_t, 1208
inner_in, 1210
inner_out, 1210
inner_process, 1209
k_inner, 1210
k_outer, 1211
min, 1210
outer_out, 1210
outer_process, 1209
this_outer_out, 1210
MHASignal::fft_t, 1211
~fft_t, 1212
backward, 1213
backward_scale, 1214
buf_in, 1215
buf_out, 1215
fft_t, 1212
fftw_plan_fft, 1215
fftw_plan_ifft, 1215
fftw_plan_spec2wave, 1215
fftw_plan_wave2spec, 1215
forward, 1213
forward_scale, 1213
n_im, 1214
n_re, 1214
nfft, 1214
scale, 1215
sort_fftw2spec, 1214
sort_spec2fftw, 1214
spec2wave, 1212, 1213
spec2wave_scale, 1213
wave2spec, 1212
wave2spec_scale, 1213
MHASignal::hilbert_fftw_t, 1216
~hilbert_fftw_t, 1216
buf_c_in, 1217
buf_c_out, 1217
buf_r_in, 1217
buf_r_out, 1217
hilbert, 1217
hilbert_fftw_t, 1216
n, 1217
p1, 1217
p2, 1217
sc, 1218
MHASignal::hilbert_t, 1218
~hilbert_t, 1219
h, 1219
hilbert_t, 1219

© 2005-2021 HörTech gGmbH, Oldenburg

operator(), 1219
MHASignal::loop_wavefragment_t, 1220
add, 1222
b_loop, 1225
get_mapping, 1223
input, 1222
intern_level, 1225
is_playback_active, 1224
level_mode_t, 1221
locate_end, 1224
loop_wavefragment_t, 1222
mute, 1222
peak, 1222
playback, 1223, 1224
playback_channels, 1224
playback_mode_t, 1222
pos, 1225
relative, 1222
replace, 1222
rewind, 1224
rms, 1222
rms_limit40, 1222
set_level_db, 1224
set_level_lin, 1224
MHASignal::matrix_t, 1225
~matrix_t, 1230
cdata, 1236
complex_ofs, 1236
dimension, 1230
get_cdata, 1236
get_comm_var, 1230
get_index, 1235
get_nelements, 1231
get_nreals, 1235
get_rdata, 1235
imag, 1232–1234
is_same_size, 1231
iscomplex, 1231
matrix_t, 1227, 1229
nelements, 1236
numbytes, 1235
operator(), 1232–1234
operator=, 1230
rdata, 1236
real, 1231–1234
size, 1231
write, 1235
MHASignal::minphase_t, 1237
minphase_t, 1237
operator(), 1238
phase, 1238
MHASignal::quantizer_t, 1238
downscale, 1240
limit, 1240
operator(), 1239
quantizer_t, 1239
up_limit, 1240
upscale, 1240
MHASignal::ringbuffer_t, 1240
contained_frames, 1242
discard, 1243
next_read_frame_index, 1244
next_write_frame_index, 1244
ringbuffer_t, 1241
value, 1242
write, 1243
MHASignal::schroeder_t, 1244
down, 1246
groupdelay_t, 1245
identity, 1247
log_down, 1248
log_up, 1247
schroeder_t, 1246, 1247
sign_t, 1246
up, 1246
MHASignal::spectrum_t, 1248
~spectrum_t, 1250
copy, 1251
copy_channel, 1252
export_to, 1252
operator(), 1250
operator[], 1251
scale, 1252
scale_channel, 1254
spectrum_t, 1249, 1250
value, 1251
MHASignal::stat_t, 1254
mean, 1255
mean_std, 1255
n, 1256
push, 1255, 1256
stat_t, 1255
sum, 1256
sum2, 1256
MHASignal::subsample_delay_t, 1256
last_complex_bin, 1258
phase_gains, 1258
process, 1257, 1258
subsample_delay_t, 1257
MHASignal::uint_vector_t, 1259
~uint_vector_t, 1260
data, 1262

© 2005-2021 HörTech gGmbH, Oldenburg

get_length, 1261
getdata, 1262
length, 1262
numbytes, 1262
operator=, 1261
operator==, 1261
operator[], 1261
uint_vector_t, 1260
write, 1262
MHASignal::waveform_t, 1263
~waveform_t, 1266
assign, 1270, 1271
assign_channel, 1271
assign_frame, 1271
copy, 1271, 1272
copy_channel, 1272
copy_from_at, 1272
export_to, 1273
flatten, 1266
get_size, 1275
limit, 1273
operator std::vector< mha_real_t >, 1266
operator(), 1267, 1268
operator=, 1267
operator[], 1267
power, 1273
powspec, 1274
scale, 1274
scale_channel, 1275
scale_frame, 1275
sum, 1269
sum_channel, 1270
sumsqr, 1270
value, 1267, 1268
waveform_t, 1265, 1266
MHASndFile, 150
mhasndfile.cpp, 1693
validator_channels, 1694
validator_length, 1694
write_wave, 1693
mhasndfile.h, 1694
write_wave, 1694
MHASndFile::sf_t, 1276
~sf_t, 1276
sf, 1277
sf_t, 1276
MHASndFile::sf_wave_t, 1277
sf_wave_t, 1278
mhastrdomain
PluginLoader, 157
MHAStrError_cb
PluginLoader::mhapluginloader_t, 1375
MHAStrError_t
mha.hh, 1584
MHATableLookup, 150
MHATableLookup::linear_table_t, 1278
~linear_table_t, 1280
add_entry, 1281
clear, 1282
interp, 1280
len, 1282
linear_table_t, 1280
lookup, 1280
prepare, 1281
scalefac, 1283
set_xmax, 1281
set_xmin, 1281
vec_y, 1282
vy, 1282
xmax, 1283
xmin, 1282
MHATableLookup::table_t, 1283
~table_t, 1284
clear, 1284
interp, 1284
lookup, 1284
table_t, 1284
MHATableLookup::xy_table_t, 1285
add_entry, 1287
clear, 1288
get_xlimits, 1289
interp, 1287
lookup, 1286
mXY, 1289
set_xfun, 1288
set_xyfun, 1289
set_yfun, 1288
xfun, 1289
xy_table_t, 1286
xyfun, 1289
yfun, 1289
MHAUtils, 150
is_denormal, 151, 152
is_multiple_of, 151
is_multiple_of_by_power_of_two, 151
is_power_of_two, 151
remove, 151
spl2hl, 153
strip, 151
MHAWindow, 153
bartlett, 154
blackman, 155

© 2005-2021 HörTech gGmbH, Oldenburg

hamming, 155
hanning, 155
rect, 154
MHAWindow::bartlett_t, 1290
bartlett_t, 1291
MHAWindow::base_t, 1291
base_t, 1292
operator(), 1292
ramp_begin, 1293
ramp_end, 1293
MHAWindow::blackman_t, 1293
blackman_t, 1294
MHAWindow::fun_t, 1295
fun_t, 1295
MHAWindow::hamming_t, 1296
hamming_t, 1297
MHAWindow::hanning_t, 1297
hanning_t, 1298
MHAWindow::rect_t, 1299
rect_t, 1300
MHAWindow::user_t, 1300
user_t, 1301
mic_azimuth_degrees_vec
rohBeam::rohBeam, 1404
MIN
mha_defs.h, 1589
min
MHASignal::doublebuffer_t, 1210
spec2wave.cpp, 1704
Vector and matrix processing toolbox, 56
min_sleep_time
dropgen_t, 439
MIN_TCP_PORT
MHAIOAsterisk.cpp, 1658
MHAIOTCP.cpp, 1685
MIN_TCP_PORT_STR
MHAIOAsterisk.cpp, 1658
MHAIOTCP.cpp, 1685
minimum_fill_count
mha_drifter_fifo_t< T >, 761
minlen
plingploing::if_t, 1343
minlen_
plingploing::plingploing_t, 1346
minLim
rohBeam::rohConfig, 1412
minphase
MHAFilter::smoothspec_t, 934
minphase_t
MHASignal::minphase_t, 1237
mint_mon_t
MHAParser::mint_mon_t, 1096
mint_t
MHAParser::mint_t, 1098
minw_
wavwriter_t, 1508
minwrite
plugins::hoertech::acrec::acrec_t, 1381
wavrec_t, 1505
mismatch
level_matching::channel_pair, 651
mix
sine_cfg_t, 1434
mixer
matrixmixer::matmix_t, 716
mixw_ref
fshift_hilbert::hilbert_shifter_t, 521
mixw_shift
fshift_hilbert::hilbert_shifter_t, 521
mode
ac2osc_t, 183
addsndfile::addsndfile_if_t, 253
audiometerbackend::audiometer_if_t, 317
levelmeter_t, 660
MHA_TCP::OS_EVENT_TYPE, 815
noise_t, 1320
sine_t, 1437
smoothgains_bridge::overlapadd_if_t, 1455
modified
dc::dc_vars_t, 390
dc_simple::dc_if_t, 396
modulename
dynamiclib_t, 450
MHAParser::c_ifc_parser_t, 1052
mon
acmon::ac_monitor_t, 205
mon_complex
acmon::ac_monitor_t, 206
mon_dump
MHAParser, 127
mon_g
dc_simple::dc_if_t, 396
dc_simple::dc_t, 401
mon_l
dc_simple::dc_if_t, 396
dc_simple::dc_t, 401
mon_mat
acmon::ac_monitor_t, 206
mon_mat_complex
acmon::ac_monitor_t, 206
mon_t

© 2005-2021 HörTech gGmbH, Oldenburg

rmslevel::mon_t, 1391
monitor variable, 4
monitor_t
MHAParser::monitor_t, 1100, 1101
monitors
matlab_wrapper::matlab_wrapper_t, 703
rmslevel::rmslevel_t, 1397
mpo
DynComp::dc_afterburn_vars_t, 457
mpo_inv
DynComp::dc_afterburn_rt_t, 452
msg
MHA_Error, 766
mu
gsc_adaptive_stage::gsc_adaptive_stage, 541
gsc_adaptive_stage::gsc_adaptive_stage_if, 548
MHAFilter::adapt_filter_param_t, 864
MHAFilter::adapt_filter_t, 868
mu_beta
adm_if_t, 274
mul4f
gtfb_simd.cpp, 1563
multibandcompressor, 155
multibandcompressor.cpp, 1695
multibandcompressor::fftfb_plug_t, 1302
bwv, 1303
cfv, 1303
efv, 1303
fftfb_plug_t, 1302
insert, 1303
multibandcompressor::interface_t, 1304
algo, 1306
burn, 1306
interface_t, 1305
num_channels, 1306
patchbay, 1306
plug, 1306
plug_sigs, 1306
prepare, 1305
process, 1305
release, 1305
update_cfg, 1305
multibandcompressor::plugin_signals_t, 1307
apply_gains, 1307
gain, 1308
plug_level, 1308
plug_output, 1308
plugin_signals_t, 1307
update_levels, 1307
mute
MHAJack::port_t, 987
MHASignal::loop_wavefragment_t, 1222
mutex
mha_fifo_posix_threads_t, 776
MHAPlugin_Split::posix_threads_t, 1174
MXCSR_DAZ
gtfb_simd.cpp, 1563
MXCSR_FTZ
gtfb_simd.cpp, 1563
mXY
MHATableLookup::xy_table_t, 1289
mylogf
dc_afterburn.cpp, 1544

N
lpc_config, 678
n
MHAJack::client_avg_t, 970
MHASignal::hilbert_fftw_t, 1217
MHASignal::stat_t, 1256
n_channels
mha_audio_descriptor_t, 742
N_ERRNO
MHA_TCP, 102
n_freqs
mha_audio_descriptor_t, 742
n_im
MHASignal::fft_t, 1214
n_new_samples
lsl2ac::save_var_t, 693
n_no_update
adaptive_feedback_canceller, 244
nlms_t, 1312
n_no_update_
adaptive_feedback_canceller_config, 247
rt_nlms_t, 1421
n_pad1
overlapadd::overlapadd_t, 1337
n_pad2
overlapadd::overlapadd_t, 1338
n_re
MHASignal::fft_t, 1214
n_samples
mha_audio_descriptor_t, 742
n_zero
overlapadd::overlapadd_t, 1337
name
ac2lsl::type_info, 179
ac2wave_if_t, 187
ac2wave_t, 190
acmon::ac_monitor_t, 205

© 2005-2021 HörTech gGmbH, Oldenburg

acsave::save_var_t, 227
fftfbpow::fftfbpow_interface_t, 496
lsl2ac::save_var_t, 692
MHA_AC::ac2matrix_helper_t, 722
MHA_AC::double_t, 731
MHA_AC::float_t, 733
MHA_AC::int_t, 734
MHA_AC::spectrum_t, 737
MHA_AC::waveform_t, 741
MHAIOPortAudio::device_info_t, 957
MHAJack::client_avg_t, 970
MHAJack::client_noncont_t, 974
MHAMultiSrc::channel_t, 997
MHAParser::entry_t, 1059
noise_psd_estimator::noise_psd_estimator_if_t,1314
plugindescription_t, 1359
rmslevel::rmslevel_if_t, 1394
shadowfilter_end::cfg_t, 1430
name_
AuditoryProfile::parser_t::fmap_t, 330
gtfb_simple_t, 574
osc_variable_t, 1330
name_b
lpc_bl_predictor, 667
lpc_bl_predictor_config, 670
lpc_burglattice, 673
lpc_burglattice_config, 676
name_con_AC
acConcat_wave, 201
name_d
nlms_t, 1311
name_d_
adaptive_feedback_canceller_config, 247
rt_nlms_t, 1421
name_e
adaptive_feedback_canceller, 243
nlms_t, 1311
name_e_
rt_nlms_t, 1421
name_f
adaptive_feedback_canceller, 243
lpc_bl_predictor, 667
lpc_bl_predictor_config, 669
lpc_burglattice, 673
lpc_burglattice_config, 675
nlms_t, 1312
name_kappa
lpc_bl_predictor, 667
lpc_burglattice, 673
name_km
lpc_bl_predictor_config, 669
name_lpc
adaptive_feedback_canceller, 243
name_lpc_
adaptive_feedback_canceller_config, 247
name_lpc_b
lpc_bl_predictor, 667
name_lpc_f
lpc_bl_predictor, 667
name_u
nlms_t, 1311
name_u_
rt_nlms_t, 1421
namelen
acsave::mat4head_t, 225
names
MHAOvlFilter::scale_var_t, 1030
nangle
acSteer_config, 232
steerbf_config, 1479
native_thread_platform_type
split.cpp, 1710
naudiochannels
dc::dc_t, 386
nbands
coherence::cohflt_t, 351
combc_t, 359
dc::dc_t, 386
dc_simple::dc_t, 400
dc_simple::level_smoother_t, 409
DynComp::gaintable_t, 461
fftfilterbank::fftfb_interface_t, 506
gtfb_simple_rt_t, 568
MHAFilter::thirdoctave_analyzer_t, 935
MHAOvlFilter::fspacing_t, 1020
nbits
calibrator_variables_t, 342
nch
dc::dc_t, 386
shadowfilter_begin::cfg_t, 1426
shadowfilter_begin::shadowfilter_begin_t, 1429
spec_fader_t, 1471
nch_out
shadowfilter_end::cfg_t, 1430
nchan
acSteer_config, 232
gsc_adaptive_stage::gsc_adaptive_stage, 541
smooth_cepstrum::smooth_cepstrum_t, 1446

© 2005-2021 HörTech gGmbH, Oldenburg

steerbf_config, 1479
nchan_block
rohBeam::rohConfig, 1409
nchannels
DynComp::gaintable_t, 461
equalize::cfg_t, 464
fftfilterbank::fftfb_interface_t, 506
lsl2ac::lsl2ac_t, 685
lsl2ac::save_var_t, 692
MHAFilter::adapt_filter_state_t, 865
MHAFilter::adapt_filter_t, 868
MHAFilter::iir_filter_t, 902
MHAFilter::smoothspec_t, 933
MHAFilter::thirdoctave_analyzer_t, 935
nchannels_file_in
io_file_t, 608
nchannels_in
io_file_t, 608
io_parser_t, 619
mconv::MConv, 720
MHAFilter::partitioned_convolution_t, 919
MHAIOPortAudio::io_portaudio_t, 963
MHAJack::client_t, 982
MHAPlugin_Resampling::resampling_t, 1161
nchannels_out
fw_t, 529
io_file_t, 608
io_parser_t, 619
mconv::MConv, 720
MHAFilter::partitioned_convolution_t, 919
MHAIOPortAudio::io_portaudio_t, 963
MHAJack::client_t, 982
MHAPlugin_Resampling::resampling_t, 1162
NDEBUG
rohBeam.hh, 1701
ndim
acsave::save_var_t, 227
needs_write
MHA_TCP::Connection, 810
neigh
acPooling_wave_config, 217
neighbourhood
acPooling_wave, 214
nelements
MHASignal::matrix_t, 1236
nested_lock
MHAParser::base_t, 1043
new_name
lsl2ac::save_var_t, 691
newgains
fader_if_t, 489
next
mha_rt_fifo_element_t< T >, 789
MHAPlugin::cfg_node_t< runtime_cfg_t >, 1144
next_except_str
mha_errno.c, 1591
next_message
mha_tcp::buffered_socket_t, 801
next_read_frame_index
MHASignal::ringbuffer_t, 1244
next_write_frame_index
MHASignal::ringbuffer_t, 1244
nextXpYf
rohBeam::rohConfig, 1411
nfft
MHASignal::fft_t, 1214
overlapadd::overlapadd_if_t, 1333
shadowfilter_end::cfg_t, 1430
spec2wave_t, 1470
wave2spec_if_t, 1496
nfft_
MHAOvlFilter::fspacing_t, 1021
nframes
acsave::save_var_t, 227
nfreq
acSteer_config, 232
gsc_adaptive_stage::gsc_adaptive_stage, 541
rohBeam::rohConfig, 1409
smooth_cepstrum::smooth_cepstrum_t, 1445
steerbf_config, 1479
nlms_t, 1309
algo, 1312
c, 1311
estimtype, 1311
lambda_smoothing_power, 1311
n_no_update, 1312
name_d, 1311
name_e, 1311
name_f, 1312
name_u, 1311
nlms_t, 1310
normtype, 1311
ntaps, 1311
patchbay, 1312
prepare, 1310
process, 1310
release, 1310

© 2005-2021 HörTech gGmbH, Oldenburg

rho, 1311
update, 1310
nlms_wave.cpp, 1695
ESTIM_CUR, 1696
ESTIM_PREV, 1696
ESTIMATION_TYPES, 1696
make_friendly_number_by_limiting, 1696
NORM_DEFAULT, 1696
NORM_NONE, 1696
NORM_SUM, 1696
NORMALIZATION_TYPES, 1696
nm
lpc_burglattice_config, 675
no_iter
adaptive_feedback_canceller_config, 247
rt_nlms_t, 1421
noise.cpp, 1697
noise_field_model
rohBeam::rohBeam, 1404
noise_integrate_hrtf
rohBeam::rohBeam, 1402
noise_psd_estimator, 155
noise_psd_estimator.cpp, 1697
POWSPEC_FACTOR, 1697
noise_psd_estimator::noise_psd_estimator_if_t, 1312
alphaPH1mean, 1314
alphaPSD, 1314
name, 1314
noise_psd_estimator_if_t, 1313
patchbay, 1314
prepare, 1313
process, 1313
q, 1314
update_cfg, 1314
xiOptDb, 1314
noise_psd_estimator::noise_psd_estimator_t, 1315
alphaPH1mean_, 1317
alphaPSD_, 1317
estimateDebug, 1317
frameno, 1318
GLRDebug, 1317
GLRexp, 1318
inputPow, 1316
inputSpec, 1317
insert, 1316
logGLRFact, 1318
noise_psd_estimator_t, 1315
noisePow, 1316
noisyPer, 1316
PH1Debug, 1317
PH1mean, 1316
priorFact, 1317
process, 1316
snrPost1Debug, 1317
xiOpt, 1318
noise_psd_estimator_if_t
noise_psd_estimator::noise_psd_estimator_if_t, 1313
noise_psd_estimator_t
noise_psd_estimator::noise_psd_estimator_t, 1315
noise_t, 1318
frozennoise_length, 1320
lev, 1320
mode, 1320
noise_t, 1319
patchbay, 1320
prepare, 1319
process, 1319
seed, 1320
update_cfg, 1320
noise_type_t
speechnoise_t, 1473
noiseFuncPtr
rohBeam::rohBeam, 1403
noiseModelExport
rohBeam::rohBeam, 1406
noisePow
noise_psd_estimator::noise_psd_estimator_t, 1316
smooth_cepstrum::smooth_cepstrum_t, 1447
noisePow_name
smooth_cepstrum::smooth_cepstrum_if_t, 1442
smooth_cepstrum::smooth_params, 1453
noisyPer
noise_psd_estimator::noise_psd_estimator_t, 1316
non_empty_partitions
MHAFilter::transfer_function_t, 939
MHAFilter::transfer_matrix_t, 941
nondefault_labels
altplugs_t, 304
NORELEASE_WARNING
mhamain.cpp, 1692
norm
lpc, 664
lpc_config, 677
NORM_DEFAULT

© 2005-2021 HörTech gGmbH, Oldenburg

nlms_wave.cpp, 1696
NORM_NONE
nlms_wave.cpp, 1696
norm_phase
gtfb_analyzer::gtfb_analyzer_cfg_t, 552
gtfb_analyzer::gtfb_analyzer_t, 556
gtfb_simd_cfg_t, 560
gtfb_simd_t, 564
NORM_SUM
nlms_wave.cpp, 1696
NORMALIZATION_TYPES
nlms_wave.cpp, 1696
normalize
Complex arithmetics in the openMHA, 68
MHAOvlFilter::fftfb_vars_t, 1013
normtype
nlms_t, 1311
not_in_use
MHAPlugin::cfg_node_t< runtime_cfg_t >, 1145
not_zero
dc_simple, 88
notify
MHAParser::base_t, 1041
notify_release
io_asterisk_t, 604
io_tcp_t, 642
notify_start
io_asterisk_t, 604
io_tcp_t, 642
notify_stop
io_asterisk_t, 604
io_tcp_t, 642
now_index
MHAFilter::polyphase_resampling_t, 927
npad1
spec2wave_t, 1469
wave2spec_t, 1502
npad2
spec2wave_t, 1469
wave2spec_t, 1502
nperiods
alsa_dev_par_parser_t, 290
nrefmic
acSteer, 230
acSteer_config, 232
nrep
MHAJack::client_avg_t, 970
nsamples
lsl2ac::lsl2ac_t, 685
lsl2ac::save_var_t, 692
nsteerchan
acSteer, 230
acSteer_config, 232
ntaps
adaptive_feedback_canceller, 243
adaptive_feedback_canceller_config, 246
MHAFilter::adapt_filter_state_t, 865
MHAFilter::adapt_filter_t, 868
nlms_t, 1311
rt_nlms_t, 1419
ntoh
io_asterisk_sound_t, 599
io_tcp_sound_t, 636
ntracks
shadowfilter_begin::cfg_t, 1427
shadowfilter_begin::shadowfilter_begin_t, 1429
shadowfilter_end::cfg_t, 1430
null_data
mha_drifter_fifo_t< T >, 763
num_AC
acConcat_wave, 201
num_accepted_connections
mha_tcp::server_t, 824
num_adms
adm_rtconfig_t, 277
num_bins
equalize::cfg_t, 464
num_brackets
MHAParser::StrCnv, 129
num_channels
ac_mul_t, 195
calibrator_variables_t, 343
DynComp::gaintable_t, 462
mha_spec_t, 795
mha_wave_t, 841
MHAFilter::blockprocessing_polyphase_resampling_t, 872
multibandcompressor::interface_t, 1306
plugins::hoertech::acrec::acwriter_t, 1387
NUM_ENTR_LTASS
speechnoise.cpp, 1706
NUM_ENTR_MHAORIG
speechnoise.cpp, 1706
NUM_ENTR_OLNOISE
speechnoise.cpp, 1706
num_entries
ac2lsl::save_var_base_t, 171
ac2lsl::save_var_t< mha_complex_t >, 177
ac2lsl::save_var_t< T >, 174

© 2005-2021 HörTech gGmbH, Oldenburg

comm_var_t, 360
testplugin::ac_parser_t, 1482
num_F
DynComp::gaintable_t, 462
num_frames
ac_mul_t, 195
mha_spec_t, 795
mha_wave_t, 841
num_inchannels
io_asterisk_sound_t, 600
io_tcp_sound_t, 637
num_L
DynComp::gaintable_t, 462
num_outchannels
io_asterisk_sound_t, 600
io_tcp_sound_t, 637
num_xruns
MHAJack::client_t, 981
numbytes
MHASignal::matrix_t, 1235
MHASignal::uint_vector_t, 1262
numchannels
acConcat_wave, 201
addsndfile::addsndfile_if_t, 253
numDevices
MHAIOPortAudio::device_info_t, 957
numsamples
acPooling_wave, 213
acTransform_wave, 237
numSamples_AC
acConcat_wave_config, 203
nupsample
doasvm_feature_extraction, 429
nvars
acsave::cfg_t, 223
nwnd
overlapadd::overlapadd_if_t, 1333
wave2spec_if_t, 1496
wave2spec_t, 1501
nwndshift
spec2wave_t, 1470
wave2spec_t, 1501
nyquist_ratio
MHAPlugin_Resampling::resampling_if_t, 1159

o1_ar_filter_t
MHAFilter::o1_ar_filter_t, 907
o1_lp_coeffs
MHAFilter, 106
o1flt_lowpass_t
MHAFilter::o1flt_lowpass_t, 911
o1flt_maxtrack_t
MHAFilter::o1flt_maxtrack_t, 914
o1flt_mintrack_t
MHAFilter::o1flt_mintrack_t, 916
ob
lsl2ac::save_var_t, 692
observe
MHA_TCP::Event_Watcher, 813
observed_by
MHA_TCP::Wakeup_Event, 837
observers
MHA_TCP::Wakeup_Event, 838
od
MHAFilter::adapt_filter_state_t, 866
offset
acTransform_wave_config, 239
dc::dc_t, 385
dc::dc_vars_t, 389
ola_powspec_scale
smooth_cepstrum::smooth_cepstrum_t, 1446
old_algos
mhachain::chain_base_t, 844
olnoise
speechnoise_t, 1473
on_configuration_update
double2acvar::double2acvar_t, 436
on_model_param_valuechanged
gsc_adaptive_stage::gsc_adaptive_stage_if, 548
rohBeam::rohBeam, 1403
smooth_cepstrum::smooth_cepstrum_if_t, 1440
on_prereadaccess
example3_t, 475
example4_t, 479
on_received_line
mha_tcp::server_t, 822
mhaserver_t, 1195
mhaserver_t::tcp_server_t, 1198
on_scale_ch_readaccess
example3_t, 475
example4_t, 479
on_scale_ch_valuechanged
example3_t, 475
example4_t, 479
on_scale_ch_writeaccess
example3_t, 475
example4_t, 479
on_set_algos
altconfig_t, 297

© 2005-2021 HörTech gGmbH, Oldenburg

on_set_select
altconfig_t, 297
on_writeaccess
matlab_wrapper::callback, 694
op
MHAParser::expression_t, 1061
op_query
MHAParser::base_t, 1037
MHAParser::c_ifc_parser_t, 1051
MHAParser::monitor_t, 1101
MHAParser::parser_t, 1106
op_setval
MHAParser::base_t, 1037
MHAParser::bool_t, 1049
MHAParser::c_ifc_parser_t, 1051
MHAParser::complex_t, 1058
MHAParser::float_t, 1065
MHAParser::int_t, 1070
MHAParser::kw_t, 1077
MHAParser::mcomplex_t, 1082
MHAParser::mfloat_t, 1087
MHAParser::mint_t, 1099
MHAParser::parser_t, 1106
MHAParser::string_t, 1117
MHAParser::variable_t, 1119
MHAParser::vcomplex_t, 1123
MHAParser::vfloat_t, 1129
MHAParser::vint_t, 1133
MHAParser::vstring_t, 1137
op_subparse
MHAParser::base_t, 1036
MHAParser::c_ifc_parser_t, 1051
MHAParser::parser_t, 1105
opact_map_t
MHAParser, 125
opact_t
MHAParser, 125
operator std::vector< mha_real_t >
MHASignal::waveform_t, 1266
operator!=
Complex arithmetics in the openMHA, 67
operator<
Complex arithmetics in the openMHA, 69
operator<<
mha_signal.hh, 1642
operator>>
mha_signal.hh, 1643
operator*
Complex arithmetics in the openMHA, 64, 65
operator*=
Complex arithmetics in the openMHA, 64
Vector and matrix processing toolbox, 50, 51
operator^=
Vector and matrix processing toolbox, 52
operator()
dc_simple::dc_t::line_t, 402
hanning_ramps_t, 575
MHAEvents::emitter_t, 860
MHAFilter::gamma_flt_t, 894
MHAFilter::iir_ord1_real_t, 904, 905
MHAFilter::o1_ar_filter_t, 908, 909
MHASignal::hilbert_t, 1219
MHASignal::matrix_t, 1232–1234
MHASignal::minphase_t, 1238
MHASignal::quantizer_t, 1239
MHASignal::spectrum_t, 1250
MHASignal::waveform_t, 1267, 1268
MHAWindow::base_t, 1292
operator+
Complex arithmetics in the openMHA, 62, 63
operator+=
Complex arithmetics in the openMHA, 62
Vector and matrix processing toolbox, 50, 52
operator-
Complex arithmetics in the openMHA, 63, 66
operator-=
Complex arithmetics in the openMHA, 63
Vector and matrix processing toolbox, 50
operator/
Complex arithmetics in the openMHA, 65, 66
operator/=
Complex arithmetics in the openMHA, 65, 66
Vector and matrix processing toolbox, 51, 52
operator=
equalize::cfg_t, 463
gtfb_simd_cfg_t, 559
lsl2ac::save_var_t, 689
MHA_AC::acspace2matrix_t, 727
MHA_Error, 766
mha_fifo_t< T >, 781
mha_fifo_thread_platform_t, 787
MHAFilter::filter_t, 890
MHAParser::base_t, 1035
MHAParser::monitor_t, 1101

© 2005-2021 HörTech gGmbH, Oldenburg

MHAPlugin_Split::domain_handler_t, 1165
MHAPlugin_Split::splitted_part_t, 1185
MHAPlugin_Split::thread_platform_t, 1190
MHASignal::matrix_t, 1230
MHASignal::uint_vector_t, 1261
MHASignal::waveform_t, 1267
rmslevel::mon_t, 1391
rohBeam::rohConfig, 1408
smooth_cepstrum::smooth_cepstrum_t, 1445
operator==
Complex arithmetics in the openMHA, 66
MHASignal::uint_vector_t, 1261
operator[]
MHA_AC::acspace2matrix_t, 727, 728
MHASignal::spectrum_t, 1251
MHASignal::uint_vector_t, 1261
MHASignal::waveform_t, 1267
operators
MHAParser::base_t, 1043
oplist
MHAParser::base_t, 1041
order
gtfb_analyzer::gtfb_analyzer_cfg_t, 552
gtfb_analyzer::gtfb_analyzer_t, 556
gtfb_simd_cfg_t, 559
gtfb_simd_t, 564
gtfb_simple_t, 573
lpc_config, 678
original_content
mha_stash_environment_variable_t, 797
origname
PluginLoader::config_file_splitter_t, 1364
os_event
MHA_TCP::Wakeup_Event, 838
os_event_valid
MHA_TCP::Wakeup_Event, 839
osc2ac.cpp, 1697
osc2ac_t, 1321
host, 1323
osc2ac_t, 1322
patchbay, 1323
port, 1323
prepare, 1322
process, 1322
release, 1322
setlock, 1323
size, 1323
srv, 1323
vars, 1323
osc_data
osc_variable_t, 1330
osc_server_t, 1324
~osc_server_t, 1324
ac_insert, 1325
error_h, 1325
insert_variable, 1325
is_running, 1326
lost, 1326
osc_server_t, 1324
pVars, 1325
server_start, 1325
server_stop, 1325
sync_osc2ac, 1325
osc_variable_t, 1326
ac_data, 1329
ac_insert, 1328
acname, 1329
handler, 1328, 1329
name_, 1330
osc_data, 1330
osc_variable_t, 1327
oscaddr, 1329
sync_osc2ac, 1328
oscaddr
osc_variable_t, 1329
out
adm_if_t, 273
delaysum::delaysum_wave_t, 417
out_buf
overlapadd::overlapadd_t, 1337
spec2wave_t, 1470
out_cfg
rohBeam::rohConfig, 1409
out_chunk
MHAFilter::thirdoctave_analyzer_t, 936
out_chunk_im
MHAFilter::thirdoctave_analyzer_t, 937
out_spec
shadowfilter_begin::cfg_t, 1426
shadowfilter_end::cfg_t, 1431
outbuf
MHA_TCP::Connection, 810
outch
mconv::MConv, 720
MHAJack::client_t, 983
outchannel
audiometerbackend::audiometer_if_t, 317
outchannels
combc_if_t, 357

© 2005-2021 HörTech gGmbH, Oldenburg

outer2inner_resampling
MHAPlugin_Resampling::resampling_t, 1162
outer_ac
analysepath_t, 308
outer_ac_copy
analysepath_t, 308
outer_error
mha_dblbuf_t< FIFO >, 755
outer_fragsize
MHAPlugin_Resampling::resampling_t, 1161
outer_out
MHASignal::doublebuffer_t, 1210
outer_output
dbasync_native::dbasync_t, 378
outer_process
dbasync_native::dbasync_t, 377
MHASignal::doublebuffer_t, 1209
outer_size
mha_dblbuf_t< FIFO >, 753
outer_srate
MHAPlugin_Resampling::resampling_t, 1161
outfile
plugins::hoertech::acrec::acwriter_t, 1387
output
delaysum_spec::delaysum_t, 421
gtfb_simple_rt_t, 568
io_parser_t, 620
mha_dblbuf_t< FIFO >, 753
MHAJack::port_t, 985
output_cfg
MHAPlugin::plugin_t< runtime_cfg_t >, 1155
output_cfg_
MHAPlugin::plugin_t< runtime_cfg_t >, 1156
output_channels
mha_dblbuf_t< FIFO >, 754
output_data
io_asterisk_sound_t, 600
output_domain
PluginLoader::mhapluginloader_t, 1371
output_fifo
mha_dblbuf_t< FIFO >, 754
output_partitions
MHAFilter::partitioned_convolution_t, 919
output_portnames
MHAJack::client_t, 984
output_sample_format
io_file_t, 609
wavrec_t, 1505
output_signal
MHAPlugin_Resampling::resampling_t, 1162
output_signal_spec
MHAFilter::partitioned_convolution_t, 920
output_signal_wave
MHAFilter::partitioned_convolution_t, 921
output_spec
testplugin::signal_parser_t, 1490
output_type
plugins::hoertech::acrec::acwriter_t, 1383
output_wave
testplugin::signal_parser_t, 1490
outputchannels
MHAFilter::fftfilterbank_t, 886
outSpec
rohBeam::rohConfig, 1410
steerbf_config, 1479
overlap_save_filterbank_analytic_t
MHAOvlFilter::overlap_save_filterbank_analytic_t, 1022
overlap_save_filterbank_t
MHAOvlFilter::overlap_save_filterbank_t, 1025
overlapadd, 156
overlapadd.cpp, 1698
overlapadd.hh, 1698
overlapadd::overlapadd_if_t, 1330
~overlapadd_if_t, 1332
algo, 1334
cf_in, 1334
cf_out, 1334
nfft, 1333
nwnd, 1333
overlapadd_if_t, 1331
plugloader, 1334
postscale, 1334
prepare, 1332
prescale, 1334
process, 1332
release, 1332
setlock, 1332
strict_window_ratio, 1333
update, 1332
window, 1333
wndexp, 1333
wndpos, 1333
zerowindow, 1333
overlapadd::overlapadd_t, 1335

© 2005-2021 HörTech gGmbH, Oldenburg

~overlapadd_t, 1335
calc_out, 1337
fft, 1336
n_pad1, 1337
n_pad2, 1338
n_zero, 1337
out_buf, 1337
overlapadd_t, 1335
postwnd, 1337
prewnd, 1336
spec2wave, 1336
spec_in, 1337
wave2spec, 1336
wave2spec_apply_window, 1336
wave2spec_compute_fft, 1336
wave2spec_hop_forward, 1336
wave_in1, 1337
wave_out1, 1337
write_buf, 1337
overlapadd_if_t
overlapadd::overlapadd_if_t, 1331
smoothgains_bridge::overlapadd_if_t, 1454
overlapadd_t
overlapadd::overlapadd_t, 1335
overrun_behavior
lsl2ac, 97
lsl2ac::lsl2ac_t, 685
ovltype
MHAOvlFilter::fftfb_vars_t, 1012
oy
MHAFilter::adapt_filter_state_t, 866

P
gsc_adaptive_stage::gsc_adaptive_stage, 544
p
acPooling_wave_config, 216
doasvm_classification_config, 426
pluginbrowser_t, 1358
p1
MHASignal::hilbert_fftw_t, 1217
p2
MHASignal::hilbert_fftw_t, 1217
p_biased
acPooling_wave_config, 216
p_biased_name
acPooling_wave, 214
p_in
io_alsa_t, 584
p_max
acPooling_wave_config, 217
doasvm_classification_config, 426
p_name
acPooling_wave, 214
doasvm_classification, 424
p_out
io_alsa_t, 584
p_parser
acmon::ac_monitor_t, 206
P_Sum
rt_nlms_t, 1421
pa22dbspl
MHASignal, 142
pa2dbspl
MHASignal, 141, 142
paInputLatency
MHAIOPortAudio::stream_info_t, 966
pairings
level_matching::level_matching_config_t, 653
paOutputLatency
MHAIOPortAudio::stream_info_t, 966
params
smooth_cepstrum::smooth_cepstrum_t, 1445
parent
matlab_wrapper::callback, 695
MHAParser::base_t, 1043
parent_
MHAParser::mhapluginloader_t, 1093
parse
altplugs_t, 302
io_asterisk_t, 603
io_tcp_t, 641
io_wrapper, 644
MHAParser::base_t, 1035, 1036
MHAPlugin_Split::splitted_part_t, 1186
plug_wrapper, 1351
plug_wrapperI, 1353
PluginLoader::fourway_processor_t, 1368
PluginLoader::mhapluginloader_t, 1371
parse_1_complex
mha_parser.cpp, 1617
parse_1_float
mha_parser.cpp, 1615
parser
io_asterisk_t, 603
io_tcp_t, 641
mhachain::plugs_t, 849
parser_algos
altconfig_t, 299
parser_int_dyn, 1338

© 2005-2021 HörTech gGmbH, Oldenburg

parser_int_dyn, 1339
set_max_angle_ind, 1339
parser_plugs
altplugs_t, 303
parser_t
AuditoryProfile::parser_t, 326
MHAParser::parser_t, 1104
parserFriendlyName
MHAIOPortAudio, 110
parsername
latex_doc_t, 647
parserstate
fw_t, 529
partitioned_convolution_t
MHAFilter::partitioned_convolution_t, 918
partitions
MHAFilter::transfer_function_t, 938
MHAFilter::transfer_matrix_t, 941
paSampleRate
MHAIOPortAudio::stream_info_t, 966
PASCALE
levelmeter.cpp, 1570
PATCH_VAR
acConcat_wave.cpp, 1525
acPooling_wave.cpp, 1526
acSteer.cpp, 1529
acTransform_wave.cpp, 1529
adaptive_feedback_canceller.cpp, 1530
doasvm_classification.cpp, 1547
doasvm_feature_extraction.cpp, 1548
level_matching.cpp, 1570
lpc.cpp, 1571
lpc_bl_predictor.cpp, 1572
lpc_burg-lattice.cpp, 1573
smooth_cepstrum.cpp, 1703
steerbf.cpp, 1710
patchbay
ac2lsl::ac2lsl_t, 165
ac2osc_t, 184
ac2wave_if_t, 188
acConcat_wave, 202
acmon::acmon_t, 210
acPooling_wave, 214
acsave::acsave_t, 222
acSteer, 231
acTransform_wave, 237
adaptive_feedback_canceller, 244
addsndfile::addsndfile_if_t, 254
adm_if_t, 275
altconfig_t, 299
altplugs_t, 304
analysispath_if_t, 312
audiometerbackend::audiometer_if_t, 317
AuditoryProfile::parser_t::fmap_t, 330
calibrator_t, 340
coherence::cohflt_if_t, 348
complex_scale_channel_t, 363
cpuload::cpuload_if_t, 368
db_if_t, 370
dc::dc_if_t, 382
dc_simple::dc_if_t, 396
delay::interface_t, 412
delaysum::delaysum_wave_if_t, 415
delaysum_spec::delaysum_spec_if_t, 420
doasvm_classification, 424
doasvm_feature_extraction, 429
double2acvar::double2acvar_t, 436
dropgen_t, 439
DynComp::dc_afterburn_t, 455
equalize::freqgains_t, 466
example3_t, 477
example4_t, 481
example6_t, 484
fader_if_t, 488
fader_wave::fader_wave_if_t, 491
fftfbpow::fftfbpow_interface_t, 497
fftfilter::interface_t, 503
fftfilterbank::fftfb_interface_t, 506
fshift::fshift_t, 514
fshift_hilbert::frequency_translator_t, 517
fw_t, 531
gain::gain_if_t, 535
gsc_adaptive_stage::gsc_adaptive_stage_if, 548
gtfb_analyzer::gtfb_analyzer_t, 556
gtfb_simd_t, 563
io_alsa_t, 584
io_parser_t, 621
level_matching::level_matching_t, 657
levelmeter_t, 660
lpc, 664
lpc_bl_predictor, 667
lpc_burglattice, 673
lsl2ac::lsl2ac_t, 685
matlab_wrapper::matlab_wrapper_t, 702
matrixmixer::matmix_t, 716
mconv::MConv, 720
mhachain::chain_base_t, 845
MHAIOJack::io_jack_t, 947
MHAIOJackdb::io_jack_t, 955
MHAIOPortAudio::io_portaudio_t, 964
MHAPlugin_Split::split_t, 1180

© 2005-2021 HörTech gGmbH, Oldenburg

multibandcompressor::interface_t, 1306
nlms_t, 1312
noise_psd_estimator::noise_psd_estimator_if_t, 1314
noise_t, 1320
osc2ac_t, 1323
plingploing::if_t, 1342
plugin_interface_t, 1356
plugins::hoertech::acrec::acrec_t, 1381
rmslevel::rmslevel_if_t, 1394
rohBeam::rohBeam, 1405
route::interface_t, 1414
sine_t, 1437
smooth_cepstrum::smooth_cepstrum_if_t, 1443
smoothgains_bridge::overlapadd_if_t, 1455
softclip_t, 1460
steerbf, 1477
testplugin::ac_parser_t, 1483
testplugin::if_t, 1488
wavrec_t, 1505
windnoise::if_t, 1516
windowselector_t, 1522
path
addsndfile::addsndfile_if_t, 252
pcm
alsa_base_t, 288
pcm_format
alsa_t< T >, 294
pcmlink
io_alsa_t, 584
peak
levelmeter_t, 660
MHASignal::loop_wavefragment_t, 1222
peaklevel
calibrator_variables_t, 341
mha_channel_info_t, 745
MHASignal::async_rmslevel_t, 1201
peek_config
MHAPlugin::config_t< runtime_cfg_t >, 1149
peer_addr
MHA_TCP::Connection, 811
peer_address
io_asterisk_parser_t, 596
io_tcp_parser_t, 632
peer_port
io_asterisk_parser_t, 596
io_tcp_parser_t, 632
period
droptect_t, 443
permute
ac_proc::interface_t, 198
pfragmentsize
fw_vars_t, 533
PH1Debug
noise_psd_estimator::noise_psd_estimator_t, 1317
PH1mean
noise_psd_estimator::noise_psd_estimator_t, 1316
phase
cpuload::cpuload_cfg_t, 365
MHASignal::minphase_t, 1238
phase_correction
MHAFilter::gamma_flt_t, 894
phase_div_2pi
sine_t, 1437
phase_gains
MHASignal::subsample_delay_t, 1258
phase_increment_div_2pi
sine_cfg_t, 1434
phasemode
fshift_hilbert::frequency_translator_t, 518
phasemodel
MHAOvlFilter::overlap_save_filterbank_t::vars_t, 1027
phasereconstruction
rohBeam::rohConfig, 1408
PI
ADM, 82
hann.cpp, 1567
pid_mon
mhaserver_t, 1197
pinchannels
fw_vars_t, 533
pink
speechnoise_t, 1473
pipe
MHA_TCP::Async_Notify, 799
pitch
plingploing::if_t, 1342
pitch_
plingploing::plingploing_t, 1346
pitch_set_first
smooth_cepstrum::smooth_cepstrum_t, 1449
pitch_set_last
smooth_cepstrum::smooth_cepstrum_t, 1449
plan_spec2analytic

© 2005-2021 HörTech gGmbH, Oldenburg

fshift_hilbert::hilbert_shifter_t, 521
plateau
MHAOvlFilter::fftfb_vars_t, 1012
playback
MHASignal::loop_wavefragment_t, 1223, 1224
playback_channels
MHASignal::loop_wavefragment_t, 1224
playback_mode_t
MHASignal::loop_wavefragment_t, 1222
plingploing, 156
drand, 156
plingploing.cpp, 1698
plingploing::if_t, 1340
bassmod, 1343
bassperiod, 1344
bpm, 1343
fun1_key, 1342
fun1_range, 1343
fun2_key, 1343
fun2_range, 1343
if_t, 1341
level, 1342
maxlen, 1343
minlen, 1343
patchbay, 1342
pitch, 1342
prepare, 1342
process, 1342
update, 1342
plingploing::plingploing_t, 1344
alph, 1348
bass, 1346
bassmod_, 1348
bassperiod_, 1348
bt, 1346
cf, 1346
dist, 1347
dist1, 1347
dur_, 1346
freq, 1347
fun1, 1347
fun1_key, 1347
fun1_range, 1347
fun2, 1347
fun2_key, 1347
fun2_range, 1347
hann1, 1348
hann2, 1348
len, 1346
level, 1348
maxlen_, 1346
minlen_, 1346
pitch_, 1346
plingploing_t, 1345
process, 1345
rms, 1348
t, 1346
plingploing_t
plingploing::plingploing_t, 1345
plug
ac_proc::interface_t, 198
analysispath_if_t, 312
gtfb_simple_t, 573
matlab_wrapper::matlab_wrapper_t, 702
MHAParser::mhapluginloader_t, 1093
MHAPlugin_Split::splitted_part_t, 1187
multibandcompressor::interface_t, 1306
testplugin::if_t, 1488
plug_level
multibandcompressor::plugin_signals_t, 1308
plug_output
multibandcompressor::plugin_signals_t, 1308
plug_sigs
multibandcompressor::interface_t, 1306
plug_t, 1349
~plug_t, 1349
get_ac, 1350
get_handle, 1350
get_process_spec, 1350
get_process_wave, 1349
plug_t, 1349
plug_wrapper, 1350
~plug_wrapper, 1351
get_categories, 1351
get_documentation, 1351
has_parser, 1351
has_process, 1352
parse, 1351
plug_wrapper, 1351
plug_wrapperI, 1352
~plug_wrapperI, 1353
get_categories, 1353
get_documentation, 1353
has_parser, 1353
has_process, 1354
parse, 1353
plug_wrapperI, 1353
plugin_categories
io_lib_t, 616

© 2005-2021 HörTech gGmbH, Oldenburg

PluginLoader::mhapluginloader_t, 1376
plugin_documentation
io_lib_t, 616
PluginLoader::mhapluginloader_t, 1376
plugin_extension
pluginbrowser_t, 1358
plugin_interface_t, 1354
factor, 1356
patchbay, 1356
plugin_interface_t, 1355
prepare, 1355
process, 1355
scale_ch, 1356
update_cfg, 1356
plugin_macro
latex_doc_t, 648
plugin_paths
fw_t, 530
plugin_signals_t
multibandcompressor::plugin_signals_t, 1307
plugin_t
MHAPlugin::plugin_t< runtime_cfg_t >, 1153
pluginbrowser.cpp, 1699
pluginbrowser.h, 1699
pluginbrowser_t, 1356
add_plugin, 1357
add_plugins, 1357
clear_plugins, 1357
get_paths, 1357
get_plugins, 1358
library_paths, 1358
p, 1358
plugin_extension, 1358
pluginbrowser_t, 1357
plugins, 1358
scan_plugin, 1357
scan_plugins, 1357
plugindescription_t, 1359
categories, 1359
documentation, 1359
fullname, 1359
name, 1359
queries, 1360
query_cmds, 1360
spec2spec, 1360
spec2wave, 1360
wave2spec, 1359
wave2wave, 1359
pluginlib_t, 1360
~pluginlib_t, 1361
pluginlib_t, 1361
resolve, 1362
PluginLoader, 157
mhaconfig_compare, 157
mhastrdomain, 157
PluginLoader::config_file_splitter_t, 1362
config_file_splitter_t, 1363
configfile, 1364
configname, 1364
get_configfile, 1364
get_configname, 1363
get_libname, 1363
get_origname, 1363
libname, 1364
origname, 1364
PluginLoader::fourway_processor_t, 1365
~fourway_processor_t, 1366
parse, 1368
prepare, 1367
process, 1366, 1367
release, 1368
PluginLoader::mhapluginloader_t, 1368
~mhapluginloader_t, 1371
ac, 1374
b_check_version, 1376
b_is_prepared, 1376
cf_input, 1375
cf_output, 1376
get_categories, 1373
get_documentation, 1373
getfullname, 1372
has_parser, 1371
has_process, 1371
input_domain, 1371
is_prepared, 1373
lib_data, 1374
lib_err, 1374
lib_handle, 1374
mha_test_struct_size, 1373
MHADestroy_cb, 1374
MHAGetVersion_cb, 1374
MHAInit_cb, 1374
mhapluginloader_t, 1370
MHAPrepare_cb, 1374
MHAProc_spec2spec_cb, 1375
MHAProc_spec2wave_cb, 1375
MHAProc_wave2spec_cb, 1375
MHAProc_wave2wave_cb, 1375
MHARelease_cb, 1374
MHASet_cb, 1375

© 2005-2021 HörTech gGmbH, Oldenburg

MHASetcpp_cb, 1375
MHAStrError_cb, 1375
output_domain, 1371
parse, 1371
plugin_categories, 1376
plugin_documentation, 1376
prepare, 1371
process, 1372
release, 1372
resolve_and_init, 1373
test_error, 1373
test_version, 1373
pluginloader_t, 1376
~pluginloader_t, 1377
pluginloader_t, 1377
plugins, 158
fw_t, 530
pluginbrowser_t, 1358
plugins::hoertech, 158
plugins::hoertech::acrec, 158
to_iso8601, 158
plugins::hoertech::acrec::acrec_t, 1377
ac, 1382
acrec_t, 1379
cv, 1381
fifolen, 1381
minwrite, 1381
patchbay, 1381
prefix, 1381
prepare, 1380
process, 1379
record, 1381
release, 1380
start_new_session, 1380
use_date, 1381
varname, 1381
plugins::hoertech::acrec::acwriter_t, 1382
~acwriter_t, 1384
active, 1386
acwriter_t, 1384
close_session, 1386
create_datafile, 1385
disk_write_threshold_min_num_samples, 1386
diskbuffer, 1387
exit_request, 1385
fifo, 1386
get_varname, 1385
is_complex, 1387
is_num_channels_known, 1387
num_channels, 1387
outfile, 1387
output_type, 1383
process, 1384
varname, 1387
write_thread, 1385
writethread, 1386
plugloader
bbcalib_interface_t, 335
db_if_t, 371
db_t, 372
dbasync_native::db_if_t, 375
dbasync_native::dbasync_t, 378
MHAPlugin_Resampling::resampling_if_t, 1159
MHAPlugin_Resampling::resampling_t, 1162
overlapadd::overlapadd_if_t, 1334
smoothgains_bridge::overlapadd_if_t, 1455
plugname
latex_doc_t, 647
MHAParser::mhapluginloader_t, 1094
plugname_name_
MHAParser::mhapluginloader_t, 1094
plugs
altplugs_t, 304
plugs_t
mhachain::plugs_t, 847
pmode
audiometerbackend::audiometer_if_t, 317
calibrator_runtime_layer_t, 338
poll
mha_rt_fifo_t< T >, 792
poll_1
mha_rt_fifo_t< T >, 792
poll_config
MHAPlugin::config_t< runtime_cfg_t >, 1149
poll_latest_value_and_reinsert
double2acvar::double2acvar_t, 435
polyphase_resampling_t
MHAFilter::polyphase_resampling_t, 925
pool
acPooling_wave_config, 218
pool_name
acPooling_wave, 214
pooling_ind
acPooling_wave_config, 217
pooling_option
acPooling_wave_config, 217
pooling_size

© 2005-2021 HörTech gGmbH, Oldenburg

acPooling_wave_config, 217
pooling_type
acPooling_wave, 213
pooling_wndlen
acPooling_wave, 213
port
ac2osc_t, 183
MHA_TCP::Server, 818
MHAJack::port_t, 987
mhaserver_t, 1197
osc2ac_t, 1323
port_t
MHAJack::port_t, 985, 986
portaudio_callback
MHAIOPortAudio.cpp, 1681, 1682
MHAIOPortAudio::io_portaudio_t, 962
portaudio_stream
MHAIOPortAudio::io_portaudio_t, 964
portnames_in
MHAIOJack::io_jack_t, 946
MHAIOJackdb::io_jack_t, 953
portnames_out
MHAIOJack::io_jack_t, 946
MHAIOJackdb::io_jack_t, 953
ports_in_all
MHAIOJack::io_jack_t, 946
MHAIOJackdb::io_jack_t, 954
ports_in_physical
MHAIOJack::io_jack_t, 946
MHAIOJackdb::io_jack_t, 954
ports_out_all
MHAIOJack::io_jack_t, 946
MHAIOJackdb::io_jack_t, 954
ports_out_physical
MHAIOJack::io_jack_t, 946
MHAIOJackdb::io_jack_t, 954
ports_parser
MHAIOJack::io_jack_t, 947
MHAIOJackdb::io_jack_t, 954
pos
addsndfile::level_adapt_t, 256
audiometerbackend::level_adapt_t, 319
cfg_t, 346
fader_wave::level_adapt_t, 493
MHAJack::client_avg_t, 970
MHAJack::client_noncont_t, 973
MHASignal::async_rmslevel_t, 1201
MHASignal::delay_spec_t, 1203
MHASignal::delay_t, 1205
MHASignal::delay_wave_t, 1207
MHASignal::loop_wavefragment_t, 1225
posix_threads_t
MHAPlugin_Split::posix_threads_t, 1173
posixthreads
split.cpp, 1710
post_plugin
gtfb_simple_rt_t, 566
post_trigger_read_line
mha_tcp::server_t, 823
postfilter
rohBeam::rohConfig, 1408
postscale
overlapadd::overlapadd_if_t, 1334
postwindow
spec2wave_t, 1470
postwnd
overlapadd::overlapadd_t, 1337
power
MHASignal::waveform_t, 1273
powSpec
smooth_cepstrum::smooth_cepstrum_t, 1447
powspec
MHASignal::waveform_t, 1274
windnoise::cfg_t, 1513
POWSPEC_FACTOR
noise_psd_estimator.cpp, 1697
pre_plugin
gtfb_simple_rt_t, 566
prefix
plugins::hoertech::acrec::acrec_t, 1381
wavrec_t, 1505
prefix_
MHAParser::mhapluginloader_t, 1094
prefix_names_AC
acConcat_wave, 201
prepare
ac2lsl::ac2lsl_t, 163
ac2osc_t, 182
ac2wave_if_t, 187
ac_proc::interface_t, 197
acConcat_wave, 200
acmon::acmon_t, 208
acPooling_wave, 212
acsave::acsave_t, 220
acSteer, 229
acTransform_wave, 235
adaptive_feedback_canceller, 242
addsndfile::addsndfile_if_t, 251
adm_if_t, 272
altconfig_t, 297
altplugs_t, 301

© 2005-2021 HörTech gGmbH, Oldenburg

analysispath_if_t, 311
attenuate20_t, 314
audiometerbackend::audiometer_if_t, 316
bbcalib_interface_t, 334
calibrator_t, 339
coherence::cohflt_if_t, 348
combc_if_t, 356
complex_scale_channel_t, 362
cpuload::cpuload_if_t, 367
db_if_t, 370
dbasync_native::db_if_t, 374
dc::dc_if_t, 381
dc_simple::dc_if_t, 393
delay::interface_t, 411
delaysum::delaysum_wave_if_t, 414
delaysum_spec::delaysum_spec_if_t, 419
doasvm_classification, 423
doasvm_feature_extraction, 428
dropgen_t, 438
droptect_t, 441
ds_t, 445
equalize::freqgains_t, 465
example1_t, 468
example2_t, 471
example3_t, 475
example4_t, 479
example6_t, 483
example7_t, 486
fader_if_t, 488
fader_wave::fader_wave_if_t, 490
fftfbpow::fftfbpow_interface_t, 495
fftfilter::interface_t, 502
fftfilterbank::fftfb_interface_t, 505
fshift::fshift_t, 513
fshift_hilbert::frequency_translator_t, 517
fw_t, 526
gain::gain_if_t, 535
gsc_adaptive_stage::gsc_adaptive_stage_if, 547
gtfb_analyzer::gtfb_analyzer_t, 555
gtfb_simd_t, 563
gtfb_simple_t, 572
identity_t, 577
io_alsa_t, 581, 582
io_asterisk_sound_t, 598
io_asterisk_t, 602
io_file_t, 607
io_lib_t, 613
io_parser_t, 618
io_tcp_sound_t, 635
io_tcp_t, 640
level_matching::level_matching_t, 656
levelmeter_t, 659
lpc, 662
lpc_bl_predictor, 666
lpc_burglattice, 672
lsl2ac::lsl2ac_t, 683
matlab_wrapper::matlab_wrapper_t, 700
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 707
matrixmixer::matmix_t, 715
mconv::MConv, 719
mhachain::chain_base_t, 844
mhachain::plugs_t, 848
MHAIOJack::io_jack_t, 943
MHAIOJackdb::io_jack_t, 950
MHAJack::client_t, 977
MHAParser::mhapluginloader_t, 1092
MHAPlugin::plugin_t< runtime_cfg_t >, 1153
MHAPlugin_Resampling::resampling_if_t, 1158
MHAPlugin_Split::splitted_part_t, 1185
MHATableLookup::linear_table_t, 1281
multibandcompressor::interface_t, 1305
nlms_t, 1310
noise_psd_estimator::noise_psd_estimator_if_t, 1313
noise_t, 1319
osc2ac_t, 1322
overlapadd::overlapadd_if_t, 1332
plingploing::if_t, 1342
plugin_interface_t, 1355
PluginLoader::fourway_processor_t, 1367
PluginLoader::mhapluginloader_t, 1371
plugins::hoertech::acrec::acrec_t, 1380
rmslevel::rmslevel_if_t, 1393
rohBeam::rohBeam, 1401
route::interface_t, 1413
save_spec_t, 1423
save_wave_t, 1425
shadowfilter_begin::shadowfilter_begin_t, 1428
shadowfilter_end::shadowfilter_end_t, 1432
sine_t, 1436
smooth_cepstrum::smooth_cepstrum_if_t, 1440
smoothgains_bridge::overlapadd_if_t, 1454
softclip_t, 1459
spec2wave_if_t, 1467

© 2005-2021 HörTech gGmbH, Oldenburg

steerbf, 1476
testplugin::if_t, 1487
us_t, 1491
wave2spec_if_t, 1494
wavrec_t, 1504
windnoise::if_t, 1515
prepare_
ac_mul_t, 193
double2acvar::double2acvar_t, 435
iirfilter_t, 579
MHAPlugin::plugin_t< runtime_cfg_t >, 1155
MHAPlugin_Split::split_t, 1179
proc_counter_t, 1389
prepare_impl
MHAJack::client_t, 980
prepare_vars
fw_t, 528
PREPARED
MHA_TCP::Thread, 830
prepared
ac2wave_if_t, 188
altplugs_t, 305
calibrator_t, 340
dc_simple::dc_if_t, 397
example3_t, 476
example4_t, 481
fader_wave::fader_wave_if_t, 491
fftfilterbank::fftfb_interface_t, 506
gtfb_analyzer::gtfb_analyzer_t, 556
gtfb_simd_t, 563
mhachain::plugs_t, 848
rohBeam::rohBeam, 1405
route::interface_t, 1415
smooth_cepstrum::smooth_cepstrum_if_t, 1443
prereadaccess
MHAParser::base_t, 1042
prescale
overlapadd::overlapadd_if_t, 1334
preset
dc::dc_vars_t, 390
dc_simple::dc_if_t, 396
prestages
gtfb_simple_t, 573
prewnd
overlapadd::overlapadd_t, 1336
print_ac
analysemhaplugin.cpp, 1535
print_plugin_references
generatemhaplugindoc.cpp, 1557
prior_q
smooth_cepstrum::smooth_cepstrum_if_t, 1442
smooth_cepstrum::smooth_params, 1452
priorFact
noise_psd_estimator::noise_psd_estimator_t, 1317
smooth_cepstrum::smooth_cepstrum_t, 1449
priority
analysepath_t, 309
analysispath_if_t, 312
dbasync_native::dbasync_t, 378
io_alsa_t, 584
MHAPlugin_Split::posix_threads_t, 1175
prob_bias
acPooling_wave, 214
prob_bias_func
acPooling_wave_config, 218
proc
MHAJack::client_avg_t, 969
MHAJack::client_noncont_t, 972, 973
proc_1
smoothgains_bridge::smoothspec_wrap_t, 1457
proc_2
smoothgains_bridge::smoothspec_wrap_t, 1457
proc_cnt
mhachain::plugs_t, 850
proc_counter.cpp, 1699
proc_counter_t, 1388
~proc_counter_t, 1389
ac, 1390
prepare_, 1389
proc_counter_t, 1389
process, 1389
release_, 1389
proc_err
io_asterisk_fwcb_t, 588
io_tcp_fwcb_t, 625
proc_error
fw_t, 531
proc_error_string
fw_t, 531
proc_event
io_alsa_t, 583
io_asterisk_fwcb_t, 588
io_file_t, 609
io_parser_t, 619
io_tcp_fwcb_t, 624

© 2005-2021 HörTech gGmbH, Oldenburg

MHAIOJackdb::io_jack_t, 952
MHAIOPortAudio::io_portaudio_t, 963
MHAJack::client_t, 982
proc_handle
io_alsa_t, 583
io_asterisk_fwcb_t, 588
io_file_t, 609
io_parser_t, 619
io_tcp_fwcb_t, 625
MHAIOJackdb::io_jack_t, 952
MHAIOPortAudio::io_portaudio_t, 963
MHAJack::client_t, 982
proc_lib
fw_t, 530
proc_name
fw_t, 529
proc_ramp
altplugs_t, 303
proc_thread
io_alsa_t, 583
proc_wave
doasvm_feature_extraction_config, 432
process
ac2lsl::ac2lsl_t, 164
ac2lsl::cfg_t, 168
ac2osc_t, 182
ac2wave_if_t, 186, 187
ac2wave_t, 189
ac_mul_t, 193, 194
ac_proc::interface_t, 197, 198
acConcat_wave, 200
acConcat_wave_config, 203
acmon::acmon_t, 209
acPooling_wave, 212
acPooling_wave_config, 216
acsave::acsave_t, 220, 221
acSteer, 229
acTransform_wave, 234
acTransform_wave_config, 238
adaptive_feedback_canceller, 241
adaptive_feedback_canceller_config, 245
addsndfile::addsndfile_if_t, 251
ADM::ADM< F >, 263
ADM::Delay< F >, 267
ADM::Linearphase_FIR< F >, 270
adm_if_t, 272
altplugs_t, 302
analysispath_if_t, 311
attenuate20_t, 314
audiometerbackend::audiometer_if_t, 316
bbcalib_interface_t, 334
calibrator_runtime_layer_t, 336
calibrator_t, 339
cfg_t, 345
coherence::cohflt_if_t, 348
coherence::cohflt_t, 350
combc_if_t, 356
combc_t, 358
complex_scale_channel_t, 362
cpuload::cpuload_cfg_t, 364
cpuload::cpuload_if_t, 367
db_if_t, 370
dbasync_native::db_if_t, 374
dc::dc_if_t, 381
dc::dc_t, 384
dc_simple::dc_if_t, 394
dc_simple::dc_t, 399
dc_simple::level_smoother_t, 408, 409
delay::interface_t, 411
delaysum::delaysum_wave_if_t, 414
delaysum::delaysum_wave_t, 417
delaysum_spec::delaysum_spec_if_t, 419
delaysum_spec::delaysum_t, 420
doasvm_classification, 423
doasvm_classification_config, 426
doasvm_feature_extraction, 428
doasvm_feature_extraction_config, 431
double2acvar::double2acvar_t, 435
dropgen_t, 438
droptect_t, 442
ds_t, 445
equalize::freqgains_t, 465
example1_t, 468
example2_t, 472
example3_t, 476
example4_t, 480
example5_t, 482
example6_t, 483
example7_t, 486
fader_if_t, 488
fader_wave::fader_wave_if_t, 490
ffttfbpow::ffttfbpow_interface_t, 496
fftfilter::fftfilter_t, 500
fftfilter::interface_t, 502
fftfilterbank::fftfb_interface_t, 505, 506
fftfilterbank::fftfb_plug_t, 508
fshift::fshift_config_t, 510
fshift::fshift_t, 513
fshift_hilbert::frequency_translator_t, 516
fshift_hilbert::hilbert_shifter_t, 520
fw_t, 527
gain::gain_if_t, 534

© 2005-2021 HörTech gGmbH, Oldenburg

gsc_adaptive_stage::gsc_adaptive_stage, 539
gsc_adaptive_stage::gsc_adaptive_stage_if, 547
gtfb_analyzer::gtfb_analyzer_t, 555
gtfb_simd_cfg_t, 559
gtfb_simd_t, 563
gtfb_simple_t, 572
identity_t, 577
iirfilter_t, 579
io_alsa_t, 582
io_asterisk_fwcb_t, 586
io_tcp_fwcb_t, 623
level_matching::level_matching_config_t, 652, 653
level_matching::level_matching_t, 655, 656
levelmeter_t, 659
lpc, 662
lpc_bl_predictor, 666
lpc_bl_predictor_config, 668
lpc_burglattice, 672
lpc_burglattice_config, 674
lpc_config, 677
lsl2ac::cfg_t, 680
lsl2ac::lsl2ac_t, 683
matlab_wrapper::matlab_wrapper_t, 699, 700
matrixmixer::cfg_t, 713
matrixmixer::matmix_t, 715, 716
mconv::MConv, 719
mha_dblbuf_t< FIFO >, 752
mhachain::chain_base_t, 843, 844
mhachain::plugs_t, 848
MHAFilter::partitioned_convolution_t, 919
MHAFilter::thirdoctave_analyzer_t, 935
MHAParser::mhapluginloader_t, 1092, 1093
MHAPlugin_Resampling::resampling_if_t, 1158
MHAPlugin_Resampling::resampling_t, 1161
MHAPlugin_Split::domain_handler_t, 1168
MHAPlugin_Split::split_t, 1179
MHAPlugin_Split::uni_processor_t, 1192
MHASignal::async_rmslevel_t, 1201
MHASignal::delay_spec_t, 1203
MHASignal::delay_t, 1205
MHASignal::delay_wave_t, 1207
MHASignal::subsample_delay_t, 1257, 1258
multibandcompressor::interface_t, 1305
nlms_t, 1310
noise_psd_estimator::noise_psd_estimator_if_t, 1313
noise_psd_estimator::noise_psd_estimator_t, 1316
noise_t, 1319
osc2ac_t, 1322
overlapadd::overlapadd_if_t, 1332
plingploing::if_t, 1342
plingploing::plingploing_t, 1345
plugin_interface_t, 1355
PluginLoader::fourway_processor_t, 1366, 1367
PluginLoader::mhapluginloader_t, 1372
plugins::hoertech::acrec::acrec_t, 1379
plugins::hoertech::acrec::acwriter_t, 1384
proc_counter_t, 1389
rmslevel::rmslevel_if_t, 1393
rmslevel::rmslevel_t, 1396
rohBeam::rohBeam, 1401
rohBeam::rohConfig, 1408
route::interface_t, 1414
route::process_t, 1416
rt_nlms_t, 1419
save_spec_t, 1423
save_wave_t, 1425
shadowfilter_begin::cfg_t, 1426
shadowfilter_begin::shadowfilter_begin_t, 1428
shadowfilter_end::cfg_t, 1430
shadowfilter_end::shadowfilter_end_t, 1432
sine_t, 1436
smooth_cepstrum::smooth_cepstrum_if_t, 1439
smooth_cepstrum::smooth_cepstrum_t, 1445
smoothgains_bridge::overlapadd_if_t, 1455
softclip_t, 1459
softclipper_t, 1462
spec2wave_if_t, 1467
spec2wave_t, 1469
steerbf, 1476
steerbf_config, 1478
testplugin::if_t, 1487
us_t, 1491
wave2spec_if_t, 1494, 1495
wave2spec_t, 1500

© 2005-2021 HörTech gGmbH, Oldenburg

wavrec_t, 1504
wavwriter_t, 1507
windnoise::cfg_t, 1511
windnoise::if_t, 1516
process_cc
ac_mul_t, 194
process_cr
ac_mul_t, 194
process_frame
io_parser_t, 619
process_rc
ac_mul_t, 194
process_rr
ac_mul_t, 194
process_ss
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 706
process_sw
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 707
process_t
route::process_t, 1416
process_ws
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 706
process_ww
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 705
processing_done
MHAPlugin_Split::posix_threads_t, 1175
ProcessMutex
analysepath_t, 309
processor
MHAPlugin_Split::domain_handler_t, 1169
MHAPlugin_Split::thread_platform_t, 1191
prof_algos
mhachain::plugs_t, 849
prof_cfg
mhachain::plugs_t, 850
prof_init
mhachain::plugs_t, 850
prof_load_con
mhachain::plugs_t, 851
prof_prepare
mhachain::plugs_t, 850
prof_process
mhachain::plugs_t, 850
prof_process_load
mhachain::plugs_t, 850
prof_process_tt
mhachain::plugs_t, 850
prof_release
mhachain::plugs_t, 850
prof_tt_con
mhachain::plugs_t, 851
profiling
mhachain::plugs_t, 849
prop_type
rohBeam::rohBeam, 1403
propExport
rohBeam::rohBeam, 1406
provoke_inner_error
mha_dblbuf_t< FIFO >, 751
provoke_outer_error
mha_dblbuf_t< FIFO >, 752
PSD_Lowpass
windnoise::cfg_t, 1513
PSD_val
adaptive_feedback_canceller_config, 247
psrate
fw_vars_t, 533
Psum
gsc_adaptive_stage::gsc_adaptive_stage, 544
Pu
adaptive_feedback_canceller_config, 246
rt_nlms_t, 1420
publish_ac_variables
wave2spec_t, 1500
pull_samples_discard
lsl2ac::save_var_t, 689
pull_samples_ignore
lsl2ac::save_var_t, 689
push
mha_rt_fifo_t< T >, 792
MHASignal::stat_t, 1255, 1256
push_config
MHAPlugin::config_t< runtime_cfg_t >, 1150
put_signal
MHAPlugin_Split::domain_handler_t, 1166
pVars
osc_server_t, 1325
pwinner_out
MHAIOJackdb::io_jack_t, 955

q
noise_psd_estimator::noise_psd_estimator_if_t, 1314
q_high

© 2005-2021 HörTech gGmbH, Oldenburg

smooth_cepstrum::smooth_cepstrum_t, 1446
q_low
smooth_cepstrum::smooth_cepstrum_t, 1446
quant
calibrator_runtime_layer_t, 337
quantile
MHASignal, 147
quantizer_t
MHASignal::quantizer_t, 1239
queries
MHAParser::base_t, 1042
plugindescription_t, 1360
query_addsubst
MHAParser::base_t, 1040
query_cmds
MHAParser::base_t, 1040
plugindescription_t, 1360
query_dump
MHAParser::base_t, 1037
MHAParser::monitor_t, 1101
MHAParser::parser_t, 1106
query_entries
MHAParser::base_t, 1037
MHAParser::parser_t, 1106
query_help
MHAParser::base_t, 1040
query_id
MHAParser::base_t, 1039
query_listids
MHAParser::base_t, 1039
MHAParser::parser_t, 1107
query_map_t
MHAParser, 126
query_peak
levelmeter_t, 660
query_perm
MHAParser::base_t, 1037
MHAParser::monitor_t, 1101
MHAParser::variable_t, 1119
query_range
MHAParser::base_t, 1038
MHAParser::kw_t, 1078
MHAParser::range_var_t, 1110
query_readfile
MHAParser::base_t, 1038
MHAParser::parser_t, 1106
query_rms
levelmeter_t, 659
query_savefile
MHAParser::base_t, 1039
MHAParser::parser_t, 1107
query_savefile_compact
MHAParser::base_t, 1039
MHAParser::parser_t, 1107
query_savemons
MHAParser::base_t, 1039
MHAParser::parser_t, 1107
query_subst
MHAParser::base_t, 1040
query_t
MHAParser, 125
query_type
MHAParser::base_t, 1038
MHAParser::bool_mon_t, 1046
MHAParser::bool_t, 1049
MHAParser::complex_mon_t, 1056
MHAParser::complex_t, 1058
MHAParser::float_mon_t, 1063
MHAParser::float_t, 1065
MHAParser::int_mon_t, 1068
MHAParser::int_t, 1071
MHAParser::kw_t, 1078
MHAParser::mcomplex_mon_t, 1080
MHAParser::mcomplex_t, 1082
MHAParser::mfloat_mon_t, 1084
MHAParser::mfloat_t, 1087
MHAParser::mint_mon_t, 1096
MHAParser::mint_t, 1099
MHAParser::parser_t, 1106
MHAParser::string_mon_t, 1114
MHAParser::string_t, 1117
MHAParser::vcomplex_mon_t, 1121
MHAParser::vcomplex_t, 1124
MHAParser::vfloat_mon_t, 1126
MHAParser::vfloat_t, 1129
MHAParser::vint_mon_t, 1131
MHAParser::vint_t, 1133
MHAParser::vstring_mon_t, 1136
MHAParser::vstring_t, 1137
query_val
MHAParser::base_t, 1038
MHAParser::bool_mon_t, 1046
MHAParser::bool_t, 1049
MHAParser::complex_mon_t, 1056
MHAParser::complex_t, 1058
MHAParser::float_mon_t, 1062
MHAParser::float_t, 1066
MHAParser::int_mon_t, 1068
MHAParser::int_t, 1071
MHAParser::kw_t, 1078

© 2005-2021 HörTech gGmbH, Oldenburg

MHAParser::mcomplex_mon_t, 1080
MHAParser::mcomplex_t, 1082
MHAParser::mfloat_mon_t, 1084
MHAParser::mfloat_t, 1087
MHAParser::mint_mon_t, 1096
MHAParser::mint_t, 1099
MHAParser::parser_t, 1107
MHAParser::string_mon_t, 1114
MHAParser::string_t, 1117
MHAParser::vcomplex_mon_t, 1121
MHAParser::vcomplex_t, 1124
MHAParser::vfloat_mon_t, 1126
MHAParser::vfloat_t, 1129
MHAParser::vint_mon_t, 1131
MHAParser::vint_t, 1133
MHAParser::vstring_mon_t, 1135
MHAParser::vstring_t, 1138
query_version
MHAParser::base_t, 1039
queue_write
mha_tcp::buffered_socket_t, 800
quit
fw_t, 526

R
AuditoryProfile::parser_t, 327
AuditoryProfile::profile_t, 332
lpc_config, 678
MHA_TCP::OS_EVENT_TYPE, 814
r
dropgen_t, 439
mha_real_test_array_t, 788
rad2smp
Vector and matrix processing toolbox, 40
ramp_a
hanning_ramps_t, 575
ramp_b
hanning_ramps_t, 576
ramp_begin
MHAWindow::base_t, 1293
ramp_counter
altplugs_t, 305
ramp_end
MHAWindow::base_t, 1293
ramp_len
altplugs_t, 305
ramplen
addsndfile::addsndfile_if_t, 253
altplugs_t, 304
audiometerbackend::audiometer_if_t, 317
fader_wave::fader_wave_if_t, 491
spec2wave_if_t, 1468
ramps
spec2wave_t, 1470
rand_dist
cfg_t, 346
random
audiometerbackend::lnn3rdoct_t, 322
random_engine
dropgen_t, 439
range
level_matching::level_matching_config_t, 653
level_matching::level_matching_t, 657
Vector and matrix processing toolbox, 38
range_var_t
MHAParser::range_var_t, 1110
ratio
ds_t, 445
us_t, 1492
raw_p_max_name
acTransform_wave, 236
acTransform_wave_config, 238
raw_p_name
acPooling_wave_config, 216
acTransform_wave, 236
acTransform_wave_config, 238
rb_f_t
mha_ruby.cpp, 1628
rcoefficients
gtfb_simd_cfg_t, 560
rdata
mha_audio_t, 744
MHASignal::matrix_t, 1236
re
mha_complex_t, 746
read
alsa_base_t, 287
alsa_t< T >, 293
mha_drifter_fifo_t< T >, 759
mha_fifo_lf_t< T >, 769
mha_fifo_lw_t< T >, 772
mha_fifo_t< T >, 780
MHAFilter::blockprocessing_polyphase_resampling_t, 871
MHAFilter::polyphase_resampling_t, 926
MHAJack::port_t, 986
read_bytes
MHA_TCP::Connection, 809
read_event
MHA_TCP::Connection, 810
read_get_cpu_load
MHAIOJack::io_jack_t, 944

© 2005-2021 HörTech gGmbH, Oldenburg

MHAIOJackdb::io_jack_t, 951
read_get_scheduler
MHAIOJack::io_jack_t, 945
MHAIOJackdb::io_jack_t, 952
read_get_xruns
MHAIOJack::io_jack_t, 945
MHAIOJackdb::io_jack_t, 951
read_levels
calibrator_t, 340
read_line
MHA_TCP::Connection, 808
read_modified
dc_simple::dc_if_t, 395
read_ptr
mha_fifo_t< T >, 783
readable_frames
MHAFilter::polyphase_resampling_t, 926
readaccess
MHAParser::base_t, 1042
reader_started
mha_drifter_fifo_t< T >, 762
reader_xruns_in_succession
mha_drifter_fifo_t< T >, 763
reader_xruns_since_start
mha_drifter_fifo_t< T >, 763
reader_xruns_total
mha_drifter_fifo_t< T >, 762
real
MHASignal::matrix_t, 1231–1234
rear_channel
adm_rtconfig_t, 278
rear_channels
adm_if_t, 273
adm_rtconfig_t, 278
rec_frames
acsave::cfg_t, 224
receive_frame
lsl2ac::save_var_t, 689
receiver
MHAEvents::connector_t< receiver_t >, 858
reciprocal
Complex arithmetics in the openMHA, 67
reclen
acsave::acsave_t, 221
recmode
acmon::acmon_t, 210
reconnect_inports
MHAIOJack::io_jack_t, 943
MHAIOJackdb::io_jack_t, 951
reconnect_outports
MHAIOJack::io_jack_t, 944
MHAIOJackdb::io_jack_t, 951
record
plugins::hoertech::acrec::acrec_t, 1381
wavrec_t, 1505
rect
MHAOvlFilter::ShapeFun, 120
MHAWindow, 154
rect_t
MHAWindow::rect_t, 1300
refL
rohBeam, 160
refR
rohBeam, 160
register_configuration_variable
windnoise.cpp, 1714
relative
MHASignal::loop_wavefragment_t, 1222
release
ac2lsl::ac2lsl_t, 164
ac2osc_t, 182
ac2wave_if_t, 187
ac_proc::interface_t, 197
acConcat_wave, 201
acmon::acmon_t, 208
acPooling_wave, 213
acsave::acsave_t, 220
acSteer, 230
acTransform_wave, 236
adaptive_feedback_canceller, 242
addsndfile::addsndfile_if_t, 252
adm_if_t, 273
altconfig_t, 297
altplugs_t, 301
analysispath_if_t, 311
attenuate20_t, 314
bbcalib_interface_t, 335
calibrator_t, 339
coherence::cohflt_if_t, 348
db_if_t, 370
dbasync_native::db_if_t, 375
dc_simple::dc_if_t, 394
delaysum::delaysum_wave_if_t, 414
doasvm_classification, 423
doasvm_feature_extraction, 429
dropgen_t, 438
droptect_t, 442
ds_t, 445
example1_t, 468
example2_t, 472
example3_t, 475

© 2005-2021 HörTech gGmbH, Oldenburg

example4_t, 480
example7_t, 486
fader_wave::fader_wave_if_t, 490
fftfilterbank::fftfb_interface_t, 505
fshift::fshift_t, 514
fshift_hilbert::frequency_translator_t, 517
fw_t, 526
gain::gain_if_t, 535
gsc_adaptive_stage::gsc_adaptive_stage_if, 547
gtfb_analyzer::gtfb_analyzer_t, 555
gtfb_simple_t, 572
identity_t, 577
io_alsa_t, 581
io_asterisk_sound_t, 598
io_asterisk_t, 602
io_file_t, 607
io_lib_t, 614
io_parser_t, 618
io_tcp_sound_t, 635
io_tcp_t, 640
level_matching::level_matching_t, 656
lpc, 663
lpc_bl_predictor, 666
lpc_burglattice, 672
lsl2ac::lsl2ac_t, 683
matlab_wrapper::matlab_wrapper_t, 701
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 707
mconv::MConv, 719
mhachain::chain_base_t, 844
mhachain::plugs_t, 848
MHAIOJack::io_jack_t, 943
MHAIOJackdb::io_jack_t, 950
MHAJack::client_t, 978
MHAParser::mhapluginloader_t, 1092
MHAPlugin::plugin_t< runtime_cfg_t >, 1154
MHAPlugin_Resampling::resampling_if_t, 1159
MHAPlugin_Split::splitted_part_t, 1186
multibandcompressor::interface_t, 1305
nlms_t, 1310
osc2ac_t, 1322
overlapadd::overlapadd_if_t, 1332
PluginLoader::fourway_processor_t, 1368
PluginLoader::mhapluginloader_t, 1372
plugins::hoertech::acrec::acrec_t, 1380
rohBeam::rohBeam, 1401
route::interface_t, 1414
smooth_cepstrum::smooth_cepstrum_if_t, 1440
smoothgains_bridge::overlapadd_if_t, 1454
spec2wave_if_t, 1467
steerbf, 1477
us_t, 1491
wave2spec_if_t, 1494
wavrec_t, 1504
windnoise::if_t, 1516
release_
ac_mul_t, 193
double2acvar::double2acvar_t, 436
iirfilter_t, 579
MHAPlugin::plugin_t< runtime_cfg_t >, 1155
MHAPlugin_Split::split_t, 1179
proc_counter_t, 1389
release_mutex
mha_fifo_posix_threads_t, 775
mha_fifo_thread_platform_t, 786
remapping
windnoise::cfg_t, 1512
remove
MHAUtils, 151
remove_abandonned
mha_rt_fifo_t< T >, 793
remove_all
mha_rt_fifo_t< T >, 793
remove_all_cfg
MHAPlugin::config_t< runtime_cfg_t >, 1150
remove_item
MHAParser::parser_t, 1104, 1105
remove_ref
algo_comm_t, 282
MHAKernel::algo_comm_class_t, 991
remove_var
algo_comm_t, 282
MHAKernel::algo_comm_class_t, 991
repl_list
MHAParser::base_t, 1043
repl_list_t
MHAParser::base_t, 1034
replace
MHAParser::base_t::replace_t, 1044
MHASignal::loop_wavefragment_t, 1222
replace_
cfg_t, 345
replace_t
MHAParser::base_t::replace_t, 1044
res_c

© 2005-2021 HörTech gGmbH, Oldenburg

ac_mul_t, 195
res_r
ac_mul_t, 195
resampled_num_frames
addsndfile, 81
resampled_soundfile_t
addsndfile::resampled_soundfile_t, 258
resampling
MHAFilter::blockprocessing_polyphase_resampling_t, 871
resampling.cpp, 1699
resampling_factors
MHAFilter, 109
resampling_filter_t
MHAFilter::resampling_filter_t, 929
resampling_if_t
MHAPlugin_Resampling::resampling_if_t, 1158
resampling_t
MHAPlugin_Resampling::resampling_t, 1160
resamplingmode
addsndfile::addsndfile_if_t, 253
reset
droptect_t, 442
MHA_TCP::Async_Notify, 799
MHA_TCP::Wakeup_Event, 838
reset_state
MHAFilter::gamma_flt_t, 895
resize
MHAFilter::iir_filter_t, 900
resolution
acTransform_wave_config, 239
resolve
dynamiclib_t, 448
pluginlib_t, 1362
resolve_and_init
PluginLoader::mhapluginloader_t, 1373
resolve_checked
dynamiclib_t, 448
restore_state
altconfig_t, 298
result
cpuload::cpuload_cfg_t, 365
resynthesis_gain
gtfb_simple_t, 574
MHAFilter::gamma_flt_t, 896
ret_size
MHAParser::c_ifc_parser_t, 1052
return_imag
fftfilterbank::fftfb_interface_t, 506
return_imag_
fftfilterbank::fftfb_plug_t, 509
return_sig
audiometerbackend, 83
return_value
MHA_TCP::Thread, 832
return_wave
wave2spec_if_t, 1496
rewind
MHAParser::c_ifc_parser_t, 1052
rewind
MHASignal::loop_wavefragment_t, 1224
rho
adaptive_feedback_canceller, 242
nlms_t, 1311
ringbuffer
MHAFilter::polyphase_resampling_t, 928
ringbuffer_t
MHASignal::ringbuffer_t, 1241
rinputs
gtfb_simd_cfg_t, 560
rm_parent_on_remove
MHAParser::base_t, 1041
rms
levelmeter_t, 660
MHASignal::loop_wavefragment_t, 1222
plingploing::plingploing_t, 1348
rms_limit40
MHASignal::loop_wavefragment_t, 1222
rmsdb
example6_t, 484
rmslevel, 158
calibrator_variables_t, 342
dc::dc_t, 385
HL, 159
MHASignal::async_rmslevel_t, 1201
SPL, 159
UNIT, 159
Vector and matrix processing toolbox, 53, 55
rmslevel.cpp, 1699
rmslevel::mon_t, 1390
mon_t, 1391
operator=, 1391
store, 1391
rmslevel::rmslevel_if_t, 1392
name, 1394
patchbay, 1394
prepare, 1393
process, 1393
rmslevel_if_t, 1393

© 2005-2021 HörTech gGmbH, Oldenburg

unit, 1394
update, 1393
rmslevel::rmslevel_t, 1394
~rmslevel_t, 1395
domain, 1397
fftlen, 1397
fill, 1396
freq_offsets, 1397
insert, 1396
monitors, 1397
process, 1396
rmslevel_t, 1395
sum_and_fill, 1396
unit, 1397
rmslevel_if_t
rmslevel::rmslevel_if_t, 1393
rmslevel_t
rmslevel::rmslevel_t, 1395
rmslevelmeter
transducers.cpp, 1712
rng
audiometerbackend::lnn3rdoct_t, 322
cfg_t, 346
rohBeam, 159
CONST_C, 160
j0, 159
refL, 160
refR, 160
rohBeam::rohBeam, 1400
scalarify, 160
rohBeam.cpp, 1700
rohBeam.hh, 1700
NDEBUG, 1701
rohBeam::configOptions, 1397
alpha_blocking_XkXi, 1398
alpha_blocking_XkY, 1398
alpha_postfilter, 1398
binaural_type_index, 1398
enable_adaptive_beam, 1398
rohBeam::rohBeam, 1399
~rohBeam, 1400
beamExport, 1405
binaural_type, 1404
compute_beamW, 1402
compute_delaycomp_vec, 1402
compute_diff2D, 1402
compute_diff3D, 1402
compute_head_model_alpha, 1401
compute_head_model_mat, 1401
compute_head_model_T, 1401
compute_uncorr, 1402
compute_wng, 1403
diag_loading_mu, 1404
enable_adaptive_beam, 1404
enable_export, 1405
enable_wng_optimization, 1405
export_beam_design, 1403
get_noise_model_func, 1403
head_model_sphere_radius_cm, 1404
intermic_distance_cm, 1404
mic_azimuth_degrees_vec, 1404
noise_field_model, 1404
noise_integrate_hrtf, 1402
noiseFuncPtr, 1403
noiseModelExport, 1406
on_model_param_valuechanged, 1403
patchbay, 1405
prepare, 1401
prepared, 1405
process, 1401
prop_type, 1403
propExport, 1406
release, 1401
rohBeam, 1400
sampled_hrir_path, 1403
solve_MVDR, 1402
source_azimuth_degrees, 1404
tau_blocking_XkXi_ms, 1405
tau_blocking_XkY_ms, 1405
tau_postfilter_ms, 1405
update_cfg, 1401
rohBeam::rohConfig, 1406
~rohConfig, 1408
alpha_blocking_XkXi, 1410
alpha_blocking_XkY, 1410
alpha_postfilter, 1410
beam1, 1410
beamA, 1410
beamW, 1410
binaural_type_index, 1409
blockSpec, 1410
blockXp, 1411
copyfixedbfoutput, 1409
corrLL, 1411
corrRR, 1411
corrXpXp, 1411
corrXpYf, 1411
corrZZ, 1411
delayComp, 1410
enable_adaptive_beam, 1409
freqResp, 1411
headModel, 1409

© 2005-2021 HörTech gGmbH, Oldenburg

hhCorrXpXp, 1411
in_cfg, 1409
init_dynamic, 1408
magResp, 1412
maxLim, 1412
minLim, 1412
nchan_block, 1409
nextXpYf, 1411
nfreq, 1409
operator=, 1408
out_cfg, 1409
outSpec, 1410
phasereconstruction, 1408
postfilter, 1408
process, 1408
rohConfig, 1407, 1408
rohConfig
rohBeam::rohConfig, 1407, 1408
root
mha_rt_fifo_t< T >, 793
rotated_i
acTransform_wave_config, 239
rotated_p
acTransform_wave_config, 239
rotated_p_max_name
acTransform_wave, 236
rotated_p_name
acTransform_wave, 236
route, 160
route.cpp, 1701
route::interface_t, 1412
algo, 1415
cfac, 1415
cfin, 1415
cfout, 1415
interface_t, 1413
patchbay, 1414
prepare, 1413
prepared, 1415
process, 1414
release, 1414
route_ac, 1414
route_out, 1414
stopped, 1415
update, 1414
route::process_t, 1415
process, 1416
process_t, 1416
sout, 1417
sout_ac, 1417
wout, 1416
wout_ac, 1417
route_ac
route::interface_t, 1414
route_out
route::interface_t, 1414
rows
acsave::mat4head_t, 224
rstates
gtfb_simd_cfg_t, 560
rt_nlms_t, 1417
~rt_nlms_t, 1418
ac, 1419
channels, 1419
F, 1419
frames, 1419
fu, 1420
fu_previous, 1420
fuflt, 1420
insert, 1419
n_no_update_, 1421
name_d_, 1421
name_e_, 1421
name_u_, 1421
no_iter, 1421
ntaps, 1419
P_Sum, 1421
process, 1419
Pu, 1420
rt_nlms_t, 1418
s_E, 1421
U, 1420
Uflt, 1420
y_previous, 1420
rt_process
analysepath_t, 307
rt_strict
ac2lsl::ac2lsl_t, 165
ac2osc_t, 184
rtcalibrator
transducers.cpp, 1712
rtmem
ac2osc_t, 184
run
mha_tcp::server_t, 822
MHA_TCP::Thread, 831
mhaserver_t, 1196
RUNNING
MHA_TCP::Thread, 830
runtime configuration, 4
rval
MHAParser::expression_t, 1061

© 2005-2021 HörTech gGmbH, Oldenburg

s_b
lpc_bl_predictor_config, 670
lpc_burglattice_config, 676
s_E
adaptive_feedback_canceller_config, 246
rt_nlms_t, 1421
s_E_afc_delay
adaptive_feedback_canceller_config, 247
s_f
lpc_bl_predictor_config, 670
lpc_burglattice_config, 676
s_file_in
io_file_t, 610
s_in
ac_proc::interface_t, 199
io_asterisk_sound_t, 600
io_file_t, 610
io_parser_t, 620
io_tcp_sound_t, 637
MHAIOPortAudio::io_portaudio_t, 962
MHAJack::client_t, 983
s_in_perm
ac_proc::interface_t, 198
s_LPC
adaptive_feedback_canceller_config, 249
s_out
ac_proc::interface_t, 198
coherence::cohflt_t, 352
combc_t, 359
fftfilterbank::fftfb_plug_t, 509
gtfb_analyzer::gtfb_analyzer_cfg_t, 552
gtfb_simd_cfg_t, 561
io_file_t, 610
io_parser_t, 620
MHAIOPortAudio::io_portaudio_t, 962
MHAJack::client_t, 983
s_U
adaptive_feedback_canceller_config, 247
s_U_delay
adaptive_feedback_canceller_config, 248
s_U_delayflt
adaptive_feedback_canceller_config, 248
s_Usmpl
adaptive_feedback_canceller_config, 249
s_W
adaptive_feedback_canceller_config, 248
s_Wflt
adaptive_feedback_canceller_config, 248
s_Y_delay
adaptive_feedback_canceller_config, 248
s_Y_delayflt
adaptive_feedback_canceller_config, 248
safe_div
Complex arithmetics in the openMHA, 66
mha_signal.cpp, 1632
mha_signal.hh, 1642
sample
lpc_config, 678
sampled_hrir_path
rohBeam::rohBeam, 1403
samplerate
io_asterisk_sound_t, 599
io_file_t, 608
io_tcp_sound_t, 637
MHAIOPortAudio::io_portaudio_t, 962
MHAJack::client_t, 982
samples_AC
acConcat_wave, 201
samplingrate
MHAOvlFilter::fftfb_t, 1010
save_m
acsave::save_var_t, 226
save_mat4
acsave::save_var_t, 226
save_spec.cpp, 1701
save_spec_t, 1422
basename, 1423
prepare, 1423
process, 1423
save_spec_t, 1422
save_state
altconfig_t, 298
save_txt
acsave::save_var_t, 226
save_var_t
ac2lsl::save_var_t< mha_complex_t >, 176
ac2lsl::save_var_t< T >, 172
acsave::save_var_t, 226
lsl2ac::save_var_t, 688
save_vars
acmon::acmon_t, 209
save_wave.cpp, 1701
save_wave_t, 1424
basename, 1425
prepare, 1425
process, 1425
save_wave_t, 1424
saveas_mat4
MHASignal, 148, 149
sc
MHASignal::hilbert_fftw_t, 1218

© 2005-2021 HörTech gGmbH, Oldenburg

spec2wave_t, 1470
scalarify
rohBeam, 160
scale
cfg_t, 345
delaysum_spec::delaysum_t, 421
example5_t, 482
MHASignal, 144
MHASignal::fft_t, 1215
MHASignal::spectrum_t, 1252
MHASignal::waveform_t, 1274
scale_ch
complex_scale_channel_t, 363
example2_t, 472
example3_t, 476
example4_t, 480
plugin_interface_t, 1356
scale_channel
MHASignal::spectrum_t, 1254
MHASignal::waveform_t, 1275
scale_frame
MHASignal::waveform_t, 1275
scale_fun_t
MHAOvlFilter, 116
scale_var_t
MHAOvlFilter::scale_var_t, 1029
scalefac
MHATableLookup::linear_table_t, 1283
scaler_t
gain::scaler_t, 536
scan_dir
addsndfile::addsndfile_if_t, 252
scan_plugin
pluginbrowser_t, 1357
scan_plugins
pluginbrowser_t, 1357
scan_syntax
ac_mul_t, 193
scheduler
analysepath_t, 309
dbasync_native::dbasync_t, 378
MHAPlugin_Split::posix_threads_t, 1175
schroeder_t
MHASignal::schroeder_t, 1246, 1247
search_pattern
addsndfile::addsndfile_if_t, 254
search_result
addsndfile::addsndfile_if_t, 254
sec2smp
MHASignal, 144
seed
noise_t, 1320
select_plug
altconfig_t, 299
altplugs_t, 304
select_source
MHAMultiSrc::base_t, 996
selectall
altconfig_t, 299
selected_plug
altplugs_t, 304
send_frame
ac2lsl::save_var_base_t, 170
ac2lsl::save_var_t< mha_complex_t >, 178
ac2lsl::save_var_t< T >, 174
send_osc_float
ac2osc_t, 182
send_port_announcement
mhaserver_t, 1195
Server
MHA_TCP::Server, 816
server
io_asterisk_t, 604
io_tcp_t, 642
server_fragsize
MHAIOJackdb::io_jack_t, 954
server_port_open
io_asterisk_parser_t, 595
io_tcp_parser_t, 632
server_srate
MHAIOJackdb::io_jack_t, 954
server_start
osc_server_t, 1325
server_stop
osc_server_t, 1325
server_t
mha_tcp::server_t, 820
servername
MHAIOJack::io_jack_t, 945
MHAIOJackdb::io_jack_t, 953
serversocket
MHA_TCP::Server, 818
set
Complex arithmetics in the openMHA, 60, 61
MHA_TCP::Async_Notify, 799
testplugin::config_parser_t, 1484
set_announce_port
mhaserver_t, 1195
set_buf_address
ac2lsl::save_var_base_t, 170

© 2005-2021 HörTech gGmbH, Oldenburg

ac2lsl::save_var_t< mha_complex_t >, 177
ac2lsl::save_var_t< T >, 173
set_channelcnt
MHAFilter::adapt_filter_t, 867
set_connected
io_asterisk_parser_t, 593
io_tcp_parser_t, 630
set_entries
MHAParser::keyword_list_t, 1073
set_errnos
io_asterisk_fwcb_t, 587
io_tcp_fwcb_t, 623
set_error
mha_fifo_lw_t< T >, 773
set_fb_pars
DynComp::dc_afterburn_t, 454
set_format
wavwriter_t, 1507
set_help
MHAParser::base_t, 1040
set_id_string
MHAParser::parser_t, 1107
set_index
MHAParser::keyword_list_t, 1074
set_input_domain
MHAPlugin_Split::domain_handler_t, 1165
set_input_portnames
MHAJack::client_t, 979
set_level
addsndfile::addsndfile_if_t, 252
audiometerbackend::audiometer_if_t, 316
fader_wave::fader_wave_if_t, 491
set_level_db
MHASignal::loop_wavefragment_t, 1224
set_level_lin
MHASignal::loop_wavefragment_t, 1224
set_local_port
io_asterisk_parser_t, 592
io_tcp_parser_t, 628
set_locate
MHAIOJackdb::io_jack_t, 952
set_max_angle_ind
parser_int_dyn, 1339
set_minabs
mha_signal.cpp, 1632
mha_signal.hh, 1642
set_new_peer
io_asterisk_parser_t, 594
io_tcp_parser_t, 630
set_node_id
MHAParser::base_t, 1040
set_output_domain
MHAPlugin_Split::domain_handler_t, 1165
set_output_portnames
MHAJack::client_t, 979
set_parse_cb
MHAParser::c_ifc_parser_t, 1051
set_range
MHAParser::kw_t, 1077
MHAParser::range_var_t, 1110
set_server_port_open
io_asterisk_parser_t, 592
io_tcp_parser_t, 629
set_state
MHAFilter::complex_bandpass_t, 874
MHAFilter::iir_ord1_real_t, 904
set_tau
MHAFilter::o1flt_lowpass_t, 912
MHAFilter::o1flt_maxtrack_t, 914
MHAFilter::o1flt_mintrack_t, 916
set_tau_attack
MHAFilter::o1_ar_filter_t, 908
set_tau_release
MHAFilter::o1_ar_filter_t, 908
set_use_jack_transport
MHAIOJackdb::io_jack_t, 952
MHAJack::client_t, 980
set_value
MHAParser::keyword_list_t, 1073
set_weights
MHAFilter::complex_bandpass_t, 874
MHAFilter::gamma_flt_t, 895
set_xfun
MHATableLookup::xy_table_t, 1288
set_xmax
MHATableLookup::linear_table_t, 1281
set_xmin
MHATableLookup::linear_table_t, 1281
set_xyfun
MHATableLookup::xy_table_t, 1289
set_yfun
MHATableLookup::xy_table_t, 1288
setlock
gtfb_simple_t, 572
io_file_t, 608
lsl2ac::lsl2ac_t, 684
MHAParser::variable_t, 1119
MHAParser::window_t, 1141
osc2ac_t, 1323

© 2005-2021 HörTech gGmbH, Oldenburg

overlapadd::overlapadd_if_t, 1332
spec2wave_if_t, 1467
testplugin::config_parser_t, 1484
wave2spec_if_t, 1495
windowselector_t, 1520
sf
MHASndFile::sf_t, 1277
wavwriter_t, 1508
sf_in
io_file_t, 610
sf_out
io_file_t, 611
sf_t
MHASndFile::sf_t, 1276
sf_wave_t
MHASndFile::sf_wave_t, 1278
sfinf_in
io_file_t, 611
sfinf_out
io_file_t, 611
shadowfilter_begin, 160
shadowfilter_begin.cpp, 1702
shadowfilter_begin::cfg_t, 1425
cfg_t, 1426
in_spec_copy, 1426
nch, 1426
ntracks, 1427
out_spec, 1426
process, 1426
shadowfilter_begin::shadowfilter_begin_t, 1427
basename, 1428
nch, 1429
ntracks, 1429
prepare, 1428
process, 1428
shadowfilter_begin_t, 1428
shadowfilter_begin_t
shadowfilter_begin::shadowfilter_begin_t, 1428
shadowfilter_end, 161
shadowfilter_end.cpp, 1702
shadowfilter_end::cfg_t, 1429
ac, 1430
cfg_t, 1429
gains, 1431
in_spec, 1430
name, 1430
nch_out, 1430
nfft, 1430
ntracks, 1430
out_spec, 1431
process, 1430
shadowfilter_end::shadowfilter_end_t, 1431
basename, 1432
prepare, 1432
process, 1432
shadowfilter_end_t, 1432
shadowfilter_end_t
shadowfilter_end::shadowfilter_end_t, 1432
shape
MHAOvlFilter::fftfb_t, 1010
shapes
MHAOvlFilter::fftfb_vars_t, 1014
shift
lpc, 663
lpc_config, 677
shifted
fshift_hilbert::hilbert_shifter_t, 520
shutdown
mha_tcp::server_t, 823
side
mha_channel_info_t, 745
sign_t
MHASignal::schroeder_t, 1246
signal
testplugin::if_t, 1488
signal_counter
MHASignal, 149
signal_dimensions
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 710
signal_gen_t
audiometerbackend::signal_gen_t, 323
signal_out
MHAPlugin_Split::split_t, 1180
signal_parser_t
testplugin::signal_parser_t, 1489
signal_type
matlab_wrapper::types< MHA_SPECTRUM >, 711
matlab_wrapper::types< MHA_WAVEFORM >, 712
sigtype
audiometerbackend::audiometer_if_t, 317
sin125
speechnoise_t, 1473
sin1k
speechnoise_t, 1473
sin250
speechnoise_t, 1473

© 2005-2021 HörTech gGmbH, Oldenburg

sin2k
speechnoise_t, 1473
sin4k
speechnoise_t, 1473
sin500
speechnoise_t, 1473
sin8k
speechnoise_t, 1473
sinc
MHAFilter, 109
sine.cpp, 1702
sine_cfg_t, 1433
amplitude, 1434
channels, 1434
mix, 1434
phase_increment_div_2pi, 1434
sine_cfg_t, 1433
sine_t, 1435
audiometerbackend::sine_t, 324
channels, 1437
frequency, 1437
lev, 1437
mode, 1437
patchbay, 1437
phase_div_2pi, 1437
prepare, 1436
process, 1436
sine_t, 1436
update_cfg, 1437
sInput
MHAFilter::fftfilter_t, 881
size
MHA_AC::ac2matrix_helper_t, 722
MHA_AC::acspace2matrix_t, 728
MHAKernel::algo_comm_class_t, 993
MHASignal::matrix_t, 1231
osc2ac_t, 1323
Vector and matrix processing toolbox, 43, 44
size_t
MHAParser::keyword_list_t, 1072
skip
ac2lsl::ac2lsl_t, 165
ac2lsl::cfg_t, 168
ac2osc_t, 184
lsl2ac::save_var_t, 692
skipcnt
ac2lsl::cfg_t, 168
ac2osc_t, 184
Sleep
mha_tcp.hh, 1648
slope
softclipper_t, 1463
softclipper_variables_t, 1465
slope_db
softclip_t, 1460
smooth_cepstrum, 161
smooth_cepstrum.cpp, 1702
INSERT_PATCH, 1703
INSERT_VAR, 1703
PATCH_VAR, 1703
smooth_cepstrum.hh, 1703
smooth_cepstrum::smooth_cepstrum_if_t, 1438
alpha_const_limits_hz, 1442
alpha_const_vals, 1442
alpha_pitch, 1441
beta_const, 1441
delta_pitch, 1441
f0_high, 1441
f0_low, 1441
gain_min_db, 1442
kappa_const, 1441
lambda_thresh, 1441
noisePow_name, 1442
on_model_param_valuechanged, 1440
patchbay, 1443
prepare, 1440
prepared, 1443
prior_q, 1442
process, 1439
release, 1440
smooth_cepstrum_if_t, 1439
spp, 1442
update_cfg, 1440
win_f0, 1442
xi_min_db, 1441
xi_opt_db, 1442
smooth_cepstrum::smooth_cepstrum_t, 1443
~smooth_cepstrum_t, 1444
ac, 1445
alpha_const, 1446
alpha_frame, 1448
alpha_hat, 1447
alpha_prev, 1446
fftlen, 1445
gain_min, 1446
gain_wiener, 1448
gamma_post, 1447
GLR, 1449
GLRexp, 1449
lambda_ceps, 1448

© 2005-2021 HörTech gGmbH, Oldenburg

lambda_ceps_prev, 1448
lambda_ml_ceps, 1447
lambda_ml_full, 1447
lambda_ml_smooth, 1447
lambda_spec, 1448
log_lambda_spec, 1448
logGLRFact, 1449
max_q, 1449
max_val, 1449
mha_fft, 1445
nchan, 1446
nfreq, 1445
noisePow, 1447
ola_powspec_scale, 1446
operator=, 1445
params, 1445
pitch_set_first, 1449
pitch_set_last, 1449
powSpec, 1447
priorFact, 1449
process, 1445
q_high, 1446
q_low, 1446
smooth_cepstrum_t, 1444
spec_out, 1448
winF0, 1446
xi_est, 1448
xi_min, 1446
xi_ml, 1447
xiOpt, 1449
smooth_cepstrum::smooth_params, 1450
alpha_const_limits_hz, 1452
alpha_const_vals, 1452
alpha_pitch, 1452
beta_const, 1452
delta_pitch, 1451
f0_high, 1451
f0_low, 1451
gain_min_db, 1452
in_cfg, 1451
kappa_const, 1452
lambda_thresh, 1451
noisePow_name, 1453
prior_q, 1452
smooth_params, 1450
winF0, 1452
xi_min_db, 1451
xi_opt_db, 1452
smooth_cepstrum_if_t
smooth_cepstrum::smooth_cepstrum_if_t, 1439
smooth_cepstrum_t
smooth_cepstrum::smooth_cepstrum_t, 1444
smooth_params
smooth_cepstrum::smooth_params, 1450
smoothgains_bridge, 161
smoothgains_bridge.cpp, 1703
smoothgains_bridge::overlapadd_if_t, 1453
~overlapadd_if_t, 1454
algo, 1456
cf_in, 1456
cf_out, 1456
epsilon, 1455
irswnd, 1455
mode, 1455
overlapadd_if_t, 1454
patchbay, 1455
plugloader, 1455
prepare, 1454
process, 1455
release, 1454
update, 1455
smoothgains_bridge::smoothspec_wrap_t, 1456
proc_1, 1457
proc_2, 1457
smoothspec, 1457
smoothspec_epsilon, 1458
smoothspec_wrap_t, 1457
spec_in_copy, 1457
use_smoothspec, 1457
smoothspec
MHAFilter::smoothspec_t, 932
smoothgains_bridge::smoothspec_wrap_t, 1457
smoothspec_epsilon
smoothgains_bridge::smoothspec_wrap_t, 1458
smoothspec_t
MHAFilter::smoothspec_t, 931
smoothspec_wrap_t
smoothgains_bridge::smoothspec_wrap_t, 1457
smp2rad
Vector and matrix processing toolbox, 40
smp2sec
MHASignal, 143
smpl
adaptive_feedback_canceller_config, 249
sn_in
MHAJack::client_avg_t, 970

© 2005-2021 HörTech gGmbH, Oldenburg

MHAJack::client_noncont_t, 973
sn_out
MHAJack::client_avg_t, 970
MHAJack::client_noncont_t, 973
sndfile_t
addsndfile::sndfile_t, 259
snprintf_required_length
mha_error_helpers, 100
snrPost1Debug
noise_psd_estimator::noise_psd_estimator_t, 1317
sock_addr
MHA_TCP::Server, 818
sock_init_t
MHA_TCP::sock_init_t, 825
sock_initializer
MHA_TCP, 103
Sockaccept_Event
MHA_TCP::Sockaccept_Event, 826
SOCKET
MHA_TCP, 102
mha_tcp.cpp, 1646
SOCKET_ERROR
mha_tcp.cpp, 1645
Sockread_Event
MHA_TCP::Sockread_Event, 827
Sockwrite_Event
MHA_TCP::Sockwrite_Event, 828
softclip
calibrator_runtime_layer_t, 337
calibrator_variables_t, 343
softclip.cpp, 1704
softclip_t, 1458
attack, 1460
decay, 1460
patchbay, 1460
prepare, 1459
process, 1459
slope_db, 1460
softclip_t, 1459
start_limit, 1460
tftype, 1460
update, 1459
softclipper_t, 1460
attack, 1462
clipmeter, 1462
decay, 1462
hardlimit, 1463
linear, 1463
process, 1462
slope, 1463
softclipper_t, 1461
threshold, 1462
softclipper_variables_t, 1463
clipped, 1465
hardlimit, 1465
linear, 1465
max_clipped, 1465
slope, 1465
softclipper_variables_t, 1464
tau_attack, 1464
tau_clip, 1465
tau_decay, 1464
threshold, 1465
solve_MVDR
rohBeam::rohBeam, 1402
sort_fftw2spec
MHASignal::fft_t, 1214
sort_spec2fftw
MHASignal::fft_t, 1214
sound
io_asterisk_t, 603
io_tcp_t, 641
source_azimuth_degrees
rohBeam::rohBeam, 1404
source_channel_index
MHAFilter::partitioned_convolution_t::index_t, 923
MHAFilter::transfer_function_t, 940
source_id
ac2lsl::ac2lsl_t, 165
ac2lsl::cfg_t, 168
sout
matrixmixer::cfg_t, 714
route::process_t, 1417
sout_ac
route::process_t, 1417
sout_buf
gtfb_simd_cfg_t, 561
spec2fir
MHAFilter, 108
MHAFilter::smoothspec_t, 933
spec2spec
plugindescription_t, 1360
spec2wave
MHASignal::fft_t, 1212, 1213
overlapadd::overlapadd_t, 1336
plugindescription_t, 1360
spec2wave.cpp, 1704
max, 1704
min, 1704
spec2wave_if_t, 1466

© 2005-2021 HörTech gGmbH, Oldenburg

prepare, 1467
process, 1467
ramplen, 1468
release, 1467
setlock, 1467
spec2wave_if_t, 1467
update, 1467
window_config, 1468
spec2wave_scale
MHASignal::fft_t, 1213
spec2wave_t, 1468
~spec2wave_t, 1469
calc_out, 1470
ft, 1469
nfft, 1470
npad1, 1469
npad2, 1469
nwndshift, 1470
out_buf, 1470
postwindow, 1470
process, 1469
ramps, 1470
sc, 1470
spec2wave_t, 1469
write_buf, 1470
spec_fader_t, 1471
~spec_fader_t, 1471
fr, 1471
gains, 1471
nch, 1471
spec_fader_t, 1471
spec_in
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 710
MHAPlugin_Split::domain_handler_t, 1169
overlapadd::overlapadd_t, 1337
wave2spec_t, 1502
spec_in_copy
smoothgains_bridge::smoothspec_wrap_t, 1457
spec_out
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 710
MHAPlugin_Split::domain_handler_t, 1169
MHAPlugin_Split::split_t, 1182
smooth_cepstrum::smooth_cepstrum_t, 1448
specSteer1
acSteer_config, 232
specSteer2
acSteer_config, 232
spectrum_t
MHA_AC::spectrum_t, 736
MHAMultiSrc::spectrum_t, 999
MHASignal::spectrum_t, 1249, 1250
speechnoise
calibrator_runtime_layer_t, 337
speechnoise.cpp, 1704
bandw_correction, 1707
erb_hz_f_hz, 1706
fhz2bandno, 1706
hz2hz, 1706
NUM_ENTR_LTASS, 1706
NUM_ENTR_MHAORIG, 1706
NUM_ENTR_OLNOISE, 1706
vLTASS_combined_lev, 1708
vLTASS_female_lev, 1708
vLTASS_freq, 1707
vLTASS_male_lev, 1708
vMHAOrigFreq, 1707
vMHAOrigSpec, 1707
vOlnoiseFreq, 1708
vOlnoiseLev, 1708
speechnoise.h, 1708
speechnoise_t, 1472
brown, 1473
creator, 1474
LTASS_combined, 1473
LTASS_female, 1473
LTASS_male, 1473
mha, 1473
noise_type_t, 1473
olnoise, 1473
pink, 1473
sin125, 1473
sin1k, 1473
sin250, 1473
sin2k, 1473
sin4k, 1473
sin500, 1473
sin8k, 1473
speechnoise_t, 1473, 1474
TEN_SPL, 1473
TEN_SPL_250_8k, 1473
TEN_SPL_50_16k, 1473
white, 1473
SPL
rmslevel, 159
spl2hl
MHAUtils, 153

© 2005-2021 HörTech gGmbH, Oldenburg

split.cpp, 1709
MHAPLUGIN_OVERLOAD_OUTDOMAIN, 1710
native_thread_platform_type, 1710
posixthreads, 1710
split_t
MHAPlugin_Split::split_t, 1178
splitted_part_t
MHAPlugin_Split::splitted_part_t, 1184, 1185
spnoise_channels
calibrator_variables_t, 342
spnoise_level
calibrator_variables_t, 342
spnoise_mode
calibrator_variables_t, 342
spnoise_parser
calibrator_variables_t, 342
spp
smooth_cepstrum::smooth_cepstrum_if_t, 1442
sq2db
MHASignal, 140
srate
ac2lsl::cfg_t, 168
adm_if_t, 275
calibrator_variables_t, 342
mhaconfig_t, 853
MHAParser::mhaconfig_mon_t, 1090
MHAPlugin_Resampling::resampling_if_t, 1159
testplugin::config_parser_t, 1485
srate_
MHAFilter::gamma_flt_t, 896
srv
osc2ac_t, 1323
start
alsa_base_t, 287
alsa_t< T >, 293
fw_t, 526
io_alsa_t, 581
io_asterisk_fwcb_t, 586
io_asterisk_t, 602
io_file_t, 607
io_lib_t, 614
io_parser_t, 618
io_tcp_fwcb_t, 623
io_tcp_t, 640
MHAJack::client_t, 978
START_BETA
ADM, 82
start_event
io_alsa_t, 583
io_asterisk_fwcb_t, 588
io_file_t, 609
io_parser_t, 620
io_tcp_fwcb_t, 624
MHAIOPortAudio::io_portaudio_t, 963
MHAJack::client_t, 982
start_handle
io_alsa_t, 583
io_asterisk_fwcb_t, 588
io_file_t, 609
io_parser_t, 620
io_tcp_fwcb_t, 625
MHAIOPortAudio::io_portaudio_t, 963
MHAJack::client_t, 982
start_limit
softclip_t, 1460
start_lin
cfg_t, 346
start_new_session
plugins::hoertech::acrec::acrec_t, 1380
wavrec_t, 1504
start_stdin_thread
mhaserver_t, 1195
started
fw_t, 527
io_parser_t, 619
starting
mha_drifter_fifo_t< T >, 761
startpos
addsndfile::addsndfile_if_t, 253
startsample
io_file_t, 610
startup_zeros
mha_drifter_fifo_t< T >, 764
stat_t
MHA_AC::stat_t, 738
MHASignal::stat_t, 1255
state
fw_t, 531
gtfb_analyzer::gtfb_analyzer_cfg_t, 553
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 708
MHA_TCP::Thread, 832
MHAFilter::filter_t, 892
state_cpuload
MHAIOJack::io_jack_t, 947
MHAIOJackdb::io_jack_t, 955
state_parser
MHAIOJack::io_jack_t, 947

© 2005-2021 HörTech gGmbH, Oldenburg

MHAIOJackdb::io_jack_t, 955
state_priority
MHAIOJack::io_jack_t, 947
MHAIOJackdb::io_jack_t, 955
state_scheduler
MHAIOJack::io_jack_t, 947
MHAIOJackdb::io_jack_t, 955
state_t
fw_t, 525
state_xruns
MHAIOJack::io_jack_t, 947
MHAIOJackdb::io_jack_t, 955
states
gtfb_analyzer::gtfb_analyzer_cfg_t, 551
staticgain
coherence::cohflt_t, 353
coherence::vars_t, 355
status
MHA_TCP::Wakeup_Event, 838
std
MHA_AC::stat_t, 739
std_vector_float
Vector and matrix processing toolbox, 49
std_vector_vector_complex
Vector and matrix processing toolbox, 49
std_vector_vector_float
Vector and matrix processing toolbox, 49
stdcomplex
Complex arithmetics in the openMHA, 61
steerbf, 1475
~steerbf, 1476
angle_ind, 1477
angle_src, 1477
bf_src, 1477
patchbay, 1477
prepare, 1476
process, 1476
release, 1477
steerbf, 1476
update_cfg, 1477
steerbf.cpp, 1710
INSERT_PATCH, 1710
PATCH_VAR, 1710
steerbf.h, 1711
steerbf_config, 1478
_steerbf, 1479
~steerbf_config, 1478
ac, 1479
bf_src_copy, 1479
bf_vec, 1479
nangle, 1479
nchan, 1479
nfreq, 1479
outSpec, 1479
process, 1478
steerbf_config, 1478
steerFile
acSteer, 230
stime
MHA_TCP, 103
stop
alsa_base_t, 287
alsa_t< T >, 293
fw_t, 526
io_alsa_t, 582
io_asterisk_fwcb_t, 587
io_asterisk_t, 602
io_file_t, 607
io_lib_t, 614
io_parser_t, 618
io_tcp_fwcb_t, 624
io_tcp_t, 640
mha_drifter_fifo_t< T >, 761
MHAJack::client_t, 978
stop_event
io_alsa_t, 583
io_asterisk_fwcb_t, 588
io_file_t, 609
io_parser_t, 620
io_tcp_fwcb_t, 624
MHAIOPortAudio::io_portaudio_t, 963
MHAJack::client_t, 982
stop_handle
io_alsa_t, 583
io_asterisk_fwcb_t, 588
io_file_t, 609
io_parser_t, 620
io_tcp_fwcb_t, 625
MHAIOPortAudio::io_portaudio_t, 963
MHAJack::client_t, 982
stopped
fw_t, 526, 527
io_file_t, 607
io_parser_t, 619
MHAJack::client_t, 981
route::interface_t, 1415
store
rmslevel::mon_t, 1391
store_frame
acsave::cfg_t, 223
acsave::save_var_t, 226
str2val

© 2005-2021 HörTech gGmbH, Oldenburg

MHAParser::StrCnv, 130, 131
str2val< mha_real_t >
MHAParser::StrCnv, 131
str_a
ac_mul_t, 195
str_b
ac_mul_t, 195
str_error
MHAJack::client_t, 979
strdom
analysemhaplugin.cpp, 1535
latex_doc_t, 646
stream
ac2lsl::save_var_t< mha_complex_t >, 178
ac2lsl::save_var_t< T >, 175
lsl2ac::save_var_t, 690
stream_dir
alsa_dev_par_parser_t, 290
stream_info
MHAIOPortAudio::io_portaudio_t, 962
stream_info_t
MHAIOPortAudio::stream_info_t, 965
streambuf
mha_tcp::buffered_socket_t, 801
streams
lsl2ac::lsl2ac_t, 684
STRERROR
MHA_TCP, 102
strict_channel_match
io_file_t, 610
strict_srate_match
io_file_t, 610
strict_window_ratio
overlapadd::overlapadd_if_t, 1333
wave2spec_if_t, 1496
stride
comm_var_t, 361
testplugin::ac_parser_t, 1482
string_mon_t
MHAParser::string_mon_t, 1114
string_t
MHAParser::string_t, 1116
strip
MHAUtils, 151
STRLEN
mha_errno.c, 1590
strNames_AC
acConcat_wave_config, 203
strreplace
MHAParser, 127
structVersion
MHAIOPortAudio::device_info_t, 957
sub4f
gtfb_simd.cpp, 1562
sub_ac
dbasync_native::db_if_t, 375
subsample_delay_t
MHASignal::subsample_delay_t, 1257
subsampledelay_coeff
ADM, 81
suggestedInputLatency
MHAIOPortAudio::io_portaudio_t, 964
suggestedOutputLatency
MHAIOPortAudio::io_portaudio_t, 964
sum
MHASignal::stat_t, 1256
MHASignal::waveform_t, 1269
sum2
MHASignal::stat_t, 1256
sum_and_fill
rmslevel::rmslevel_t, 1396
sum_channel
MHASignal::waveform_t, 1270
sumsqr
MHASignal::waveform_t, 1270
sumsqr_channel
Vector and matrix processing toolbox, 56
sumsqr_frame
Vector and matrix processing toolbox, 57
svc
analysepath_t, 307
dbasync_native::dbasync_t, 378
sWeights
MHAFilter::fftfilter_t, 881
symmetry_scale
MHAOvlFilter::fspacing_t, 1021
sync
mha_fifo_lw_t< T >, 773
mha_fifo_thread_guard_t, 784
sync_osc2ac
osc_server_t, 1325
osc_variable_t, 1328
sysread
MHA_TCP::Connection, 806
syswrite
MHA_TCP::Connection, 806

T
MHA_TCP::OS_EVENT_TYPE, 814
t
acsave::mat4head_t, 224
mha_tictoc_t, 839

© 2005-2021 HörTech gGmbH, Oldenburg

plingploing::plingploing_t, 1346
table
cpuload::cpuload_cfg_t, 365
table_size
cpuload::cpuload_if_t, 368
table_t
MHATableLookup::table_t, 1284
tail
MHAFilter::fftfilterbank_t, 887
target_channel_index
MHAFilter::partitioned_convolution_t::index_t, 923
MHAFilter::transfer_function_t, 940
tau
coherence::vars_t, 354
droptect_t, 443
fader_if_t, 488
levelmeter_t, 660
tau_attack
softclipper_variables_t, 1464
tau_beta
adm_if_t, 274
tau_blocking_XkXi_ms
rohBeam::rohBeam, 1405
tau_blocking_XkY_ms
rohBeam::rohBeam, 1405
tau_clip
softclipper_variables_t, 1465
tau_decay
softclipper_variables_t, 1464
tau_level
calibrator_variables_t, 342
tau_Lowpass
windnoise::if_t, 1517
tau_postfilter_ms
rohBeam::rohBeam, 1405
tau_unit
coherence::vars_t, 354
tauattack
dc::dc_vars_t, 388
dc_simple::dc_vars_t, 405
taudecay
dc::dc_vars_t, 389
dc_simple::dc_vars_t, 405
taugain
DynComp::dc_afterburn_vars_t, 457
taurmslevel
dc::dc_vars_t, 388
tc
lsl2ac::save_var_t, 691
tc_name
lsl2ac::save_var_t, 691
tcp_connect_to
mha_tcp.cpp, 1646
tcp_connect_to_with_timeout
mha_tcp.cpp, 1646
tcp_server_t
mhaserver_t::tcp_server_t, 1198
tcpserver
mhaserver_t, 1196
TEN_SPL
speechnoise_t, 1473
TEN_SPL_250_8k
speechnoise_t, 1473
TEN_SPL_50_16k
speechnoise_t, 1473
termination_request
MHAPlugin_Split::posix_threads_t, 1176
test_error
io_lib_t, 614
MHAParser::c_ifc_parser_t, 1052
PluginLoader::mhapluginloader_t, 1373
test_fail
dc_simple, 87
test_prepare
testplugin::if_t, 1487
test_process
testplugin::if_t, 1487
test_version
PluginLoader::mhapluginloader_t, 1373
testalsadevice.c, 1711
main, 1711
testplugin, 161
testplugin.cpp, 1711
testplugin::ac_parser_t, 1480
_MHA_AC_CHAR, 1481
_MHA_AC_DOUBLE, 1481
_MHA_AC_FLOAT, 1481
_MHA_AC_INT, 1481
_MHA_AC_MHACOMPLEX, 1481
_MHA_AC_MHAREAL, 1481
_unknown, 1481
ac_parser_t, 1481
char_data, 1482
complex_data, 1482
data_type, 1482
data_type_t, 1481
do_get_var, 1481
do_insert_var, 1481
float_data, 1482
get_var, 1482
insert_var, 1482

© 2005-2021 HörTech gGmbH, Oldenburg

int_data, 1482
num_entries, 1482
patchbay, 1483
stride, 1482
testplugin::config_parser_t, 1483
channels, 1484
config_parser_t, 1484
domain, 1485
fftlen, 1485
fragsize, 1485
get, 1484
set, 1484
setlock, 1484
srate, 1485
wndlen, 1485
testplugin::if_t, 1486
_prepare, 1488
ac, 1488
config_in, 1488
config_out, 1488
if_t, 1487
patchbay, 1488
plug, 1488
prepare, 1487
process, 1487
signal, 1488
test_prepare, 1487
test_process, 1487
testplugin::signal_parser_t, 1489
input_spec, 1490
input_wave, 1490
output_spec, 1490
output_wave, 1490
signal_parser_t, 1489
tftype
MHAPlugin::plugin_t< runtime_cfg_t >, 1156
softclip_t, 1460
The MHA Framework interface, 22
The openMHA configuration language, 30
The openMHA Toolbox library, 31
thefullname
MHAParser::base_t, 1043
thirdoctave_analyzer_t
MHAFilter::thirdoctave_analyzer_t, 935
this_outer_out
MHASignal::doublebuffer_t, 1210
thr_f
MHA_TCP::Thread, 829
Thread
MHA_TCP::Thread, 830
thread
analysepath_t, 309
dbasync_native::dbasync_t, 378
io_asterisk_t, 604
io_tcp_t, 642
MHAPlugin_Split::posix_threads_t, 1175
MHAPlugin_Split::splitted_part_t, 1188
thread_arg
MHA_TCP::Thread, 832
thread_attr
MHA_TCP::Thread, 831
thread_finish_event
MHA_TCP::Thread, 832
thread_func
MHA_TCP::Thread, 832
thread_handle
MHA_TCP::Thread, 831
thread_platform
MHAPlugin_Split::split_t, 1181
thread_platform_t
MHAPlugin_Split::thread_platform_t, 1189
thread_start
analysispath.cpp, 1536
dbasync_native, 86
io_alsa_t, 582
MHAPlugin_Split::posix_threads_t, 1174
thread_start_func
mha_tcp.cpp, 1646
thread_startup_function
MHAIOAsterisk.cpp, 1660
MHAIOTCP.cpp, 1686
threshold
droptect_t, 443
softclipper_t, 1462
softclipper_variables_t, 1465
threshold_compare
windnoise::cfg_t, 1511
tic
lsl2ac::save_var_t, 692
tictoc
mhachain::plugs_t, 851
timeout
MHA_TCP::OS_EVENT_TYPE, 815
MHA_TCP::Timeout_Watcher, 835
Timeout_Event
MHA_TCP::Timeout_Event, 833
Timeout_Watcher
MHA_TCP::Timeout_Watcher, 835
timeshift
Vector and matrix processing toolbox, 45

© 2005-2021 HörTech gGmbH, Oldenburg

tmp
level_matching::level_matching_config_t, 653
tmp_spec
MHAFilter::smoothspec_t, 934
tmp_wave
MHAFilter::smoothspec_t, 934
to_from
acTransform_wave, 237
acTransform_wave_config, 239
to_int_clamped
mhaioutils, 111
to_iso8601
plugins::hoertech::acrec, 158
total_read
io_file_t, 611
transducers.cpp, 1712
kw_index2type, 1712
rmslevelmeter, 1712
rtcalibrator, 1712
vint_0123n1, 1713
transfer_function_t
MHAFilter::transfer_function_t, 938
trigger_accept
mha_tcp::server_t, 823
trigger_processing
MHAPlugin_Split::split_t, 1180
MHAPlugin_Split::splitted_part_t, 1186
trigger_read_line
mha_tcp::server_t, 823
trim
MHAParser, 126
try_accept
MHA_TCP::Server, 817
try_write
MHA_TCP::Connection, 809
ts
lsl2ac::save_var_t, 691
ts_buf
lsl2ac::save_var_t, 690
ts_name
lsl2ac::save_var_t, 691
ttl
ac2osc_t, 183
tv1
mha_tictoc_t, 839
tv2
mha_tictoc_t, 839
types
ac2lsl, 78
tz
mha_tictoc_t, 839
U
rt_nlms_t, 1420
UbufferPrew
adaptive_feedback_canceller_config, 248
UCL
AuditoryProfile::parser_t::ear_t, 328
AuditoryProfile::profile_t::ear_t, 333
Uflt
rt_nlms_t, 1420
uint_mode
addsndfile::addsndfile_if_t, 254
uint_vector_t
MHASignal::uint_vector_t, 1260
underflow
MHAFilter::polyphase_resampling_t, 927
UNIT
rmslevel, 159
unit
MHAOvlFilter::fscale_t, 1017
rmslevel::rmslevel_if_t, 1394
rmslevel::rmslevel_t, 1397
unit2hz
MHAOvlFilter::scale_var_t, 1029
unlock_channels
fw_vars_t, 532
unlock_srate_fragsize
fw_vars_t, 532
unset_fb_pars
DynComp::dc_afterburn_t, 454
up
MHASignal::schroeder_t, 1246
up_incl
MHAParser::range_var_t, 1112
up_limit
MHAParser::range_var_t, 1112
MHASignal::quantizer_t, 1240
up_thresh
acPooling_wave_config, 217
update
ac2lsl::ac2lsl_t, 165
ac2wave_if_t, 187
addsndfile::addsndfile_if_t, 252
adm_if_t, 273
audiometerbackend::audiometer_if_t, 316
calibrator_t, 340
coherence::cohflt_if_t, 348
cpuload::cpuload_if_t, 367
dc::dc_if_t, 381
delay::interface_t, 412
DynComp::dc_afterburn_t, 454

© 2005-2021 HörTech gGmbH, Oldenburg

DynComp::gaintable_t, 460
fftfilter::interface_t, 503
fshift_hilbert::frequency_translator_t, 517
lsl2ac::lsl2ac_t, 684
matlab_wrapper::matlab_wrapper_t, 701
mconv::MConv, 719
MHA_AC::ac2matrix_t, 724
MHA_AC::acspace2matrix_t, 728
MHA_AC::stat_t, 738
mhachain::chain_base_t, 844
MHAMultiSrc::spectrum_t, 999
MHAMultiSrc::waveform_t, 1001
MHAParser::mhaconfig_mon_t, 1089
MHAPlugin_Split::split_t, 1179
nlms_t, 1310
overlapadd::overlapadd_if_t, 1332
plingploing::if_t, 1342
rmslevel::rmslevel_if_t, 1393
route::interface_t, 1414
smoothgains_bridge::overlapadd_if_t, 1455
softclip_t, 1459
spec2wave_if_t, 1467
wave2spec_if_t, 1495
windnoise::if_t, 1516
update_burner
DynComp::dc_afterburn_t, 454
update_cfg
acConcat_wave, 201
acPooling_wave, 213
acSteer, 230
acTransform_wave, 236
adaptive_feedback_canceller, 242
complex_scale_channel_t, 363
delaysum::delaysum_wave_if_t, 414
delaysum_spec::delaysum_spec_if_t, 419
doasvm_classification, 423
doasvm_feature_extraction, 429
example6_t, 484
fader_if_t, 488
fftfbpow::fftfbpow_interface_t, 496
fftfilterbank::fftfb_interface_t, 506
fshift::fshift_t, 514
gsc_adaptive_stage::gsc_adaptive_stage_if_t, 548
gtfb_analyzer::gtfb_analyzer_t, 555
gtfb_simd_t, 563
level_matching::level_matching_t, 656
lpc, 663
lpc_bl_predictor, 666
lpc_burglattice, 672
multibandcompressor::interface_t, 1305
noise_psd_estimator::noise_psd_estimator_if_t, 1314
noise_t, 1320
plugin_interface_t, 1356
rohBeam::rohBeam, 1401
sine_t, 1437
smooth_cepstrum::smooth_cepstrum_if_t, 1440
steerbf, 1477
update_coeffs
MHAFilter::fftfilter_t, 879
MHAFilter::fftfilterbank_t, 884
update_dc
dc_simple::dc_if_t, 395
update_filter
MHAFilter::iir_filter_t, 902
update_frame
addsndfile::level_adapt_t, 256
audiometerbackend::level_adapt_t, 319
fader_wave::level_adapt_t, 493
update_gain
gain::gain_if_t, 535
update_gain_mon
dc_simple::dc_if_t, 395
update_gains
equalize::freqgains_t, 466
update_hz
MHAOvlFilter::fscale_bw_t, 1015
MHAOvlFilter::fscale_t, 1017
update_id
equalize::freqgains_t, 466
update_irs
mconv::MConv, 719
update_level
dc_simple::dc_if_t, 395
update_level_mon
dc_simple::dc_if_t, 395
update_levels
multibandcompressor::plugin_signals_t, 1307
update_m
matrixmixer::matmix_t, 716
update_minmax
gain::gain_if_t, 535
update_mismatch
level_matching::channel_pair, 649, 650
update_mode
ac2osc_t, 183
update_monitors
dc::dc_if_t, 381

© 2005-2021 HörTech gGmbH, Oldenburg

matlab_wrapper::matlab_wrapper_t, 701
update_mu
MHAFilter::adapt_filter_t, 867
update_ntaps
MHAFilter::adapt_filter_t, 868
update_parser
windowselector_t, 1521
update_proc_load
mhachain::plugs_t, 849
update_PSD_Lowpass
windnoise::cfg_t, 1511
update_ramplen
altplugs_t, 303
update_recmode
acmon::acmon_t, 209
update_selector_list
altplugs_t, 303
update_tau
levelmeter_t, 659
update_tau_level
calibrator_t, 340
update_varlist
ac2lsl::cfg_t, 168
updated
windowselector_t, 1521
updater
MHAOvlFilter::fscale_bw_t, 1016
MHAOvlFilter::fscale_t, 1018
upper_threshold
acPooling_wave, 213
UPrew
adaptive_feedback_canceller_config, 249
UPrewW
adaptive_feedback_canceller_config, 249
upsample.cpp, 1713
upsampling_factor
MHAFilter::polyphase_resampling_t, 927
upscale
MHASignal::quantizer_t, 1240
us_t, 1490
antialias, 1492
prepare, 1491
process, 1491
ratio, 1492
release, 1491
us_t, 1491
use_date
plugins::hoertech::acrec::acrec_t, 1381
wavrec_t, 1505
use_frozen_
cfg_t, 345
use_jack_transport
MHAIOJackdb::io_jack_t, 953
MHAJack::client_t, 983
use_mat
acmon::ac_monitor_t, 206
use_own_ac
altplugs_t, 303
use_sine
cpuload::cpuload_cfg_t, 365
cpuload::cpuload_if_t, 368
use_smoothspec
smoothgains_bridge::smoothspec_wrap_t, 1457
UseChannel_LF_attenuation
windnoise::cfg_t, 1512
windnoise::if_t, 1517
user
MHAParser::window_t, 1142
user_config
matlab_wrapper::callback, 694
matlab_wrapper::matlab_wrapper_rt_cfg_t, 696
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 708
user_err_msg
MHAIOalsa.cpp, 1656
MHAIOAsterisk.cpp, 1661
MHAIOFile.cpp, 1665
MHAIOJack.cpp, 1670
MHAIOJackdb.cpp, 1674
MHAIOParser.cpp, 1678
MHAIOPortAudio.cpp, 1682
MHAIOTCP.cpp, 1688
user_t
MHAWindow::user_t, 1301
username
MHA_AC::ac2matrix_helper_t, 722
userwnd
windowselector_t, 1522
useVAD
gsc_adaptive_stage::gsc_adaptive_stage, 542
gsc_adaptive_stage::gsc_adaptive_stage_if, 549

v_G
adaptive_feedback_canceller_config, 247
vadName
gsc_adaptive_stage::gsc_adaptive_stage, 542
gsc_adaptive_stage::gsc_adaptive_stage_if, 549

© 2005-2021 HörTech gGmbH, Oldenburg

val2str
MHAParser::StrCnv, 132–134
VAL_COMPLEX
ac_mul.hh, 1525
VAL_REAL
ac_mul.hh, 1525
val_type_t
ac_mul.hh, 1524
validate
AuditoryProfile::parser_t::fmap_t, 330
MHAParser::keyword_list_t, 1074
MHAParser::kw_t, 1077
MHAParser::range_var_t, 1110, 1111
validator_channels
mhasndfile.cpp, 1694
validator_length
mhasndfile.cpp, 1694
value
AuditoryProfile::parser_t::fmap_t, 330
mha_signal.hh, 1642
MHASignal::ringbuffer_t, 1242
MHASignal::spectrum_t, 1251
MHASignal::waveform_t, 1267, 1268
Vector and matrix processing toolbox, 46–49
value_type
mha_dblbuf_t< FIFO >, 749
mha_fifo_t< T >, 778
valuechanged
MHAParser::base_t, 1042
var
matlab_wrapper::callback, 695
variable, 4
variable_name
mha_stash_environment_variable_t, 797
variable_t
MHAParser::variable_t, 1119
variables, 4
acsave::acsave_t, 221
varlist
ac2lsl::cfg_t, 168
acmon::acmon_t, 209
acsave::acsave_t, 221
acsave::cfg_t, 223
lsl2ac::cfg_t, 680
varlist_t
acsave::acsave_t, 219
varname
plugins::hoertech::acrec::acrec_t, 1381
plugins::hoertech::acrec::acwriter_t, 1387
vars
ac2lsl::ac2lsl_t, 165
ac2osc_t, 183
acmon::acmon_t, 210
analysispath_if_t, 312
calibrator_t, 340
coherence::cohflt_if_t, 349
matlab_wrapper::matlab_wrapper_t, 702
MHAKernel::algo_comm_class_t, 994
osc2ac_t, 1323
vars_t
coherence::vars_t, 354
MHAOvlFilter::overlap_save_filterbank_t::vars_t, 1026
vbark
MHAOvlFilter::barkscale, 117
vbin1
MHAOvlFilter::fftfb_t, 1009
vbin2
MHAOvlFilter::fftfb_t, 1009
vcomplex_mon_t
MHAParser::vcomplex_mon_t, 1121
vcomplex_t
MHAParser::vcomplex_t, 1123
vec_y
MHATableLookup::linear_table_t, 1282
Vector and matrix processing toolbox, 33
assign, 44, 45
bin2freq, 39
channels, 38
clear, 44
colored_intensity, 54
conjugate, 57
copy_channel, 52, 53
dupvec, 41
dupvec_chk, 41
equal_dim, 42
freq2bin, 39
integrate, 43
max, 56
maxabs, 54, 55
mha_real_t, 37
min, 56
operator∗=, 50, 51
operator^=, 52
operator+=, 50, 52
operator-=, 50
operator/=, 51, 52
rad2smp, 40
range, 38
rmslevel, 53, 55
size, 43, 44

© 2005-2021 HörTech gGmbH, Oldenburg

smp2rad, 40
std_vector_float, 49
std_vector_vector_complex, 49
std_vector_vector_float, 49
sumsqr_channel, 56
sumsqr_frame, 57
timeshift, 45
value, 46–49
VERSION_EXTENSION
mhamain.cpp, 1692
vF
DynComp::gaintable_t, 462
vfloat_mon_t
MHAParser::vfloat_mon_t, 1126
vfloat_t
MHAParser::vfloat_t, 1128
vFlog
DynComp::gaintable_t, 462
vfreq
MHAOvlFilter::barkscale, 117
vGCC
acConcat_wave_config, 203
doasvm_feature_extraction_config, 432
vGCC_ac
doasvm_feature_extraction_config, 432
vGCC_con
acConcat_wave_config, 203
vGCC_name
doasvm_classification, 424
doasvm_feature_extraction, 429
vint_0123n1
transducers.cpp, 1713
vint_mon_t
MHAParser::vint_mon_t, 1131
vint_t
MHAParser::vint_t, 1133
vL
DynComp::gaintable_t, 462
vLTASS_combined_lev
speechnoise.cpp, 1708
vLTASS_female_lev
speechnoise.cpp, 1708
vLTASS_freq
speechnoise.cpp, 1707
vLTASS_male_lev
speechnoise.cpp, 1708
vmax
gain::gain_if_t, 536
vMHAOrigFreq
speechnoise.cpp, 1707
vMHAOrigSpec
speechnoise.cpp, 1707
vmin
gain::gain_if_t, 535
vOlnoiseFreq
speechnoise.cpp, 1708
vOlnoiseLev
speechnoise.cpp, 1708
vstring_mon_t
MHAParser::vstring_mon_t, 1135
vstring_t
MHAParser::vstring_t, 1137
vy
MHATableLookup::linear_table_t, 1282

W
gsc_adaptive_stage::gsc_adaptive_stage, 542
MHA_TCP::OS_EVENT_TYPE, 814
MHAFilter::adapt_filter_state_t, 865
w
ac2wave_t, 190
doasvm_classification, 424
MHAOvlFilter::fftfb_t, 1009
w_out
combc_t, 359
wait
MHA_TCP::Event_Watcher, 813
wait_for_decrease
mha_fifo_posix_threads_t, 775
mha_fifo_thread_platform_t, 786
wait_for_increase
mha_fifo_posix_threads_t, 775
mha_fifo_thread_platform_t, 787
Wakeup_Event
MHA_TCP::Wakeup_Event, 837
wave
alsa_t< T >, 294
wave2spec
MHASignal::fft_t, 1212
overlapadd::overlapadd_t, 1336
plugindescription_t, 1359
wave2spec.cpp, 1713
wave2spec.hh, 1713
MHAPLUGIN_OVERLOAD_OUTDOMAIN, 1713
wave2spec_apply_window
overlapadd::overlapadd_t, 1336
wave2spec_compute_fft
overlapadd::overlapadd_t, 1336
wave2spec_hop_forward
overlapadd::overlapadd_t, 1336
wave2spec_if_t, 1492

© 2005-2021 HörTech gGmbH, Oldenburg

algo, 1497
nfft, 1496
nwnd, 1496
prepare, 1494
process, 1494, 1495
release, 1494
return_wave, 1496
setlock, 1495
strict_window_ratio, 1496
update, 1495
wave2spec_if_t, 1494
window_config, 1496
wndpos, 1496
zeropadding, 1497
wave2spec_scale
MHASignal::fft_t, 1213
wave2spec_t, 1497
~wave2spec_t, 1499
ac_wndshape_name, 1502
calc_in, 1502
calc_pre_wnd, 1501
ft, 1501
get_zeropadding, 1500
in_buf, 1502
npad1, 1502
npad2, 1502
nwnd, 1501
nwndshift, 1501
process, 1500
publish_ac_variables, 1500
spec_in, 1502
wave2spec_t, 1499
window, 1502
wave2wave
plugindescription_t, 1359
wave_fifo
analysepath_t, 308
wave_in
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 710
MHAPlugin_Split::domain_handler_t, 1168
wave_in1
overlapadd::overlapadd_t, 1337
wave_out
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 710
MHAPlugin_Split::domain_handler_t, 1168
MHAPlugin_Split::split_t, 1182
wave_out1
overlapadd::overlapadd_t, 1337
wave_reader
addsndfile, 80
waveform_proxy_t
addsndfile::waveform_proxy_t, 261
waveform_t
MHA_AC::waveform_t, 740
MHAMultiSrc::waveform_t, 1000
MHASignal::waveform_t, 1265, 1266
wavrec.cpp, 1714
wavrec_t, 1503
fifolen, 1505
minwrite, 1505
output_sample_format, 1505
patchbay, 1505
prefix, 1505
prepare, 1504
process, 1504
record, 1505
release, 1504
start_new_session, 1504
use_date, 1505
wavrec_t, 1504
wavwriter_t, 1506
~wavwriter_t, 1507
act_, 1508
cf_, 1508
close_session, 1508
create_soundfile, 1507
data, 1509
exit_request, 1507
fifo, 1508
format_name, 1509
minw_, 1508
process, 1507
set_format, 1507
sf, 1508
wavwriter_t, 1506
writethread, 1507
writethread, 1508
weights
delaysum::delaysum_wave_if_t, 415
delaysum::delaysum_wave_t, 417
what
MHA_Error, 766
whitenplugin_t,
speechnoise_t, 1473
win_f0
smooth_cepstrum::smooth_cepstrum_if_t, 1442
WINAPI

© 2005-2021 HörTech gGmbH, Oldenburg

mha_plugin.hh, 1623
windnoise, 161
windnoise.cpp, 1714
register_configuration_variable, 1714
windnoise.hh, 1714
windnoise::cfg_t, 1509
alpha_Lowpass, 1512
cfg_t, 1510
compensation, 1512
FrequencyBinLowPass, 1512
LowPassFraction, 1513
LowPassWindGain, 1513
powspec, 1513
process, 1511
PSD_Lowpass, 1513
remapping, 1512
threshold_compare, 1511
update_PSD_Lowpass, 1511
UseChannel_LF_attenuation, 1512
windnoise::if_t, 1514
detected, 1517
detected_acname, 1517
if_t, 1515
insert, 1516
lowpass_quotient, 1517
lowpass_quotient_acname, 1518
LowPassCutOffFrequency, 1517
LowPassFraction, 1517
LowPassWindGain, 1517
patchbay, 1516
prepare, 1515
process, 1516
release, 1516
tau_Lowpass, 1517
update, 1516
UseChannel_LF_attenuation, 1517
WindNoiseDetector, 1517
WindNoiseDetector
windnoise::if_t, 1517
window
MHAFilter::smoothspec_t, 933
overlapadd::overlapadd_if_t, 1333
wave2spec_t, 1502
window_config
spec2wave_if_t, 1468
wave2spec_if_t, 1496
window_t
MHAParser::window_t, 1140
windowselector.cpp, 1714
windowselector.h, 1714
windowselector_t, 1518
~windowselector_t, 1520
get_window_data, 1520
insert_items, 1520
invalidate_window_data, 1521
patchbay, 1522
setlock, 1520
update_parser, 1521
updated, 1521
userwnd, 1522
windowselector_t, 1519
wnd, 1521
wndexp, 1522
wndtype, 1521
winF0
smooth_cepstrum::smooth_cepstrum_t, 1446
smooth_cepstrum::smooth_params, 1452
wInput
MHAFilter::fftfilter_t, 881
wInput_fft
MHAFilter::fftfilter_t, 881
wIRS_fft
MHAFilter::fftfilter_t, 881
wnd
addsndfile::level_adapt_t, 256
audiometerbackend::level_adapt_t, 320
fader_wave::level_adapt_t, 493
windowselector_t, 1521
wnd_bartlett
MHAParser::window_t, 1140
wnd_blackman
MHAParser::window_t, 1140
wnd_funs
mha_windowparser.cpp, 1650
wnd_hamming
MHAParser::window_t, 1140
wnd_hann
MHAParser::window_t, 1140
wnd_rect
MHAParser::window_t, 1140
wnd_user
MHAParser::window_t, 1140
wndexp
overlapadd::overlapadd_if_t, 1333
windowselector_t, 1522
wndlen
doasvm_feature_extraction_config, 431
mhaconfig_t, 853
MHAParser::mhaconfig_mon_t, 1089
testplugin::config_parser_t, 1485
wndpos

© 2005-2021 HörTech gGmbH, Oldenburg

overlapadd::overlapadd_if_t, 1333
wave2spec_if_t, 1496
wndtype
windowselector_t, 1521
worker_thread_priority
dbasync_native::db_if_t, 375
MHAPlugin_Split::split_t, 1181
worker_thread_scheduler
dbasync_native::db_if_t, 375
MHAPlugin_Split::split_t, 1181
wout
matrixmixer::cfg_t, 714
route::process_t, 1416
wout_ac
route::process_t, 1417
wOutput
MHAFilter::fftfilter_t, 881
wOutput_fft
MHAFilter::fftfilter_t, 881
wrapped_plugin_t
matlab_wrapper::matlab_wrapper_t::wrapped_plugin_t, 705
write
alsa_base_t, 288
alsa_t< T >, 293
mha_drifter_fifo_t< T >, 759
mha_fifo_lf_t< T >, 768
mha_fifo_lw_t< T >, 772
mha_fifo_t< T >, 779
MHA_TCP::Connection, 809
MHAFilter::blockprocessing_polyphase_resampling_t, 870
MHAFilter::polyphase_resampling_t, 926
MHAJack::port_t, 986
MHASignal::matrix_t, 1235
MHASignal::ringbuffer_t, 1243
MHASignal::uint_vector_t, 1262
write_buf
overlapadd::overlapadd_t, 1337
spec2wave_t, 1470
write_event
MHA_TCP::Connection, 810
write_float
mha_parser.cpp, 1615
write_ptr
mha_fifo_t< T >, 782
write_thread
plugins::hoertech::acrec::acwriter_t, 1385
wavwriter_t, 1507
write_to_table
cpuload::cpuload_cfg_t, 365
write_wave
mhasndfile.cpp, 1693
mhasndfile.h, 1694
writeaccess
MHAParser::base_t, 1042
writer_started
mha_drifter_fifo_t< T >, 762
writer_xruns_in_succession
mha_drifter_fifo_t< T >, 763
writer_xruns_since_start
mha_drifter_fifo_t< T >, 762
writer_xruns_total
mha_drifter_fifo_t< T >, 762
writethread
plugins::hoertech::acrec::acwriter_t, 1386
wavwriter_t, 1508
Writing openMHA Plugins. A step-by-step tutorial, 6
wtype
MHAParser::window_t, 1142
wtype_t
MHAParser::window_t, 1140

X
gsc_adaptive_stage::gsc_adaptive_stage, 542
MHA_TCP::OS_EVENT_TYPE, 814
MHAFilter::adapt_filter_state_t, 865
x
doasvm_classification, 424
gsc_adaptive_stage::gsc_adaptive_stage, 542
xfun
MHATableLookup::xy_table_t, 1289
xi_est
smooth_cepstrum::smooth_cepstrum_t, 1448
xi_min
smooth_cepstrum::smooth_cepstrum_t, 1446
xi_min_db
smooth_cepstrum::smooth_cepstrum_if_t, 1441
smooth_cepstrum::smooth_params, 1451
xi_ml
smooth_cepstrum::smooth_cepstrum_t, 1447
xi_opt_db
smooth_cepstrum::smooth_cepstrum_if_t, 1442
smooth_cepstrum::smooth_params, 1452
xiOpt

© 2005-2021 HörTech gGmbH, Oldenburg

noise_psd_estimator::noise_psd_estimator_t, 1318
smooth_cepstrum::smooth_cepstrum_t, 1449
xiOptDb
noise_psd_estimator::noise_psd_estimator_if_t, 1314
xmax
MHATableLookup::linear_table_t, 1283
xmin
MHATableLookup::linear_table_t, 1282
Xs
MHAFilter::fftfilterbank_t, 886
xw
MHAFilter::fftfilterbank_t, 886
xy_table_t
MHATableLookup::xy_table_t, 1286
xyfun
MHATableLookup::xy_table_t, 1289

Y
gsc_adaptive_stage::gsc_adaptive_stage, 542
y
doasvm_classification, 424
gsc_adaptive_stage::gsc_adaptive_stage, 543
y0
dc_simple::dc_t::line_t, 403
y_previous
rt_nlms_t, 1420
yfun
MHATableLookup::xy_table_t, 1289
Yn
MHAFilter::complex_bandpass_t, 876
MHAFilter::iir_ord1_real_t, 905
YPrew
adaptive_feedback_canceller_config, 249
Ys
MHAFilter::fftfilterbank_t, 887
yw
MHAFilter::fftfilterbank_t, 887
yw_temp
MHAFilter::fftfilterbank_t, 887

zeropadding
wave2spec_if_t, 1497
zeros
ac2wave_if_t, 188
zerowindow
overlapadd::overlapadd_if_t, 1333

© 2005-2021 HörTech gGmbH, Oldenburg