

M

Generated by Doxygen 1.12.0



<b>1 Data Structure Index</b>	<b>1</b>
1.1 Data Structures	1
<b>2 File Index</b>	<b>3</b>
2.1 File List	3
<b>3 Data Structure Documentation</b>	<b>7</b>
3.1 m_delay_buffer Struct Reference	7
3.1.1 Field Documentation	7
3.1.1.1 buffer_array_size	7
3.1.1.2 buffers	7
3.1.1.3 index	7
3.1.1.4 n_buffers	7
3.1.1.5 pos	8
3.1.1.6 valid	8
3.2 m_eng_3_band_eq_str Struct Reference	8
3.2.1 Field Documentation	8
3.2.1.1 coefs	8
3.2.1.2 control_mode	8
3.2.1.3 filters	8
3.2.1.4 high	8
3.2.1.5 low	9
3.2.1.6 mid	9
3.3 m_eng_3_band_splitter_str Struct Reference	9
3.3.1 Field Documentation	9
3.3.1.1 filters	9
3.4 m_eng_adaptive_waveshaper_str Struct Reference	9
3.4.1 Field Documentation	10
3.4.1.1 coefficient	10
3.4.1.2 local_amplitude	10
3.4.1.3 shape	10
3.5 m_eng_amplifier_str Struct Reference	10
3.5.1 Field Documentation	10
3.5.1.1 db	10
3.5.1.2 g	10
3.5.1.3 gain	10
3.5.1.4 mode	11
3.6 m_eng_band_pass_filter_str Struct Reference	11
3.6.1 Field Documentation	11
3.6.1.1 a0	11
3.6.1.2 a1	11
3.6.1.3 a2	11
3.6.1.4 a3	11

3.6.1.5 a4	12
3.6.1.6 bandwidth	12
3.6.1.7 center	12
3.6.1.8 x1	12
3.6.1.9 x2	12
3.6.1.10 y1	12
3.6.1.11 y2	12
3.7 m_eng_biquad_str Struct Reference	12
3.7.1 Field Documentation	13
3.7.1.1 a0	13
3.7.1.2 a1	13
3.7.1.3 a2	13
3.7.1.4 a3	13
3.7.1.5 a4	13
3.7.1.6 bandwidth	13
3.7.1.7 cutoff	13
3.7.1.8 db_gain	14
3.7.1.9 type	14
3.7.1.10 x1	14
3.7.1.11 x2	14
3.7.1.12 y1	14
3.7.1.13 y2	14
3.8 m_eng_compressor_str Struct Reference	14
3.8.1 Field Documentation	15
3.8.1.1 alpha	15
3.8.1.2 attack	15
3.8.1.3 e_final	15
3.8.1.4 ratio	15
3.8.1.5 release	15
3.8.1.6 rho	15
3.8.1.7 threshold	15
3.9 m_eng_context Struct Reference	15
3.9.1 Field Documentation	16
3.9.1.1 active_profile	16
3.9.1.2 dc_blocker_avg	16
3.9.1.3 declick_buffer	16
3.9.1.4 input_lpf	16
3.9.1.5 n_profiles	16
3.9.1.6 new_profile	16
3.9.1.7 output_amp	17
3.9.1.8 output_hpf	17
3.9.1.9 prev_block	17

3.9.1.10 profile_array_size . . . . .	17
3.9.1.11 profile_maintainance_index . . . . .	17
3.9.1.12 profile_switch_progress . . . . .	17
3.9.1.13 profile_switch_samples . . . . .	17
3.9.1.14 profile_switch_triggered . . . . .	17
3.9.1.15 profile_switch_type . . . . .	17
3.9.1.16 profiles . . . . .	17
3.9.1.17 profiles_switching . . . . .	18
3.9.1.18 runs . . . . .	18
3.9.1.19 status_flags . . . . .	18
3.10 m_eng_delay_str Struct Reference . . . . .	18
3.10.1 Field Documentation . . . . .	18
3.10.1.1 buf . . . . .	18
3.10.1.2 delay_gain . . . . .	18
3.10.1.3 delay_samples . . . . .	18
3.10.1.4 g . . . . .	19
3.10.1.5 note . . . . .	19
3.10.1.6 tempo . . . . .	19
3.11 m_eng_dirty_octave_str Struct Reference . . . . .	19
3.11.1 Field Documentation . . . . .	19
3.11.1.1 dc_average . . . . .	19
3.11.1.2 fuzz . . . . .	19
3.11.1.3 last_out_sample . . . . .	19
3.11.1.4 lpf_alpha . . . . .	20
3.12 m_eng_distortion_str Struct Reference . . . . .	20
3.12.1 Field Documentation . . . . .	20
3.12.1.1 bass_cutoff . . . . .	20
3.12.1.2 bass_mix . . . . .	20
3.12.1.3 dist . . . . .	20
3.12.1.4 function . . . . .	20
3.12.1.5 low_pass . . . . .	20
3.12.1.6 wet_mix . . . . .	21
3.13 m_eng_envelope_str Struct Reference . . . . .	21
3.13.1 Field Documentation . . . . .	21
3.13.1.1 alpha . . . . .	21
3.13.1.2 chunk_size . . . . .	21
3.13.1.3 e . . . . .	21
3.13.1.4 filter . . . . .	21
3.13.1.5 max_center . . . . .	22
3.13.1.6 min_center . . . . .	22
3.13.1.7 sensitivity . . . . .	22
3.13.1.8 smoothness . . . . .	22

3.13.1.9 speed	22
3.13.1.10 width	22
3.14 m_eng_flanger_str Struct Reference	22
3.14.1 Field Documentation	23
3.14.1.1 buf	23
3.14.1.2 d	23
3.14.1.3 depth	23
3.14.1.4 dry_mix	23
3.14.1.5 mix	23
3.14.1.6 note	23
3.14.1.7 period	23
3.14.1.8 r	23
3.14.1.9 range	23
3.14.1.10 s	23
3.14.1.11 t	24
3.14.1.12 tempo	24
3.14.1.13 wet_mix	24
3.15 m_eng_high_pass_filter_str Struct Reference	24
3.15.1 Field Documentation	24
3.15.1.1 a0	24
3.15.1.2 a1	24
3.15.1.3 a2	25
3.15.1.4 a3	25
3.15.1.5 a4	25
3.15.1.6 cutoff_frequency	25
3.15.1.7 x1	25
3.15.1.8 x2	25
3.15.1.9 y1	25
3.15.1.10 y2	25
3.16 m_eng_log_entry Struct Reference	25
3.16.1 Field Documentation	26
3.16.1.1 cycle	26
3.16.1.2 data	26
3.16.1.3 data_type	26
3.16.1.4 file_name	26
3.16.1.5 function	26
3.16.1.6 line	26
3.16.1.7 message	26
3.16.1.8 trace_depth	27
3.16.1.9 type	27
3.17 m_eng_low_end_compressor_str Struct Reference	27
3.17.1 Field Documentation	27

3.17.1.1 bass_comp	27
3.17.1.2 low_pass	27
3.17.1.3 mid_pass	27
3.17.1.4 mids_comp	27
3.18 m_eng_low_pass_filter_str Struct Reference	28
3.18.1 Field Documentation	28
3.18.1.1 a0	28
3.18.1.2 a1	28
3.18.1.3 a2	28
3.18.1.4 a3	28
3.18.1.5 a4	28
3.18.1.6 cutoff_frequency	28
3.18.1.7 x1	29
3.18.1.8 x2	29
3.18.1.9 y1	29
3.18.1.10 y2	29
3.19 m_eng_mixer_str Struct Reference	29
3.19.1 Field Documentation	29
3.19.1.1 ratio	29
3.20 m_eng_n_band_splitter_str Struct Reference	29
3.20.1 Field Documentation	30
3.20.1.1 filters	30
3.21 m_eng_noise_suppressor_str Struct Reference	30
3.21.1 Field Documentation	30
3.21.1.1 e_final	30
3.21.1.2 max_reduction	30
3.21.1.3 r	30
3.21.1.4 ratio	30
3.21.1.5 threshold	31
3.22 m_eng_percussifier_str Struct Reference	31
3.22.1 Field Documentation	31
3.22.1.1 alpha_long	31
3.22.1.2 alpha_short	31
3.22.1.3 arm_threshold	32
3.22.1.4 decay_rate	32
3.22.1.5 fade_alpha	32
3.22.1.6 fade_in	32
3.22.1.7 fade_in_samples	32
3.22.1.8 fade_out	32
3.22.1.9 gain	32
3.22.1.10 hold_samples	32
3.22.1.11 note	32

3.22.1.12 r	32
3.22.1.13 refractory_period	33
3.22.1.14 refractory_samples	33
3.22.1.15 rms_long	33
3.22.1.16 rms_short	33
3.22.1.17 state	33
3.22.1.18 tempo	33
3.22.1.19 timer	33
3.22.1.20 trigger_threshold	33
3.23 m_eng_profile Struct Reference	33
3.23.1 Field Documentation	34
3.23.1.1 active	34
3.23.1.2 back_pipeline	34
3.23.1.3 back_pipeline_warmed_up	34
3.23.1.4 blocked_jobs	34
3.23.1.5 front_pipeline	34
3.23.1.6 jobs	34
3.23.1.7 output_amp	35
3.23.1.8 pipeline_swap_progress	35
3.23.1.9 pipeline_swap_samples	35
3.23.1.10 pipeline_swap_type	35
3.23.1.11 pipelines_swapping	35
3.23.1.12 prev_block	35
3.23.1.13 runs	35
3.23.1.14 transition_policy	35
3.24 m_eng_profiler_entry Struct Reference	35
3.24.1 Field Documentation	36
3.24.1.1 calls	36
3.24.1.2 function_name	36
3.24.1.3 open_cycle	36
3.24.1.4 peak_cycles	36
3.24.1.5 ra_cycles	36
3.24.1.6 total_cycles	36
3.25 m_eng_simple_distortion_str Struct Reference	36
3.25.1 Field Documentation	37
3.25.1.1 postgain	37
3.25.1.2 pregain	37
3.26 m_eng_warbler_str Struct Reference	37
3.26.1 Field Documentation	37
3.26.1.1 alpha	37
3.26.1.2 center	37
3.26.1.3 e	38



3.26.1.4 filter	38
3.26.1.5 max_rate	38
3.26.1.6 min_rate	38
3.26.1.7 rate	38
3.26.1.8 reactivity	38
3.26.1.9 sensitivity	38
3.26.1.10 t	38
3.26.1.11 width	38
3.27 m_eng_waveshaper_str Struct Reference	39
3.27.1 Field Documentation	39
3.27.1.1 coefficient	39
3.27.1.2 shape	39
3.28 m_lr_high_pass_filter_str Struct Reference	39
3.28.1 Field Documentation	40
3.28.1.1 a0	40
3.28.1.2 a1	40
3.28.1.3 a2	40
3.28.1.4 a3	40
3.28.1.5 a4	40
3.28.1.6 cutoff_frequency	40
3.28.1.7 x_11	40
3.28.1.8 x_12	40
3.28.1.9 x_21	40
3.28.1.10 x_22	40
3.28.1.11 y_11	41
3.28.1.12 y_12	41
3.28.1.13 y_21	41
3.28.1.14 y_22	41
3.29 m_lr_low_pass_filter_str Struct Reference	41
3.29.1 Field Documentation	41
3.29.1.1 a0	41
3.29.1.2 a1	42
3.29.1.3 a2	42
3.29.1.4 a3	42
3.29.1.5 a4	42
3.29.1.6 cutoff_frequency	42
3.29.1.7 x_11	42
3.29.1.8 x_12	42
3.29.1.9 x_21	42
3.29.1.10 x_22	42
3.29.1.11 y_11	42
3.29.1.12 y_12	43

3.29.1.13 y_21 . . . . .	43
3.29.1.14 y_22 . . . . .	43
3.30 m_message Struct Reference . . . . .	43
3.30.1 Field Documentation . . . . .	43
3.30.1.1 callback . . . . .	43
3.30.1.2 cb_arg . . . . .	43
3.30.1.3 data . . . . .	43
3.30.1.4 retries . . . . .	44
3.30.1.5 type . . . . .	44
3.31 m_parameter Struct Reference . . . . .	44
3.31.1 Field Documentation . . . . .	44
3.31.1.1 max . . . . .	44
3.31.1.2 min . . . . .	44
3.31.1.3 new_value . . . . .	44
3.31.1.4 old_value . . . . .	44
3.31.1.5 scale . . . . .	45
3.31.1.6 updated . . . . .	45
3.31.1.7 value . . . . .	45
3.32 m_parameter_id Struct Reference . . . . .	45
3.32.1 Field Documentation . . . . .	45
3.32.1.1 parameter_id . . . . .	45
3.32.1.2 profile_id . . . . .	45
3.32.1.3 transformer_id . . . . .	45
3.33 m_pipeline Struct Reference . . . . .	46
3.34 m_pipeline_mod Struct Reference . . . . .	46
3.34.1 Field Documentation . . . . .	46
3.34.1.1 data . . . . .	46
3.34.1.2 sdata . . . . .	46
3.34.1.3 tid . . . . .	46
3.34.1.4 type . . . . .	46
3.35 m_profile Struct Reference . . . . .	47
3.35.1 Field Documentation . . . . .	47
3.35.1.1 active . . . . .	47
3.36 m_response Struct Reference . . . . .	47
3.36.1 Field Documentation . . . . .	47
3.36.1.1 data . . . . .	47
3.36.1.2 extra . . . . .	47
3.36.1.3 type . . . . .	47
3.37 m_setting Struct Reference . . . . .	48
3.37.1 Field Documentation . . . . .	48
3.37.1.1 new_value . . . . .	48
3.37.1.2 old_value . . . . .	48

---

3.37.1.3 updated . . . . .	48
3.37.1.4 value . . . . .	48
3.38 m_setting_id Struct Reference . . . . .	48
3.38.1 Field Documentation . . . . .	49
3.38.1.1 profile_id . . . . .	49
3.38.1.2 setting_id . . . . .	49
3.38.1.3 transformer_id . . . . .	49
3.39 m_transformer Struct Reference . . . . .	49
3.39.1 Field Documentation . . . . .	49
3.39.1.1 band_center . . . . .	49
3.39.1.2 band_hp_cutoff . . . . .	49
3.39.1.3 band_lp_cutoff . . . . .	50
3.39.1.4 band_mode . . . . .	50
3.39.1.5 band_width . . . . .	50
3.39.1.6 id . . . . .	50
3.39.1.7 type . . . . .	50
3.39.1.8 wet_mix . . . . .	50
3.40 m_transformer_str Struct Reference . . . . .	50
3.40.1 Field Documentation . . . . .	50
3.40.1.1 param . . . . .	50
<b>4 File Documentation . . . . .</b>	<b>51</b>
4.1 m_eng.h File Reference . . . . .	51
4.1.1 Macro Definition Documentation . . . . .	53
4.1.1.1 ALLOW_PRINTLINES . . . . .	53
4.1.1.2 AUDIO_BLOCK_MS . . . . .	53
4.1.1.3 AUDIO_BLOCK_SAMPLES . . . . .	53
4.1.1.4 AUDIO_SAMPLE_RATE . . . . .	53
4.1.1.5 AUDIO_SAMPLE_RATE_EXACT . . . . .	53
4.1.1.6 binary_max . . . . .	53
4.1.1.7 binary_min . . . . .	53
4.1.1.8 LL_FREE . . . . .	53
4.1.1.9 LL_MALLOC . . . . .	54
4.1.1.10 LN_2 . . . . .	54
4.1.1.11 M_ENGINE . . . . .	54
4.1.1.12 M_VOICE_COMMS . . . . .	54
4.1.1.13 M_VOICE_CXT . . . . .	54
4.1.1.14 M_VOICE_ERR . . . . .	54
4.1.1.15 M_VOICE_LOG . . . . .	54
4.1.1.16 M_VOICE_PL . . . . .	54
4.1.1.17 M_VOICE_PR . . . . .	54
4.1.1.18 M_VOICE_PRF . . . . .	54

4.1.1.19 M_VOICE_TR . . . . .	55
4.1.1.20 MS_TO_SAMPLES . . . . .	55
4.1.1.21 NUM_MASKS . . . . .	55
4.1.1.22 S_TO_SAMPLES . . . . .	55
4.1.1.23 SAMPLE_FREQUENCY . . . . .	55
4.1.1.24 SAMPLES_TO_MS . . . . .	55
4.1.1.25 SAMPLES_TO_S . . . . .	55
4.1.1.26 sqr . . . . .	56
4.1.2 Function Documentation . . . . .	56
4.1.2.1 trig_transition_function() . . . . .	56
4.2 m_eng.h . . . . .	56
4.3 m_eng_adaptive_waveshaper.h File Reference . . . . .	58
4.3.1 Macro Definition Documentation . . . . .	58
4.3.1.1 ADAPTIVE_WAVESHAPER_ENVELOPE_ATTACK . . . . .	58
4.3.1.2 ADAPTIVE_WAVESHAPER_ENVELOPE_RELEASE . . . . .	58
4.3.2 Function Documentation . . . . .	59
4.3.2.1 calc_adaptive_waveshaper() . . . . .	59
4.3.2.2 init_adaptive_waveshaper_str() . . . . .	59
4.4 m_eng_adaptive_waveshaper.h . . . . .	59
4.5 m_eng_audio_block.h File Reference . . . . .	59
4.5.1 Macro Definition Documentation . . . . .	59
4.5.1.1 AUDIO_BLOCK_SAMPLES . . . . .	59
4.6 m_eng_audio_block.h . . . . .	60
4.7 m_eng_band_splitter.h File Reference . . . . .	60
4.7.1 Function Documentation . . . . .	60
4.7.1.1 calc_3_band_splitter() . . . . .	60
4.7.1.2 calc_n_band_splitter() . . . . .	60
4.7.1.3 init_3_band_splitter_str() . . . . .	61
4.7.1.4 init_n_band_splitter_str() . . . . .	61
4.7.1.5 reconfigure_3_band_splitter() . . . . .	61
4.7.1.6 reconfigure_n_band_splitter() . . . . .	61
4.8 m_eng_band_splitter.h . . . . .	61
4.9 m_eng_biquad.h File Reference . . . . .	61
4.9.1 Function Documentation . . . . .	62
4.9.1.1 calc_biquad() . . . . .	62
4.9.1.2 init_biquad_str() . . . . .	62
4.9.1.3 reconfigure_biquad() . . . . .	62
4.10 m_eng_biquad.h . . . . .	62
4.11 m_eng_buffer_mixer_amp.h File Reference . . . . .	62
4.11.1 Macro Definition Documentation . . . . .	63
4.11.1.1 M_ENG_AMPLIFIER_DB . . . . .	63
4.11.1.2 M_ENG_AMPLIFIER_LINEAR . . . . .	63

4.11.2 Function Documentation . . . . .	63
4.11.2.1 calc_amplifier() . . . . .	63
4.11.2.2 calc_buffer() . . . . .	63
4.11.2.3 init_amplifier_str() . . . . .	63
4.11.2.4 reconfigure_amplifier() . . . . .	64
4.12 m_eng_buffer_mixer_amp.h . . . . .	64
4.13 m_eng_comms.h File Reference . . . . .	64
4.13.1 Macro Definition Documentation . . . . .	64
4.13.1.1 TEENSY_I2C_SLAVE_ADDR . . . . .	64
4.13.2 Function Documentation . . . . .	65
4.13.2.1 esp32_message_check_handle() . . . . .	65
4.13.2.2 i2c_receive_isr() . . . . .	65
4.13.2.3 i2c_request_isr() . . . . .	65
4.13.2.4 init_esp32_link() . . . . .	65
4.14 m_eng_comms.h . . . . .	65
4.15 m_eng_compressor.h File Reference . . . . .	65
4.15.1 Function Documentation . . . . .	66
4.15.1.1 calc_compressor() . . . . .	66
4.15.1.2 init_compressor_str() . . . . .	66
4.15.1.3 reconfigure_compressor() . . . . .	66
4.16 m_eng_compressor.h . . . . .	66
4.17 m_eng_context.h File Reference . . . . .	66
4.17.1 Macro Definition Documentation . . . . .	68
4.17.1.1 CLICK_SLOPE_THRESHOLD . . . . .	68
4.17.1.2 DC_BLOCKER_ALPHA . . . . .	68
4.17.1.3 DECLICK_BUFSIZE . . . . .	68
4.17.1.4 M_PROFILE_SWITCH_SAMPLES . . . . .	68
4.17.1.5 PROFILE_ARRAY_INITIAL_SIZE . . . . .	68
4.17.1.6 PROFILES_MALLOC_CHUNK_SIZE . . . . .	68
4.17.1.7 SILENCE_BLOCKS_THRESHOLD . . . . .	68
4.17.1.8 SILENCE_ENERGY_THRESHOLD . . . . .	68
4.17.2 Function Documentation . . . . .	68
4.17.2.1 cxt_append_transformer_to_profile() . . . . .	68
4.17.2.2 cxt_get_n_profile_transformers() . . . . .	69
4.17.2.3 cxt_get_n_transformer_params() . . . . .	69
4.17.2.4 cxt_get_n_transformer_settings() . . . . .	69
4.17.2.5 cxt_get_parameter_by_id() . . . . .	69
4.17.2.6 cxt_get_setting_by_id() . . . . .	69
4.17.2.7 cxt_get_tid_by_pos() . . . . .	69
4.17.2.8 cxt_get_transformer_by_id() . . . . .	69
4.17.2.9 cxt_get_transformer_type() . . . . .	70
4.17.2.10 cxt_insert_transformer_to_profile() . . . . .	70

4.17.2.11 cxt_move_transformer()	70
4.17.2.12 cxt_parameter_id_valid()	70
4.17.2.13 cxt_prepend_transformer_to_profile()	70
4.17.2.14 cxt_process()	70
4.17.2.15 cxt_profile_id_valid()	70
4.17.2.16 cxt_remove_transformer_from_profile()	71
4.17.2.17 cxt_run_scheduled_maintenance()	71
4.17.2.18 cxt_set_active_profile()	71
4.17.2.19 cxt_switch_to_profile()	71
4.17.2.20 cxt_transformer_id_valid()	71
4.17.2.21 cxt_update_parameter_value_by_id()	71
4.17.2.22 cxt_update_setting_value_by_id()	71
4.17.2.23 init_m_eng_context()	72
4.17.2.24 m_eng_context_new_profile()	72
4.17.2.25 m_eng_safe_reboot()	72
4.17.2.26 reset_context()	72
4.17.3 Variable Documentation	72
4.17.3.1 global_cxt	72
4.18 m_eng_context.h	72
4.19 m_eng_debugging.h File Reference	73
4.19.1 Function Documentation	74
4.19.1.1 full_debug_print()	74
4.19.1.2 print_context_info()	74
4.19.1.3 print_pipeline_info()	74
4.19.1.4 print_profile_info()	74
4.19.1.5 print_transformer_info()	74
4.20 m_eng_debugging.h	74
4.21 m_eng_delay.h File Reference	75
4.21.1 Function Documentation	75
4.21.1.1 calc_delay()	75
4.21.1.2 init_delay_str()	75
4.21.1.3 reconfigure_delay()	75
4.22 m_eng_delay.h	75
4.23 m_eng_delay_buffer.h File Reference	76
4.23.1 Function Documentation	76
4.23.1.1 init_delay_buffer()	76
4.23.1.2 m_delay_buffer_advance()	76
4.23.1.3 m_delay_buffer_get_delayed_sample()	76
4.23.1.4 m_delay_buffer_get_delayed_sample_ptr()	76
4.23.1.5 m_delay_buffer_get_fractional_delayed_sample()	77
4.23.1.6 m_delay_buffer_resize_milliseconds()	77
4.23.1.7 m_delay_buffer_resize_samples()	77

4.23.1.8 m_delay_buffer_resize_seconds()	77
4.23.1.9 m_delay_buffer_tick()	77
4.24 m_eng_delay_buffer.h	77
4.25 m_eng_dirty_octave.h File Reference	78
4.25.1 Function Documentation	78
4.25.1.1 calc_dirty_octave()	78
4.25.1.2 init_dirty_octave_str()	78
4.25.1.3 reconfigure_dirty_octave()	78
4.26 m_eng_dirty_octave.h	78
4.27 m_eng_distortion.h File Reference	79
4.27.1 Macro Definition Documentation	79
4.27.1.1 M_DISTORTION_ARCTAN	79
4.27.1.2 M_DISTORTION_CLIP	79
4.27.1.3 M_DISTORTION_FOLD	79
4.27.1.4 M_DISTORTION_TANH	79
4.27.1.5 USE_GLOBAL_TEMP_BUFFERS	79
4.27.2 Function Documentation	80
4.27.2.1 calc_distortion()	80
4.27.2.2 init_distortion_str()	80
4.27.2.3 reconfigure_distortion()	80
4.28 m_eng_distortion.h	80
4.29 m_eng_envelope.h File Reference	80
4.29.1 Function Documentation	81
4.29.1.1 calc_envelope()	81
4.29.1.2 init_envelope_str()	81
4.29.1.3 reconfigure_envelope()	81
4.30 m_eng_envelope.h	81
4.31 m_eng_equaliser.h File Reference	82
4.31.1 Macro Definition Documentation	82
4.31.1.1 M_ENG_EQ_CONTROL_DB_GAIN	82
4.31.1.2 M_ENG_EQ_CONTROL_DIRECT	82
4.31.2 Function Documentation	82
4.31.2.1 calc_3_band_eq()	82
4.31.2.2 init_3_band_eq_str()	82
4.31.2.3 reconfigure_3_band_eq()	82
4.32 m_eng_equaliser.h	83
4.33 m_eng_flanger.h File Reference	83
4.33.1 Function Documentation	83
4.33.1.1 calc_flanger()	83
4.33.1.2 init_flanger_str()	83
4.33.1.3 reconfigure_flanger()	84
4.34 m_eng_flanger.h	84

4.35 m_eng_flops.h File Reference . . . . .	84
4.35.1 Macro Definition Documentation . . . . .	84
4.35.1.1 RESTRICT . . . . .	84
4.36 m_eng_flops.h . . . . .	85
4.37 m_eng_globals.h File Reference . . . . .	87
4.37.1 Function Documentation . . . . .	88
4.37.1.1 current_cycle() . . . . .	88
4.37.1.2 cycles_to_seconds() . . . . .	88
4.37.1.3 update_upper_cycles() . . . . .	88
4.37.2 Variable Documentation . . . . .	88
4.37.2.1 cpu_cycles_total . . . . .	88
4.37.2.2 cpu_cycles_total_max . . . . .	88
4.37.2.3 memory_used . . . . .	88
4.37.2.4 memory_used_max . . . . .	88
4.37.2.5 trace_depth . . . . .	89
4.37.2.6 update_scheduled . . . . .	89
4.38 m_eng_globals.h . . . . .	89
4.39 m_eng_i2s_dma.h File Reference . . . . .	89
4.39.1 Typedef Documentation . . . . .	90
4.39.1.1 raw_sample_t . . . . .	90
4.39.2 Function Documentation . . . . .	90
4.39.2.1 i2s_input_update() . . . . .	90
4.39.2.2 i2s_output_transmit_mono_float() . . . . .	90
4.39.2.3 i2s_output_transmit_mono_int() . . . . .	90
4.39.2.4 i2s_output_update() . . . . .	90
4.39.2.5 init_i2s_dma() . . . . .	90
4.39.2.6 m_eng_i2s_input_isr() . . . . .	90
4.39.2.7 m_eng_i2s_output_isr() . . . . .	90
4.39.3 Variable Documentation . . . . .	90
4.39.3.1 i2s_input_blocks . . . . .	90
4.40 m_eng_i2s_dma.h . . . . .	91
4.41 m_eng_linkowitz_riley.h File Reference . . . . .	91
4.41.1 Function Documentation . . . . .	91
4.41.1.1 calc_lr_high_pass_filter() . . . . .	91
4.41.1.2 calc_lr_low_pass_filter() . . . . .	91
4.41.1.3 init_lr_high_pass_filter_str() . . . . .	92
4.41.1.4 init_lr_low_pass_filter_str() . . . . .	92
4.41.1.5 reconfigure_lr_high_pass_filter() . . . . .	92
4.41.1.6 reconfigure_lr_low_pass_filter() . . . . .	92
4.42 m_eng_linkowitz_riley.h . . . . .	92
4.43 m_eng_logging.h File Reference . . . . .	93
4.43.1 Macro Definition Documentation . . . . .	94



4.43.1.1 CYCLES_TO_SECONDS . . . . .	94
4.43.1.2 FUNCTION_START . . . . .	94
4.43.1.3 M_ENG_LOG_ENTRY_ERROR . . . . .	94
4.43.1.4 M_ENG_LOG_ENTRY_MESSAGE . . . . .	94
4.43.1.5 M_ENG_LOG_ENTRY_RETURN . . . . .	94
4.43.1.6 M_ENG_LOG_ENTRY_RETURN_ERR . . . . .	95
4.43.1.7 M_ENG_LOG_ENTRY_RETURN_INT . . . . .	95
4.43.1.8 M_ENG_LOG_ENTRY_RETURN_PTR . . . . .	95
4.43.1.9 M_ENG_LOG_ERROR . . . . .	95
4.43.1.10 M_ENG_LOG_ERRORS . . . . .	95
4.43.1.11 M_ENG_LOG_EVERYTHING . . . . .	95
4.43.1.12 M_ENG_LOG_INDENT_TRACE . . . . .	95
4.43.1.13 M_ENG_LOG_LEVEL . . . . .	95
4.43.1.14 M_ENG_LOG_RETURN . . . . .	95
4.43.1.15 M_ENG_LOG_RETURN_ERR_CODE . . . . .	96
4.43.1.16 M_ENG_LOG_RETURN_INT . . . . .	96
4.43.1.17 M_ENG_LOG_RETURN_PTR . . . . .	96
4.43.1.18 M_ENG_LOG_RETURNS . . . . .	96
4.43.1.19 M_ENG_LOG_SPIKES . . . . .	96
4.43.1.20 M_ENG_LOG_TRACE . . . . .	96
4.43.1.21 M_ENG_PROFILER_LOG_ENTRY . . . . .	96
4.43.1.22 M_ENG_PROFILER_LOG_RETURN . . . . .	96
4.43.1.23 M_ENG_TRACE_FUNCTION_ENTER . . . . .	96
4.43.1.24 M_ENG_TRACE_FUNCTION_RETURN . . . . .	96
4.43.1.25 M_ENG_TRACE_LOG_ENTRY . . . . .	97
4.43.1.26 M_ENG_TRACE_LOG_RETURN . . . . .	97
4.43.1.27 M_LOG . . . . .	97
4.43.1.28 M_LOG_ERROR . . . . .	97
4.43.1.29 RETURN . . . . .	97
4.43.1.30 RETURN_ERR_CODE . . . . .	97
4.43.1.31 RETURN_INT . . . . .	97
4.43.1.32 RETURN_NEG_ERR_CODE . . . . .	98
4.43.1.33 RETURN_PTR . . . . .	98
4.43.1.34 RETURN_VOID . . . . .	98
4.43.1.35 SECONDS_TO_CYCLES . . . . .	98
4.43.1.36 STR . . . . .	98
4.43.1.37 XSTR . . . . .	98
4.43.2 Function Documentation . . . . .	99
4.43.2.1 m_eng_log() . . . . .	99
4.43.2.2 m_eng_log_error_code() . . . . .	99
4.43.2.3 m_eng_log_message() . . . . .	99
4.43.2.4 m_eng_log_return_() . . . . .	99

4.43.2.5 m_eng_log_return_err()	99
4.43.2.6 m_eng_log_return_int()	99
4.43.2.7 m_eng_log_return_ptr()	100
4.43.2.8 m_eng_print_flush_log()	100
4.43.2.9 m_eng_print_log()	100
4.43.2.10 m_eng_trace_log_begin()	100
4.43.2.11 m_eng_trace_log_return()	100
4.44 m_eng_logging.h	100
4.45 m_eng_low_end_compressor.h File Reference	102
4.45.1 Function Documentation	103
4.45.1.1 calc_low_end_compressor()	103
4.45.1.2 init_low_end_compressor_str()	103
4.45.1.3 reconfigure_low_end_compressor()	103
4.46 m_eng_low_end_compressor.h	103
4.47 m_eng_memcpy_audio.h File Reference	103
4.47.1 Function Documentation	104
4.47.1.1 memcpy_tointerleaveL()	104
4.47.1.2 memcpy_tointerleaveLR()	104
4.47.1.3 memcpy_tointerleaveQuad()	104
4.47.1.4 memcpy_tointerleaveR()	104
4.48 m_eng_memcpy_audio.h	104
4.49 m_eng_mempool.h File Reference	105
4.49.1 Macro Definition Documentation	105
4.49.1.1 M_BUFFER_POOL_SIZE	105
4.49.1.2 MAX_AUDIO_MEMORY	105
4.49.1.3 MEM_SIZE	105
4.49.2 Function Documentation	106
4.49.2.1 allocate_buffer()	106
4.49.2.2 init_mem_pools()	106
4.49.2.3 print_mempool_info()	106
4.49.2.4 release_buffer()	106
4.49.3 Variable Documentation	106
4.49.3.1 sink_buffer	106
4.49.3.2 zero_buffer	106
4.50 m_eng_mempool.h	106
4.51 m_eng_noise_suppressor.h File Reference	107
4.51.1 Function Documentation	107
4.51.1.1 calc_noise_suppressor()	107
4.51.1.2 init_noise_suppressor_str()	107
4.51.1.3 reconfigure_noise_suppressor()	107
4.52 m_eng_noise_suppressor.h	107
4.53 m_eng_parameter.h File Reference	108

4.53.1 Macro Definition Documentation . . . . .	108
4.53.1.1 DEFAULT_MAX_JUMP . . . . .	108
4.53.1.2 PARAM_NAM_ENG_MAX_LEN . . . . .	108
4.53.1.3 PARAMETER_UPDATE_BIBLOCK_LINEAR . . . . .	108
4.53.1.4 PARAMETER_UPDATE_INSTANT . . . . .	108
4.53.1.5 PARAMETER_UPDATE_MONOBLOCK_LINEAR . . . . .	108
4.53.1.6 PARAMETER_UPDATE_QUADBLOCK_LINEAR . . . . .	109
4.53.2 Function Documentation . . . . .	109
4.53.2.1 init_parameter() . . . . .	109
4.53.2.2 init_setting() . . . . .	109
4.54 m_eng_parameter.h . . . . .	109
4.55 m_eng_pass_filter.h File Reference . . . . .	109
4.55.1 Function Documentation . . . . .	110
4.55.1.1 calc_band_pass_filter() . . . . .	110
4.55.1.2 calc_high_pass_filter() . . . . .	110
4.55.1.3 calc_low_pass_filter() . . . . .	110
4.55.1.4 init_band_pass_filter_str() . . . . .	110
4.55.1.5 init_high_pass_filter_str() . . . . .	110
4.55.1.6 init_low_pass_filter_str() . . . . .	111
4.55.1.7 reconfigure_band_pass_filter() . . . . .	111
4.55.1.8 reconfigure_high_pass_filter() . . . . .	111
4.55.1.9 reconfigure_low_pass_filter() . . . . .	111
4.56 m_eng_pass_filter.h . . . . .	111
4.57 m_eng_percussifier.h File Reference . . . . .	112
4.57.1 Macro Definition Documentation . . . . .	112
4.57.1.1 PERCUSSIFIER_FADE_IN . . . . .	112
4.57.1.2 PERCUSSIFIER_FADE_OUT . . . . .	112
4.57.1.3 PERCUSSIFIER_HOLD . . . . .	112
4.57.1.4 PERCUSSIFIER_MUTE . . . . .	112
4.57.1.5 PERCUSSIFIER_REFRACTORY . . . . .	112
4.57.2 Function Documentation . . . . .	113
4.57.2.1 calc_percussifier() . . . . .	113
4.57.2.2 init_percussifier_str() . . . . .	113
4.57.2.3 reconfigure_percussifier() . . . . .	113
4.58 m_eng_percussifier.h . . . . .	113
4.59 m_eng_pipeline.h File Reference . . . . .	114
4.59.1 Macro Definition Documentation . . . . .	114
4.59.1.1 INITIAL_TRANSFORMER_ARRAY_LENGTH . . . . .	114
4.59.2 Function Documentation . . . . .	114
4.59.2.1 clone_pipeline() . . . . .	114
4.59.2.2 compute_pipeline() . . . . .	115
4.59.2.3 gut_pipeline() . . . . .	115

4.59.2.4	<a href="#">init_pipeline()</a>	115
4.59.2.5	<a href="#">pipeline_append_transformer()</a>	115
4.59.2.6	<a href="#">pipeline_append_transformer_type()</a>	115
4.59.2.7	<a href="#">pipeline_change_transformer_setting()</a>	115
4.59.2.8	<a href="#">pipeline_clone_transformer_into_position()</a>	115
4.59.2.9	<a href="#">pipeline_compare()</a>	116
4.59.2.10	<a href="#">pipeline_expand_transformer_array()</a>	116
4.59.2.11	<a href="#">pipeline_expand_transformer_array_to()</a>	116
4.59.2.12	<a href="#">pipeline_get_transformer_by_id()</a>	116
4.59.2.13	<a href="#">pipeline_get_transformer_position()</a>	116
4.59.2.14	<a href="#">pipeline_insert_transformer()</a>	116
4.59.2.15	<a href="#">pipeline_insert_transformer_type()</a>	116
4.59.2.16	<a href="#">pipeline_move_transformer()</a>	117
4.59.2.17	<a href="#">pipeline_prepend_transformer()</a>	117
4.59.2.18	<a href="#">pipeline_prepend_transformer_type()</a>	117
4.59.2.19	<a href="#">pipeline_remove_transformer()</a>	117
4.59.2.20	<a href="#">pipeline_swap_transformers()</a>	117
4.59.2.21	<a href="#">pipeline_update_transition_policy()</a>	117
4.59.2.22	<a href="#">pipeline_valid()</a>	117
4.60	<a href="#">m_eng_pipeline.h</a>	118
4.61	<a href="#">m_eng_pipeline_mod.h</a> File Reference	118
4.61.1	Macro Definition Documentation	119
4.61.1.1	<a href="#">PIPELINE_MOD_APPEND_TRANSFORMER</a>	119
4.61.1.2	<a href="#">PIPELINE_MOD_CHANGE_TRANSFORMER_SETTING</a>	119
4.61.1.3	<a href="#">PIPELINE_MOD_MOVE_TRANSFORMER</a>	119
4.61.1.4	<a href="#">PIPELINE_MOD_REMOVE_TRANSFORMER</a>	119
4.61.2	Function Documentation	119
4.61.2.1	<a href="#">apply_pipeline_mod()</a>	119
4.61.2.2	<a href="#">create_pipeline_mod_append_transformer()</a>	119
4.61.2.3	<a href="#">create_pipeline_mod_change_transformer_setting()</a>	119
4.61.2.4	<a href="#">create_pipeline_mod_move_transformer()</a>	119
4.61.2.5	<a href="#">create_pipeline_mod_remove_transformer()</a>	120
4.61.2.6	<a href="#">DECLARE_LINKED_LIST()</a>	120
4.61.2.7	<a href="#">init_pipeline_mod()</a>	120
4.61.2.8	<a href="#">pipeline_mod_type_string()</a>	120
4.62	<a href="#">m_eng_pipeline_mod.h</a>	120
4.63	<a href="#">m_eng_printf.h</a> File Reference	121
4.63.1	Function Documentation	121
4.63.1.1	<a href="#">m_mute_voice()</a>	121
4.63.1.2	<a href="#">m_printf()</a>	121
4.63.1.3	<a href="#">m_unmute_voice()</a>	121
4.63.1.4	<a href="#">m_voice_printf()</a>	121

4.63.1.5 pretty_print_block()	121
4.63.1.6 pretty_print_block_float()	121
4.63.1.7 serial_print_blocks()	122
4.64 m_eng_printf.h	122
4.65 m_eng_profile.h File Reference	122
4.65.1 Function Documentation	122
4.65.1.1 init_bypass_profile()	122
4.65.1.2 init_profile()	123
4.65.1.3 nullify_profile()	123
4.65.1.4 profile_apply_pipeline_mod()	123
4.65.1.5 profile_process()	123
4.65.1.6 profile_scheduled_maintenance()	123
4.65.1.7 profile_trigger_pipeline_swap()	124
4.65.1.8 profile_update()	124
4.66 m_eng_profile.h	124
4.67 m_eng_sgtl5000.h File Reference	124
4.67.1 Function Documentation	125
4.67.1.1 calc_vol()	125
4.67.1.2 sgtl5000_adc_high_pass_filter_disable()	125
4.67.1.3 sgtl5000_adc_high_pass_filter_enable()	125
4.67.1.4 sgtl5000_adc_high_pass_filter_freeze()	125
4.67.1.5 sgtl5000_automate()	125
4.67.1.6 sgtl5000_dap_audio_eq_band()	126
4.67.1.7 sgtl5000_enable()	126
4.67.1.8 sgtl5000_healthy()	126
4.67.1.9 sgtl5000_kill_automation()	126
4.67.1.10 sgtl5000_line_in_level()	126
4.67.1.11 sgtl5000_line_out_level()	126
4.67.1.12 sgtl5000_mic_gain()	126
4.67.1.13 sgtl5000_modify_reg()	126
4.67.1.14 sgtl5000_mute_headphone()	126
4.67.1.15 sgtl5000_mute_line_out()	127
4.67.1.16 sgtl5000_read_reg()	127
4.67.1.17 sgtl5000_set_address()	127
4.67.1.18 sgtl5000_set_master_mode()	127
4.67.1.19 sgtl5000_soft_reboot()	127
4.67.1.20 sgtl5000_start()	127
4.67.1.21 sgtl5000_unmute_headphone()	127
4.67.1.22 sgtl5000_unmute_line_out()	127
4.67.1.23 sgtl5000_volum_eng_integer()	127
4.67.1.24 sgtl5000_volume()	128
4.67.1.25 sgtl5000_write_reg()	128

4.68 m_eng_sgtl5000.h . . . . .	128
4.69 m_eng_sgtl5000_defs.h File Reference . . . . .	128
4.69.1 Macro Definition Documentation . . . . .	130
4.69.1.1 AUDIO_HEADPHONE_DAC . . . . .	130
4.69.1.2 AUDIO_HEADPHONE_LINEIN . . . . .	130
4.69.1.3 CHIP_ADCDAC_CTRL . . . . .	130
4.69.1.4 CHIP_ANA_ADC_CTRL . . . . .	130
4.69.1.5 CHIP_ANA_CTRL . . . . .	130
4.69.1.6 CHIP_ANA_HP_CTRL . . . . .	130
4.69.1.7 CHIP_ANA_POWER . . . . .	130
4.69.1.8 CHIP_ANA_STATUS . . . . .	130
4.69.1.9 CHIP_ANA_TEST1 . . . . .	131
4.69.1.10 CHIP_ANA_TEST2 . . . . .	131
4.69.1.11 CHIP_CLK_CTRL . . . . .	131
4.69.1.12 CHIP_CLK_TOP_CTRL . . . . .	131
4.69.1.13 CHIP_DAC_VOL . . . . .	131
4.69.1.14 CHIP_DIG_POWER . . . . .	131
4.69.1.15 CHIP_I2S_CTRL . . . . .	131
4.69.1.16 CHIP_ID . . . . .	131
4.69.1.17 CHIP_LINE_OUT_CTRL . . . . .	131
4.69.1.18 CHIP_LINE_OUT_VOL . . . . .	131
4.69.1.19 CHIP_LINREG_CTRL . . . . .	132
4.69.1.20 CHIP_MIC_CTRL . . . . .	132
4.69.1.21 CHIP_PAD_STRENGTH . . . . .	132
4.69.1.22 CHIP_PLL_CTRL . . . . .	132
4.69.1.23 CHIP_REF_CTRL . . . . .	132
4.69.1.24 CHIP_SHORT_CTRL . . . . .	132
4.69.1.25 CHIP_SSS_CTRL . . . . .	132
4.69.1.26 DAP_AUDIO_EQ . . . . .	132
4.69.1.27 DAP_AUDIO_EQ_BAND1 . . . . .	132
4.69.1.28 DAP_AUDIO_EQ_BAND2 . . . . .	132
4.69.1.29 DAP_AUDIO_EQ_BAND3 . . . . .	133
4.69.1.30 DAP_AUDIO_EQ_BASS_BAND0 . . . . .	133
4.69.1.31 DAP_AUDIO_EQ_TREBLE_BAND4 . . . . .	133
4.69.1.32 DAP_AVC_ATTACK . . . . .	133
4.69.1.33 DAP_AVC_CTRL . . . . .	133
4.69.1.34 DAP_AVC_DECAY . . . . .	133
4.69.1.35 DAP_AVC_THRESHOLD . . . . .	133
4.69.1.36 DAP_BASS_ENHANCE . . . . .	133
4.69.1.37 DAP_BASS_ENHANCE_CTRL . . . . .	133
4.69.1.38 DAP_COEF_WR_A1_LSB . . . . .	133
4.69.1.39 DAP_COEF_WR_A1_MSB . . . . .	134

4.69.1.40 DAP_COEF_WR_A2_LSB . . . . .	134
4.69.1.41 DAP_COEF_WR_A2_MSB . . . . .	134
4.69.1.42 DAP_COEF_WR_B0_LSB . . . . .	134
4.69.1.43 DAP_COEF_WR_B0_MSB . . . . .	134
4.69.1.44 DAP_COEF_WR_B1_LSB . . . . .	134
4.69.1.45 DAP_COEF_WR_B1_MSB . . . . .	134
4.69.1.46 DAP_COEF_WR_B2_LSB . . . . .	134
4.69.1.47 DAP_COEF_WR_B2_MSB . . . . .	134
4.69.1.48 DAP_CONTROL . . . . .	134
4.69.1.49 DAP_FILTER_COEF_ACCESS . . . . .	135
4.69.1.50 DAP_MAIN_CHAN . . . . .	135
4.69.1.51 DAP_MIX_CHAN . . . . .	135
4.69.1.52 DAP_PEQ . . . . .	135
4.69.1.53 DAP_SGTL_SURROUND . . . . .	135
4.69.1.54 FILTER_BANDPASS . . . . .	135
4.69.1.55 FILTER_HIPASS . . . . .	135
4.69.1.56 FILTER_HISHELF . . . . .	135
4.69.1.57 FILTER_LOPASS . . . . .	135
4.69.1.58 FILTER_LOSHELF . . . . .	135
4.69.1.59 FILTER_NOTCH . . . . .	136
4.69.1.60 FILTER_PARAEQ . . . . .	136
4.69.1.61 FLAT_FREQUENCY . . . . .	136
4.69.1.62 GRAPHIC_EQUALIZER . . . . .	136
4.69.1.63 PARAMETRIC_EQUALIZER . . . . .	136
4.69.1.64 SGTL5000_I2C_ADDR_CS_HIGH . . . . .	136
4.69.1.65 SGTL5000_I2C_ADDR_CS_LOW . . . . .	136
4.69.1.66 TONE_CONTROLS . . . . .	136
4.70 m_eng_sgtl5000_defs.h . . . . .	137
4.71 m_eng_simple_distortion.h File Reference . . . . .	142
4.71.1 Function Documentation . . . . .	143
4.71.1.1 calc_simple_distortion() . . . . .	143
4.71.1.2 init_simple_distortion_str() . . . . .	143
4.71.1.3 reconfigure_simple_distortion() . . . . .	143
4.72 m_eng_simple_distortion.h . . . . .	143
4.73 m_eng_transformer.h File Reference . . . . .	143
4.73.1 Macro Definition Documentation . . . . .	144
4.73.1.1 FADER_FADE_IN . . . . .	144
4.73.1.2 FADER_FADE_OUT . . . . .	144
4.73.1.3 N_NATIVE_PARAMETERS . . . . .	144
4.73.1.4 N_NATIVE_SETTINGS . . . . .	144
4.73.1.5 PRINT_TRANSFORMER_INFO . . . . .	145
4.73.1.6 TRANSFORMER_MAX_INPUTS . . . . .	145

4.73.1.7 TRANSFORMER_MAX_OUTPUTS . . . . .	145
4.73.1.8 TRANSFORMER_SWITCH_ACTION_BYPASS . . . . .	145
4.73.1.9 TRANSFORMER_TRANSITION_BIBLOCK_LINEAR . . . . .	145
4.73.1.10 TRANSFORMER_TRANSITION_INSTANT . . . . .	145
4.73.1.11 TRANSFORMER_TRANSITION_MONOBLOCK_LINEAR . . . . .	145
4.73.1.12 TRANSFORMER_TRANSITION_QUADBLOCK_LINEAR . . . . .	145
4.73.1.13 TRANSFORMER_TRANSITION_TAIL . . . . .	145
4.73.2 Function Documentation . . . . .	145
4.73.2.1 clone_transformer() . . . . .	145
4.73.2.2 free_transformer() . . . . .	146
4.73.2.3 run_transformer() . . . . .	146
4.73.2.4 transformer_add_parameter() . . . . .	146
4.73.2.5 transformer_add_setting() . . . . .	146
4.73.2.6 transformer_get_parameter() . . . . .	146
4.73.2.7 transformer_get_setting() . . . . .	146
4.73.2.8 transformer_init_controls() . . . . .	146
4.73.2.9 transformer_init_parameter_array() . . . . .	146
4.73.2.10 transformer_init_setting_array() . . . . .	147
4.73.2.11 transformer_type_to_string() . . . . .	147
4.74 m_eng_transformer.h . . . . .	147
4.75 m_eng_transformer_init.h File Reference . . . . .	147
4.75.1 Function Documentation . . . . .	148
4.75.1.1 init_transformer() . . . . .	148
4.76 m_eng_transformer_init.h . . . . .	148
4.77 m_eng_transformer_template.h File Reference . . . . .	148
4.77.1 Function Documentation . . . . .	148
4.77.1.1 calc_transformer() . . . . .	148
4.77.1.2 init_transformer_str() . . . . .	148
4.77.1.3 reconfigure_transformer() . . . . .	149
4.78 m_eng_transformer_template.h . . . . .	149
4.79 m_eng_transition.h File Reference . . . . .	149
4.79.1 Macro Definition Documentation . . . . .	149
4.79.1.1 TAIL_INPUT_FADE_SAMPLES . . . . .	149
4.79.1.2 TAIL_NEW_FADE_IN_SAMPLES . . . . .	149
4.79.1.3 TRANSITION_MONOBLOCK_COS2 . . . . .	149
4.79.1.4 TRANSITION_OCTOBLOCK_COS2 . . . . .	150
4.79.1.5 TRANSITION_QUADBLOCK_COS2 . . . . .	150
4.79.1.6 TRANSITION_TAIL . . . . .	150
4.80 m_eng_transition.h . . . . .	150
4.81 m_eng_update.h File Reference . . . . .	150
4.81.1 Macro Definition Documentation . . . . .	150
4.81.1.1 m_eng_disable_software_interrupts . . . . .	150



4.81.1.2 m_eng_enable_software_interrupts . . . . .	151
4.81.2 Function Documentation . . . . .	151
4.81.2.1 m_eng_software_isr() . . . . .	151
4.81.2.2 update_all() . . . . .	151
4.81.2.3 update_setup() . . . . .	151
4.81.2.4 update_stop() . . . . .	151
4.82 m_eng_update.h . . . . .	151
4.83 m_eng_useful_functions.h File Reference . . . . .	152
4.83.1 Function Documentation . . . . .	152
4.83.1.1 convert_block_float_to_int() . . . . .	152
4.83.1.2 convert_block_int_to_float() . . . . .	152
4.83.1.3 hard_clip() . . . . .	152
4.83.1.4 identity_function() . . . . .	152
4.83.1.5 normalised_arctan() . . . . .	152
4.83.1.6 soft_fold() . . . . .	152
4.83.1.7 trig_transition_function() . . . . .	153
4.84 m_eng_useful_functions.h . . . . .	153
4.85 m_eng_warbler.h File Reference . . . . .	153
4.85.1 Function Documentation . . . . .	153
4.85.1.1 calc_warbler() . . . . .	153
4.85.1.2 init_warbler_str() . . . . .	154
4.85.1.3 reconfigure_warbler() . . . . .	154
4.86 m_eng_warbler.h . . . . .	154
4.87 m_eng_waveshaper.h File Reference . . . . .	154
4.87.1 Macro Definition Documentation . . . . .	155
4.87.1.1 WAVESHAPER_ENVELOPE_ATTACK . . . . .	155
4.87.1.2 WAVESHAPER_ENVELOPE_RELEASE . . . . .	155
4.87.2 Function Documentation . . . . .	155
4.87.2.1 calc_waveshaper() . . . . .	155
4.87.2.2 init_waveshaper_str() . . . . .	155
4.88 m_eng_waveshaper.h . . . . .	155
4.89 m_eng.cpp File Reference . . . . .	155
4.89.1 Macro Definition Documentation . . . . .	156
4.89.1.1 DEBUG_PRINT_MILLIS . . . . .	156
4.89.1.2 LED_BLINK_MILLIS . . . . .	156
4.89.1.3 LOG_PRINT_MILLIS . . . . .	156
4.89.1.4 MEM_REPORT_MILLIS . . . . .	156
4.89.1.5 PRINT_LOG . . . . .	156
4.89.1.6 PROFILER_PRINT_MILLIS . . . . .	156
4.89.1.7 SCHEDULED_MAINTAINANCE . . . . .	157
4.89.1.8 SCHEDULED_MAINTAINANCE_MILLIS . . . . .	157
4.89.1.9 SGT5000_CHECK_PERIOD . . . . .	157

4.89.2 Function Documentation	157
4.89.2.1 main()	157
4.90 m_eng_adaptive_waveshaper.c File Reference	157
4.90.1 Function Documentation	157
4.90.1.1 calc_adaptive_waveshaper()	157
4.90.1.2 init_adaptive_waveshaper_str()	157
4.91 m_eng_audio_block.c File Reference	158
4.92 m_eng_band_splitter.c File Reference	158
4.93 m_eng_biquad.c File Reference	158
4.93.1 Function Documentation	158
4.93.1.1 calc_biquad()	158
4.93.1.2 init_biquad_str()	158
4.93.1.3 reconfigure_biquad()	158
4.94 m_eng_buffer_mixer_amp.c File Reference	158
4.94.1 Function Documentation	159
4.94.1.1 calc_amplifier()	159
4.94.1.2 calc_buffer()	159
4.94.1.3 init_amplifier_str()	159
4.94.1.4 reconfigure_amplifier()	159
4.95 m_eng_comms.cpp File Reference	159
4.95.1 Macro Definition Documentation	160
4.95.1.1 PRINT_RESPONSE_BYTES	160
4.95.2 Enumeration Type Documentation	160
4.95.2.1 comms_fsm_eng_state_t	160
4.95.3 Function Documentation	161
4.95.3.1 esp32_message_check_handle()	161
4.95.3.2 handle_esp32_message()	161
4.95.3.3 i2c_receive_isr()	161
4.95.3.4 i2c_request_isr()	161
4.95.3.5 init_esp32_link()	161
4.95.3.6 m_message_sanity_check()	161
4.95.4 Variable Documentation	161
4.95.4.1 comms_fsm_eng_state	161
4.95.4.2 message_pending	161
4.95.4.3 prev_response	161
4.95.4.4 receive_buffer	162
4.95.4.5 received	162
4.95.4.6 received_length	162
4.95.4.7 response	162
4.95.4.8 response_buffer	162
4.95.4.9 response_length	162
4.95.4.10 response_ready	162

4.95.4.11 string_in . . . . .	162
4.95.4.12 string_in_pos . . . . .	162
4.95.4.13 string_out . . . . .	162
4.95.4.14 string_out_pos . . . . .	163
4.95.4.15 wait_message . . . . .	163
4.96 m_eng_compressor.c File Reference . . . . .	163
4.96.1 Macro Definition Documentation . . . . .	163
4.96.1.1 EPSILON . . . . .	163
4.96.2 Function Documentation . . . . .	163
4.96.2.1 calc_compressor() . . . . .	163
4.96.2.2 init_compressor_str() . . . . .	163
4.96.2.3 reconfigure_compressor() . . . . .	164
4.97 m_eng_context.c File Reference . . . . .	164
4.97.1 Macro Definition Documentation . . . . .	165
4.97.1.1 AVG_DURATION_UPDATE_COEF . . . . .	165
4.97.2 Function Documentation . . . . .	165
4.97.2.1 cxt_append_transformer_to_profile() . . . . .	165
4.97.2.2 cxt_get_back_parameter_by_id() . . . . .	165
4.97.2.3 cxt_get_front_parameter_by_id() . . . . .	165
4.97.2.4 cxt_get_n_profile_transformers() . . . . .	165
4.97.2.5 cxt_get_n_transformer_params() . . . . .	165
4.97.2.6 cxt_get_n_transformer_settings() . . . . .	166
4.97.2.7 cxt_get_parameter_by_id() . . . . .	166
4.97.2.8 cxt_get_setting_by_id() . . . . .	166
4.97.2.9 cxt_get_tid_by_pos() . . . . .	166
4.97.2.10 cxt_get_transformer_by_id() . . . . .	166
4.97.2.11 cxt_get_transformer_type() . . . . .	166
4.97.2.12 cxt_insert_transformer_to_profile() . . . . .	167
4.97.2.13 cxt_move_transformer() . . . . .	167
4.97.2.14 cxt_prepend_transformer_to_profile() . . . . .	167
4.97.2.15 cxt_process() . . . . .	167
4.97.2.16 cxt_profile_id_valid() . . . . .	167
4.97.2.17 cxt_remove_transformer_from_profile() . . . . .	167
4.97.2.18 cxt_run_scheduled_maintenance() . . . . .	167
4.97.2.19 cxt_set_active_profile() . . . . .	168
4.97.2.20 cxt_switch_to_profile() . . . . .	168
4.97.2.21 cxt_transformer_id_valid() . . . . .	168
4.97.2.22 cxt_update_parameter_value_by_id() . . . . .	168
4.97.2.23 cxt_update_setting_value_by_id() . . . . .	168
4.97.2.24 init_m_eng_context() . . . . .	168
4.97.2.25 m_eng_context_new_profile() . . . . .	168
4.97.2.26 m_eng_safe_reboot() . . . . .	169

4.97.2.27 pcxt_profile_id_valid()	169
4.97.2.28 reset_context()	169
4.98 m_eng_debugging.c File Reference	169
4.98.1 Function Documentation	169
4.98.1.1 full_debug_print()	169
4.98.1.2 print_binary()	169
4.98.1.3 print_context_info()	170
4.98.1.4 print_pipeline_info()	170
4.98.1.5 print_profile_info()	170
4.98.1.6 print_transformer_info()	170
4.99 m_eng_delay.c File Reference	170
4.99.1 Function Documentation	170
4.99.1.1 calc_delay()	170
4.99.1.2 init_delay_str()	171
4.99.1.3 reconfigure_delay()	171
4.100 m_eng_delay_buffer.c File Reference	171
4.100.1 Function Documentation	171
4.100.1.1 init_delay_buffer()	171
4.100.1.2 m_delay_buffer_advance()	171
4.100.1.3 m_delay_buffer_get_delayed_sample()	171
4.100.1.4 m_delay_buffer_get_delayed_sample_ptr()	172
4.100.1.5 m_delay_buffer_get_fractional_delayed_sample()	172
4.100.1.6 m_delay_buffer_resize_milliseconds()	172
4.100.1.7 m_delay_buffer_resize_samples()	172
4.100.1.8 m_delay_buffer_resize_seconds()	172
4.100.1.9 m_delay_buffer_tick()	172
4.101 m_eng_dirty_octave.c File Reference	172
4.101.1 Function Documentation	173
4.101.1.1 calc_dirty_octave()	173
4.101.1.2 init_dirty_octave_str()	173
4.101.1.3 reconfigure_dirty_octave()	173
4.102 m_eng_distortion.c File Reference	173
4.102.1 Function Documentation	173
4.102.1.1 calc_distortion()	173
4.102.1.2 init_distortion_str()	173
4.102.1.3 reconfigure_distortion()	174
4.103 m_eng_envelope.c File Reference	174
4.103.1 Function Documentation	174
4.103.1.1 calc_envelope()	174
4.103.1.2 init_envelope_str()	174
4.103.1.3 reconfigure_envelope()	174
4.104 m_eng_equaliser.c File Reference	174

4.104.1 Function Documentation	175
4.104.1.1 calc_3_band_eq()	175
4.104.1.2 init_3_band_eq_str()	175
4.104.1.3 reconfigure_3_band_eq()	175
4.105 m_eng_flanger.c File Reference	175
4.105.1 Function Documentation	175
4.105.1.1 calc_flanger()	175
4.105.1.2 free_flanger_struct()	176
4.105.1.3 init_flanger_str()	176
4.105.1.4 reconfigure_flanger()	176
4.106 m_eng_globals.c File Reference	176
4.106.1 Function Documentation	176
4.106.1.1 current_cycle()	176
4.106.1.2 cycles_to_seconds()	176
4.106.2 Variable Documentation	177
4.106.2.1 cpu_cycles_total	177
4.106.2.2 cpu_cycles_total_max	177
4.106.2.3 cycles_upper	177
4.106.2.4 global_cxt	177
4.106.2.5 memory_used	177
4.106.2.6 memory_used_max	177
4.106.2.7 trace_depth	177
4.106.2.8 update_scheduled	177
4.107 m_eng_i2s_dma.cpp File Reference	177
4.107.1 Function Documentation	178
4.107.1.1 __attribute__()	178
4.107.1.2 configure_i2s_dma()	178
4.107.1.3 i2s_in_dma()	178
4.107.1.4 i2s_in_transmit()	179
4.107.1.5 i2s_input_update()	179
4.107.1.6 i2s_out_dma()	179
4.107.1.7 i2s_output_transmit_mono_float()	179
4.107.1.8 i2s_output_transmit_mono_int()	179
4.107.1.9 i2s_output_update()	179
4.107.1.10 init_i2s_dma()	179
4.107.1.11 m_eng_i2s_input_isr()	179
4.107.1.12 m_eng_i2s_output_isr()	179
4.107.2 Variable Documentation	180
4.107.2.1 i2s_in_block_left	180
4.107.2.2 i2s_in_block_offset	180
4.107.2.3 i2s_in_block_right	180
4.107.2.4 i2s_in_update_responsibility	180

4.107.2.5 i2s_input_blocks . . . . .	180
4.107.2.6 i2s_out_block_left_1st . . . . .	180
4.107.2.7 i2s_out_block_left_2nd . . . . .	180
4.107.2.8 i2s_out_block_left_offset . . . . .	180
4.107.2.9 i2s_out_block_right_1st . . . . .	180
4.107.2.10 i2s_out_block_right_2nd . . . . .	180
4.107.2.11 i2s_out_block_right_offset . . . . .	181
4.107.2.12 i2s_out_update_responsibility . . . . .	181
4.107.2.13 i2s_output_blocks . . . . .	181
4.108 m_eng_linkowitz_riley.c File Reference . . . . .	181
4.108.1 Function Documentation . . . . .	181
4.108.1.1 calc_lr_high_pass_filter() . . . . .	181
4.108.1.2 calc_lr_low_pass_filter() . . . . .	181
4.108.1.3 init_lr_high_pass_filter_str() . . . . .	182
4.108.1.4 init_lr_low_pass_filter_str() . . . . .	182
4.108.1.5 reconfigure_lr_high_pass_filter() . . . . .	182
4.108.1.6 reconfigure_lr_low_pass_filter() . . . . .	182
4.109 m_eng_logging.cpp File Reference . . . . .	182
4.109.1 Macro Definition Documentation . . . . .	183
4.109.1.1 LOG_ENTRIES_PRINT_BUF_LEN . . . . .	183
4.109.1.2 M_ENG_LOG_ENTRIES_N . . . . .	183
4.109.1.3 M_ENG_PROFILER_ARRAY_N . . . . .	183
4.109.1.4 M_ENG_PROFILER_RA_CYCLES_ALPHA . . . . .	183
4.109.1.5 MESSAGE_BEGIN_COL . . . . .	183
4.109.2 Function Documentation . . . . .	184
4.109.2.1 format_log_entry() . . . . .	184
4.109.2.2 m_eng_init_profiler() . . . . .	184
4.109.2.3 m_eng_log_error_code() . . . . .	184
4.109.2.4 m_eng_log_message() . . . . .	184
4.109.2.5 m_eng_log_return_() . . . . .	184
4.109.2.6 m_eng_log_return_err() . . . . .	184
4.109.2.7 m_eng_log_return_int() . . . . .	185
4.109.2.8 m_eng_log_return_ptr() . . . . .	185
4.109.2.9 m_eng_print_flush_log() . . . . .	185
4.109.2.10 m_eng_profiler_log_entry() . . . . .	185
4.109.2.11 m_eng_profiler_log_return() . . . . .	185
4.109.2.12 m_eng_profiler_print() . . . . .	185
4.109.2.13 m_eng_profiler_sort() . . . . .	185
4.109.2.14 m_eng_trace_log_begin() . . . . .	185
4.109.2.15 m_eng_trace_log_return() . . . . .	186
4.110 m_eng_low_end_compressor.c File Reference . . . . .	186
4.110.1 Function Documentation . . . . .	186

4.110.1.1 calc_low_end_compressor()	186
4.110.1.2 init_low_end_compressor_str()	186
4.110.1.3 reconfigure_low_end_compressor()	186
4.111 m_eng_mempool.c File Reference	186
4.111.1 Macro Definition Documentation	187
4.111.1.1 BUFFER_QUEUE_STATIC	187
4.111.1.2 MEMPOOL_MALLOC_TRIES	187
4.111.2 Function Documentation	187
4.111.2.1 allocate_buffer()	187
4.111.2.2 init_mem_pools()	187
4.111.2.3 print_mempool_info()	188
4.111.2.4 release_buffer()	188
4.111.3 Variable Documentation	188
4.111.3.1 buffer_buffer	188
4.111.3.2 buffer_head	188
4.111.3.3 buffer_pool	188
4.111.3.4 buffer_tail	188
4.111.3.5 head	188
4.111.3.6 mem_pools_initialised	188
4.111.3.7 sink_buffer	188
4.111.3.8 tail	188
4.111.3.9 zero_buffer	189
4.112 m_eng_noise_suppressor.c File Reference	189
4.112.1 Function Documentation	189
4.112.1.1 calc_noise_suppressor()	189
4.112.1.2 init_noise_suppressor_str()	189
4.112.1.3 reconfigure_noise_suppressor()	189
4.113 m_eng_parameter.c File Reference	189
4.113.1 Function Documentation	190
4.113.1.1 init_parameter()	190
4.113.1.2 init_setting()	190
4.113.1.3 update_setting()	190
4.114 m_eng_pass_filter.c File Reference	190
4.114.1 Function Documentation	190
4.114.1.1 calc_band_pass_filter()	190
4.114.1.2 calc_high_pass_filter()	191
4.114.1.3 calc_low_pass_filter()	191
4.114.1.4 init_band_pass_filter_str()	191
4.114.1.5 init_high_pass_filter_str()	191
4.114.1.6 init_low_pass_filter_str()	191
4.114.1.7 reconfigure_band_pass_filter()	191
4.114.1.8 reconfigure_high_pass_filter()	191

4.114.1.9 reconfigure_low_pass_filter()	191
4.115 m_eng_percussifier.c File Reference	192
4.115.1 Function Documentation	192
4.115.1.1 calc_percussifier()	192
4.115.1.2 init_percussifier_str()	192
4.115.1.3 reconfigure_percussifier()	192
4.116 m_eng_pipeline.c File Reference	192
4.116.1 Macro Definition Documentation	193
4.116.1.1 TRANSFORMERS_MALLOC_CHUNK_SIZE	193
4.116.2 Function Documentation	193
4.116.2.1 clone_pipeline()	193
4.116.2.2 compute_pipeline()	193
4.116.2.3 gut_pipeline()	194
4.116.2.4 init_pipeline()	194
4.116.2.5 pipeline_append_transformer()	194
4.116.2.6 pipeline_append_transformer_type()	194
4.116.2.7 pipeline_change_transformer_setting()	194
4.116.2.8 pipeline_clone_transformer_into_position()	194
4.116.2.9 pipeline_compare()	194
4.116.2.10 pipeline_expand_transformer_array()	194
4.116.2.11 pipeline_expand_transformer_array_to()	195
4.116.2.12 pipeline_get_transformer_by_id()	195
4.116.2.13 pipeline_get_transformer_position()	195
4.116.2.14 pipeline_insert_transformer()	195
4.116.2.15 pipeline_insert_transformer_type()	195
4.116.2.16 pipeline_move_transformer()	195
4.116.2.17 pipeline_prepend_transformer()	195
4.116.2.18 pipeline_prepend_transformer_type()	196
4.116.2.19 pipeline_print_transformer_array()	196
4.116.2.20 pipeline_remove_transformer()	196
4.116.2.21 pipeline_swap_transformers()	196
4.116.2.22 pipeline_update_transition_policy()	196
4.116.2.23 pipeline_valid()	196
4.117 m_eng_pipeline_mod.c File Reference	196
4.117.1 Function Documentation	197
4.117.1.1 apply_pipeline_mod()	197
4.117.1.2 create_pipeline_mod_append_transformer()	197
4.117.1.3 create_pipeline_mod_change_transformer_setting()	197
4.117.1.4 create_pipeline_mod_move_transformer()	197
4.117.1.5 create_pipeline_mod_remove_transformer()	197
4.117.1.6 IMPLEMENT_LINKED_LIST()	197
4.117.1.7 pipeline_mod_type_string()	197



4.118 m_eng_printf.cpp File Reference . . . . .	197
4.118.1 Macro Definition Documentation . . . . .	198
4.118.1.1 SPACING [1/2] . . . . .	198
4.118.1.2 SPACING [2/2] . . . . .	198
4.118.2 Function Documentation . . . . .	198
4.118.2.1 m_mute_voice() . . . . .	198
4.118.2.2 m_printf() . . . . .	198
4.118.2.3 m_unmute_voice() . . . . .	198
4.118.2.4 m_voice_printf() . . . . .	199
4.118.2.5 pretty_print_block() . . . . .	199
4.118.2.6 pretty_print_block_float() . . . . .	199
4.118.2.7 serial_print_blocks() . . . . .	199
4.118.3 Variable Documentation . . . . .	199
4.118.3.1 voice_colour_table . . . . .	199
4.119 m_eng_profile.c File Reference . . . . .	199
4.119.1 Function Documentation . . . . .	200
4.119.1.1 init_profile() . . . . .	200
4.119.1.2 nullify_profile() . . . . .	200
4.119.1.3 profile_apply_pipeline_mod() . . . . .	200
4.119.1.4 profile_print_job_list() . . . . .	201
4.119.1.5 profile_print_ujob_list() . . . . .	201
4.119.1.6 profile_process() . . . . .	201
4.119.1.7 profile_regenerate_back_pipeline() . . . . .	201
4.119.1.8 profile_scheduled_maintenance() . . . . .	201
4.119.1.9 profile_trigger_pipeline_swap() . . . . .	201
4.119.1.10 profile_update() . . . . .	201
4.120 m_eng_sgtl5000.cpp File Reference . . . . .	202
4.120.1 Function Documentation . . . . .	202
4.120.1.1 calc_vol() . . . . .	202
4.120.1.2 sgtl5000_adc_high_pass_filter_disable() . . . . .	202
4.120.1.3 sgtl5000_adc_high_pass_filter_enable() . . . . .	202
4.120.1.4 sgtl5000_adc_high_pass_filter_freeze() . . . . .	203
4.120.1.5 sgtl5000_automate() . . . . .	203
4.120.1.6 sgtl5000_dap_audio_eq_band() . . . . .	203
4.120.1.7 sgtl5000_enable() . . . . .	203
4.120.1.8 sgtl5000_eq_select() . . . . .	203
4.120.1.9 sgtl5000_kill_automation() . . . . .	203
4.120.1.10 sgtl5000_line_in_level() . . . . .	203
4.120.1.11 sgtl5000_line_out_level() . . . . .	203
4.120.1.12 sgtl5000_modify_reg() . . . . .	203
4.120.1.13 sgtl5000_mute_headphone() . . . . .	204
4.120.1.14 sgtl5000_mute_line_out() . . . . .	204

4.120.1.15 sgtl5000_read_reg()	204
4.120.1.16 sgtl5000_set_address()	204
4.120.1.17 sgtl5000_start()	204
4.120.1.18 sgtl5000_unmute_headphone()	204
4.120.1.19 sgtl5000_unmute_line_out()	204
4.120.1.20 sgtl5000_volum_eng_integer()	204
4.120.1.21 sgtl5000_volume()	204
4.120.1.22 sgtl5000_write_reg()	205
4.121 m_eng_simple_distortion.c File Reference	205
4.121.1 Function Documentation	205
4.121.1.1 calc_simple_distortion()	205
4.121.1.2 init_simple_distortion_str()	205
4.121.1.3 reconfigure_simple_distortion()	205
4.122 m_eng_transformer.c File Reference	205
4.122.1 Macro Definition Documentation	206
4.122.1.1 MAX_BLOCK_DIVIDER	206
4.122.2 Function Documentation	206
4.122.2.1 clone_transformer()	206
4.122.2.2 free_transformer()	206
4.122.2.3 run_bypass()	206
4.122.2.4 run_transformer()	206
4.122.2.5 transformer_add_parameter()	207
4.122.2.6 transformer_add_setting()	207
4.122.2.7 transformer_get_parameter()	207
4.122.2.8 transformer_get_setting()	207
4.122.2.9 transformer_init_controls()	207
4.122.2.10 transformer_init_parameter_array()	207
4.122.2.11 transformer_init_setting_array()	207
4.123 m_eng_transformer_init.c File Reference	207
4.123.1 Function Documentation	208
4.123.1.1 init_3_band_eq()	208
4.123.1.2 init_amplifier()	208
4.123.1.3 init_band_pass_filter()	208
4.123.1.4 init_compressor()	208
4.123.1.5 init_delay()	208
4.123.1.6 init_dirty_octave()	209
4.123.1.7 init_distortion()	209
4.123.1.8 init_envelope()	209
4.123.1.9 init_flanger()	209
4.123.1.10 init_high_pass_filter()	209
4.123.1.11 init_low_end_compressor()	209
4.123.1.12 init_low_pass_filter()	209

4.123.1.13 init_noise_suppressor()	209
4.123.1.14 init_percussifier()	209
4.123.1.15 init_transformer()	210
4.123.1.16 init_warbler()	210
4.124 m_eng_transformer_template.c File Reference	210
4.125 m_eng_update.c File Reference	210
4.125.1 Function Documentation	210
4.125.1.1 m_eng_software_isr()	210
4.125.1.2 update_all()	210
4.125.1.3 update_setup()	210
4.125.1.4 update_stop()	210
4.126 m_eng_useful_functions.c File Reference	211
4.126.1 Macro Definition Documentation	211
4.126.1.1 DENORMAL_THRESHOLD	211
4.126.1.2 FLOAT_TO_INT16_MAX	211
4.126.1.3 MAX_INT	211
4.126.1.4 SCALE_FACTOR	211
4.126.2 Function Documentation	211
4.126.2.1 convert_block_float_to_int()	211
4.126.2.2 convert_block_int_to_float()	212
4.126.2.3 hard_clip()	212
4.126.2.4 identity_function()	212
4.126.2.5 normalised_arctan()	212
4.126.2.6 soft_fold()	212
4.126.2.7 trig_transition_function()	212
4.127 m_eng_warbler.c File Reference	212
4.127.1 Function Documentation	213
4.127.1.1 calc_warbler()	213
4.127.1.2 init_warbler_str()	213
4.127.1.3 reconfigure_warbler()	213
4.128 m_eng_waveshaper.c File Reference	213
4.128.1 Function Documentation	213
4.128.1.1 calc_waveshaper()	213
4.128.1.2 init_waveshaper_str()	213
4.129 m_alloc.c File Reference	214
4.129.1 Function Documentation	214
4.129.1.1 m_alloc()	214
4.129.1.2 m_free()	214
4.129.1.3 m_realloc()	214
4.129.1.4 m_strndup()	214
4.129.1.5 print_memory_report()	214
4.130 m_alloc.h File Reference	214

4.130.1 Function Documentation	215
4.130.1.1 m_alloc()	215
4.130.1.2 m_free()	215
4.130.1.3 m_int_lv_free()	215
4.130.1.4 m_int_lv_malloc()	215
4.130.1.5 m_realloc()	215
4.130.1.6 m_strndup()	215
4.130.1.7 print_memory_report()	215
4.131 m_alloc.h	216
4.132 m_comms.c File Reference	216
4.132.1 Function Documentation	216
4.132.1.1 crc_8()	216
4.132.1.2 create_m_message()	217
4.132.1.3 create_m_message_nodata()	217
4.132.1.4 create_m_response()	217
4.132.1.5 create_m_response_error()	217
4.132.1.6 create_m_response_nodata()	217
4.132.1.7 create_m_response_ok()	217
4.132.1.8 create_m_response_parameter_value()	217
4.132.1.9 create_m_response_profile_id()	217
4.132.1.10 create_m_response_transformer_id()	218
4.132.1.11 decode_m_message()	218
4.132.1.12 decode_m_response()	218
4.132.1.13 encode_m_message()	218
4.132.1.14 encode_m_response()	218
4.132.1.15 et_message_data_length()	218
4.132.1.16 et_message_default_retries()	218
4.132.1.17 m_message_code_to_string()	218
4.132.1.18 m_response_code_to_string()	219
4.132.1.19 te_message_data_length()	219
4.132.1.20 valid_m_message_type()	219
4.132.1.21 valid_m_response_type()	219
4.133 m_comms.h File Reference	219
4.133.1 Macro Definition Documentation	221
4.133.1.1 M_MESSAGE_APPEND_TRANSFORMER	221
4.133.1.2 M_MESSAGE_CRC_FAIL	221
4.133.1.3 M_MESSAGE_CREATE_PROFILE	221
4.133.1.4 M_MESSAGE_DELETE_PROFILE	221
4.133.1.5 M_MESSAGE_ENTER_TUNER_MODE	221
4.133.1.6 M_MESSAGE_EXIT_TUNER_MODE	221
4.133.1.7 M_MESSAGE_GET_N_PARAMETERS	221
4.133.1.8 M_MESSAGE_GET_N_PROFILES	221

4.133.1.9 M_MESSAGE_GET_N_SETTINGS . . . . .	222
4.133.1.10 M_MESSAGE_GET_N_TRANSFORMERS . . . . .	222
4.133.1.11 M_MESSAGE_GET_PARAM_VALUE . . . . .	222
4.133.1.12 M_MESSAGE_GET_SETTING_VALUE . . . . .	222
4.133.1.13 M_MESSAGE_GET_TRANSFORMER_ID . . . . .	222
4.133.1.14 M_MESSAGE_GET_TRANSFORMER_TYPE . . . . .	222
4.133.1.15 M_MESSAGE_HI . . . . .	222
4.133.1.16 M_MESSAGE_INVALID . . . . .	222
4.133.1.17 M_MESSAGE_MAX_DATA_LEN . . . . .	222
4.133.1.18 M_MESSAGE_MAX_TRANSFER_LEN . . . . .	222
4.133.1.19 M_MESSAGE_MOVE_TRANSFORMER . . . . .	223
4.133.1.20 M_MESSAGE_NO_MESSAGE . . . . .	223
4.133.1.21 M_MESSAGE_REBOOT . . . . .	223
4.133.1.22 M_MESSAGE_REMOVE_TRANSFORMER . . . . .	223
4.133.1.23 M_MESSAGE_REPEAT_MESSAGE . . . . .	223
4.133.1.24 M_MESSAGE_RESET . . . . .	223
4.133.1.25 M_MESSAGE_SET_PARAM_VALUE . . . . .	223
4.133.1.26 M_MESSAGE_SET_SETTING_VALUE . . . . .	223
4.133.1.27 M_MESSAGE_STRING_CONTINUE . . . . .	223
4.133.1.28 M_MESSAGE_STRING_CONTINUING . . . . .	223
4.133.1.29 M_MESSAGE_SWITCH_PROFILE . . . . .	224
4.133.1.30 M_MESSAGE_TYPE_MAX . . . . .	224
4.133.1.31 M_RESPONSE_BAD_MESSAGE . . . . .	224
4.133.1.32 M_RESPONSE_BAD_REQUEST . . . . .	224
4.133.1.33 M_RESPONSE_CRC_FAIL . . . . .	224
4.133.1.34 M_RESPONSE_DELETED_PROFILE . . . . .	224
4.133.1.35 M_RESPONSE_ERROR . . . . .	224
4.133.1.36 M_RESPONSE_HI . . . . .	224
4.133.1.37 M_RESPONSE_INVALID . . . . .	224
4.133.1.38 M_RESPONSE_MAX_DATA_LEN . . . . .	224
4.133.1.39 M_RESPONSE_MAX_TRANSFER_LEN . . . . .	225
4.133.1.40 M_RESPONSE_N_PARAMETERS . . . . .	225
4.133.1.41 M_RESPONSE_N_PROFILES . . . . .	225
4.133.1.42 M_RESPONSE_N_SETTINGS . . . . .	225
4.133.1.43 M_RESPONSE_N_TRANSFORMERS . . . . .	225
4.133.1.44 M_RESPONSE_NO_MESSAGE . . . . .	225
4.133.1.45 M_RESPONSE_OK . . . . .	225
4.133.1.46 M_RESPONSE_PARAM_VALUE . . . . .	225
4.133.1.47 M_RESPONSE_PROFILE_ID . . . . .	225
4.133.1.48 M_RESPONSE_REPEAT_MESSAGE . . . . .	225
4.133.1.49 M_RESPONSE_SETTING_VALUE . . . . .	226
4.133.1.50 M_RESPONSE_START_OVER . . . . .	226

4.133.1.51 M_RESPONSE_STRING_CONTINUING . . . . .	226
4.133.1.52 M_RESPONSE_SWITCHING_PROFILE . . . . .	226
4.133.1.53 M_RESPONSE_TRANSFORMER_ID . . . . .	226
4.133.1.54 M_RESPONSE_TRANSFORMER_TYPE . . . . .	226
4.133.1.55 M_RESPONSE_TRY_AGAIN . . . . .	226
4.133.1.56 M_RESPONSE_TYPE_MAX . . . . .	226
4.133.1.57 M_RESPONSE_WAIT . . . . .	226
4.133.1.58 MESSAGE_LEN_VARIABLE . . . . .	226
4.133.1.59 TEENSY_ADDR . . . . .	227
4.133.2 Function Documentation . . . . .	227
4.133.2.1 create_m_message() . . . . .	227
4.133.2.2 create_m_message_nodata() . . . . .	227
4.133.2.3 create_m_response() . . . . .	227
4.133.2.4 create_m_response_error() . . . . .	227
4.133.2.5 create_m_response_nodata() . . . . .	227
4.133.2.6 create_m_response_ok() . . . . .	227
4.133.2.7 create_m_response_parameter_value() . . . . .	227
4.133.2.8 create_m_response_profile_id() . . . . .	228
4.133.2.9 create_m_response_transformer_id() . . . . .	228
4.133.2.10 decode_m_message() . . . . .	228
4.133.2.11 decode_m_response() . . . . .	228
4.133.2.12 encode_m_message() . . . . .	228
4.133.2.13 encode_m_response() . . . . .	228
4.133.2.14 et_message_data_length() . . . . .	228
4.133.2.15 m_message_code_to_string() . . . . .	228
4.133.2.16 m_response_code_to_string() . . . . .	229
4.133.2.17 te_message_data_length() . . . . .	229
4.133.2.18 valid_m_message_type() . . . . .	229
4.133.2.19 valid_m_response_type() . . . . .	229
4.134 m_comms.h . . . . .	229
4.135 m_error_codes.c File Reference . . . . .	231
4.135.1 Function Documentation . . . . .	231
4.135.1.1 m_error_code_to_string() . . . . .	231
4.136 m_error_codes.h File Reference . . . . .	231
4.136.1 Macro Definition Documentation . . . . .	232
4.136.1.1 ERR_ALLOC_FAIL . . . . .	232
4.136.1.2 ERR_ARRAY_MALFORMED . . . . .	232
4.136.1.3 ERR_BAD_ARGS . . . . .	232
4.136.1.4 ERR_BAD_REQUEST . . . . .	232
4.136.1.5 ERR_BUSTED_MSG . . . . .	232
4.136.1.6 ERR_COMMS_FAIL . . . . .	232
4.136.1.7 ERR_FIXED_ARRAY_FULL . . . . .	233

4.136.1.8	ERR_FOPEN_FAIL	233
4.136.1.9	ERR_I2C_FAIL	233
4.136.1.10	ERR_INCONSISTENT_BACK_PIPELINE	233
4.136.1.11	ERR_INVALID_MESSAGE	233
4.136.1.12	ERR_INVALID_PARAMETER_ID	233
4.136.1.13	ERR_INVALID_PROFILE_ID	233
4.136.1.14	ERR_INVALID_SETTING_ID	233
4.136.1.15	ERR_INVALID_TRANSFORMER_ID	233
4.136.1.16	ERR_LOOP_DETECTED	233
4.136.1.17	ERR_MANGLED_FILE	234
4.136.1.18	ERR_MUTEX_UNAVAILABLE	234
4.136.1.19	ERR_NO_RESPONSE	234
4.136.1.20	ERR_NODE_PRIVATE	234
4.136.1.21	ERR_NULL_PTR	234
4.136.1.22	ERR_PIPELINE_BUSTED	234
4.136.1.23	ERR_PIPELINE_FULL	234
4.136.1.24	ERR_PIPELINE_NULL	234
4.136.1.25	ERR_POSITION_ILLEGAL	234
4.136.1.26	ERR_POSITION_OCCUPIED	234
4.136.1.27	ERR_POT_LINK_MALFORMED	235
4.136.1.28	ERR_QUEUE_FULL	235
4.136.1.29	ERR_QUEUE_SEND_FAILED	235
4.136.1.30	ERR_SD_INIT_FAIL	235
4.136.1.31	ERR_SD_MOUNT_FAIL	235
4.136.1.32	ERR_SGTL5000_WRITE_FAIL	235
4.136.1.33	ERR_SPI_INIT_FAIL	235
4.136.1.34	ERR_SWITCH_LINK_MALFORMED	235
4.136.1.35	ERR_TRANSFORMER_MALFORMED	235
4.136.1.36	ERR_UNFINISHED_WRITE	235
4.136.1.37	ERR_UNIMPLEMENTED	236
4.136.1.38	ERR_UNKNOWN_ERR	236
4.136.1.39	ERR_VALUE_OUT_OF_BOUNDS	236
4.136.1.40	NO_ERROR	236
4.136.2	Function Documentation	236
4.136.2.1	m_error_code_to_string()	236
4.137	m_error_codes.h	236
4.138	m_linked_list.h File Reference	237
4.138.1	Macro Definition Documentation	237
4.138.1.1	DECLARE_LINKED_LIST	237
4.138.1.2	DECLARE_LINKED_PTR_LIST	238
4.138.1.3	IMPLEMENT_LINKED_LIST	238
4.138.1.4	IMPLEMENT_LINKED_PTR_LIST	238

4.138.1.5 LL_FREE	238
4.138.1.6 LL_MALLOC	239
4.139 m_linked_list.h	239
4.140 m_parameter.h File Reference	247
4.140.1 Macro Definition Documentation	248
4.140.1.1 PARAMETER_SCALE_LINEAR	248
4.140.1.2 PARAMETER_SCALE_LOGARITHMIC	248
4.140.1.3 TRANSFORMER_SETTING_BOOL	248
4.140.1.4 TRANSFORMER_SETTING_ENUM	248
4.140.1.5 TRANSFORMER_SETTING_INT	248
4.140.1.6 TRANSFORMER_SETTING_PAGE_MAIN	248
4.140.1.7 TRANSFORMER_SETTING_PAGE_SETTINGS	248
4.140.2 Function Documentation	248
4.140.2.1 DECLARE_LINKED_PTR_LIST() [1/2]	248
4.140.2.2 DECLARE_LINKED_PTR_LIST() [2/2]	248
4.141 m_parameter.h	249
4.142 m_pipeline.h File Reference	250
4.143 m_pipeline.h	250
4.144 m_profile.h File Reference	250
4.145 m_profile.h	251
4.146 m_status.h File Reference	251
4.146.1 Macro Definition Documentation	251
4.146.1.1 M_STATUS_BOOTING	251
4.146.1.2 M_STATUS_FRESH_BOOT	252
4.146.1.3 M_STATUS_OK	252
4.147 m_status.h	252
4.148 m_transformer.h File Reference	252
4.148.1 Function Documentation	252
4.148.1.1 DECLARE_LINKED_PTR_LIST()	252
4.149 m_transformer.h	253
4.150 m_transformer_enum.c File Reference	253
4.150.1 Function Documentation	254
4.150.1.1 transformer_type_to_string()	254
4.150.1.2 transformer_type_valid()	254
4.151 m_transformer_enum.h File Reference	254
4.151.1 Macro Definition Documentation	255
4.151.1.1 DISTORTION_ARCTAN	255
4.151.1.2 DISTORTION_CLIP	255
4.151.1.3 DISTORTION_SOFT_FOLD	255
4.151.1.4 DISTORTION_TANH	255
4.151.1.5 TRANSFORMER_3_BAND_EQ	255
4.151.1.6 TRANSFORMER_AMPLIFIER	255



4.151.1.7 TRANSFORMER_BAND_HP_CUTOFF_PID . . . . .	255
4.151.1.8 TRANSFORMER_BAND_LP_CUTOFF_PID . . . . .	256
4.151.1.9 TRANSFORMER_BAND_MODE_SID . . . . .	256
4.151.1.10 TRANSFORMER_BAND_PASS_FILTER . . . . .	256
4.151.1.11 TRANSFORMER_COMPRESSOR . . . . .	256
4.151.1.12 TRANSFORMER_DELAY . . . . .	256
4.151.1.13 TRANSFORMER_DIRTY_OCTAVE . . . . .	256
4.151.1.14 TRANSFORMER_DISTORTION . . . . .	256
4.151.1.15 TRANSFORMER_ENVELOPE . . . . .	256
4.151.1.16 TRANSFORMER_FLANGER . . . . .	256
4.151.1.17 TRANSFORMER_HIGH_PASS_FILTER . . . . .	256
4.151.1.18 TRANSFORMER_LOW_END_COMPRESSOR . . . . .	257
4.151.1.19 TRANSFORMER_LOW_PASS_FILTER . . . . .	257
4.151.1.20 TRANSFORMER_MODE_BAND . . . . .	257
4.151.1.21 TRANSFORMER_MODE_FULL_SPECTRUM . . . . .	257
4.151.1.22 TRANSFORMER_MODE_LOWER_SPECTRUM . . . . .	257
4.151.1.23 TRANSFORMER_MODE_UPPER_SPECTRUM . . . . .	257
4.151.1.24 TRANSFORMER_NOISE_SUPPRESSOR . . . . .	257
4.151.1.25 TRANSFORMER_PERCUSSIFIER . . . . .	257
4.151.1.26 TRANSFORMER_WARBLER . . . . .	257
4.151.1.27 TRANSFORMER_WET_MIX_PID . . . . .	257
4.151.2 Enumeration Type Documentation . . . . .	257
4.151.2.1 biquad_type . . . . .	257
4.151.3 Function Documentation . . . . .	258
4.151.3.1 transformer_type_to_string() . . . . .	258
4.151.3.2 transformer_type_valid() . . . . .	258
4.152 m_transformer_enum.h . . . . .	258



# Chapter 1

## Data Structure Index

### 1.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">m_delay_buffer</a>	7
<a href="#">m_eng_3_band_eq_str</a>	8
<a href="#">m_eng_3_band_splitter_str</a>	9
<a href="#">m_eng_adaptive_waveshaper_str</a>	9
<a href="#">m_eng_amplifier_str</a>	10
<a href="#">m_eng_band_pass_filter_str</a>	11
<a href="#">m_eng_biquad_str</a>	12
<a href="#">m_eng_compressor_str</a>	14
<a href="#">m_eng_context</a>	15
<a href="#">m_eng_delay_str</a>	18
<a href="#">m_eng_dirty_octave_str</a>	19
<a href="#">m_eng_distortion_str</a>	20
<a href="#">m_eng_envelope_str</a>	21
<a href="#">m_eng_flanger_str</a>	22
<a href="#">m_eng_high_pass_filter_str</a>	24
<a href="#">m_eng_log_entry</a>	25
<a href="#">m_eng_low_end_compressor_str</a>	27
<a href="#">m_eng_low_pass_filter_str</a>	28
<a href="#">m_eng_mixer_str</a>	29
<a href="#">m_eng_n_band_splitter_str</a>	29
<a href="#">m_eng_noise_suppressor_str</a>	30
<a href="#">m_eng_percussifier_str</a>	31
<a href="#">m_eng_profile</a>	33
<a href="#">m_eng_profiler_entry</a>	35
<a href="#">m_eng_simple_distortion_str</a>	36
<a href="#">m_eng_warbler_str</a>	37
<a href="#">m_eng_waveshaper_str</a>	39
<a href="#">m_lr_high_pass_filter_str</a>	39
<a href="#">m_lr_low_pass_filter_str</a>	41
<a href="#">m_message</a>	43
<a href="#">m_parameter</a>	44
<a href="#">m_parameter_id</a>	45
<a href="#">m_pipeline</a>	46
<a href="#">m_pipeline_mod</a>	46
<a href="#">m_profile</a>	47

<a href="#">m_response</a> . . . . .	<a href="#">47</a>
<a href="#">m_setting</a> . . . . .	<a href="#">48</a>
<a href="#">m_setting_id</a> . . . . .	<a href="#">48</a>
<a href="#">m_transformer</a> . . . . .	<a href="#">49</a>
<a href="#">m_transformer_str</a> . . . . .	<a href="#">50</a>

# Chapter 2

## File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

<a href="#">m_eng.h</a>	51
<a href="#">m_eng_adaptive_waveshaper.h</a>	58
<a href="#">m_eng_audio_block.h</a>	59
<a href="#">m_eng_band_splitter.h</a>	60
<a href="#">m_eng_biquad.h</a>	61
<a href="#">m_eng_buffer_mixer_amp.h</a>	62
<a href="#">m_eng_comms.h</a>	64
<a href="#">m_eng_compressor.h</a>	65
<a href="#">m_eng_context.h</a>	66
<a href="#">m_eng_debugging.h</a>	73
<a href="#">m_eng_delay.h</a>	75
<a href="#">m_eng_delay_buffer.h</a>	76
<a href="#">m_eng_dirty_octave.h</a>	78
<a href="#">m_eng_distortion.h</a>	79
<a href="#">m_eng_envelope.h</a>	80
<a href="#">m_eng_equaliser.h</a>	82
<a href="#">m_eng_flanger.h</a>	83
<a href="#">m_eng_flops.h</a>	84
<a href="#">m_eng_globals.h</a>	87
<a href="#">m_eng_i2s_dma.h</a>	89
<a href="#">m_eng_linkowitz_riley.h</a>	91
<a href="#">m_eng_logging.h</a>	93
<a href="#">m_eng_low_end_compressor.h</a>	102
<a href="#">m_eng_memcpy_audio.h</a>	103
<a href="#">m_eng_mempool.h</a>	105
<a href="#">m_eng_noise_suppressor.h</a>	107
<a href="#">m_eng_parameter.h</a>	108
<a href="#">m_eng_pass_filter.h</a>	109
<a href="#">m_eng_percussifier.h</a>	112
<a href="#">m_eng_pipeline.h</a>	114
<a href="#">m_eng_pipeline_mod.h</a>	118
<a href="#">m_eng_printf.h</a>	121
<a href="#">m_eng_profile.h</a>	122
<a href="#">m_eng_sgtl5000.h</a>	124
<a href="#">m_eng_sgtl5000_defs.h</a>	128

<a href="#">m_eng_simple_distortion.h</a>	142
<a href="#">m_eng_transformer.h</a>	143
<a href="#">m_eng_transformer_init.h</a>	147
<a href="#">m_eng_transformer_template.h</a>	148
<a href="#">m_eng_transition.h</a>	149
<a href="#">m_eng_update.h</a>	150
<a href="#">m_eng_useful_functions.h</a>	152
<a href="#">m_eng_warbler.h</a>	153
<a href="#">m_eng_waveshaper.h</a>	154
<a href="#">m_eng.cpp</a>	155
<a href="#">m_eng_adaptive_waveshaper.c</a>	157
<a href="#">m_eng_audio_block.c</a>	158
<a href="#">m_eng_band_splitter.c</a>	158
<a href="#">m_eng_biquad.c</a>	158
<a href="#">m_eng_buffer_mixer_amp.c</a>	158
<a href="#">m_eng_comms.cpp</a>	159
<a href="#">m_eng_compressor.c</a>	163
<a href="#">m_eng_context.c</a>	164
<a href="#">m_eng_debugging.c</a>	169
<a href="#">m_eng_delay.c</a>	170
<a href="#">m_eng_delay_buffer.c</a>	171
<a href="#">m_eng_dirty_octave.c</a>	172
<a href="#">m_eng_distortion.c</a>	173
<a href="#">m_eng_envelope.c</a>	174
<a href="#">m_eng_equaliser.c</a>	174
<a href="#">m_eng_flanger.c</a>	175
<a href="#">m_eng_globals.c</a>	176
<a href="#">m_eng_i2s_dma.cpp</a>	177
<a href="#">m_eng_linkowitz_riley.c</a>	181
<a href="#">m_eng_logging.cpp</a>	182
<a href="#">m_eng_low_end_compressor.c</a>	186
<a href="#">m_eng_mempool.c</a>	186
<a href="#">m_eng_noise_suppressor.c</a>	189
<a href="#">m_eng_parameter.c</a>	189
<a href="#">m_eng_pass_filter.c</a>	190
<a href="#">m_eng_percussifier.c</a>	192
<a href="#">m_eng_pipeline.c</a>	192
<a href="#">m_eng_pipeline_mod.c</a>	196
<a href="#">m_eng_printf.cpp</a>	197
<a href="#">m_eng_profile.c</a>	199
<a href="#">m_eng_sgtl5000.cpp</a>	202
<a href="#">m_eng_simple_distortion.c</a>	205
<a href="#">m_eng_transformer.c</a>	205
<a href="#">m_eng_transformer_init.c</a>	207
<a href="#">m_eng_transformer_template.c</a>	210
<a href="#">m_eng_update.c</a>	210
<a href="#">m_eng_useful_functions.c</a>	211
<a href="#">m_eng_warbler.c</a>	212
<a href="#">m_eng_waveshaper.c</a>	213
<a href="#">m_alloc.c</a>	214
<a href="#">m_alloc.h</a>	214
<a href="#">m_comms.c</a>	216
<a href="#">m_comms.h</a>	219
<a href="#">m_error_codes.c</a>	231
<a href="#">m_error_codes.h</a>	231
<a href="#">m_linked_list.h</a>	237
<a href="#">m_parameter.h</a>	247
<a href="#">m_pipeline.h</a>	250

<a href="#">m_profile.h</a>	250
<a href="#">m_status.h</a>	251
<a href="#">m_transformer.h</a>	252
<a href="#">m_transformer_enum.c</a>	253
<a href="#">m_transformer_enum.h</a>	254





## Chapter 3

# Data Structure Documentation

### 3.1 m\_delay\_buffer Struct Reference

```
#include <m_eng_delay_buffer.h>
```

#### Data Fields

- int [valid](#)
- int [pos](#)
- int [index](#)
- int [n\\_buffers](#)
- int [buffer\\_array\\_size](#)
- float \*\* [buffers](#)

#### 3.1.1 Field Documentation

##### 3.1.1.1 buffer\_array\_size

```
int m_delay_buffer::buffer_array_size
```

##### 3.1.1.2 buffers

```
float** m_delay_buffer::buffers
```

##### 3.1.1.3 index

```
int m_delay_buffer::index
```

##### 3.1.1.4 n\_buffers

```
int m_delay_buffer::n_buffers
```

### 3.1.1.5 pos

```
int m_delay_buffer::pos
```

### 3.1.1.6 valid

```
int m_delay_buffer::valid
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_delay\\_buffer.h](#)

## 3.2 m\_eng\_3\_band\_eq\_str Struct Reference

```
#include <m_eng_equaliser.h>
```

### Data Fields

- [m\\_parameter low](#)
- [m\\_parameter mid](#)
- [m\\_parameter high](#)
- float [coefs](#) [3]
- int [control\\_mode](#)
- [m\\_lr\\_low\\_pass\\_filter\\_str filters](#) [2]

### 3.2.1 Field Documentation

#### 3.2.1.1 coefs

```
float m_eng_3_band_eq_str::coefs[3]
```

#### 3.2.1.2 control\_mode

```
int m_eng_3_band_eq_str::control_mode
```

#### 3.2.1.3 filters

```
m\_lr\_low\_pass\_filter\_str m_eng_3_band_eq_str::filters[2]
```

#### 3.2.1.4 high

```
m\_parameter m_eng_3_band_eq_str::high
```

#### 3.2.1.5 low

`m_parameter m_eng_3_band_eq_str::low`

#### 3.2.1.6 mid

`m_parameter m_eng_3_band_eq_str::mid`

The documentation for this struct was generated from the following file:

- [m\\_eng\\_equaliser.h](#)

### 3.3 m\_eng\_3\_band\_splitter\_str Struct Reference

```
#include <m_eng_band_splitter.h>
```

#### Data Fields

- [m\\_lr\\_low\\_pass\\_filter\\_str](#) filters [3]

#### 3.3.1 Field Documentation

##### 3.3.1.1 filters

`m_lr_low_pass_filter_str m_eng_3_band_splitter_str::filters[3]`

The documentation for this struct was generated from the following file:

- [m\\_eng\\_band\\_splitter.h](#)

### 3.4 m\_eng\_adaptive\_waveshaper\_str Struct Reference

```
#include <m_eng_adaptive_waveshaper.h>
```

#### Data Fields

- [m\\_parameter](#) coefficient
- `float(* shape)(float x)`
- `float local\_amplitude`

### 3.4.1 Field Documentation

#### 3.4.1.1 coefficient

`m_parameter` `m_eng_adaptive_waveshaper_str::coefficient`

#### 3.4.1.2 local\_amplitude

`float` `m_eng_adaptive_waveshaper_str::local_amplitude`

#### 3.4.1.3 shape

`float(* m_eng_adaptive_waveshaper_str::shape) (float x)`

The documentation for this struct was generated from the following file:

- [m\\_eng\\_adaptive\\_waveshaper.h](#)

## 3.5 m\_eng\_amplifier\_str Struct Reference

```
#include <m_eng_buffer_mixer_amp.h>
```

### Data Fields

- [m\\_parameter](#) gain
- `float` `g`
- [m\\_setting](#) mode
- `int` `db`

### 3.5.1 Field Documentation

#### 3.5.1.1 db

`int` `m_eng_amplifier_str::db`

#### 3.5.1.2 g

`float` `m_eng_amplifier_str::g`

#### 3.5.1.3 gain

`m_parameter` `m_eng_amplifier_str::gain`

#### 3.5.1.4 mode

`m_setting m_eng_amplifier_str::mode`

The documentation for this struct was generated from the following file:

- `m_eng_buffer_mixer_amp.h`

## 3.6 m\_eng\_band\_pass\_filter\_str Struct Reference

```
#include <m_eng_pass_filter.h>
```

### Data Fields

- `m_parameter center`
- `m_parameter bandwidth`
- float `a0`
- float `a1`
- float `a2`
- float `a3`
- float `a4`
- float `x1`
- float `x2`
- float `y1`
- float `y2`

### 3.6.1 Field Documentation

#### 3.6.1.1 a0

```
float m_eng_band_pass_filter_str::a0
```

#### 3.6.1.2 a1

```
float m_eng_band_pass_filter_str::a1
```

#### 3.6.1.3 a2

```
float m_eng_band_pass_filter_str::a2
```

#### 3.6.1.4 a3

```
float m_eng_band_pass_filter_str::a3
```

#### 3.6.1.5 a4

```
float m_eng_band_pass_filter_str::a4
```

#### 3.6.1.6 bandwidth

```
m_parameter m_eng_band_pass_filter_str::bandwidth
```

#### 3.6.1.7 center

```
m_parameter m_eng_band_pass_filter_str::center
```

#### 3.6.1.8 x1

```
float m_eng_band_pass_filter_str::x1
```

#### 3.6.1.9 x2

```
float m_eng_band_pass_filter_str::x2
```

#### 3.6.1.10 y1

```
float m_eng_band_pass_filter_str::y1
```

#### 3.6.1.11 y2

```
float m_eng_band_pass_filter_str::y2
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_pass\\_filter.h](#)

## 3.7 m\_eng\_biquad\_str Struct Reference

```
#include <m_eng_biquad.h>
```

## Data Fields

- float [a0](#)
- float [a1](#)
- float [a2](#)
- float [a3](#)
- float [a4](#)
- float [x1](#)
- float [x2](#)
- float [y1](#)
- float [y2](#)
- [m\\_setting](#) type
- [m\\_parameter](#) cutoff
- [m\\_parameter](#) bandwidth
- [m\\_parameter](#) db\_gain

## 3.7.1 Field Documentation

### 3.7.1.1 a0

`float m_eng_biquad_str::a0`

### 3.7.1.2 a1

`float m_eng_biquad_str::a1`

### 3.7.1.3 a2

`float m_eng_biquad_str::a2`

### 3.7.1.4 a3

`float m_eng_biquad_str::a3`

### 3.7.1.5 a4

`float m_eng_biquad_str::a4`

### 3.7.1.6 bandwidth

[m\\_parameter](#) `m_eng_biquad_str::bandwidth`

### 3.7.1.7 cutoff

[m\\_parameter](#) `m_eng_biquad_str::cutoff`

### 3.7.1.8 db\_gain

[m\\_parameter](#) m\_eng\_biquad\_str::db\_gain

### 3.7.1.9 type

[m\\_setting](#) m\_eng\_biquad\_str::type

### 3.7.1.10 x1

float m\_eng\_biquad\_str::x1

### 3.7.1.11 x2

float m\_eng\_biquad\_str::x2

### 3.7.1.12 y1

float m\_eng\_biquad\_str::y1

### 3.7.1.13 y2

float m\_eng\_biquad\_str::y2

The documentation for this struct was generated from the following file:

- [m\\_eng\\_biquad.h](#)

## 3.8 m\_eng\_compressor\_str Struct Reference

```
#include <m_eng_compressor.h>
```

### Data Fields

- [m\\_parameter](#) ratio
- [m\\_parameter](#) threshold
- [m\\_parameter](#) attack
- [m\\_parameter](#) release
- float [alpha](#)
- float [rho](#)
- float [e\\_final](#)



### 3.8.1 Field Documentation

#### 3.8.1.1 alpha

```
float m_eng_compressor_str::alpha
```

#### 3.8.1.2 attack

```
m_parameter m_eng_compressor_str::attack
```

#### 3.8.1.3 e\_final

```
float m_eng_compressor_str::e_final
```

#### 3.8.1.4 ratio

```
m_parameter m_eng_compressor_str::ratio
```

#### 3.8.1.5 release

```
m_parameter m_eng_compressor_str::release
```

#### 3.8.1.6 rho

```
float m_eng_compressor_str::rho
```

#### 3.8.1.7 threshold

```
m_parameter m_eng_compressor_str::threshold
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_compressor.h](#)

## 3.9 m\_eng\_context Struct Reference

```
#include <m_eng_context.h>
```

## Data Fields

- `uint16_t status_flags`
- `int n_profiles`
- `int profile_array_size`
- `m_eng_profile * profiles`
- `int active_profile`
- `int profile_switch_triggered`
- `int new_profile`
- `int profiles_switching`
- `int profile_switch_progress`
- `int profile_switch_samples`
- `int profile_switch_type`
- `float declick_buffer [DECLICK_BUFSIZE]`
- `int profile_maintenance_index`
- `float prev_block [AUDIO_BLOCK_SAMPLES]`
- `int runs`
- `m_eng_high_pass_filter_str output_hpf`
- `m_eng_low_pass_filter_str input_lpf`
- `m_transformer output_amp`
- `float dc_blocker_avg`

### 3.9.1 Field Documentation

#### 3.9.1.1 active\_profile

```
int m_eng_context::active_profile
```

#### 3.9.1.2 dc\_blocker\_avg

```
float m_eng_context::dc_blocker_avg
```

#### 3.9.1.3 declick\_buffer

```
float m_eng_context::declick_buffer[DECLICK_BUFSIZE]
```

#### 3.9.1.4 input\_lpf

```
m_eng_low_pass_filter_str m_eng_context::input_lpf
```

#### 3.9.1.5 n\_profiles

```
int m_eng_context::n_profiles
```

#### 3.9.1.6 new\_profile

```
int m_eng_context::new_profile
```

### 3.9.1.7 output\_amp

`m_transformer` m\_eng\_context::output\_amp

### 3.9.1.8 output\_hpf

`m_eng_high_pass_filter_str` m\_eng\_context::output\_hpf

### 3.9.1.9 prev\_block

`float` m\_eng\_context::prev\_block[AUDIO\_BLOCK\_SAMPLES]

### 3.9.1.10 profile\_array\_size

`int` m\_eng\_context::profile\_array\_size

### 3.9.1.11 profile\_maintainance\_index

`int` m\_eng\_context::profile\_maintainance\_index

### 3.9.1.12 profile\_switch\_progress

`int` m\_eng\_context::profile\_switch\_progress

### 3.9.1.13 profile\_switch\_samples

`int` m\_eng\_context::profile\_switch\_samples

### 3.9.1.14 profile\_switch\_triggered

`int` m\_eng\_context::profile\_switch\_triggered

### 3.9.1.15 profile\_switch\_type

`int` m\_eng\_context::profile\_switch\_type

### 3.9.1.16 profiles

`m_eng_profile*` m\_eng\_context::profiles

### 3.9.1.17 profiles\_switching

```
int m_eng_context::profiles_switching
```

### 3.9.1.18 runs

```
int m_eng_context::runs
```

### 3.9.1.19 status\_flags

```
uint16_t m_eng_context::status_flags
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_context.h](#)

## 3.10 m\_eng\_delay\_str Struct Reference

```
#include <m_eng_delay.h>
```

### Data Fields

- [m\\_setting tempo](#)
- [m\\_setting note](#)
- [m\\_parameter delay\\_gain](#)
- float [delay\\_samples](#)
- float [g](#)
- [m\\_delay\\_buffer](#) buf

### 3.10.1 Field Documentation

#### 3.10.1.1 buf

```
m_delay_buffer m_eng_delay_str::buf
```

#### 3.10.1.2 delay\_gain

```
m_parameter m_eng_delay_str::delay_gain
```

#### 3.10.1.3 delay\_samples

```
float m_eng_delay_str::delay_samples
```

#### 3.10.1.4 g

```
float m_eng_delay_str::g
```

#### 3.10.1.5 note

```
m_setting m_eng_delay_str::note
```

#### 3.10.1.6 tempo

```
m_setting m_eng_delay_str::tempo
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_delay.h](#)

## 3.11 m\_eng\_dirty\_octave\_str Struct Reference

```
#include <m_eng_dirty_octave.h>
```

### Data Fields

- [m\\_parameter fuzz](#)
- float [dc\\_average](#)
- float [lpf\\_alpha](#)
- float [last\\_out\\_sample](#)

### 3.11.1 Field Documentation

#### 3.11.1.1 dc\_average

```
float m_eng_dirty_octave_str::dc_average
```

#### 3.11.1.2 fuzz

```
m_parameter m_eng_dirty_octave_str::fuzz
```

#### 3.11.1.3 last\_out\_sample

```
float m_eng_dirty_octave_str::last_out_sample
```

#### 3.11.1.4 lpf\_alpha

```
float m_eng_dirty_octave_str::lpf_alpha
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_dirty\\_octave.h](#)

## 3.12 m\_eng\_distortion\_str Struct Reference

```
#include <m_eng_distortion.h>
```

### Data Fields

- [m\\_setting](#) function
- [m\\_lr\\_low\\_pass\\_filter\\_str](#) low\_pass
- [m\\_eng\\_waveshaper\\_str](#) dist
- [m\\_parameter](#) wet\_mix
- [m\\_parameter](#) bass\_mix
- [m\\_parameter](#) bass\_cutoff

### 3.12.1 Field Documentation

#### 3.12.1.1 bass\_cutoff

```
m_parameter m_eng_distortion_str::bass_cutoff
```

#### 3.12.1.2 bass\_mix

```
m_parameter m_eng_distortion_str::bass_mix
```

#### 3.12.1.3 dist

```
m_eng_waveshaper_str m_eng_distortion_str::dist
```

#### 3.12.1.4 function

```
m_setting m_eng_distortion_str::function
```

#### 3.12.1.5 low\_pass

```
m_lr_low_pass_filter_str m_eng_distortion_str::low_pass
```

### 3.12.1.6 wet\_mix

`m_parameter m_eng_distortion_str::wet_mix`

The documentation for this struct was generated from the following file:

- [m\\_eng\\_distortion.h](#)

## 3.13 m\_eng\_envelope\_str Struct Reference

```
#include <m_eng_envelope.h>
```

### Data Fields

- [m\\_parameter min\\_center](#)
- [m\\_parameter max\\_center](#)
- [m\\_parameter width](#)
- [m\\_parameter speed](#)
- [m\\_parameter sensitivity](#)
- [m\\_parameter smoothness](#)
- float [alpha](#)
- float [e](#)
- int [chunk\\_size](#)
- [m\\_eng\\_band\\_pass\\_filter\\_str](#) [filter](#)

### 3.13.1 Field Documentation

#### 3.13.1.1 alpha

`float m_eng_envelope_str::alpha`

#### 3.13.1.2 chunk\_size

`int m_eng_envelope_str::chunk_size`

#### 3.13.1.3 e

`float m_eng_envelope_str::e`

#### 3.13.1.4 filter

`m_eng_band_pass_filter_str m_eng_envelope_str::filter`

#### 3.13.1.5 max\_center

[m\\_parameter](#) [m\\_eng\\_envelope\\_str::max\\_center](#)

#### 3.13.1.6 min\_center

[m\\_parameter](#) [m\\_eng\\_envelope\\_str::min\\_center](#)

#### 3.13.1.7 sensitivity

[m\\_parameter](#) [m\\_eng\\_envelope\\_str::sensitivity](#)

#### 3.13.1.8 smoothness

[m\\_parameter](#) [m\\_eng\\_envelope\\_str::smoothness](#)

#### 3.13.1.9 speed

[m\\_parameter](#) [m\\_eng\\_envelope\\_str::speed](#)

#### 3.13.1.10 width

[m\\_parameter](#) [m\\_eng\\_envelope\\_str::width](#)

The documentation for this struct was generated from the following file:

- [m\\_eng\\_envelope.h](#)

## 3.14 m\_eng\_flanger\_str Struct Reference

```
#include <m_eng_flanger.h>
```

### Data Fields

- [m\\_parameter](#) range
- [m\\_parameter](#) tempo
- [m\\_parameter](#) depth
- [m\\_parameter](#) mix
- [m\\_setting](#) note
- float [r](#)
- float [s](#)
- float [d](#)
- float [wet\\_mix](#)
- float [dry\\_mix](#)
- float [period](#)
- float [t](#)
- [m\\_delay\\_buffer](#) buf



### 3.14.1 Field Documentation

#### 3.14.1.1 buf

`m_delay_buffer` m\_eng\_flanger\_str::buf

#### 3.14.1.2 d

`float` m\_eng\_flanger\_str::d

#### 3.14.1.3 depth

`m_parameter` m\_eng\_flanger\_str::depth

#### 3.14.1.4 dry\_mix

`float` m\_eng\_flanger\_str::dry\_mix

#### 3.14.1.5 mix

`m_parameter` m\_eng\_flanger\_str::mix

#### 3.14.1.6 note

`m_setting` m\_eng\_flanger\_str::note

#### 3.14.1.7 period

`float` m\_eng\_flanger\_str::period

#### 3.14.1.8 r

`float` m\_eng\_flanger\_str::r

#### 3.14.1.9 range

`m_parameter` m\_eng\_flanger\_str::range

#### 3.14.1.10 s

`float` m\_eng\_flanger\_str::s

#### 3.14.1.11 t

```
float m_eng_flanger_str::t
```

#### 3.14.1.12 tempo

```
m_parameter m_eng_flanger_str::tempo
```

#### 3.14.1.13 wet\_mix

```
float m_eng_flanger_str::wet_mix
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_flanger.h](#)

### 3.15 m\_eng\_high\_pass\_filter\_str Struct Reference

```
#include <m_eng_pass_filter.h>
```

#### Data Fields

- [m\\_parameter cutoff\\_frequency](#)
- float [a0](#)
- float [a1](#)
- float [a2](#)
- float [a3](#)
- float [a4](#)
- float [x1](#)
- float [x2](#)
- float [y1](#)
- float [y2](#)

#### 3.15.1 Field Documentation

##### 3.15.1.1 a0

```
float m_eng_high_pass_filter_str::a0
```

##### 3.15.1.2 a1

```
float m_eng_high_pass_filter_str::a1
```

### 3.15.1.3 a2

```
float m_eng_high_pass_filter_str::a2
```

### 3.15.1.4 a3

```
float m_eng_high_pass_filter_str::a3
```

### 3.15.1.5 a4

```
float m_eng_high_pass_filter_str::a4
```

### 3.15.1.6 cutoff\_frequency

```
m_parameter m_eng_high_pass_filter_str::cutoff_frequency
```

### 3.15.1.7 x1

```
float m_eng_high_pass_filter_str::x1
```

### 3.15.1.8 x2

```
float m_eng_high_pass_filter_str::x2
```

### 3.15.1.9 y1

```
float m_eng_high_pass_filter_str::y1
```

### 3.15.1.10 y2

```
float m_eng_high_pass_filter_str::y2
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_pass\\_filter.h](#)

## 3.16 m\_eng\_log\_entry Struct Reference

```
#include <m_eng_logging.h>
```

## Data Fields

- int [type](#)
- const char \* [file\\_name](#)
- const char \* [line](#)
- const char \* [data\\_type](#)
- const char \* [function](#)
- const char \* [message](#)
- uint64\_t [cycle](#)
- uint32\_t [data](#)
- uint32\_t [trace\\_depth](#)

### 3.16.1 Field Documentation

#### 3.16.1.1 cycle

```
uint64_t m_eng_log_entry::cycle
```

#### 3.16.1.2 data

```
uint32_t m_eng_log_entry::data
```

#### 3.16.1.3 data\_type

```
const char* m_eng_log_entry::data_type
```

#### 3.16.1.4 file\_name

```
const char* m_eng_log_entry::file_name
```

#### 3.16.1.5 function

```
const char* m_eng_log_entry::function
```

#### 3.16.1.6 line

```
const char* m_eng_log_entry::line
```

#### 3.16.1.7 message

```
const char* m_eng_log_entry::message
```

### 3.16.1.8 trace\_depth

```
uint32_t m_eng_log_entry::trace_depth
```

### 3.16.1.9 type

```
int m_eng_log_entry::type
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_logging.h](#)

## 3.17 m\_eng\_low\_end\_compressor\_str Struct Reference

```
#include <m_eng_low_end_compressor.h>
```

### Data Fields

- [m\\_eng\\_compressor\\_str bass\\_comp](#)
- [m\\_eng\\_compressor\\_str mids\\_comp](#)
- [m\\_lr\\_low\\_pass\\_filter\\_str low\\_pass](#)
- [m\\_lr\\_low\\_pass\\_filter\\_str mid\\_pass](#)

### 3.17.1 Field Documentation

#### 3.17.1.1 bass\_comp

```
m_eng_compressor_str m_eng_low_end_compressor_str::bass_comp
```

#### 3.17.1.2 low\_pass

```
m_lr_low_pass_filter_str m_eng_low_end_compressor_str::low_pass
```

#### 3.17.1.3 mid\_pass

```
m_lr_low_pass_filter_str m_eng_low_end_compressor_str::mid_pass
```

#### 3.17.1.4 mids\_comp

```
m_eng_compressor_str m_eng_low_end_compressor_str::mids_comp
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_low\\_end\\_compressor.h](#)

## 3.18 m\_eng\_low\_pass\_filter\_str Struct Reference

```
#include <m_eng_pass_filter.h>
```

### Data Fields

- [m\\_parameter](#) [cutoff\\_frequency](#)
- float [a0](#)
- float [a1](#)
- float [a2](#)
- float [a3](#)
- float [a4](#)
- float [x1](#)
- float [x2](#)
- float [y1](#)
- float [y2](#)

### 3.18.1 Field Documentation

#### 3.18.1.1 a0

```
float m_eng_low_pass_filter_str::a0
```

#### 3.18.1.2 a1

```
float m_eng_low_pass_filter_str::a1
```

#### 3.18.1.3 a2

```
float m_eng_low_pass_filter_str::a2
```

#### 3.18.1.4 a3

```
float m_eng_low_pass_filter_str::a3
```

#### 3.18.1.5 a4

```
float m_eng_low_pass_filter_str::a4
```

#### 3.18.1.6 cutoff\_frequency

```
m\_parameter m_eng_low_pass_filter_str::cutoff_frequency
```

### 3.18.1.7 x1

```
float m_eng_low_pass_filter_str::x1
```

### 3.18.1.8 x2

```
float m_eng_low_pass_filter_str::x2
```

### 3.18.1.9 y1

```
float m_eng_low_pass_filter_str::y1
```

### 3.18.1.10 y2

```
float m_eng_low_pass_filter_str::y2
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_pass\\_filter.h](#)

## 3.19 m\_eng\_mixer\_str Struct Reference

```
#include <m_eng_buffer_mixer_amp.h>
```

### Data Fields

- [m\\_parameter](#) ratio

### 3.19.1 Field Documentation

#### 3.19.1.1 ratio

```
m\_parameter m_eng_mixer_str::ratio
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_buffer\\_mixer\\_amp.h](#)

## 3.20 m\_eng\_n\_band\_splitter\_str Struct Reference

```
#include <m_eng_band_splitter.h>
```

## Data Fields

- [m\\_lr\\_low\\_pass\\_filter\\_str](#) \* [filters](#)

### 3.20.1 Field Documentation

#### 3.20.1.1 filters

[m\\_lr\\_low\\_pass\\_filter\\_str](#)\* [m\\_eng\\_n\\_band\\_splitter\\_str::filters](#)

The documentation for this struct was generated from the following file:

- [m\\_eng\\_band\\_splitter.h](#)

## 3.21 m\_eng\_noise\_suppressor\_str Struct Reference

```
#include <m_eng_noise_suppressor.h>
```

## Data Fields

- [m\\_parameter](#) threshold
- [m\\_parameter](#) ratio
- [m\\_parameter](#) max\_reduction
- float [e\\_final](#)
- int [r](#)

### 3.21.1 Field Documentation

#### 3.21.1.1 e\_final

float [m\\_eng\\_noise\\_suppressor\\_str::e\\_final](#)

#### 3.21.1.2 max\_reduction

[m\\_parameter](#) [m\\_eng\\_noise\\_suppressor\\_str::max\\_reduction](#)

#### 3.21.1.3 r

int [m\\_eng\\_noise\\_suppressor\\_str::r](#)

#### 3.21.1.4 ratio

[m\\_parameter](#) [m\\_eng\\_noise\\_suppressor\\_str::ratio](#)



### 3.21.1.5 threshold

`m_parameter m_eng_noise_suppressor_str::threshold`

The documentation for this struct was generated from the following file:

- [m\\_eng\\_noise\\_suppressor.h](#)

## 3.22 m\_eng\_percussifier\_str Struct Reference

```
#include <m_eng_percussifier.h>
```

### Data Fields

- [m\\_parameter tempo](#)
- [m\\_parameter note](#)
- [m\\_parameter trigger\\_threshold](#)
- [m\\_parameter arm\\_threshold](#)
- [m\\_parameter refractory\\_period](#)
- [m\\_parameter fade\\_in](#)
- [m\\_parameter fade\\_out](#)
- [int state](#)
- [int fade\\_in\\_samples](#)
- [int hold\\_samples](#)
- [int refractory\\_samples](#)
- [int timer](#)
- [float decay\\_rate](#)
- [float rms\\_short](#)
- [float rms\\_long](#)
- [float alpha\\_short](#)
- [float alpha\\_long](#)
- [float gain](#)
- [float fade\\_alpha](#)
- [int r](#)

### 3.22.1 Field Documentation

#### 3.22.1.1 alpha\_long

`float m_eng_percussifier_str::alpha_long`

#### 3.22.1.2 alpha\_short

`float m_eng_percussifier_str::alpha_short`

### 3.22.1.3 arm\_threshold

`m_parameter m_eng_percussifier_str::arm_threshold`

### 3.22.1.4 decay\_rate

`float m_eng_percussifier_str::decay_rate`

### 3.22.1.5 fade\_alpha

`float m_eng_percussifier_str::fade_alpha`

### 3.22.1.6 fade\_in

`m_parameter m_eng_percussifier_str::fade_in`

### 3.22.1.7 fade\_in\_samples

`int m_eng_percussifier_str::fade_in_samples`

### 3.22.1.8 fade\_out

`m_parameter m_eng_percussifier_str::fade_out`

### 3.22.1.9 gain

`float m_eng_percussifier_str::gain`

### 3.22.1.10 hold\_samples

`int m_eng_percussifier_str::hold_samples`

### 3.22.1.11 note

`m_parameter m_eng_percussifier_str::note`

### 3.22.1.12 r

`int m_eng_percussifier_str::r`

#### 3.22.1.13 refractory\_period

`m_parameter m_eng_percussifier_str::refractory_period`

#### 3.22.1.14 refractory\_samples

`int m_eng_percussifier_str::refractory_samples`

#### 3.22.1.15 rms\_long

`float m_eng_percussifier_str::rms_long`

#### 3.22.1.16 rms\_short

`float m_eng_percussifier_str::rms_short`

#### 3.22.1.17 state

`int m_eng_percussifier_str::state`

#### 3.22.1.18 tempo

`m_parameter m_eng_percussifier_str::tempo`

#### 3.22.1.19 timer

`int m_eng_percussifier_str::timer`

#### 3.22.1.20 trigger\_threshold

`m_parameter m_eng_percussifier_str::trigger_threshold`

The documentation for this struct was generated from the following file:

- [m\\_eng\\_percussifier.h](#)

### 3.23 m\_eng\_profile Struct Reference

```
#include <m_eng_profile.h>
```

## Data Fields

- [m\\_pipeline](#) \* [front\\_pipeline](#)
- [m\\_pipeline](#) \* [back\\_pipeline](#)
- int [pipelines\\_swapping](#)
- int [pipeline\\_swap\\_progress](#)
- int [pipeline\\_swap\\_samples](#)
- int [pipeline\\_swap\\_type](#)
- int [back\\_pipeline\\_warmed\\_up](#)
- [m\\_pipeline\\_mod\\_ll](#) \* [jobs](#)
- [m\\_pipeline\\_mod\\_ll](#) \* [blocked\\_jobs](#)
- int [active](#)
- int [transition\\_policy](#)
- float \* [prev\\_block](#)
- [m\\_transformer](#) [output\\_amp](#)
- int [runs](#)

### 3.23.1 Field Documentation

#### 3.23.1.1 active

```
int m_eng_profile::active
```

#### 3.23.1.2 back\_pipeline

```
m\_pipeline* m_eng_profile::back_pipeline
```

#### 3.23.1.3 back\_pipeline\_warmed\_up

```
int m_eng_profile::back_pipeline_warmed_up
```

#### 3.23.1.4 blocked\_jobs

```
m\_pipeline\_mod\_ll* m_eng_profile::blocked_jobs
```

#### 3.23.1.5 front\_pipeline

```
m\_pipeline* m_eng_profile::front_pipeline
```

#### 3.23.1.6 jobs

```
m\_pipeline\_mod\_ll* m_eng_profile::jobs
```

### 3.23.1.7 output\_amp

```
m_transformer m_eng_profile::output_amp
```

### 3.23.1.8 pipeline\_swap\_progress

```
int m_eng_profile::pipeline_swap_progress
```

### 3.23.1.9 pipeline\_swap\_samples

```
int m_eng_profile::pipeline_swap_samples
```

### 3.23.1.10 pipeline\_swap\_type

```
int m_eng_profile::pipeline_swap_type
```

### 3.23.1.11 pipelines\_swapping

```
int m_eng_profile::pipelines_swapping
```

### 3.23.1.12 prev\_block

```
float* m_eng_profile::prev_block
```

### 3.23.1.13 runs

```
int m_eng_profile::runs
```

### 3.23.1.14 transition\_policy

```
int m_eng_profile::transition_policy
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_profile.h](#)

## 3.24 m\_eng\_profiler\_entry Struct Reference

```
#include <m_eng_logging.h>
```

## Data Fields

- const char \* [function\\_name](#)
- uint64\_t [open\\_cycle](#)
- uint64\_t [calls](#)
- uint64\_t [total\\_cycles](#)
- uint64\_t [peak\\_cycles](#)
- double [ra\\_cycles](#)

### 3.24.1 Field Documentation

#### 3.24.1.1 calls

```
uint64_t m_eng_profiler_entry::calls
```

#### 3.24.1.2 function\_name

```
const char* m_eng_profiler_entry::function_name
```

#### 3.24.1.3 open\_cycle

```
uint64_t m_eng_profiler_entry::open_cycle
```

#### 3.24.1.4 peak\_cycles

```
uint64_t m_eng_profiler_entry::peak_cycles
```

#### 3.24.1.5 ra\_cycles

```
double m_eng_profiler_entry::ra_cycles
```

#### 3.24.1.6 total\_cycles

```
uint64_t m_eng_profiler_entry::total_cycles
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_logging.h](#)

## 3.25 m\_eng\_simple\_distortion\_str Struct Reference

```
#include <m_eng_simple_distortion.h>
```

## Data Fields

- [m\\_parameter pregain](#)
- [m\\_parameter postgain](#)

### 3.25.1 Field Documentation

#### 3.25.1.1 postgain

[m\\_parameter](#) m\_eng\_simple\_distortion\_str::postgain

#### 3.25.1.2 pregain

[m\\_parameter](#) m\_eng\_simple\_distortion\_str::pregain

The documentation for this struct was generated from the following file:

- [m\\_eng\\_simple\\_distortion.h](#)

## 3.26 m\_eng\_warbler\_str Struct Reference

```
#include <m_eng_warbler.h>
```

## Data Fields

- [m\\_parameter center](#)
- [m\\_parameter width](#)
- [m\\_parameter reactivity](#)
- [m\\_parameter sensitivity](#)
- [m\\_parameter max\\_rate](#)
- [m\\_parameter min\\_rate](#)
- float [alpha](#)
- float [e](#)
- float [t](#)
- float [rate](#)
- [m\\_eng\\_band\\_pass\\_filter\\_str](#) filter

### 3.26.1 Field Documentation

#### 3.26.1.1 alpha

float m\_eng\_warbler\_str::alpha

#### 3.26.1.2 center

[m\\_parameter](#) m\_eng\_warbler\_str::center

### 3.26.1.3 e

`float m_eng_warbler_str::e`

### 3.26.1.4 filter

`m_eng_band_pass_filter_str m_eng_warbler_str::filter`

### 3.26.1.5 max\_rate

`m_parameter m_eng_warbler_str::max_rate`

### 3.26.1.6 min\_rate

`m_parameter m_eng_warbler_str::min_rate`

### 3.26.1.7 rate

`float m_eng_warbler_str::rate`

### 3.26.1.8 reactivity

`m_parameter m_eng_warbler_str::reactivity`

### 3.26.1.9 sensitivity

`m_parameter m_eng_warbler_str::sensitivity`

### 3.26.1.10 t

`float m_eng_warbler_str::t`

### 3.26.1.11 width

`m_parameter m_eng_warbler_str::width`

The documentation for this struct was generated from the following file:

- [m\\_eng\\_warbler.h](#)



## 3.27 m\_eng\_waveshaper\_str Struct Reference

```
#include <m_eng_waveshaper.h>
```

### Data Fields

- [m\\_parameter](#) coefficient
- float(\* [shape](#) )(float x)

### 3.27.1 Field Documentation

#### 3.27.1.1 coefficient

[m\\_parameter](#) m\_eng\_waveshaper\_str::coefficient

#### 3.27.1.2 shape

float(\* m\_eng\_waveshaper\_str::shape) (float x)

The documentation for this struct was generated from the following file:

- [m\\_eng\\_waveshaper.h](#)

## 3.28 m\_lr\_high\_pass\_filter\_str Struct Reference

```
#include <m_eng_linkowitz_riley.h>
```

### Data Fields

- [m\\_parameter](#) cutoff\_frequency
- float [a0](#)
- float [a1](#)
- float [a2](#)
- float [a3](#)
- float [a4](#)
- float [x\\_11](#)
- float [x\\_12](#)
- float [y\\_11](#)
- float [y\\_12](#)
- float [x\\_21](#)
- float [x\\_22](#)
- float [y\\_21](#)
- float [y\\_22](#)

### 3.28.1 Field Documentation

#### 3.28.1.1 a0

`float m_lr_high_pass_filter_str::a0`

#### 3.28.1.2 a1

`float m_lr_high_pass_filter_str::a1`

#### 3.28.1.3 a2

`float m_lr_high_pass_filter_str::a2`

#### 3.28.1.4 a3

`float m_lr_high_pass_filter_str::a3`

#### 3.28.1.5 a4

`float m_lr_high_pass_filter_str::a4`

#### 3.28.1.6 cutoff\_frequency

`m_parameter m_lr_high_pass_filter_str::cutoff_frequency`

#### 3.28.1.7 x\_11

`float m_lr_high_pass_filter_str::x_11`

#### 3.28.1.8 x\_12

`float m_lr_high_pass_filter_str::x_12`

#### 3.28.1.9 x\_21

`float m_lr_high_pass_filter_str::x_21`

#### 3.28.1.10 x\_22

`float m_lr_high_pass_filter_str::x_22`

### 3.28.1.11 y\_11

```
float m_lr_high_pass_filter_str::y_11
```

### 3.28.1.12 y\_12

```
float m_lr_high_pass_filter_str::y_12
```

### 3.28.1.13 y\_21

```
float m_lr_high_pass_filter_str::y_21
```

### 3.28.1.14 y\_22

```
float m_lr_high_pass_filter_str::y_22
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_linkowitz\\_riley.h](#)

## 3.29 m\_lr\_low\_pass\_filter\_str Struct Reference

```
#include <m_eng_linkowitz_riley.h>
```

### Data Fields

- [m\\_parameter](#) cutoff\_frequency
- float [a0](#)
- float [a1](#)
- float [a2](#)
- float [a3](#)
- float [a4](#)
- float [x\\_11](#)
- float [x\\_12](#)
- float [y\\_11](#)
- float [y\\_12](#)
- float [x\\_21](#)
- float [x\\_22](#)
- float [y\\_21](#)
- float [y\\_22](#)

### 3.29.1 Field Documentation

#### 3.29.1.1 a0

```
float m_lr_low_pass_filter_str::a0
```

**3.29.1.2 a1**

```
float m_lr_low_pass_filter_str::a1
```

**3.29.1.3 a2**

```
float m_lr_low_pass_filter_str::a2
```

**3.29.1.4 a3**

```
float m_lr_low_pass_filter_str::a3
```

**3.29.1.5 a4**

```
float m_lr_low_pass_filter_str::a4
```

**3.29.1.6 cutoff\_frequency**

```
m_parameter m_lr_low_pass_filter_str::cutoff_frequency
```

**3.29.1.7 x\_11**

```
float m_lr_low_pass_filter_str::x_11
```

**3.29.1.8 x\_12**

```
float m_lr_low_pass_filter_str::x_12
```

**3.29.1.9 x\_21**

```
float m_lr_low_pass_filter_str::x_21
```

**3.29.1.10 x\_22**

```
float m_lr_low_pass_filter_str::x_22
```

**3.29.1.11 y\_11**

```
float m_lr_low_pass_filter_str::y_11
```

### 3.29.1.12 y\_12

```
float m_lr_low_pass_filter_str::y_12
```

### 3.29.1.13 y\_21

```
float m_lr_low_pass_filter_str::y_21
```

### 3.29.1.14 y\_22

```
float m_lr_low_pass_filter_str::y_22
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_linkowitz\\_riley.h](#)

## 3.30 m\_message Struct Reference

```
#include <m_comms.h>
```

### Data Fields

- [uint8\\_t type](#)
- [uint8\\_t data \[M\\_RESPONSE\\_MAX\\_DATA\\_LEN\]](#)
- [void\(\\* callback\)\(struct m\\_message msg, m\\_response response\)](#)
- [void \\* cb\\_arg](#)
- [int retries](#)

### 3.30.1 Field Documentation

#### 3.30.1.1 callback

```
void(* m_message::callback) (struct m\_message msg, m\_response response)
```

#### 3.30.1.2 cb\_arg

```
void* m_message::cb_arg
```

#### 3.30.1.3 data

```
uint8_t m_message::data[M_RESPONSE_MAX_DATA_LEN]
```

#### 3.30.1.4 retries

```
int m_message::retries
```

#### 3.30.1.5 type

```
uint8_t m_message::type
```

The documentation for this struct was generated from the following file:

- [m\\_comms.h](#)

## 3.31 m\_parameter Struct Reference

```
#include <m_parameter.h>
```

### Data Fields

- float [value](#)
- float [min](#)
- float [max](#)
- int [scale](#)
- int [updated](#)
- float [old\\_value](#)
- float [new\\_value](#)

### 3.31.1 Field Documentation

#### 3.31.1.1 max

```
float m_parameter::max
```

#### 3.31.1.2 min

```
float m_parameter::min
```

#### 3.31.1.3 new\_value

```
float m_parameter::new_value
```

#### 3.31.1.4 old\_value

```
float m_parameter::old_value
```

### 3.31.1.5 scale

```
int m_parameter::scale
```

### 3.31.1.6 updated

```
int m_parameter::updated
```

### 3.31.1.7 value

```
float m_parameter::value
```

The documentation for this struct was generated from the following file:

- [m\\_parameter.h](#)

## 3.32 m\_parameter\_id Struct Reference

```
#include <m_parameter.h>
```

### Data Fields

- uint16\_t [profile\\_id](#)
- uint16\_t [transformer\\_id](#)
- uint16\_t [parameter\\_id](#)

### 3.32.1 Field Documentation

#### 3.32.1.1 parameter\_id

```
uint16_t m_parameter_id::parameter_id
```

#### 3.32.1.2 profile\_id

```
uint16_t m_parameter_id::profile_id
```

#### 3.32.1.3 transformer\_id

```
uint16_t m_parameter_id::transformer_id
```

The documentation for this struct was generated from the following file:

- [m\\_parameter.h](#)

### 3.33 m\_pipeline Struct Reference

```
#include <m_pipeline.h>
```

The documentation for this struct was generated from the following file:

- [m\\_pipeline.h](#)

### 3.34 m\_pipeline\_mod Struct Reference

```
#include <m_eng_pipeline_mod.h>
```

#### Data Fields

- int [type](#)
- uint16\_t [tid](#)
- uint16\_t [data](#)
- int16\_t [sdata](#)

#### 3.34.1 Field Documentation

##### 3.34.1.1 data

```
uint16_t m_pipeline_mod::data
```

##### 3.34.1.2 sdata

```
int16_t m_pipeline_mod::sdata
```

##### 3.34.1.3 tid

```
uint16_t m_pipeline_mod::tid
```

##### 3.34.1.4 type

```
int m_pipeline_mod::type
```

The documentation for this struct was generated from the following file:

- [m\\_eng\\_pipeline\\_mod.h](#)



## 3.35 m\_profile Struct Reference

```
#include <m_profile.h>
```

### Data Fields

- int [active](#)

### 3.35.1 Field Documentation

#### 3.35.1.1 active

```
int m_profile::active
```

The documentation for this struct was generated from the following file:

- [m\\_profile.h](#)

## 3.36 m\_response Struct Reference

```
#include <m_comms.h>
```

### Data Fields

- uint8\_t [type](#)
- uint8\_t [data](#) [[M\\_RESPONSE\\_MAX\\_DATA\\_LEN](#)]
- void \* [extra](#)

### 3.36.1 Field Documentation

#### 3.36.1.1 data

```
uint8_t m_response::data[M_RESPONSE_MAX_DATA_LEN]
```

#### 3.36.1.2 extra

```
void* m_response::extra
```

#### 3.36.1.3 type

```
uint8_t m_response::type
```

The documentation for this struct was generated from the following file:

- [m\\_comms.h](#)

## 3.37 m\_setting Struct Reference

```
#include <m_parameter.h>
```

### Data Fields

- [int16\\_t value](#)
- [int updated](#)
- [int16\\_t old\\_value](#)
- [int16\\_t new\\_value](#)

### 3.37.1 Field Documentation

#### 3.37.1.1 new\_value

```
int16_t m_setting::new_value
```

#### 3.37.1.2 old\_value

```
int16_t m_setting::old_value
```

#### 3.37.1.3 updated

```
int m_setting::updated
```

#### 3.37.1.4 value

```
int16_t m_setting::value
```

The documentation for this struct was generated from the following file:

- [m\\_parameter.h](#)

## 3.38 m\_setting\_id Struct Reference

```
#include <m_parameter.h>
```

### Data Fields

- [uint16\\_t profile\\_id](#)
- [uint16\\_t transformer\\_id](#)
- [uint16\\_t setting\\_id](#)

### 3.38.1 Field Documentation

#### 3.38.1.1 profile\_id

```
uint16_t m_setting_id::profile_id
```

#### 3.38.1.2 setting\_id

```
uint16_t m_setting_id::setting_id
```

#### 3.38.1.3 transformer\_id

```
uint16_t m_setting_id::transformer_id
```

The documentation for this struct was generated from the following file:

- [m\\_parameter.h](#)

## 3.39 m\_transformer Struct Reference

```
#include <m_transformer.h>
```

### Data Fields

- [uint16\\_t type](#)
- [uint16\\_t id](#)
- [m\\_parameter wet\\_mix](#)
- [m\\_setting band\\_mode](#)
- [m\\_parameter band\\_lp\\_cutoff](#)
- [m\\_parameter band\\_hp\\_cutoff](#)
- [m\\_parameter band\\_center](#)
- [m\\_parameter band\\_width](#)

### 3.39.1 Field Documentation

#### 3.39.1.1 band\_center

```
m_parameter m_transformer::band_center
```

#### 3.39.1.2 band\_hp\_cutoff

```
m_parameter m_transformer::band_hp_cutoff
```

### 3.39.1.3 band\_lp\_cutoff

[m\\_parameter](#) m\_transformer::band\_lp\_cutoff

### 3.39.1.4 band\_mode

[m\\_setting](#) m\_transformer::band\_mode

### 3.39.1.5 band\_width

[m\\_parameter](#) m\_transformer::band\_width

### 3.39.1.6 id

uint16\_t m\_transformer::id

### 3.39.1.7 type

uint16\_t m\_transformer::type

### 3.39.1.8 wet\_mix

[m\\_parameter](#) m\_transformer::wet\_mix

The documentation for this struct was generated from the following file:

- [m\\_transformer.h](#)

## 3.40 m\_transformer\_str Struct Reference

```
#include <m_eng_transformer_template.h>
```

### Data Fields

- [m\\_parameter](#) param

### 3.40.1 Field Documentation

#### 3.40.1.1 param

[m\\_parameter](#) m\_transformer\_str::param

The documentation for this struct was generated from the following file:

- [m\\_eng\\_transformer\\_template.h](#)

## Chapter 4

# File Documentation

### 4.1 m\_eng.h File Reference

```
#include <arm_math.h>
#include "utility/imxrt_hw.h"
#include <stdint.h>
#include <string.h>
#include <stdarg.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include "m_transformer_enum.h"
#include "m_error_codes.h"
#include "m_status.h"
#include "m_comms.h"
#include "m_alloc.h"
#include "m_linked_list.h"
#include "m_parameter.h"
#include "m_transformer.h"
#include "m_pipeline.h"
#include "m_profile.h"
#include "m_eng_logging.h"
#include "m_eng_flops.h"
#include "m_eng_useful_functions.h"
#include "m_eng_audio_block.h"
#include "m_eng_mempool.h"
#include "m_eng_globals.h"
#include "m_eng_parameter.h"
#include "m_eng_transformer.h"
#include "m_eng_transformer_init.h"
#include "m_eng_delay_buffer.h"
#include "m_eng_buffer_mixer_amp.h"
#include "m_eng_linkowitz_riley.h"
#include "m_eng_equaliser.h"
#include "m_eng_pass_filter.h"
#include "m_eng_biquad.h"
#include "m_eng_compressor.h"
#include "m_eng_low_end_compressor.h"
#include "m_eng_waveshaper.h"
#include "m_eng_adaptive_waveshaper.h"
```

```

#include "m_eng_simple_distortion.h"
#include "m_eng_distortion.h"
#include "m_eng_dirty_octave.h"
#include "m_eng_noise_suppressor.h"
#include "m_eng_percussifier.h"
#include "m_eng_envelope.h"
#include "m_eng_flanger.h"
#include "m_eng_warbler.h"
#include "m_eng_delay.h"
#include "m_eng_pipeline.h"
#include "m_eng_pipeline_mod.h"
#include "m_eng_profile.h"
#include "m_eng_update.h"
#include "m_eng_i2s_dma.h"
#include "m_eng_sgtl5000.h"
#include "m_eng_memcpy_audio.h"
#include "m_eng_context.h"
#include "m_eng_comms.h"
#include "m_eng_transition.h"
#include "m_eng_printf.h"
#include "m_eng_debugging.h"

```

## Macros

- #define [AUDIO\\_BLOCK\\_SAMPLES](#) 128
- #define [AUDIO\\_SAMPLE\\_RATE\\_EXACT](#) 44100.0f
- #define [AUDIO\\_SAMPLE\\_RATE](#) [AUDIO\\_SAMPLE\\_RATE\\_EXACT](#)
- #define [SAMPLE\\_FREQUENCY](#) 0.0000226757369615
- #define [NUM\\_MASKS](#) ((([MAX\\_AUDIO\\_MEMORY](#) / [AUDIO\\_BLOCK\\_SAMPLES](#) / 2) + 31) / 32)
- #define [M\\_ENGINE](#)
- #define [LN\\_2](#) 0.69314718055994530942
- #define [S\\_TO\\_SAMPLES](#)(x)
- #define [SAMPLES\\_TO\\_S](#)(x)
- #define [MS\\_TO\\_SAMPLES](#)(x)
- #define [SAMPLES\\_TO\\_MS](#)(x)
- #define [AUDIO\\_BLOCK\\_MS](#) (((float)[AUDIO\\_BLOCK\\_SAMPLES](#) \* 1000.0) / (float)[AUDIO\\_SAMPLE\\_RATE](#))
- #define [ALLOW\\_PRINTLINES](#)
- #define [M\\_VOICE\\_ERR](#) 0
- #define [M\\_VOICE\\_COMMS](#) 1
- #define [M\\_VOICE\\_CXT](#) 2
- #define [M\\_VOICE\\_TR](#) 3
- #define [M\\_VOICE\\_PL](#) 4
- #define [M\\_VOICE\\_PR](#) 5
- #define [M\\_VOICE\\_LOG](#) 6
- #define [M\\_VOICE\\_PRF](#) 7
- #define [LL\\_MALLOC](#) [m\\_alloc](#)
- #define [LL\\_FREE](#) [m\\_free](#)
- #define [sqr](#)(x)
- #define [binary\\_max](#)(x, y)
- #define [binary\\_min](#)(x, y)

## Functions

- float [trig\\_transition\\_function](#) (float x)

## 4.1.1 Macro Definition Documentation

### 4.1.1.1 ALLOW\_PRINTLINES

```
#define ALLOW_PRINTLINES
```

### 4.1.1.2 AUDIO\_BLOCK\_MS

```
#define AUDIO_BLOCK_MS (((float)AUDIO_BLOCK_SAMPLES * 1000.0) / (float)AUDIO_SAMPLE_RATE)
```

### 4.1.1.3 AUDIO\_BLOCK\_SAMPLES

```
#define AUDIO_BLOCK_SAMPLES 128
```

### 4.1.1.4 AUDIO\_SAMPLE\_RATE

```
#define AUDIO_SAMPLE_RATE AUDIO_SAMPLE_RATE_EXACT
```

### 4.1.1.5 AUDIO\_SAMPLE\_RATE\_EXACT

```
#define AUDIO_SAMPLE_RATE_EXACT 44100.0f
```

### 4.1.1.6 binary\_max

```
#define binary_max(  
    x,  
    y)
```

**Value:**

```
((x > y) ? x : y)
```

### 4.1.1.7 binary\_min

```
#define binary_min(  
    x,  
    y)
```

**Value:**

```
((x > y) ? y : x)
```

### 4.1.1.8 LL\_FREE

```
#define LL_FREE m_free
```

#### 4.1.1.9 LL\_MALLOC

```
#define LL_MALLOC m_alloc
```

#### 4.1.1.10 LN\_2

```
#define LN_2 0.69314718055994530942
```

#### 4.1.1.11 M\_ENGINE

```
#define M_ENGINE
```

#### 4.1.1.12 M\_VOICE\_COMMS

```
#define M_VOICE_COMMS 1
```

#### 4.1.1.13 M\_VOICE\_CXT

```
#define M_VOICE_CXT 2
```

#### 4.1.1.14 M\_VOICE\_ERR

```
#define M_VOICE_ERR 0
```

#### 4.1.1.15 M\_VOICE\_LOG

```
#define M_VOICE_LOG 6
```

#### 4.1.1.16 M\_VOICE\_PL

```
#define M_VOICE_PL 4
```

#### 4.1.1.17 M\_VOICE\_PR

```
#define M_VOICE_PR 5
```

#### 4.1.1.18 M\_VOICE\_PRF

```
#define M_VOICE_PRF 7
```



#### 4.1.1.19 M\_VOICE\_TR

```
#define M_VOICE_TR 3
```

#### 4.1.1.20 MS\_TO\_SAMPLES

```
#define MS_TO_SAMPLES(  
    x)
```

**Value:**

```
((int)(ceilf((((float)(x) / 1000.0) * AUDIO_SAMPLE_RATE))))
```

#### 4.1.1.21 NUM\_MASKS

```
#define NUM_MASKS (((MAX_AUDIO_MEMORY / AUDIO_BLOCK_SAMPLES / 2) + 31) / 32)
```

#### 4.1.1.22 S\_TO\_SAMPLES

```
#define S_TO_SAMPLES(  
    x)
```

**Value:**

```
((int)(ceilf((float)(x) * AUDIO_SAMPLE_RATE)))
```

#### 4.1.1.23 SAMPLE\_FREQUENCY

```
#define SAMPLE_FREQUENCY 0.0000226757369615
```

#### 4.1.1.24 SAMPLES\_TO\_MS

```
#define SAMPLES_TO_MS(  
    x)
```

**Value:**

```
((float)(x) * 1000.0) / (float)AUDIO_SAMPLE_RATE
```

#### 4.1.1.25 SAMPLES\_TO\_S

```
#define SAMPLES_TO_S(  
    x)
```

**Value:**

```
((float)(x) / (float)AUDIO_SAMPLE_RATE)
```

#### 4.1.1.26 sqr

```
#define sqr(  
    x)
```

##### Value:

(x \* x)

### 4.1.2 Function Documentation

#### 4.1.2.1 trig\_transition\_function()

```
float trig_transition_function (  
    float x)
```

## 4.2 m\_eng.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_MAIN_H_
00002 #define M_MAIN_H_
00003
00004 #define AUDIO_BLOCK_SAMPLES      128
00005
00006 #ifndef ARM_ASSEMBLY
00007 #define AUDIO_SAMPLE_RATE_EXACT  44100.0f
00008
00009 #define AUDIO_SAMPLE_RATE        AUDIO_SAMPLE_RATE_EXACT
00010 #define SAMPLE_FREQUENCY         0.0000226757369615
00011
00012 #define NUM_MASKS                (( (MAX_AUDIO_MEMORY / AUDIO_BLOCK_SAMPLES / 2) + 31) / 32)
00013
00014 #ifndef M_ENGINE
00015 #define M_ENGINE
00016 #endif
00017
00018 // #define NO_CMSIS
00019
00020 #ifdef M_SIMULATED
00021 #ifndef NO_CMSIS
00022 #define NO_CMSIS
00023 #endif
00024 #endif
00025
00026 #ifdef M_SIMULATED
00027 #define DMAMEM
00028 #define FLASHMEM
00029 #define __enable_irq()
00030 #define __disable_irq()
00031 #define PI      3.14159265
00032 #else
00033 #include <arm_math.h>
00034 #include "utility/imxrt_hw.h"
00035 #endif
00036
00037 #ifdef __cplusplus
00038 #include <stdint.h>
00039 #include <cstring>
00040 #include <cstdint>
00041 #include <stdlib.h>
00042 #include <stdio.h>
00043 #include <cmath>
00044
00045 extern "C" {
00046 #else
00047 #include <stdint.h>
00048 #include <string.h>
00049 #include <stdarg.h>
00050 #include <stdlib.h>
00051 #include <stdio.h>
00052 #include <math.h>
00053 #endif
```

```
00054
00055 #define LN_2 0.69314718055994530942
00056
00057 #define S_TO_SAMPLES(x) ((int)(ceilf((float)(x) * AUDIO_SAMPLE_RATE)))
00058 #define SAMPLES_TO_S(x) ((float)(x) / (float)AUDIO_SAMPLE_RATE)
00059
00060 #define MS_TO_SAMPLES(x) ((int)(ceilf(((float)(x) / 1000.0) * AUDIO_SAMPLE_RATE))))
00061 #define SAMPLES_TO_MS(x) (((float)(x) * 1000.0) / (float)AUDIO_SAMPLE_RATE)
00062
00063 #define AUDIO_BLOCK_MS (((float)AUDIO_BLOCK_SAMPLES * 1000.0) / (float)AUDIO_SAMPLE_RATE)
00064
00065 #define ALLOW_PRINTLINES
00066 #define M_VOICE_ERR 0
00067 #define M_VOICE_COMMS 1
00068 #define M_VOICE_CXT 2
00069 #define M_VOICE_TR 3
00070 #define M_VOICE_PL 4
00071 #define M_VOICE_PR 5
00072 #define M_VOICE_LOG 6
00073 #define M_VOICE_PRF 7
00074
00075 // #define ENABLE_LOGGING
00076
00077 #include "m_transformer_enum.h"
00078 #include "m_error_codes.h"
00079 #include "m_status.h"
00080 #include "m_comms.h"
00081
00082 #include "m_alloc.h"
00083
00084 #define LL_MALLOC m_alloc
00085 #define LL_FREE m_free
00086
00087 #include "m_linked_list.h"
00088
00089 #include "m_parameter.h"
00090 #include "m_transformer.h"
00091 #include "m_pipeline.h"
00092 #include "m_profile.h"
00093
00094 #include "m_eng_logging.h"
00095
00096 #include "m_eng_flops.h"
00097 #include "m_eng_useful_functions.h"
00098
00099 #include "m_eng_audio_block.h"
00100 #include "m_eng_mempool.h"
00101
00102 #include "m_eng_globals.h"
00103
00104 #include "m_eng_parameter.h"
00105 #include "m_eng_transformer.h"
00106 #include "m_eng_transformer_init.h"
00107
00108 #include "m_eng_delay_buffer.h"
00109
00110 #include "m_eng_buffer_mixer_amp.h"
00111 #include "m_eng_linkowitz_riley.h"
00112 #include "m_eng_equaliser.h"
00113 #include "m_eng_pass_filter.h"
00114 #include "m_eng_biquad.h"
00115 #include "m_eng_compressor.h"
00116 #include "m_eng_low_end_compressor.h"
00117 #include "m_eng_waveshaper.h"
00118 #include "m_eng_adaptive_waveshaper.h"
00119 #include "m_eng_simple_distortion.h"
00120 #include "m_eng_distortion.h"
00121 #include "m_eng_dirty_octave.h"
00122 #include "m_eng_noise_suppressor.h"
00123 #include "m_eng_percussifier.h"
00124 #include "m_eng_envelope.h"
00125 #include "m_eng_flanger.h"
00126 #include "m_eng_warbler.h"
00127 #include "m_eng_delay.h"
00128
00129 #ifdef GRAPH_PIPELINE
00130 #include "m_eng_graph.h"
00131 #endif
00132 #include "m_eng_pipeline.h"
00133 #include "m_eng_pipeline_mod.h"
00134 #include "m_eng_profile.h"
00135 #include "m_eng_update.h"
00136
00137 #ifndef M_SIMULATED
00138 #include "m_eng_i2s_dma.h"
00139 #include "m_eng_sgtl5000.h"
00140 #include "m_eng_memcpy_audio.h"
```

```

00141 #endif
00142
00143 #include "m_eng_context.h"
00144 #include "m_eng_comms.h"
00145
00146 #include "m_eng_transition.h"
00147
00148 #include "m_eng_printf.h"
00149 #include "m_eng_debugging.h"
00150
00151 #ifdef M_SIMULATED
00152 #include "m_eng_sim.h"
00153 #endif
00154
00155 #define sqr(x) (x * x)
00156
00157 #define binary_max(x, y) ((x > y) ? x : y)
00158 #define binary_min(x, y) ((x > y) ? y : x)
00159
00160 float trig_transition_function(float x);
00161
00162 #ifdef __cplusplus
00163 }
00164 #endif
00165
00166 #endif
00167 #endif

```

## 4.3 m\_eng\_adaptive\_waveshaper.h File Reference

### Data Structures

- struct [m\\_eng\\_adaptive\\_waveshaper\\_str](#)

### Macros

- #define [ADAPTIVE\\_WAVESHAPER\\_ENVELOPE\\_ATTACK](#) `expf(-7.0/8.0)`
- #define [ADAPTIVE\\_WAVESHAPER\\_ENVELOPE\\_RELEASE](#) `expf(-7.0/2048.0)`

### Functions

- int [init\\_adaptive\\_waveshaper\\_str](#) ([m\\_eng\\_adaptive\\_waveshaper\\_str](#) \*str)
- int [calc\\_adaptive\\_waveshaper](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.3.1 Macro Definition Documentation

#### 4.3.1.1 ADAPTIVE\_WAVESHAPER\_ENVELOPE\_ATTACK

```
#define ADAPTIVE_WAVESHAPER_ENVELOPE_ATTACK expf(-7.0/8.0)
```

#### 4.3.1.2 ADAPTIVE\_WAVESHAPER\_ENVELOPE\_RELEASE

```
#define ADAPTIVE_WAVESHAPER_ENVELOPE_RELEASE expf(-7.0/2048.0)
```

## 4.3.2 Function Documentation

### 4.3.2.1 calc\_adaptive\_waveshaper()

```
int calc_adaptive_waveshaper (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

### 4.3.2.2 init\_adaptive\_waveshaper\_str()

```
int init_adaptive_waveshaper_str (
    m_eng_adaptive_waveshaper_str * str)
```

## 4.4 m\_eng\_adaptive\_waveshaper.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ADAPTIVE_WAVESHAPER_H_
00002 #define M_ADAPTIVE_WAVESHAPER_H_
00003
00004 #define ADAPTIVE_WAVESHAPER_ENVELOPE_ATTACK expf(-7.0/8.0)
00005 #define ADAPTIVE_WAVESHAPER_ENVELOPE_RELEASE expf(-7.0/2048.0)
00006
00007 typedef struct
00008 {
00009     m_parameter coefficient;
00010
00011     float (*shape)(float x);
00012
00013     float local_amplitude;
00014 } m_eng_adaptive_waveshaper_str;
00015
00016 int init_adaptive_waveshaper_str(m_eng_adaptive_waveshaper_str *str);
00017 int calc_adaptive_waveshaper(void *data_struct, float *dest, float *src, int n_samples);
00018
00019
00020 #endif
```

## 4.5 m\_eng\_audio\_block.h File Reference

### Macros

- #define [AUDIO\\_BLOCK\\_SAMPLES](#) 128

### 4.5.1 Macro Definition Documentation

#### 4.5.1.1 AUDIO\_BLOCK\_SAMPLES

```
#define AUDIO_BLOCK_SAMPLES 128
```

## 4.6 m\_eng\_audio\_block.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_AUDIO_BLOCK_H_
00002 #define M_AUDIO_BLOCK_H_
00003
00004 #ifndef AUDIO_BLOCK_SAMPLES
00005 #define AUDIO_BLOCK_SAMPLES 128
00006 #endif
00007
00008 #ifdef __cplusplus
00009 extern "C" {
00010 #endif
00011
00012
00013
00014 #ifdef __cplusplus
00015 }
00016 #endif
00017
00018 #endif
```

## 4.7 m\_eng\_band\_splitter.h File Reference

### Data Structures

- struct [m\\_eng\\_3\\_band\\_splitter\\_str](#)
- struct [m\\_eng\\_n\\_band\\_splitter\\_str](#)

### Functions

- int [init\\_3\\_band\\_splitter\\_str](#) ([m\\_eng\\_3\\_band\\_splitter\\_str](#) \*str)
- int [reconfigure\\_3\\_band\\_splitter](#) (void \*data\_struct)
- int [calc\\_3\\_band\\_splitter](#) (void \*data\_struct, float \*\*dest, float \*\*src, int n\_samples)
- int [init\\_n\\_band\\_splitter\\_str](#) ([m\\_eng\\_n\\_band\\_splitter\\_str](#) \*str)
- int [reconfigure\\_n\\_band\\_splitter](#) (void \*data\_struct)
- int [calc\\_n\\_band\\_splitter](#) (void \*data\_struct, float \*\*dest, float \*\*src, int n\_samples)

### 4.7.1 Function Documentation

#### 4.7.1.1 calc\_3\_band\_splitter()

```
int calc_3_band_splitter (
    void * data_struct,
    float ** dest,
    float ** src,
    int n_samples)
```

#### 4.7.1.2 calc\_n\_band\_splitter()

```
int calc_n_band_splitter (
    void * data_struct,
    float ** dest,
    float ** src,
    int n_samples)
```

**4.7.1.3 init\_3\_band\_splitter\_str()**

```
int init_3_band_splitter_str (
    m_eng_3_band_splitter_str * str)
```

**4.7.1.4 init\_n\_band\_splitter\_str()**

```
int init_n_band_splitter_str (
    m_eng_n_band_splitter_str * str)
```

**4.7.1.5 reconfigure\_3\_band\_splitter()**

```
int reconfigure_3_band_splitter (
    void * data_struct)
```

**4.7.1.6 reconfigure\_n\_band\_splitter()**

```
int reconfigure_n_band_splitter (
    void * data_struct)
```

**4.8 m\_eng\_band\_splitter.h**

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_TRANSFORMER_H_
00002 #define M_ENG_TRANSFORMER_H_
00003
00004 typedef struct
00005 {
00006     m_lr_low_pass_filter_str filters[3];
00007 } m_eng_3_band_splitter_str;
00008
00009 int init_3_band_splitter_str(m_eng_3_band_splitter_str *str);
00010 int reconfigure_3_band_splitter(void *data_struct);
00011 int calc_3_band_splitter(void *data_struct, float **dest, float **src, int n_samples);
00012
00013 typedef struct
00014 {
00015     m_lr_low_pass_filter_str *filters;
00016 } m_eng_n_band_splitter_str;
00017
00018 int init_n_band_splitter_str(m_eng_n_band_splitter_str *str);
00019 int reconfigure_n_band_splitter(void *data_struct);
00020 int calc_n_band_splitter(void *data_struct, float **dest, float **src, int n_samples);
00021
00022 #endif
00023
```

**4.9 m\_eng\_biquad.h File Reference****Data Structures**

- struct [m\\_eng\\_biquad\\_str](#)

## Functions

- int [init\\_biquad\\_str](#) ([m\\_eng\\_biquad\\_str](#) \*str)
- int [reconfigure\\_biquad](#) (void \*data\_struct)
- int [calc\\_biquad](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

## 4.9.1 Function Documentation

### 4.9.1.1 [calc\\_biquad\(\)](#)

```
int calc_biquad (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

### 4.9.1.2 [init\\_biquad\\_str\(\)](#)

```
int init_biquad_str (
    m\_eng\_biquad\_str * str)
```

### 4.9.1.3 [reconfigure\\_biquad\(\)](#)

```
int reconfigure_biquad (
    void * data_struct)
```

## 4.10 [m\\_eng\\_biquad.h](#)

[Go to the documentation of this file.](#)

```
00001 #ifndef M_BIQUAD_H_
00002 #define M_BIQUAD_H_
00003
00004 typedef struct
00005 {
00006     float a0, a1, a2, a3, a4;
00007     float x1, x2, y1, y2;
00008
00009     m\_setting type;
00010     m\_parameter cutoff;
00011     m\_parameter bandwidth;
00012     m\_parameter db_gain;
00013 } m\_eng\_biquad\_str;
00014
00015 int init\_biquad\_str(m\_eng\_biquad\_str *str);
00016 int reconfigure\_biquad(void *data_struct);
00017 int calc\_biquad(void *data_struct, float *dest, float *src, int n_samples);
00018
00019 #endif
```

## 4.11 [m\\_eng\\_buffer\\_mixer\\_amp.h](#) File Reference

### Data Structures

- struct [m\\_eng\\_amplifier\\_str](#)
- struct [m\\_eng\\_mixer\\_str](#)



## Macros

- `#define M_ENG_AMPLIFIER_LINEAR 0`
- `#define M_ENG_AMPLIFIER_DB 1`

## Functions

- `int calc_buffer` (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- `int init_amplifier_str` (m\_eng\_amplifier\_str \*str)
- `int reconfigure_amplifier` (void \*data\_struct)
- `int calc_amplifier` (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.11.1 Macro Definition Documentation

#### 4.11.1.1 M\_ENG\_AMPLIFIER\_DB

```
#define M_ENG_AMPLIFIER_DB 1
```

#### 4.11.1.2 M\_ENG\_AMPLIFIER\_LINEAR

```
#define M_ENG_AMPLIFIER_LINEAR 0
```

### 4.11.2 Function Documentation

#### 4.11.2.1 calc\_amplifier()

```
int calc_amplifier (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.11.2.2 calc\_buffer()

```
int calc_buffer (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.11.2.3 init\_amplifier\_str()

```
int init_amplifier_str (  
    m_eng_amplifier_str * str)
```

#### 4.11.2.4 reconfigure\_amplifier()

```
int reconfigure_amplifier (
    void * data_struct)
```

### 4.12 m\_eng\_buffer\_mixer\_amp.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_BMA_H_
00002 #define M_BMA_H_
00003
00004 #define M_ENG_AMPLIFIER_LINEAR      0
00005 #define M_ENG_AMPLIFIER_DB         1
00006
00007 typedef struct
00008 {
00009     m_parameter gain;
00010     float g;
00011
00012     m_setting mode;
00013     int db;
00014 } m_eng_amplifier_str;
00015
00016 typedef struct
00017 {
00018     m_parameter ratio;
00019 } m_eng_mixer_str;
00020
00021
00022 int calc_buffer(void *data_struct, float *dest, float *src, int n_samples);
00023
00024 int init_amplifier_str(m_eng_amplifier_str *str);
00025 int reconfigure_amplifier(void *data_struct);
00026 int calc_amplifier(void *data_struct, float *dest, float *src, int n_samples);
00027
00028 /*
00029 int init_mixer_str(m_eng_mixer_str *str);
00030 int calc_mixer(void *data_struct, float **dest, float **src, int n_samples);
00031 */
00032
00033 #endif
```

### 4.13 m\_eng\_comms.h File Reference

#### Macros

- `#define TEENSY_I2C_SLAVE_ADDR 0x08`

#### Functions

- `int init_esp32_link ()`
- `void esp32_message_check_handle ()`
- `void i2c_receive_isr (int n)`
- `void i2c_request_isr ()`

#### 4.13.1 Macro Definition Documentation

##### 4.13.1.1 TEENSY\_I2C\_SLAVE\_ADDR

```
#define TEENSY_I2C_SLAVE_ADDR 0x08
```

## 4.13.2 Function Documentation

### 4.13.2.1 esp32\_message\_check\_handle()

```
void esp32_message_check_handle ()
```

### 4.13.2.2 i2c\_receive\_isr()

```
void i2c_receive_isr (  
    int n)
```

### 4.13.2.3 i2c\_request\_isr()

```
void i2c_request_isr ()
```

### 4.13.2.4 init\_esp32\_link()

```
int init_esp32_link ()
```

## 4.14 m\_eng\_comms.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_INTERFACE_H_
00002 #define M_INTERFACE_H_
00003
00004 #ifdef __cplusplus
00005 extern "C" {
00006 #endif
00007
00008 #define TEENSY_I2C_SLAVE_ADDR 0x08
00009
00010 int init_esp32_link();
00011
00012 void esp32_message_check_handle();
00013
00014 void i2c_receive_isr(int n);
00015 void i2c_request_isr();
00016
00017 #ifdef __cplusplus
00018 }
00019 #endif
00020
00021 #endif
```

## 4.15 m\_eng\_compressor.h File Reference

### Data Structures

- struct [m\\_eng\\_compressor\\_str](#)

## Functions

- `int init_compressor_str (m_eng_compressor_str *str)`
- `int reconfigure_compressor (void *data_struct)`
- `int calc_compressor (void *data_struct, float *dest, float *src, int n_samples)`

### 4.15.1 Function Documentation

#### 4.15.1.1 calc\_compressor()

```
int calc_compressor (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

#### 4.15.1.2 init\_compressor\_str()

```
int init_compressor_str (
    m_eng_compressor_str * str)
```

#### 4.15.1.3 reconfigure\_compressor()

```
int reconfigure_compressor (
    void * data_struct)
```

## 4.16 m\_eng\_compressor.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_COMPRESSOR1_H_
00002 #define M_COMPRESSOR1_H_
00003
00004 typedef struct
00005 {
00006     m_parameter ratio;
00007     m_parameter threshold;
00008     m_parameter attack;
00009     m_parameter release;
00010
00011     float alpha;
00012     float rho;
00013
00014     float e_final;
00015 } m_eng_compressor_str;
00016
00017 int init_compressor_str(m_eng_compressor_str *str);
00018 int reconfigure_compressor(void *data_struct);
00019 int calc_compressor(void *data_struct, float *dest, float *src, int n_samples);
00020
00021 #endif
```

## 4.17 m\_eng\_context.h File Reference

```
#include "m_eng_profile.h"
```

## Data Structures

- struct [m\\_eng\\_context](#)

## Macros

- #define [M\\_PROFILE\\_SWITCH\\_SAMPLES](#) 256
- #define [PROFILES\\_MALLOC\\_CHUNK\\_SIZE](#) 16
- #define [PROFILE\\_ARRAY\\_INITIAL\\_SIZE](#) 64
- #define [SILENCE\\_ENERGY\\_THRESHOLD](#) 2000
- #define [SILENCE\\_BLOCKS\\_THRESHOLD](#) 64
- #define [DECLICK\\_BUFSIZE](#) 6
- #define [CLICK\\_SLOPE\\_THRESHOLD](#) 0.5
- #define [DC\\_BLOCKER\\_ALPHA](#) 0.999

## Functions

- int [init\\_m\\_eng\\_context](#) ([m\\_eng\\_context](#) \*cxt)
- int [m\\_eng\\_context\\_new\\_profile](#) ([m\\_eng\\_context](#) \*cxt)
- int [cxt\\_set\\_active\\_profile](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid)
- int [cxt\\_switch\\_to\\_profile](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid)
- int [cxt\\_profile\\_id\\_valid](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid)
- int [cxt\\_transformer\\_id\\_valid](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t tid)
- int [cxt\\_parameter\\_id\\_valid](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t ppid)
- [m\\_parameter](#) \* [cxt\\_get\\_parameter\\_by\\_id](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t ppid)
- int [cxt\\_update\\_parameter\\_value\\_by\\_id](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t ppid, float new\_value)
- [m\\_setting](#) \* [cxt\\_get\\_setting\\_by\\_id](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t sid)
- int [cxt\\_update\\_setting\\_value\\_by\\_id](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t sid, uint16\_t new\_value)
- int [cxt\\_append\\_transformer\\_to\\_profile](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t type)
- int [cxt\\_remove\\_transformer\\_from\\_profile](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t tid)
- int [cxt\\_insert\\_transformer\\_to\\_profile](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t type, uint16\_t pos)
- int [cxt\\_prepend\\_transformer\\_to\\_profile](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t type)
- int [cxt\\_get\\_n\\_profile\\_transformers](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid)
- int [cxt\\_get\\_n\\_transformer\\_params](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t tid)
- int [cxt\\_get\\_n\\_transformer\\_settings](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t tid)
- int [cxt\\_get\\_transformer\\_type](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t tid)
- int [cxt\\_get\\_tid\\_by\\_pos](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t pos)
- [m\\_transformer](#) \* [cxt\\_get\\_transformer\\_by\\_id](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t pid, uint16\_t tid)
- int [cxt\\_move\\_transformer](#) ([m\\_eng\\_context](#) \*cxt, uint16\_t tid, uint16\_t new\_pos)
- int [cxt\\_process](#) ([m\\_eng\\_context](#) \*cxt)
- void [m\\_eng\\_safe\\_reboot](#) ([m\\_eng\\_context](#) \*cxt)
- int [reset\\_context](#) ([m\\_eng\\_context](#) \*cxt)
- int [cxt\\_run\\_scheduled\\_maintenance](#) ([m\\_eng\\_context](#) \*cxt)

## Variables

- [m\\_eng\\_context\\_global\\_cxt](#)

## 4.17.1 Macro Definition Documentation

### 4.17.1.1 CLICK\_SLOPE\_THRESHOLD

```
#define CLICK_SLOPE_THRESHOLD 0.5
```

### 4.17.1.2 DC\_BLOCKER\_ALPHA

```
#define DC_BLOCKER_ALPHA 0.999
```

### 4.17.1.3 DECLICK\_BUFSIZE

```
#define DECLICK_BUFSIZE 6
```

### 4.17.1.4 M\_PROFILE\_SWITCH\_SAMPLES

```
#define M_PROFILE_SWITCH_SAMPLES 256
```

### 4.17.1.5 PROFILE\_ARRAY\_INITIAL\_SIZE

```
#define PROFILE_ARRAY_INITIAL_SIZE 64
```

### 4.17.1.6 PROFILES\_MALLOC\_CHUNK\_SIZE

```
#define PROFILES_MALLOC_CHUNK_SIZE 16
```

### 4.17.1.7 SILENCE\_BLOCKS\_THRESHOLD

```
#define SILENCE_BLOCKS_THRESHOLD 64
```

### 4.17.1.8 SILENCE\_ENERGY\_THRESHOLD

```
#define SILENCE_ENERGY_THRESHOLD 2000
```

## 4.17.2 Function Documentation

### 4.17.2.1 cxt\_append\_transformer\_to\_profile()

```
int cxt_append_transformer_to_profile (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t type)
```

#### 4.17.2.2 cxt\_get\_n\_profile\_transformers()

```
int cxt_get_n_profile_transformers (  
    m_eng_context * cxt,  
    uint16_t pid)
```

#### 4.17.2.3 cxt\_get\_n\_transformer\_params()

```
int cxt_get_n_transformer_params (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid)
```

#### 4.17.2.4 cxt\_get\_n\_transformer\_settings()

```
int cxt_get_n_transformer_settings (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid)
```

#### 4.17.2.5 cxt\_get\_parameter\_by\_id()

```
m_parameter * cxt_get_parameter_by_id (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid,  
    uint16_t ppid)
```

#### 4.17.2.6 cxt\_get\_setting\_by\_id()

```
m_setting * cxt_get_setting_by_id (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid,  
    uint16_t sid)
```

#### 4.17.2.7 cxt\_get\_tid\_by\_pos()

```
int cxt_get_tid_by_pos (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t pos)
```

#### 4.17.2.8 cxt\_get\_transformer\_by\_id()

```
m_transformer * cxt_get_transformer_by_id (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid)
```

#### 4.17.2.9 cxt\_get\_transformer\_type()

```
int cxt_get_transformer_type (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid)
```

#### 4.17.2.10 cxt\_insert\_transformer\_to\_profile()

```
int cxt_insert_transformer_to_profile (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t type,  
    uint16_t pos)
```

#### 4.17.2.11 cxt\_move\_transformer()

```
int cxt_move_transformer (  
    m_eng_context * cxt,  
    uint16_t tid,  
    uint16_t new_pos)
```

#### 4.17.2.12 cxt\_parameter\_id\_valid()

```
int cxt_parameter_id_valid (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid,  
    uint16_t ppid)
```

#### 4.17.2.13 cxt\_prepend\_transformer\_to\_profile()

```
int cxt_prepend_transformer_to_profile (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t type)
```

#### 4.17.2.14 cxt\_process()

```
int cxt_process (  
    m_eng_context * cxt)
```

#### 4.17.2.15 cxt\_profile\_id\_valid()

```
int cxt_profile_id_valid (  
    m_eng_context * cxt,  
    uint16_t pid)
```



**4.17.2.16 cxt\_remove\_transformer\_from\_profile()**

```
int cxt_remove_transformer_from_profile (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid)
```

**4.17.2.17 cxt\_run\_scheduled\_maintenance()**

```
int cxt_run_scheduled_maintenance (  
    m_eng_context * cxt)
```

**4.17.2.18 cxt\_set\_active\_profile()**

```
int cxt_set_active_profile (  
    m_eng_context * cxt,  
    uint16_t pid)
```

**4.17.2.19 cxt\_switch\_to\_profile()**

```
int cxt_switch_to_profile (  
    m_eng_context * cxt,  
    uint16_t pid)
```

**4.17.2.20 cxt\_transformer\_id\_valid()**

```
int cxt_transformer_id_valid (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid)
```

**4.17.2.21 cxt\_update\_parameter\_value\_by\_id()**

```
int cxt_update_parameter_value_by_id (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid,  
    uint16_t ppid,  
    float new_value)
```

**4.17.2.22 cxt\_update\_setting\_value\_by\_id()**

```
int cxt_update_setting_value_by_id (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid,  
    uint16_t sid,  
    uint16_t new_value)
```

#### 4.17.2.23 init\_m\_eng\_context()

```
int init_m_eng_context (
    m_eng_context * cxt)
```

#### 4.17.2.24 m\_eng\_context\_new\_profile()

```
int m_eng_context_new_profile (
    m_eng_context * cxt)
```

#### 4.17.2.25 m\_eng\_safe\_reboot()

```
void m_eng_safe_reboot (
    m_eng_context * cxt)
```

#### 4.17.2.26 reset\_context()

```
int reset_context (
    m_eng_context * cxt)
```

### 4.17.3 Variable Documentation

#### 4.17.3.1 global\_cxt

```
m_eng_context global_cxt [extern]
```

## 4.18 m\_eng\_context.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_CONTEXT_H_
00002 #define M_CONTEXT_H_
00003
00004 #include "m_eng_profile.h"
00005
00006 #define M_PROFILE_SWITCH_SAMPLES 256
00007 #define PROFILES_MALLOC_CHUNK_SIZE 16
00008 #define PROFILE_ARRAY_INITIAL_SIZE 64
00009
00010 #define SILENCE_ENERGY_THRESHOLD 2000
00011 #define SILENCE_BLOCKS_THRESHOLD 64
00012
00013 #define DECLICK_BUFSIZE 6
00014
00015 #define CLICK_SLOPE_THRESHOLD 0.5
00016
00017 #define DC_BLOCKER_ALPHA 0.999
00018
00019 // #define PRINT_TIMES
00020 // #define PRINT_BLOCKS
00021 // #define PRETTY_PRINT_BLOCKS
00022 // #define PRINT_PROFILE
00023
00024 typedef struct
00025 {
00026     uint16_t status_flags;
00027
00028     int n_profiles;
```

```

00029     int profile_array_size;
00030     m_eng_profile *profiles;
00031
00032     int active_profile;
00033     int profile_switch_triggered;
00034     int new_profile;
00035
00036     #ifdef GRAPH_PIPELINE
00037     m_eng_graph *unconfigured_pipeline;
00038     #endif
00039
00040     int profiles_switching;
00041     int profile_switch_progress;
00042     int profile_switch_samples;
00043     int profile_switch_type;
00044
00045     float declick_buffer[DECLICK_BUFSIZE];
00046
00047     int profile_maintenance_index;
00048     float prev_block[AUDIO_BLOCK_SAMPLES];
00049
00050     int runs;
00051
00052     m_eng_high_pass_filter_str output_hpf;
00053     m_eng_low_pass_filter_str input_lpf;
00054     m_transformer output_amp;
00055
00056     float dc_blocker_avg;
00057 } m_eng_context;
00058
00059 int init_m_eng_context(m_eng_context *cxt);
00060
00061 int m_eng_context_new_profile(m_eng_context *cxt);
00062
00063 extern m_eng_context global_cxt;
00064
00065 int cxt_set_active_profile(m_eng_context *cxt, uint16_t pid);
00066 int cxt_switch_to_profile (m_eng_context *cxt, uint16_t pid);
00067
00068 int cxt_profile_id_valid (m_eng_context *cxt, uint16_t pid);
00069 int cxt_transformer_id_valid(m_eng_context *cxt, uint16_t pid, uint16_t tid);
00070 int cxt_parameter_id_valid (m_eng_context *cxt, uint16_t pid, uint16_t tid, uint16_t ppid);
00071
00072 m_parameter *cxt_get_parameter_by_id(m_eng_context *cxt, uint16_t pid, uint16_t tid, uint16_t ppid);
00073 int cxt_update_parameter_value_by_id(m_eng_context *cxt, uint16_t pid, uint16_t tid,
uint16_t ppid, float new_value);
00074
00075 m_setting *cxt_get_setting_by_id(m_eng_context *cxt, uint16_t pid, uint16_t tid, uint16_t sid);
00076 int cxt_update_setting_value_by_id(m_eng_context *cxt, uint16_t pid, uint16_t tid, uint16_t
sid, uint16_t new_value);
00077
00078 int cxt_append_transformer_to_profile (m_eng_context *cxt, uint16_t pid, uint16_t type);
00079 int cxt_remove_transformer_from_profile(m_eng_context *cxt, uint16_t pid, uint16_t tid);
00080 int cxt_insert_transformer_to_profile (m_eng_context *cxt, uint16_t pid, uint16_t type, uint16_t pos);
00081 int cxt_prepend_transformer_to_profile(m_eng_context *cxt, uint16_t pid, uint16_t type);
00082
00083 int cxt_get_n_profile_transformers(m_eng_context *cxt, uint16_t pid);
00084 int cxt_get_n_transformer_params (m_eng_context *cxt, uint16_t pid, uint16_t tid);
00085 int cxt_get_n_transformer_settings (m_eng_context *cxt, uint16_t pid, uint16_t tid);
00086 int cxt_get_transformer_type (m_eng_context *cxt, uint16_t pid, uint16_t tid);
00087 int cxt_get_tid_by_pos (m_eng_context *cxt, uint16_t pid, uint16_t pos);
00088
00089 m_transformer *cxt_get_transformer_by_id(m_eng_context *cxt, uint16_t pid, uint16_t tid);
00090
00091 int cxt_move_transformer(m_eng_context *cxt, uint16_t tid, uint16_t new_pos);
00092
00093 int cxt_process(m_eng_context *cxt);
00094
00095 void m_eng_safe_reboot(m_eng_context *cxt);
00096 int reset_context(m_eng_context *cxt);
00097
00098 int cxt_run_scheduled_maintenance(m_eng_context *cxt);
00099
00100 #endif

```

## 4.19 m\_eng\_debugging.h File Reference

### Functions

- void [full\\_debug\\_print](#) (m\_eng\_context \*cxt)

- void [print\\_context\\_info](#) ([m\\_eng\\_context](#) \*cxt, int depth)
- void [print\\_profile\\_info](#) ([m\\_eng\\_profile](#) \*profile, int depth)
- void [print\\_pipeline\\_info](#) ([m\\_pipeline](#) \*pipeline, int depth)
- void [print\\_transformer\\_info](#) ([m\\_transformer](#) \*trans, int depth)

## 4.19.1 Function Documentation

### 4.19.1.1 full\_debug\_print()

```
void full_debug_print (
    m\_eng\_context * cxt)
```

### 4.19.1.2 print\_context\_info()

```
void print_context_info (
    m\_eng\_context * cxt,
    int depth)
```

### 4.19.1.3 print\_pipeline\_info()

```
void print_pipeline_info (
    m\_pipeline * pipeline,
    int depth)
```

### 4.19.1.4 print\_profile\_info()

```
void print_profile_info (
    m\_eng\_profile * profile,
    int depth)
```

### 4.19.1.5 print\_transformer\_info()

```
void print_transformer_info (
    m\_transformer * trans,
    int depth)
```

## 4.20 m\_eng\_debugging.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_DEBUGGING_H_
00002 #define M_DEBUGGING_H_
00003
00004 void full\_debug\_print(m\_eng\_context *cxt);
00005 void print\_context\_info(m\_eng\_context *cxt, int depth);
00006 void print\_profile\_info(m\_eng\_profile *profile, int depth);
00007 void print\_pipeline\_info(m\_pipeline *pipeline, int depth);
00008 void print\_transformer\_info(m\_transformer *trans, int depth);
00009
00010 #endif
```

## 4.21 m\_eng\_delay.h File Reference

### Data Structures

- struct [m\\_eng\\_delay\\_str](#)

### Functions

- int [init\\_delay\\_str](#) ([m\\_eng\\_delay\\_str](#) \*str)
- int [reconfigure\\_delay](#) (void \*data\_struct)
- int [calc\\_delay](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.21.1 Function Documentation

#### 4.21.1.1 calc\_delay()

```
int calc_delay (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

#### 4.21.1.2 init\_delay\_str()

```
int init_delay_str (
    m\_eng\_delay\_str * str)
```

#### 4.21.1.3 reconfigure\_delay()

```
int reconfigure_delay (
    void * data_struct)
```

## 4.22 m\_eng\_delay.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_DELAY_H_
00002 #define M_ENG_DELAY_H_
00003
00004 typedef struct
00005 {
00006     m\_setting tempo;
00007     m\_setting note;
00008     m\_parameter delay_gain;
00009
00010     float delay_samples;
00011     float g;
00012
00013     m\_delay\_buffer buf;
00014 } m\_eng\_delay\_str;
00015
00016 int init\_delay\_str(m\_eng\_delay\_str *str);
00017 int reconfigure\_delay(void *data_struct);
00018 int calc\_delay(void *data_struct, float *dest, float *src, int n_samples);
00019
00020 #endif
```

## 4.23 m\_eng\_delay\_buffer.h File Reference

### Data Structures

- struct [m\\_delay\\_buffer](#)

### Functions

- int [init\\_delay\\_buffer](#) ([m\\_delay\\_buffer](#) \*buf)
- int [m\\_delay\\_buffer\\_resize\\_samples](#) ([m\\_delay\\_buffer](#) \*buf, int n\_samples)
- int [m\\_delay\\_buffer\\_resize\\_seconds](#) ([m\\_delay\\_buffer](#) \*buf, float seconds)
- int [m\\_delay\\_buffer\\_resize\\_milliseconds](#) ([m\\_delay\\_buffer](#) \*buf, float ms)
- int [m\\_delay\\_buffer\\_tick](#) ([m\\_delay\\_buffer](#) \*buf, float new\_sample)
- int [m\\_delay\\_buffer\\_advance](#) ([m\\_delay\\_buffer](#) \*buf, float \*new\_samples, unsigned int n)
- int [m\\_delay\\_buffer\\_get\\_delayed\\_sample](#) ([m\\_delay\\_buffer](#) \*buf, float \*dest, unsigned int delay)
- float \* [m\\_delay\\_buffer\\_get\\_delayed\\_sample\\_ptr](#) ([m\\_delay\\_buffer](#) \*buf, unsigned int delay)
- int [m\\_delay\\_buffer\\_get\\_fractional\\_delayed\\_sample](#) ([m\\_delay\\_buffer](#) \*buf, float \*dest, float delay)

### 4.23.1 Function Documentation

#### 4.23.1.1 init\_delay\_buffer()

```
int init_delay_buffer (  
    m\_delay\_buffer * buf)
```

#### 4.23.1.2 m\_delay\_buffer\_advance()

```
int m_delay_buffer_advance (  
    m\_delay\_buffer * buf,  
    float * new_samples,  
    unsigned int n)
```

#### 4.23.1.3 m\_delay\_buffer\_get\_delayed\_sample()

```
int m_delay_buffer_get_delayed_sample (  
    m\_delay\_buffer * buf,  
    float * dest,  
    unsigned int delay)
```

#### 4.23.1.4 m\_delay\_buffer\_get\_delayed\_sample\_ptr()

```
float * m_delay_buffer_get_delayed_sample_ptr (  
    m\_delay\_buffer * buf,  
    unsigned int delay)
```

**4.23.1.5 m\_delay\_buffer\_get\_fractional\_delayed\_sample()**

```
int m_delay_buffer_get_fractional_delayed_sample (
    m_delay_buffer * buf,
    float * dest,
    float delay)
```

**4.23.1.6 m\_delay\_buffer\_resize\_milliseconds()**

```
int m_delay_buffer_resize_milliseconds (
    m_delay_buffer * buf,
    float ms)
```

**4.23.1.7 m\_delay\_buffer\_resize\_samples()**

```
int m_delay_buffer_resize_samples (
    m_delay_buffer * buf,
    int n_samples)
```

**4.23.1.8 m\_delay\_buffer\_resize\_seconds()**

```
int m_delay_buffer_resize_seconds (
    m_delay_buffer * buf,
    float seconds)
```

**4.23.1.9 m\_delay\_buffer\_tick()**

```
int m_delay_buffer_tick (
    m_delay_buffer * buf,
    float new_sample)
```

**4.24 m\_eng\_delay\_buffer.h**

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_DELAY_BUFFER_H_
00002 #define M_ENG_DELAY_BUFFER_H_
00003
00004 typedef struct
00005 {
00006     int valid;
00007
00008     int pos;
00009     int index;
00010
00011     int n_buffers;
00012     int buffer_array_size;
00013     float **buffers;
00014 } m_delay_buffer;
00015
00016 int init_delay_buffer(m_delay_buffer *buf);
00017
00018 int m_delay_buffer_resize_samples(m_delay_buffer *buf, int n_samples);
00019 int m_delay_buffer_resize_seconds(m_delay_buffer *buf, float seconds);
00020 int m_delay_buffer_resize_milliseconds(m_delay_buffer *buf, float ms);
00021
00022 int m_delay_buffer_tick(m_delay_buffer *buf, float new_sample);
00023 int m_delay_buffer_advance(m_delay_buffer *buf, float *new_samples, unsigned int n);
00024
00025 int m_delay_buffer_get_delayed_sample(m_delay_buffer *buf, float *dest, unsigned int delay);
00026 float *m_delay_buffer_get_delayed_sample_ptr(m_delay_buffer *buf, unsigned int delay);
00027 int m_delay_buffer_get_fractional_delayed_sample(m_delay_buffer *buf, float *dest, float delay);
00028
00029 #endif
```

## 4.25 m\_eng\_dirty\_octave.h File Reference

### Data Structures

- struct [m\\_eng\\_dirty\\_octave\\_str](#)

### Functions

- int [init\\_dirty\\_octave\\_str](#) ([m\\_eng\\_dirty\\_octave\\_str](#) \*str)
- int [reconfigure\\_dirty\\_octave](#) (void \*data\_struct)
- int [calc\\_dirty\\_octave](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.25.1 Function Documentation

#### 4.25.1.1 calc\_dirty\_octave()

```
int calc_dirty_octave (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

#### 4.25.1.2 init\_dirty\_octave\_str()

```
int init_dirty_octave_str (
    m\_eng\_dirty\_octave\_str * str)
```

#### 4.25.1.3 reconfigure\_dirty\_octave()

```
int reconfigure_dirty_octave (
    void * data_struct)
```

## 4.26 m\_eng\_dirty\_octave.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_DIRTY_OCTAVE_H_
00002 #define M_ENG_DIRTY_OCTAVE_H_
00003
00004 typedef struct
00005 {
00006     m_parameter fuzz;
00007
00008     float dc_average;
00009
00010     float lpf_alpha;
00011     float last_out_sample;
00012 } m_eng_dirty_octave_str;
00013
00014 int init_dirty_octave_str(m_eng_dirty_octave_str *str);
00015 int reconfigure_dirty_octave(void *data_struct);
00016 int calc_dirty_octave(void *data_struct, float *dest, float *src, int n_samples);
00017
00018 #endif
```



## 4.27 m\_eng\_distortion.h File Reference

### Data Structures

- struct [m\\_eng\\_distortion\\_str](#)

### Macros

- #define [USE\\_GLOBAL\\_TEMP\\_BUFFERS](#)
- #define [M\\_DISTORTION\\_CLIP](#) 0
- #define [M\\_DISTORTION\\_TANH](#) 1
- #define [M\\_DISTORTION\\_ARCTAN](#) 2
- #define [M\\_DISTORTION\\_FOLD](#) 3

### Functions

- int [init\\_distortion\\_str](#) ([m\\_eng\\_distortion\\_str](#) \*str)
- int [reconfigure\\_distortion](#) (void \*data\_struct)
- int [calc\\_distortion](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.27.1 Macro Definition Documentation

#### 4.27.1.1 M\_DISTORTION\_ARCTAN

```
#define M_DISTORTION_ARCTAN 2
```

#### 4.27.1.2 M\_DISTORTION\_CLIP

```
#define M_DISTORTION_CLIP 0
```

#### 4.27.1.3 M\_DISTORTION\_FOLD

```
#define M_DISTORTION_FOLD 3
```

#### 4.27.1.4 M\_DISTORTION\_TANH

```
#define M_DISTORTION_TANH 1
```

#### 4.27.1.5 USE\_GLOBAL\_TEMP\_BUFFERS

```
#define USE_GLOBAL_TEMP_BUFFERS
```

## 4.27.2 Function Documentation

### 4.27.2.1 calc\_distortion()

```
int calc_distortion (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

### 4.27.2.2 init\_distortion\_str()

```
int init_distortion_str (
    m_eng_distortion_str * str)
```

### 4.27.2.3 reconfigure\_distortion()

```
int reconfigure_distortion (
    void * data_struct)
```

## 4.28 m\_eng\_distortion.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_DISTORTION_H_
00002 #define M_DISTORTION_H_
00003
00004 #define USE_GLOBAL_TEMP_BUFFERS
00005
00006 #define M_DISTORTION_CLIP 0
00007 #define M_DISTORTION_TANH 1
00008 #define M_DISTORTION_ARCTAN 2
00009 #define M_DISTORTION_FOLD 3
00010
00011 typedef struct
00012 {
00013     m_setting function;
00014
00015     m_lr_low_pass_filter_str low_pass;
00016
00017     m_eng_waveshaper_str dist;
00018
00019     m_parameter wet_mix;
00020     m_parameter bass_mix;
00021     m_parameter bass_cutoff;
00022 } m_eng_distortion_str;
00023
00024 int init_distortion_str(m_eng_distortion_str *str);
00025 int reconfigure_distortion(void *data_struct);
00026 int calc_distortion(void *data_struct, float *dest, float *src, int n_samples);
00027
00028 #endif
```

## 4.29 m\_eng\_envelope.h File Reference

### Data Structures

- struct [m\\_eng\\_envelope\\_str](#)

## Functions

- int [init\\_envelope\\_str](#) (m\_eng\_envelope\_str \*str)
- int [reconfigure\\_envelope](#) (void \*data\_struct)
- int [calc\\_envelope](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

## 4.29.1 Function Documentation

### 4.29.1.1 calc\_envelope()

```
int calc_envelope (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

### 4.29.1.2 init\_envelope\_str()

```
int init_envelope_str (
    m_eng_envelope_str * str)
```

### 4.29.1.3 reconfigure\_envelope()

```
int reconfigure_envelope (
    void * data_struct)
```

## 4.30 m\_eng\_envelope.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_ENVELOPE_H_
00002 #define M_ENG_ENVELOPE_H_
00003
00004 typedef struct
00005 {
00006     m_parameter min_center;
00007     m_parameter max_center;
00008     m_parameter width;
00009     m_parameter speed;
00010     m_parameter sensitivity;
00011     m_parameter smoothness;
00012
00013     float alpha;
00014     float e;
00015
00016     int chunk_size;
00017
00018     m_eng_band_pass_filter_str filter;
00019 } m_eng_envelope_str;
00020
00021 int init_envelope_str(m_eng_envelope_str *str);
00022 int reconfigure_envelope(void *data_struct);
00023 int calc_envelope(void *data_struct, float *dest, float *src, int n_samples);
00024
00025 #endif
```

## 4.31 m\_eng\_equaliser.h File Reference

### Data Structures

- struct [m\\_eng\\_3\\_band\\_eq\\_str](#)

### Macros

- `#define M_ENG_EQ_CONTROL_DIRECT 0`
- `#define M_ENG_EQ_CONTROL_DB_GAIN 1`

### Functions

- int [init\\_3\\_band\\_eq\\_str](#) ([m\\_eng\\_3\\_band\\_eq\\_str](#) \*str)
- int [reconfigure\\_3\\_band\\_eq](#) (void \*data\_struct)
- int [calc\\_3\\_band\\_eq](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.31.1 Macro Definition Documentation

#### 4.31.1.1 M\_ENG\_EQ\_CONTROL\_DB\_GAIN

```
#define M_ENG_EQ_CONTROL_DB_GAIN 1
```

#### 4.31.1.2 M\_ENG\_EQ\_CONTROL\_DIRECT

```
#define M_ENG_EQ_CONTROL_DIRECT 0
```

### 4.31.2 Function Documentation

#### 4.31.2.1 calc\_3\_band\_eq()

```
int calc_3_band_eq (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.31.2.2 init\_3\_band\_eq\_str()

```
int init_3_band_eq_str (  
    m\_eng\_3\_band\_eq\_str * str)
```

#### 4.31.2.3 reconfigure\_3\_band\_eq()

```
int reconfigure_3_band_eq (  
    void * data_struct)
```

## 4.32 m\_eng\_equaliser.h

[Go to the documentation of this file.](#)

```

00001 #ifndef M_ENG_EQUALISER_H_
00002 #define M_ENG_EQUALISER_H_
00003
00004 #define M_ENG_EQ_CONTROL_DIRECT    0
00005 #define M_ENG_EQ_CONTROL_DB_GAIN  1
00006
00007 typedef struct
00008 {
00009     m_parameter low;
00010     m_parameter mid;
00011     m_parameter high;
00012
00013     float coefs[3];
00014     int control_mode;
00015
00016     m_lr_low_pass_filter_str filters[2];
00017 } m_eng_3_band_eq_str;
00018
00019 int init_3_band_eq_str(m_eng_3_band_eq_str *str);
00020 int reconfigure_3_band_eq(void *data_struct);
00021 int calc_3_band_eq(void *data_struct, float *dest, float *src, int n_samples);
00022
00023 #endif

```

## 4.33 m\_eng\_flanger.h File Reference

### Data Structures

- struct [m\\_eng\\_flanger\\_str](#)

### Functions

- int [init\\_flanger\\_str](#) ([m\\_eng\\_flanger\\_str](#) \*str)
- int [reconfigure\\_flanger](#) (void \*data\_struct)
- int [calc\\_flanger](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.33.1 Function Documentation

#### 4.33.1.1 calc\_flanger()

```

int calc_flanger (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)

```

#### 4.33.1.2 init\_flanger\_str()

```

int init_flanger_str (
    m_eng_flanger_str * str)

```

#### 4.33.1.3 reconfigure\_flanger()

```
int reconfigure_flanger (
    void * data_struct)
```

### 4.34 m\_eng\_flanger.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_FLANGER_H_
00002 #define M_ENG_FLANGER_H_
00003
00004 typedef struct
00005 {
00006     m_parameter range;
00007     m_parameter tempo;
00008     m_parameter depth;
00009     m_parameter mix;
00010
00011     m_setting note;
00012
00013     float r;
00014     float s;
00015     float d;
00016
00017     float wet_mix;
00018     float dry_mix;
00019
00020     float period;
00021
00022     float t;
00023
00024     m_delay_buffer buf;
00025 } m_eng_flanger_str;
00026
00027 int init_flanger_str(m_eng_flanger_str *str);
00028 int reconfigure_flanger(void *data_struct);
00029 int calc_flanger(void *data_struct, float *dest, float *src, int n_samples);
00030
00031 #endif
```

### 4.35 m\_eng\_flops.h File Reference

#### Macros

- #define [RESTRICT](#)

#### 4.35.1 Macro Definition Documentation

##### 4.35.1.1 RESTRICT

```
#define RESTRICT
```

## 4.36 m\_eng\_flops.h

[Go to the documentation of this file.](#)

```

00001 #ifndef M_ENG_FLOPS_H_
00002 #define M_ENG_FLOPS_H_
00003
00004 #ifndef RESTRICT
00005 #   if defined(__STDC_VERSION__) && __STDC_VERSION__ >= 199901L
00006 #       define RESTRICT restrict
00007 #   elif defined(__GNUC__) || defined(__clang__)
00008 #       define RESTRICT __restrict__
00009 #   else
00010 #       define RESTRICT
00011 #   endif
00012 #endif
00013
00014 static inline void m_add_f32(float *dest, float *RESTRICT x, float *RESTRICT y, uint32_t n)
00015 {
00016     if (dest && x && y)
00017     {
00018         #ifdef NO_CMSIS
00019             int i = 0;
00020             for (; i + 3 < n; i += 4)
00021             {
00022                 dest[i] = x[i] + y[i];
00023                 dest[i + 1] = x[i + 1] + y[i + 1];
00024                 dest[i + 2] = x[i + 2] + y[i + 2];
00025                 dest[i + 3] = x[i + 3] + y[i + 3];
00026             }
00027             for (; i < n; i++) dest[i] = x[i] + y[i];
00028         #else
00029             arm_add_f32(x, y, dest, n);
00030         #endif
00031     }
00032 }
00033
00034 static inline void m_sub_f32(float *dest, float *RESTRICT x, float *RESTRICT y, uint32_t n)
00035 {
00036     if (dest && x && y)
00037     {
00038         #ifdef NO_CMSIS
00039             int i = 0;
00040             for (; i + 3 < n; i += 4)
00041             {
00042                 dest[i] = x[i] - y[i];
00043                 dest[i + 1] = x[i + 1] - y[i + 1];
00044                 dest[i + 2] = x[i + 2] - y[i + 2];
00045                 dest[i + 3] = x[i + 3] - y[i + 3];
00046             }
00047             for (; i < n; i++) dest[i] = x[i] - y[i];
00048         #else
00049             arm_sub_f32(x, y, dest, n);
00050         #endif
00051     }
00052 }
00053
00054
00055 static inline void m_add_to_f32(float *dest, float *x, uint32_t n)
00056 {
00057     if (dest && x)
00058     {
00059         #ifdef NO_CMSIS
00060             int i = 0;
00061             for (; i + 3 < n; i += 4)
00062             {
00063                 dest[i] += x[i];
00064                 dest[i + 1] += x[i + 1];
00065                 dest[i + 2] += x[i + 2];
00066                 dest[i + 3] += x[i + 3];
00067             }
00068             for (; i < n; i++) dest[i] += x[i];
00069         #else
00070             arm_add_f32(dest, x, dest, n);
00071         #endif
00072     }
00073 }
00074
00075 static inline void m_sub_from_f32(float *dest, float *x, uint32_t n)
00076 {
00077     if (dest && x)
00078     {
00079         #ifdef NO_CMSIS
00080             int i = 0;
00081             for (; i + 3 < n; i += 4)
00082             {

```

```

00083         dest[i] -= x[i];
00084         dest[i + 1] -= x[i + 1];
00085         dest[i + 2] -= x[i + 2];
00086         dest[i + 3] -= x[i + 3];
00087     }
00088     for (; i < n; i++) dest[i] -= x[i];
00089     #else
00090     arm_sub_f32(dest, x, dest, n);
00091     #endif
00092 }
00093 }
00094
00095
00096 static inline void m_mul_and_add_f32(float *dest, float *RESTRICT x, float *RESTRICT y, uint32_t n)
00097 {
00098     if (dest && x && y)
00099     {
00100         #ifdef NO_CMSIS
00101         int i = 0;
00102         for (; i + 3 < n; i += 4)
00103         {
00104             dest[i] += x[i] * y[i];
00105             dest[i + 1] += x[i + 1] * y[i + 1];
00106             dest[i + 2] += x[i + 2] * y[i + 2];
00107             dest[i + 3] += x[i + 3] * y[i + 3];
00108         }
00109         for (; i < n; i++) dest[i] += x[i] * y[i];
00110         #else
00111         float temp[n];
00112         arm_mult_f32(x, y, temp, n);
00113         arm_add_f32(dest, temp, dest, n);
00114         #endif
00115     }
00116 }
00117
00118 static inline void m_scale_and_add_to_f32(float *RESTRICT dest, float *RESTRICT x, float s, uint32_t
n)
00119 {
00120     if (dest && x)
00121     {
00122         #ifdef NO_CMSIS
00123         int i = 0;
00124         for (; i + 3 < n; i += 4)
00125         {
00126             dest[i] += x[i] * s;
00127             dest[i + 1] += x[i + 1] * s;
00128             dest[i + 2] += x[i + 2] * s;
00129             dest[i + 3] += x[i + 3] * s;
00130         }
00131         for (; i < n; i++) dest[i] += x[i] * s;
00132         #else
00133         float temp[n];
00134         arm_scale_f32(x, s, temp, n);
00135         arm_add_f32(dest, temp, dest, n);
00136         #endif
00137     }
00138 }
00139
00140
00141 static inline void m_scale_f32(float *dest, float *x, float s, uint32_t n)
00142 {
00143     if (dest && x)
00144     {
00145         #ifdef NO_CMSIS
00146         int i = 0;
00147         for (; i + 3 < n; i += 4)
00148         {
00149             dest[i] = x[i] * s;
00150             dest[i + 1] = x[i + 1] * s;
00151             dest[i + 2] = x[i + 2] * s;
00152             dest[i + 3] = x[i + 3] * s;
00153         }
00154         for (; i < n; i++) dest[i] = x[i] * s;
00155         #else
00156         arm_scale_f32(x, s, dest, n);
00157         #endif
00158     }
00159 }
00160
00161 static inline void m_scale_in_place_f32(float *x, float s, uint32_t n)
00162 {
00163     if (x)
00164     {
00165         #ifdef NO_CMSIS
00166         int i = 0;
00167         for (; i + 3 < n; i += 4)
00168         {

```



```

00169         x[i    ] *= s;
00170         x[i + 1] *= s;
00171         x[i + 2] *= s;
00172         x[i + 3] *= s;
00173     }
00174     for (; i < n; i++) x[i] *= s;
00175 #else
00176     arm_scale_f32(x, s, x, n);
00177 #endif
00178 }
00179 }
00180
00181 static inline void m_linterpolate_f32(float *dest, float *x, float *y, float t, uint32_t n)
00182 {
00183     if (dest && x && y)
00184     {
00185         if (t <= 0.0)
00186         {
00187             memcpy(dest, x, n * sizeof(float));
00188         }
00189         if (t >= 1.0)
00190         {
00191             memcpy(dest, y, n * sizeof(float));
00192         }
00193         else
00194         {
00195             int i = 0;
00196             for (; i + 3 < n; i += 4)
00197             {
00198                 dest[i    ] = (1.0 - t) * x[i    ] + t * y[i    ];
00199                 dest[i + 1] = (1.0 - t) * x[i + 1] + t * y[i + 1];
00200                 dest[i + 2] = (1.0 - t) * x[i + 2] + t * y[i + 2];
00201                 dest[i + 3] = (1.0 - t) * x[i + 3] + t * y[i + 3];
00202             }
00203             for (; i < n; i++) dest[i] = (1.0 - t) * x[i] + t * y[i];
00204         }
00205     }
00206 }
00207
00208 static inline void m_mix_in_f32(float *x, float *y, float t, uint32_t n)
00209 {
00210     if (x && y)
00211     {
00212         if (t <= 0.0)
00213         {
00214             return;
00215         }
00216         if (t >= 1.0)
00217         {
00218             memcpy(x, y, n * sizeof(float));
00219         }
00220         else
00221         {
00222             int i = 0;
00223             for (; i + 3 < n; i += 4)
00224             {
00225                 x[i    ] = (1.0 - t) * x[i    ] + t * y[i    ];
00226                 x[i + 1] = (1.0 - t) * x[i + 1] + t * y[i + 1];
00227                 x[i + 2] = (1.0 - t) * x[i + 2] + t * y[i + 2];
00228                 x[i + 3] = (1.0 - t) * x[i + 3] + t * y[i + 3];
00229             }
00230             for (; i < n; i++) x[i] = (1.0 - t) * x[i] + t * y[i];
00231         }
00232     }
00233 }
00234
00235 #endif

```

## 4.37 m\_eng\_globals.h File Reference

### Functions

- void [update\\_upper\\_cycles](#) ()
- double [cycles\\_to\\_seconds](#) (uint64\_t cycles)
- uint64\_t [current\\_cycle](#) ()

## Variables

- uint16\_t [cpu\\_cycles\\_total](#)
- uint16\_t [cpu\\_cycles\\_total\\_max](#)
- uint16\_t [memory\\_used](#)
- uint16\_t [memory\\_used\\_max](#)
- int [update\\_scheduled](#)
- uint32\_t [trace\\_depth](#)

## 4.37.1 Function Documentation

### 4.37.1.1 [current\\_cycle\(\)](#)

```
uint64_t current_cycle () [inline]
```

### 4.37.1.2 [cycles\\_to\\_seconds\(\)](#)

```
double cycles_to_seconds (  
    uint64_t cycles)
```

### 4.37.1.3 [update\\_upper\\_cycles\(\)](#)

```
void update_upper_cycles ()
```

## 4.37.2 Variable Documentation

### 4.37.2.1 [cpu\\_cycles\\_total](#)

```
uint16_t cpu_cycles_total [extern]
```

### 4.37.2.2 [cpu\\_cycles\\_total\\_max](#)

```
uint16_t cpu_cycles_total_max [extern]
```

### 4.37.2.3 [memory\\_used](#)

```
uint16_t memory_used [extern]
```

### 4.37.2.4 [memory\\_used\\_max](#)

```
uint16_t memory_used_max [extern]
```

#### 4.37.2.5 trace\_depth

```
uint32_t trace_depth [extern]
```

#### 4.37.2.6 update\_scheduled

```
int update_scheduled [extern]
```

## 4.38 m\_eng\_globals.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_GLOBAL_VARS_H_
00002 #define M_GLOBAL_VARS_H_
00003
00004 extern uint16_t cpu_cycles_total;
00005 extern uint16_t cpu_cycles_total_max;
00006 extern uint16_t memory_used;
00007 extern uint16_t memory_used_max;
00008
00009 extern int update_scheduled;
00010
00011 extern uint32_t trace_depth;
00012
00013 void update_upper_cycles();
00014
00015 double cycles_to_seconds(uint64_t cycles);
00016
00017 uint64_t current_cycle();
00018
00019 #endif
```

## 4.39 m\_eng\_i2s\_dma.h File Reference

### Typedefs

- typedef int16\_t [raw\\_sample\\_t](#)

### Functions

- void [init\\_i2s\\_dma](#) ()
- void [m\\_eng\\_i2s\\_input\\_isr](#) ()
- void [m\\_eng\\_i2s\\_output\\_isr](#) ()
- void [i2s\\_input\\_update](#) ()
- void [i2s\\_output\\_update](#) ()
- void [i2s\\_output\\_transmit\\_mono\\_int](#) ([raw\\_sample\\_t](#) \*block)
- void [i2s\\_output\\_transmit\\_mono\\_float](#) (float \*block)

### Variables

- [raw\\_sample\\_t](#) [i2s\\_input\\_blocks](#) [2][[AUDIO\\_BLOCK\\_SAMPLES](#)]

### 4.39.1 Typedef Documentation

#### 4.39.1.1 raw\_sample\_t

```
typedef int16_t raw_sample_t
```

### 4.39.2 Function Documentation

#### 4.39.2.1 i2s\_input\_update()

```
void i2s_input_update ()
```

#### 4.39.2.2 i2s\_output\_transmit\_mono\_float()

```
void i2s_output_transmit_mono_float (  
    float * block)
```

#### 4.39.2.3 i2s\_output\_transmit\_mono\_int()

```
void i2s_output_transmit_mono_int (  
    raw_sample_t * block)
```

#### 4.39.2.4 i2s\_output\_update()

```
void i2s_output_update ()
```

#### 4.39.2.5 init\_i2s\_dma()

```
void init_i2s_dma ()
```

#### 4.39.2.6 m\_eng\_i2s\_input\_isr()

```
void m_eng_i2s_input_isr ()
```

#### 4.39.2.7 m\_eng\_i2s\_output\_isr()

```
void m_eng_i2s_output_isr ()
```

### 4.39.3 Variable Documentation

#### 4.39.3.1 i2s\_input\_blocks

```
raw_sample_t i2s_input_blocks[2][AUDIO_BLOCK_SAMPLES] [extern]
```

## 4.40 m\_eng\_i2s\_dma.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_I2S_DMA_H_
00002 #define M_I2S_DMA_H_
00003
00004 typedef int16_t raw_sample_t;
00005
00006 void init_i2s_dma();
00007
00008 void m_eng_i2s_input_isr();
00009 void m_eng_i2s_output_isr();
00010
00011 void i2s_input_update();
00012 void i2s_output_update();
00013
00014 void i2s_output_transmit_mono_int (raw_sample_t *block);
00015 void i2s_output_transmit_mono_float(float *block);
00016
00017 extern raw_sample_t i2s_input_blocks[2][AUDIO_BLOCK_SAMPLES];
00018
00019 #endif
```

## 4.41 m\_eng\_linkowitz\_riley.h File Reference

### Data Structures

- struct [m\\_lr\\_low\\_pass\\_filter\\_str](#)
- struct [m\\_lr\\_high\\_pass\\_filter\\_str](#)

### Functions

- int [init\\_lr\\_low\\_pass\\_filter\\_str](#) ([m\\_lr\\_low\\_pass\\_filter\\_str](#) \*str)
- int [reconfigure\\_lr\\_low\\_pass\\_filter](#) (void \*data\_struct)
- int [calc\\_lr\\_low\\_pass\\_filter](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- int [init\\_lr\\_high\\_pass\\_filter\\_str](#) ([m\\_lr\\_high\\_pass\\_filter\\_str](#) \*str)
- int [reconfigure\\_lr\\_high\\_pass\\_filter](#) (void \*data\_struct)
- int [calc\\_lr\\_high\\_pass\\_filter](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.41.1 Function Documentation

#### 4.41.1.1 calc\_lr\_high\_pass\_filter()

```
int calc_lr_high_pass_filter (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

#### 4.41.1.2 calc\_lr\_low\_pass\_filter()

```
int calc_lr_low_pass_filter (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

#### 4.41.1.3 init\_lr\_high\_pass\_filter\_str()

```
int init_lr_high_pass_filter_str (
    m_lr_high_pass_filter_str * str)
```

#### 4.41.1.4 init\_lr\_low\_pass\_filter\_str()

```
int init_lr_low_pass_filter_str (
    m_lr_low_pass_filter_str * str)
```

#### 4.41.1.5 reconfigure\_lr\_high\_pass\_filter()

```
int reconfigure_lr_high_pass_filter (
    void * data_struct)
```

#### 4.41.1.6 reconfigure\_lr\_low\_pass\_filter()

```
int reconfigure_lr_low_pass_filter (
    void * data_struct)
```

## 4.42 m\_eng\_linkowitz\_riley.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_LR_FILTER_H_
00002 #define M_ENG_LR_FILTER_H_
00003
00004 typedef struct m_lr_low_pass_filter_str
00005 {
00006     m_parameter cutoff_frequency;
00007
00008     float a0, a1, a2, a3, a4;
00009     float x_11, x_12, y_11, y_12;
00010     float x_21, x_22, y_21, y_22;
00011 } m_lr_low_pass_filter_str;
00012
00013 int init_lr_low_pass_filter_str(m_lr_low_pass_filter_str *str);
00014 int reconfigure_lr_low_pass_filter(void *data_struct);
00015 int calc_lr_low_pass_filter(void *data_struct, float *dest, float *src, int n_samples);
00016
00017 typedef struct m_lr_high_pass_filter_str
00018 {
00019     m_parameter cutoff_frequency;
00020
00021     float a0, a1, a2, a3, a4;
00022     float x_11, x_12, y_11, y_12;
00023     float x_21, x_22, y_21, y_22;
00024 } m_lr_high_pass_filter_str;
00025
00026 int init_lr_high_pass_filter_str(m_lr_high_pass_filter_str *str);
00027 int reconfigure_lr_high_pass_filter(void *data_struct);
00028 int calc_lr_high_pass_filter(void *data_struct, float *dest, float *src, int n_samples);
00029
00030 #endif
```

## 4.43 m\_eng\_logging.h File Reference

### Data Structures

- struct [m\\_eng\\_log\\_entry](#)
- struct [m\\_eng\\_profiler\\_entry](#)

### Macros

- #define [M\\_ENG\\_TRACE\\_FUNCTION\\_ENTER](#) 0
- #define [M\\_ENG\\_TRACE\\_FUNCTION\\_RETURN](#) 1
- #define [M\\_ENG\\_LOG\\_ENTRY\\_ERROR](#) 2
- #define [M\\_ENG\\_LOG\\_ENTRY\\_RETURN\\_ERR](#) 3
- #define [M\\_ENG\\_LOG\\_ENTRY\\_RETURN\\_PTR](#) 4
- #define [M\\_ENG\\_LOG\\_ENTRY\\_RETURN\\_INT](#) 5
- #define [M\\_ENG\\_LOG\\_ENTRY\\_RETURN](#) 6
- #define [M\\_ENG\\_LOG\\_ENTRY\\_MESSAGE](#) 7
- #define [M\\_ENG\\_LOG\\_ERRORS](#) 0b00001
- #define [M\\_ENG\\_LOG\\_TRACE](#) 0b00010
- #define [M\\_ENG\\_LOG\\_RETURNS](#) 0b00100
- #define [M\\_ENG\\_LOG\\_SPIKES](#) 0b01000
- #define [M\\_ENG\\_LOG\\_EVERYTHING](#) ([M\\_ENG\\_LOG\\_ERRORS](#) | [M\\_ENG\\_LOG\\_TRACE](#) | [M\\_ENG\\_LOG\\_RETURNS](#) | [M\\_ENG\\_LOG\\_SPIKES](#))
- #define [STR](#)(x)
- #define [XSTR](#)(x)
- #define [M\\_ENG\\_LOG\\_LEVEL](#) ([M\\_ENG\\_LOG\\_ERRORS](#))
- #define [M\\_ENG\\_LOG\\_INDENT\\_TRACE](#)
- #define [M\\_ENG\\_LOG\\_ERROR](#)(x)
- #define [M\\_ENG\\_PROFILER\\_LOG\\_ENTRY](#)()
- #define [M\\_ENG\\_PROFILER\\_LOG\\_RETURN](#)()
- #define [M\\_ENG\\_TRACE\\_LOG\\_ENTRY](#)()
- #define [M\\_ENG\\_TRACE\\_LOG\\_RETURN](#)()
- #define [M\\_ENG\\_LOG\\_RETURN\\_ERR\\_CODE](#)(x)
- #define [M\\_ENG\\_LOG\\_RETURN\\_PTR](#)(x)
- #define [M\\_ENG\\_LOG\\_RETURN\\_INT](#)(x)
- #define [M\\_ENG\\_LOG\\_RETURN](#)(x)
- #define [FUNCTION\\_START](#)()
- #define [RETURN\\_VOID](#)()
- #define [RETURN\\_ERR\\_CODE](#)(x)
- #define [RETURN\\_NEG\\_ERR\\_CODE](#)(x)
- #define [RETURN\\_PTR](#)(x)
- #define [RETURN\\_INT](#)(x)
- #define [RETURN](#)(x)
- #define [M\\_LOG](#)(fmt, ...)
- #define [M\\_LOG\\_ERROR](#)(x)
- #define [CYCLES\\_TO\\_SECONDS](#)(x)
- #define [SECONDS\\_TO\\_CYCLES](#)(x)

## Functions

- void [m\\_eng\\_log\\_message](#) (const char \*fname, const char \*line, const char \*function, uint64\_t cycle, const char \*fmt,...)
- void [m\\_eng\\_log](#) (const char \*fmt,...)
- void [m\\_eng\\_print\\_log](#) ()
- void [m\\_eng\\_print\\_flush\\_log](#) ()
- void [m\\_eng\\_trace\\_log\\_begin](#) (const char \*fname, const char \*line, const char \*function, int local\_trace\_↵ depth, uint64\_t cycle)
- void [m\\_eng\\_trace\\_log\\_return](#) (const char \*fname, const char \*line, const char \*function, int local\_trace\_↵ depth, uint64\_t cycle)
- void [m\\_eng\\_log\\_error\\_code](#) (const char \*fname, const char \*line, const char \*function, int error\_code, uint64\_t cycle)
- void [m\\_eng\\_log\\_return\\_err](#) (const char \*fname, const char \*line, const char \*function, int error\_code, uint64\_t cycle)
- void [m\\_eng\\_log\\_return\\_ptr](#) (const char \*fname, const char \*line, const char \*function, void \*ptr, uint64\_t cycle)
- void [m\\_eng\\_log\\_return\\_int](#) (const char \*fname, const char \*line, const char \*function, int val, uint64\_t cycle)
- void [m\\_eng\\_log\\_return\\_](#) (const char \*fname, const char \*line, const char \*function, uint64\_t cycle)

### 4.43.1 Macro Definition Documentation

#### 4.43.1.1 CYCLES\_TO\_SECONDS

```
#define CYCLES_TO_SECONDS(  
    x)
```

**Value:**

```
((double) (x) * 1.666666667e-9)
```

#### 4.43.1.2 FUNCTION\_START

```
#define FUNCTION_START()
```

**Value:**

```
M_ENG_PROFILER_LOG_ENTRY(); M_ENG_TRACE_LOG_ENTRY();
```

#### 4.43.1.3 M\_ENG\_LOG\_ENTRY\_ERROR

```
#define M_ENG_LOG_ENTRY_ERROR 2
```

#### 4.43.1.4 M\_ENG\_LOG\_ENTRY\_MESSAGE

```
#define M_ENG_LOG_ENTRY_MESSAGE 7
```

#### 4.43.1.5 M\_ENG\_LOG\_ENTRY\_RETURN

```
#define M_ENG_LOG_ENTRY_RETURN 6
```



#### 4.43.1.6 M\_ENG\_LOG\_ENTRY\_RETURN\_ERR

```
#define M_ENG_LOG_ENTRY_RETURN_ERR 3
```

#### 4.43.1.7 M\_ENG\_LOG\_ENTRY\_RETURN\_INT

```
#define M_ENG_LOG_ENTRY_RETURN_INT 5
```

#### 4.43.1.8 M\_ENG\_LOG\_ENTRY\_RETURN\_PTR

```
#define M_ENG_LOG_ENTRY_RETURN_PTR 4
```

#### 4.43.1.9 M\_ENG\_LOG\_ERROR

```
#define M_ENG_LOG_ERROR(  
    x)
```

**Value:**

```
if (x != NO_ERROR) \  
    m_eng_log_error_code(FNAME, XSTR(__LINE__), __func__, x, current_cycle());
```

#### 4.43.1.10 M\_ENG\_LOG\_ERRORS

```
#define M_ENG_LOG_ERRORS 0b00001
```

#### 4.43.1.11 M\_ENG\_LOG EVERYTHING

```
#define M_ENG_LOG EVERYTHING (M_ENG_LOG_ERRORS | M_ENG_LOG_TRACE | M_ENG_LOG_RETURNS | M_ENG_LOG_SPIKES)
```

#### 4.43.1.12 M\_ENG\_LOG\_INDENT\_TRACE

```
#define M_ENG_LOG_INDENT_TRACE
```

#### 4.43.1.13 M\_ENG\_LOG\_LEVEL

```
#define M_ENG_LOG_LEVEL (M_ENG_LOG_ERRORS)
```

#### 4.43.1.14 M\_ENG\_LOG\_RETURN

```
#define M_ENG_LOG_RETURN(  
    x)
```

**4.43.1.15 M\_ENG\_LOG\_RETURN\_ERR\_CODE**

```
#define M_ENG_LOG_RETURN_ERR_CODE(  
    x)
```

**4.43.1.16 M\_ENG\_LOG\_RETURN\_INT**

```
#define M_ENG_LOG_RETURN_INT(  
    x)
```

**4.43.1.17 M\_ENG\_LOG\_RETURN\_PTR**

```
#define M_ENG_LOG_RETURN_PTR(  
    x)
```

**4.43.1.18 M\_ENG\_LOG\_RETURNS**

```
#define M_ENG_LOG_RETURNS 0b00100
```

**4.43.1.19 M\_ENG\_LOG\_SPIKES**

```
#define M_ENG_LOG_SPIKES 0b01000
```

**4.43.1.20 M\_ENG\_LOG\_TRACE**

```
#define M_ENG_LOG_TRACE 0b00010
```

**4.43.1.21 M\_ENG\_PROFILER\_LOG\_ENTRY**

```
#define M_ENG_PROFILER_LOG_ENTRY()
```

**4.43.1.22 M\_ENG\_PROFILER\_LOG\_RETURN**

```
#define M_ENG_PROFILER_LOG_RETURN()
```

**4.43.1.23 M\_ENG\_TRACE\_FUNCTION\_ENTER**

```
#define M_ENG_TRACE_FUNCTION_ENTER 0
```

**4.43.1.24 M\_ENG\_TRACE\_FUNCTION\_RETURN**

```
#define M_ENG_TRACE_FUNCTION_RETURN 1
```

#### 4.43.1.25 M\_ENG\_TRACE\_LOG\_ENTRY

```
#define M_ENG_TRACE_LOG_ENTRY()
```

#### 4.43.1.26 M\_ENG\_TRACE\_LOG\_RETURN

```
#define M_ENG_TRACE_LOG_RETURN()
```

#### 4.43.1.27 M\_LOG

```
#define M_LOG(  
    fmt,  
    ...)
```

**Value:**

```
m_eng_log_message(FNAME, XSTR(__LINE__), __func__, current_cycle(), fmt, ##__VA_ARGS__);
```

#### 4.43.1.28 M\_LOG\_ERROR

```
#define M_LOG_ERROR(  
    x)
```

**Value:**

```
m_eng_log_error_code(FNAME, XSTR(__LINE__), __func__, x, current_cycle());
```

#### 4.43.1.29 RETURN

```
#define RETURN(  
    x)
```

**Value:**

```
M_ENG_PROFILER_LOG_RETURN(); M_ENG_TRACE_LOG_RETURN(); M_ENG_LOG_RETURN(); return x;
```

#### 4.43.1.30 RETURN\_ERR\_CODE

```
#define RETURN_ERR_CODE(  
    x)
```

**Value:**

```
M_ENG_PROFILER_LOG_RETURN(); M_ENG_LOG_ERROR(x); M_ENG_TRACE_LOG_RETURN(); M_ENG_LOG_RETURN_ERR_CODE(x);  
return x;
```

#### 4.43.1.31 RETURN\_INT

```
#define RETURN_INT(  
    x)
```

**Value:**

```
M_ENG_PROFILER_LOG_RETURN(); M_ENG_TRACE_LOG_RETURN(); M_ENG_LOG_RETURN_INT(x); return x;
```

#### 4.43.1.32 RETURN\_NEG\_ERR\_CODE

```
#define RETURN_NEG_ERR_CODE(  
    x)
```

**Value:**

```
M_ENG_PROFILER_LOG_RETURN(); M_ENG_LOG_ERROR(x); M_ENG_TRACE_LOG_RETURN(); M_ENG_LOG_RETURN_ERR_CODE(x);  
    return -x;
```

#### 4.43.1.33 RETURN\_PTR

```
#define RETURN_PTR(  
    x)
```

**Value:**

```
M_ENG_PROFILER_LOG_RETURN(); M_ENG_TRACE_LOG_RETURN(); M_ENG_LOG_RETURN_PTR(x); return x;
```

#### 4.43.1.34 RETURN\_VOID

```
#define RETURN_VOID()
```

**Value:**

```
M_ENG_PROFILER_LOG_RETURN(); M_ENG_TRACE_LOG_RETURN(); return;
```

#### 4.43.1.35 SECONDS\_TO\_CYCLES

```
#define SECONDS_TO_CYCLES(  
    x)
```

**Value:**

```
((double)(x) / 1.66666667e-9)
```

#### 4.43.1.36 STR

```
#define STR(  
    x)
```

**Value:**

```
#x
```

#### 4.43.1.37 XSTR

```
#define XSTR(  
    x)
```

**Value:**

```
(STR(x))
```

## 4.43.2 Function Documentation

### 4.43.2.1 m\_eng\_log()

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

### 4.43.2.2 m\_eng\_log\_error\_code()

```
void m_eng_log_error_code (
    const char * fname,
    const char * line,
    const char * function,
    int error_code,
    uint64_t cycle)
```

### 4.43.2.3 m\_eng\_log\_message()

```
void m_eng_log_message (
    const char * fname,
    const char * line,
    const char * function,
    uint64_t cycle,
    const char * fmt,
    ...)
```

### 4.43.2.4 m\_eng\_log\_return\_()

```
void m_eng_log_return_ (
    const char * fname,
    const char * line,
    const char * function,
    uint64_t cycle)
```

### 4.43.2.5 m\_eng\_log\_return\_err()

```
void m_eng_log_return_err (
    const char * fname,
    const char * line,
    const char * function,
    int error_code,
    uint64_t cycle)
```

### 4.43.2.6 m\_eng\_log\_return\_int()

```
void m_eng_log_return_int (
    const char * fname,
    const char * line,
    const char * function,
    int val,
    uint64_t cycle)
```

#### 4.43.2.7 m\_eng\_log\_return\_ptr()

```
void m_eng_log_return_ptr (
    const char * fname,
    const char * line,
    const char * function,
    void * ptr,
    uint64_t cycle)
```

#### 4.43.2.8 m\_eng\_print\_flush\_log()

```
void m_eng_print_flush_log ()
```

#### 4.43.2.9 m\_eng\_print\_log()

```
void m_eng_print_log ()
```

#### 4.43.2.10 m\_eng\_trace\_log\_begin()

```
void m_eng_trace_log_begin (
    const char * fname,
    const char * line,
    const char * function,
    int local_trace_depth,
    uint64_t cycle)
```

#### 4.43.2.11 m\_eng\_trace\_log\_return()

```
void m_eng_trace_log_return (
    const char * fname,
    const char * line,
    const char * function,
    int local_trace_depth,
    uint64_t cycle)
```

### 4.44 m\_eng\_logging.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_LOGGING_H_
00002 #define M_ENG_LOGGING_H_
00003
00004 #define M_ENG_TRACE_FUNCTION_ENTER      0
00005 #define M_ENG_TRACE_FUNCTION_RETURN    1
00006 #define M_ENG_LOG_ENTRY_ERROR          2
00007 #define M_ENG_LOG_ENTRY_RETURN_ERR     3
00008 #define M_ENG_LOG_ENTRY_RETURN_PTR     4
00009 #define M_ENG_LOG_ENTRY_RETURN_INT     5
00010 #define M_ENG_LOG_ENTRY_RETURN         6
00011 #define M_ENG_LOG_ENTRY_MESSAGE        7
00012
00013 typedef struct
00014 {
00015     int type;
```

```

00016     const char *file_name;
00017     const char *line;
00018     const char *data_type;
00019     const char *function;
00020     const char *message;
00021     uint64_t cycle;
00022     uint32_t data;
00023     uint32_t trace_depth;
00024 } m_eng_log_entry;
00025
00026 typedef struct
00027 {
00028     const char *function_name;
00029     uint64_t open_cycle;
00030     uint64_t calls;
00031     uint64_t total_cycles;
00032     uint64_t peak_cycles;
00033     double ra_cycles;
00034 } m_eng_profiler_entry;
00035
00036 #define M_ENG_LOG_ERRORS      0b000001
00037 #define M_ENG_LOG_TRACE      0b00010
00038 #define M_ENG_LOG_RETURNS    0b00100
00039 #define M_ENG_LOG_SPIKES     0b01000
00040
00041 #define M_ENG_LOG_EVERYTHING (M_ENG_LOG_ERRORS | M_ENG_LOG_TRACE | M_ENG_LOG_RETURNS |
    M_ENG_LOG_SPIKES)
00042
00043 #define STR(x) #x
00044 #define XSTR(x) (STR(x))
00045
00046 #ifndef M_SIMULATED
00047 #define M_ENG_LOG_LEVEL (M_ENG_LOG_ERRORS)
00048 // #define M_ENG_PROFILER
00049 #else
00050 #define M_ENG_LOG_LEVEL M_ENG_LOG_ERRORS
00051 #define M_ENG_PROFILER
00052 #endif
00053
00054 #define M_ENG_LOG_INDENT_TRACE
00055
00056 #ifdef M_ENG_LOG_LEVEL
00057     #if M_ENG_LOG_LEVEL & M_ENG_LOG_TRACE
00058         #define M_ENG_TRACE_LOG_ENTRY() m_eng_trace_log_begin (FNAME, XSTR(__LINE__), __func__,
    trace_depth, current_cycle());
00059         #define M_ENG_TRACE_LOG_RETURN() m_eng_trace_log_return(FNAME, XSTR(__LINE__), __func__,
    trace_depth, current_cycle());
00060     #endif
00061
00062     #if M_ENG_LOG_LEVEL & M_ENG_LOG_RETURNS
00063         #define M_ENG_LOG_RETURN_ERR_CODE(x) m_eng_log_return_err(FNAME, XSTR(__LINE__), __func__, x,
    current_cycle());
00064         #define M_ENG_LOG_RETURN_PTR(x) m_eng_log_return_ptr(FNAME, XSTR(__LINE__), __func__, x,
    current_cycle());
00065         #define M_ENG_LOG_RETURN_INT(x) m_eng_log_return_int(FNAME, XSTR(__LINE__), __func__, x,
    current_cycle());
00066         #define M_ENG_LOG_RETURN(x) m_eng_log_return_ (FNAME, XSTR(__LINE__), __func__,
    current_cycle());
00067     #endif
00068
00069     #if M_ENG_LOG_LEVEL & M_ENG_LOG_ERRORS
00070         #define M_ENG_LOG_ERROR(x) \
00071             if (x != NO_ERROR) \
00072                 m_eng_log_error_code(FNAME, XSTR(__LINE__), __func__, x, current_cycle());
00073     #endif
00074 #endif
00075
00076 #ifdef M_ENG_PROFILER
00077     #define M_ENG_PROFILER_LOG_ENTRY() m_eng_profiler_log_entry ( __func__);
00078     #define M_ENG_PROFILER_LOG_RETURN() m_eng_profiler_log_return( __func__, current_cycle());
00079 #else
00080     #define M_ENG_PROFILER_LOG_ENTRY()
00081     #define M_ENG_PROFILER_LOG_RETURN()
00082 #endif
00083
00084 #ifndef M_ENG_TRACE_LOG_ENTRY
00085 #define M_ENG_TRACE_LOG_ENTRY()
00086 #endif
00087
00088 #ifndef M_ENG_TRACE_LOG_RETURN
00089 #define M_ENG_TRACE_LOG_RETURN()
00090 #endif
00091
00092 #ifndef M_ENG_LOG_RETURN_ERR_CODE
00093 #define M_ENG_LOG_RETURN_ERR_CODE(x)
00094 #endif
00095

```

```

00096 #ifndef M_ENG_LOG_RETURN_PTR
00097 #define M_ENG_LOG_RETURN_PTR(x)
00098 #endif
00099
00100 #ifndef M_ENG_LOG_RETURN_INT
00101 #define M_ENG_LOG_RETURN_INT(x)
00102 #endif
00103
00104 #ifndef M_ENG_LOG_RETURN
00105 #define M_ENG_LOG_RETURN(x)
00106 #endif
00107
00108 #ifndef M_ENG_LOG_ERROR
00109 #define M_ENG_LOG_ERROR(x)
00110 #endif
00111
00112 void m_eng_log_message(const char *fname, const char *line, const char *function, uint64_t cycle,
    const char *fmt, ...);
00113
00114
00115 #define FUNCTION_START() M_ENG_PROFILER_LOG_ENTRY(); M_ENG_TRACE_LOG_ENTRY();
00116 #define RETURN_VOID() M_ENG_PROFILER_LOG_RETURN(); M_ENG_TRACE_LOG_RETURN(); return;
00117 #define RETURN_ERR_CODE(x) M_ENG_PROFILER_LOG_RETURN(); M_ENG_LOG_ERROR(x);
    M_ENG_TRACE_LOG_RETURN(); M_ENG_LOG_RETURN_ERR_CODE(x); return x;
00118 #define RETURN_NEG_ERR_CODE(x) M_ENG_PROFILER_LOG_RETURN(); M_ENG_LOG_ERROR(x);
    M_ENG_TRACE_LOG_RETURN(); M_ENG_LOG_RETURN_ERR_CODE(x); return -x;
00119 #define RETURN_PTR(x) M_ENG_PROFILER_LOG_RETURN(); M_ENG_TRACE_LOG_RETURN();
    M_ENG_LOG_RETURN_PTR(x); return x;
00120 #define RETURN_INT(x) M_ENG_PROFILER_LOG_RETURN(); M_ENG_TRACE_LOG_RETURN();
    M_ENG_LOG_RETURN_INT(x); return x;
00121 #define RETURN(x) M_ENG_PROFILER_LOG_RETURN(); M_ENG_TRACE_LOG_RETURN();
    M_ENG_LOG_RETURN(); return x;
00122
00123 void m_eng_log(const char *fmt, ...);
00124 void m_eng_print_log();
00125 void m_eng_print_flush_log();
00126
00127 void m_eng_trace_log_begin(const char *fname, const char *line, const char *function, int
    local_trace_depth, uint64_t cycle);
00128 void m_eng_trace_log_return(const char *fname, const char *line, const char *function, int
    local_trace_depth, uint64_t cycle);
00129 void m_eng_log_error_code(const char *fname, const char *line, const char *function, int
    error_code, uint64_t cycle);
00130 void m_eng_log_return_err(const char *fname, const char *line, const char *function, int
    error_code, uint64_t cycle);
00131 void m_eng_log_return_ptr(const char *fname, const char *line, const char *function, void *ptr,
    uint64_t cycle);
00132 void m_eng_log_return_int(const char *fname, const char *line, const char *function, int val,
    uint64_t cycle);
00133 void m_eng_log_return(const char *fname, const char *line, const char *function, uint64_t
    cycle);
00134
00135 #define M_LOG(fmt, ...) m_eng_log_message(FNAME, XSTR(__LINE__), __func__, current_cycle(), fmt,
    ##__VA_ARGS__);
00136 #define M_LOG_ERROR(x) m_eng_log_error_code(FNAME, XSTR(__LINE__), __func__, x, current_cycle());
00137
00138 #ifdef M_ENG_PROFILER
00139 void m_eng_init_profiler();
00140 void m_eng_profiler_log_entry(const char *function_name);
00141 void m_eng_profiler_log_return(const char *function_name, uint64_t cycle);
00142
00143 void m_eng_profiler_print();
00144 #endif
00145
00146 #define CYCLES_TO_SECONDS(x) ((double)(x) * 1.66666667e-9)
00147 #define SECONDS_TO_CYCLES(x) ((double)(x) / 1.66666667e-9)
00148
00149 #endif

```

## 4.45 m\_eng\_low\_end\_compressor.h File Reference

### Data Structures

- struct [m\\_eng\\_low\\_end\\_compressor\\_str](#)

### Functions

- int [init\\_low\\_end\\_compressor\\_str](#) ([m\\_eng\\_low\\_end\\_compressor\\_str](#) \*str)
- int [reconfigure\\_low\\_end\\_compressor](#) (void \*data\_struct)
- int [calc\\_low\\_end\\_compressor](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)



## 4.45.1 Function Documentation

### 4.45.1.1 calc\_low\_end\_compressor()

```
int calc_low_end_compressor (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

### 4.45.1.2 init\_low\_end\_compressor\_str()

```
int init_low_end_compressor_str (
    m_eng_low_end_compressor_str * str)
```

### 4.45.1.3 reconfigure\_low\_end\_compressor()

```
int reconfigure_low_end_compressor (
    void * data_struct)
```

## 4.46 m\_eng\_low\_end\_compressor.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_LOW_END_COMPRESSOR_H_
00002 #define M_ENG_LOW_END_COMPRESSOR_H_
00003
00004 typedef struct
00005 {
00006     m_eng_compressor_str bass_comp;
00007     m_eng_compressor_str mids_comp;
00008
00009     m_lr_low_pass_filter_str low_pass;
00010     m_lr_low_pass_filter_str mid_pass;
00011
00012 } m_eng_low_end_compressor_str;
00013
00014 int init_low_end_compressor_str(m_eng_low_end_compressor_str *str);
00015 int reconfigure_low_end_compressor(void *data_struct);
00016 int calc_low_end_compressor(void *data_struct, float *dest, float *src, int n_samples);
00017
00018 #endif
```

## 4.47 m\_eng\_memcpy\_audio.h File Reference

### Functions

- void [memcpy\\_tointerleaveLR](#) (int16\_t \*dst, const int16\_t \*srcL, const int16\_t \*srcR)
- void [memcpy\\_tointerleaveL](#) (int16\_t \*dst, const int16\_t \*srcL)
- void [memcpy\\_tointerleaveR](#) (int16\_t \*dst, const int16\_t \*srcR)
- void [memcpy\\_tointerleaveQuad](#) (int16\_t \*dst, const int16\_t \*src1, const int16\_t \*src2, const int16\_t \*src3, const int16\_t \*src4)

## 4.47.1 Function Documentation

### 4.47.1.1 memcpy\_tointerleaveL()

```
void memcpy_tointerleaveL (
    int16_t * dst,
    const int16_t * srcL)
```

### 4.47.1.2 memcpy\_tointerleaveLR()

```
void memcpy_tointerleaveLR (
    int16_t * dst,
    const int16_t * srcL,
    const int16_t * srcR)
```

### 4.47.1.3 memcpy\_tointerleaveQuad()

```
void memcpy_tointerleaveQuad (
    int16_t * dst,
    const int16_t * src1,
    const int16_t * src2,
    const int16_t * src3,
    const int16_t * src4)
```

### 4.47.1.4 memcpy\_tointerleaveR()

```
void memcpy_tointerleaveR (
    int16_t * dst,
    const int16_t * srcR)
```

## 4.48 m\_eng\_memcpy\_audio.h

[Go to the documentation of this file.](#)

```
00001 /* Teensyduino m_eng_Audio Memcpy
00002  * Copyright (c) 2016 Frank Bösing
00003  *
00004  * Permission is hereby granted, free of charge, to any person obtaining
00005  * a copy of this software and associated documentation files (the
00006  * "Software"), to deal in the Software without restriction, including
00007  * without limitation the rights to use, copy, modify, merge, publish,
00008  * distribute, sublicense, and/or sell copies of the Software, and to
00009  * permit persons to whom the Software is furnished to do so, subject to
00010  * the following conditions:
00011  *
00012  * 1. The above copyright notice and this permission notice shall be
00013  * included in all copies or substantial portions of the Software.
00014  *
00015  * 2. If the Software is incorporated into a build system that allows
00016  * selection among a list of target devices, then similar target
00017  * devices manufactured by PJRC.COM must be included in the list of
00018  * target devices and selectable in the same manner.
00019  *
00020  * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
00021  * EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
00022  * MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
00023  * NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS
00024  * BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN
00025  * ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
```

```

00026  * CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
00027  * SOFTWARE.
00028  */
00029
00030 #ifndef memcpy_audio_h_
00031 #define memcpy_audio_h_
00032
00033 #ifdef __cplusplus
00034 extern "C" {
00035 #endif
00036 void memcpy_tointerleaveLR(int16_t *dst, const int16_t *srcL, const int16_t *srcR);
00037 void memcpy_tointerleaveL(int16_t *dst, const int16_t *srcL);
00038 void memcpy_tointerleaveR(int16_t *dst, const int16_t *srcR);
00039 void memcpy_tointerleaveQuad(int16_t *dst, const int16_t *src1, const int16_t *src2,
00040                             const int16_t *src3, const int16_t *src4);
00041 #ifdef __cplusplus
00042 }
00043 #endif
00044
00045
00046 #endif

```

## 4.49 m\_eng\_mempool.h File Reference

### Macros

- #define [MAX\\_AUDIO\\_MEMORY](#) 229376
- #define [MEM\\_SIZE](#) 48
- #define [M\\_BUFFER\\_POOL\\_SIZE](#) 128

### Functions

- float \* [allocate\\_buffer](#) ()
- void [release\\_buffer](#) (float \*buffer)
- void [init\\_mem\\_pools](#) ()
- void [print\\_mempool\\_info](#) ()

### Variables

- float [zero\\_buffer](#) [[AUDIO\\_BLOCK\\_SAMPLES](#)]
- float [sink\\_buffer](#) [[AUDIO\\_BLOCK\\_SAMPLES](#)]

## 4.49.1 Macro Definition Documentation

### 4.49.1.1 M\_BUFFER\_POOL\_SIZE

```
#define M_BUFFER_POOL_SIZE 128
```

### 4.49.1.2 MAX\_AUDIO\_MEMORY

```
#define MAX_AUDIO_MEMORY 229376
```

### 4.49.1.3 MEM\_SIZE

```
#define MEM_SIZE 48
```

## 4.49.2 Function Documentation

### 4.49.2.1 allocate\_buffer()

```
float * allocate_buffer ()
```

### 4.49.2.2 init\_mem\_pools()

```
void init_mem_pools ()
```

### 4.49.2.3 print\_mempool\_info()

```
void print_mempool_info ()
```

### 4.49.2.4 release\_buffer()

```
void release_buffer (  
    float * buffer)
```

## 4.49.3 Variable Documentation

### 4.49.3.1 sink\_buffer

```
float sink_buffer[AUDIO_BLOCK_SAMPLES] [extern]
```

### 4.49.3.2 zero\_buffer

```
float zero_buffer[AUDIO_BLOCK_SAMPLES] [extern]
```

## 4.50 m\_eng\_mempool.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_MEMPOOL_H_
00002 #define M_MEMPOOL_H_
00003
00004 #define MAX_AUDIO_MEMORY 229376
00005
00006 #define MEM_SIZE 48
00007
00008 #ifdef M_SIMULATED
00009 #define M_BUFFER_POOL_SIZE 8192
00010 #else
00011 #define M_BUFFER_POOL_SIZE 128
00012 #endif
00013
00014 float *allocate_buffer();
00015 void release_buffer(float *buffer);
00016
00017 void init_mem_pools();
00018
00019 extern float zero_buffer[AUDIO_BLOCK_SAMPLES];
00020 extern float sink_buffer[AUDIO_BLOCK_SAMPLES];
00021
00022 void print_mempool_info();
00023
00024 #endif
```

## 4.51 m\_eng\_noise\_suppressor.h File Reference

### Data Structures

- struct [m\\_eng\\_noise\\_suppressor\\_str](#)

### Functions

- int [reconfigure\\_noise\\_suppressor](#) (void \*data\_struct)
- int [calc\\_noise\\_suppressor](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- int [init\\_noise\\_suppressor\\_str](#) ([m\\_eng\\_noise\\_suppressor\\_str](#) \*str)

### 4.51.1 Function Documentation

#### 4.51.1.1 calc\_noise\_suppressor()

```
int calc_noise_suppressor (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.51.1.2 init\_noise\_suppressor\_str()

```
int init_noise_suppressor_str (  
    m\_eng\_noise\_suppressor\_str * str)
```

#### 4.51.1.3 reconfigure\_noise\_suppressor()

```
int reconfigure_noise_suppressor (  
    void * data_struct)
```

## 4.52 m\_eng\_noise\_suppressor.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_NOISE_SUPPRESSOR_H_  
00002 #define M_ENG_NOISE_SUPPRESSOR_H_  
00003  
00004 typedef struct  
00005 {  
00006     m_parameter threshold;  
00007     m_parameter ratio;  
00008     m_parameter max_reduction;  
00009     float e_final;  
00010  
00011     int r;  
00012 } m_eng_noise_suppressor_str;  
00013  
00014 int reconfigure_noise_suppressor(void *data_struct);  
00015 int calc_noise_suppressor(void *data_struct, float *dest, float *src, int n_samples);  
00016 int init_noise_suppressor_str(m_eng_noise_suppressor_str *str);  
00017  
00018 #endif
```

## 4.53 m\_eng\_parameter.h File Reference

```
#include <stddef.h>
#include "m_parameter.h"
```

### Macros

- `#define PARAM_NAM_ENG_MAX_LEN 16`
- `#define PARAMETER_UPDATE_INSTANT 0`
- `#define PARAMETER_UPDATE_MONOBLOCK_LINEAR 1`
- `#define PARAMETER_UPDATE_BIBLOCK_LINEAR 2`
- `#define PARAMETER_UPDATE_QUADBLOCK_LINEAR 3`
- `#define DEFAULT_MAX_JUMP 0.01`

### Functions

- void `init_parameter` (`m_parameter` \*param, float initial, float min, float max, float max\_jump, int scale)
- int `init_setting` (`m_setting` \*setting, int16\_t initial)

### 4.53.1 Macro Definition Documentation

#### 4.53.1.1 DEFAULT\_MAX\_JUMP

```
#define DEFAULT_MAX_JUMP 0.01
```

#### 4.53.1.2 PARAM\_NAM\_ENG\_MAX\_LEN

```
#define PARAM_NAM_ENG_MAX_LEN 16
```

#### 4.53.1.3 PARAMETER\_UPDATE\_BIBLOCK\_LINEAR

```
#define PARAMETER_UPDATE_BIBLOCK_LINEAR 2
```

#### 4.53.1.4 PARAMETER\_UPDATE\_INSTANT

```
#define PARAMETER_UPDATE_INSTANT 0
```

#### 4.53.1.5 PARAMETER\_UPDATE\_MONOBLOCK\_LINEAR

```
#define PARAMETER_UPDATE_MONOBLOCK_LINEAR 1
```

#### 4.53.1.6 PARAMETER\_UPDATE\_QUADBLOCK\_LINEAR

```
#define PARAMETER_UPDATE_QUADBLOCK_LINEAR 3
```

### 4.53.2 Function Documentation

#### 4.53.2.1 init\_parameter()

```
void init_parameter (
    m_parameter * param,
    float initial,
    float min,
    float max,
    float max_jump,
    int scale)
```

#### 4.53.2.2 init\_setting()

```
int init_setting (
    m_setting * setting,
    int16_t initial)
```

## 4.54 m\_eng\_parameter.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_PARAMETER_H_
00002 #define M_ENG_PARAMETER_H_
00003
00004 #ifdef __cplusplus
00005 #include <cstdint>
00006 #else
00007 #include <stdint.h>
00008 #endif
00009
00010 #define PARAM_NAM_ENG_MAX_LEN 16
00011
00012 #define PARAMETER_UPDATE_INSTANT 0
00013 #define PARAMETER_UPDATE_MONOBLOCK_LINEAR 1
00014 #define PARAMETER_UPDATE_BIBLOCK_LINEAR 2
00015 #define PARAMETER_UPDATE_QUADBLOCK_LINEAR 3
00016
00017 #define DEFAULT_MAX_JUMP 0.01
00018
00019 #include "m_parameter.h"
00020
00021 void init_parameter(m_parameter *param, float initial, float min, float max, float max_jump, int
    scale);
00022 int init_setting(m_setting *setting, int16_t initial);
00023
00024 #endif
```

## 4.55 m\_eng\_pass\_filter.h File Reference

### Data Structures

- struct [m\\_eng\\_low\\_pass\\_filter\\_str](#)
- struct [m\\_eng\\_high\\_pass\\_filter\\_str](#)
- struct [m\\_eng\\_band\\_pass\\_filter\\_str](#)

## Functions

- [int init\\_low\\_pass\\_filter\\_str](#) ([m\\_eng\\_low\\_pass\\_filter\\_str](#) \*str)
- [int reconfigure\\_low\\_pass\\_filter](#) (void \*data\_struct)
- [int calc\\_low\\_pass\\_filter](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- [int init\\_high\\_pass\\_filter\\_str](#) ([m\\_eng\\_high\\_pass\\_filter\\_str](#) \*str)
- [int reconfigure\\_high\\_pass\\_filter](#) (void \*data\_struct)
- [int calc\\_high\\_pass\\_filter](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- [int init\\_band\\_pass\\_filter\\_str](#) ([m\\_eng\\_band\\_pass\\_filter\\_str](#) \*str)
- [int reconfigure\\_band\\_pass\\_filter](#) (void \*data\_struct)
- [int calc\\_band\\_pass\\_filter](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

## 4.55.1 Function Documentation

### 4.55.1.1 [calc\\_band\\_pass\\_filter\(\)](#)

```
int calc_band_pass_filter (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

### 4.55.1.2 [calc\\_high\\_pass\\_filter\(\)](#)

```
int calc_high_pass_filter (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

### 4.55.1.3 [calc\\_low\\_pass\\_filter\(\)](#)

```
int calc_low_pass_filter (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

### 4.55.1.4 [init\\_band\\_pass\\_filter\\_str\(\)](#)

```
int init_band_pass_filter_str (  
    m\_eng\_band\_pass\_filter\_str * str)
```

### 4.55.1.5 [init\\_high\\_pass\\_filter\\_str\(\)](#)

```
int init_high_pass_filter_str (  
    m\_eng\_high\_pass\_filter\_str * str)
```



**4.55.1.6 init\_low\_pass\_filter\_str()**

```
int init_low_pass_filter_str (
    m_eng_low_pass_filter_str * str)
```

**4.55.1.7 reconfigure\_band\_pass\_filter()**

```
int reconfigure_band_pass_filter (
    void * data_struct)
```

**4.55.1.8 reconfigure\_high\_pass\_filter()**

```
int reconfigure_high_pass_filter (
    void * data_struct)
```

**4.55.1.9 reconfigure\_low\_pass\_filter()**

```
int reconfigure_low_pass_filter (
    void * data_struct)
```

**4.56 m\_eng\_pass\_filter.h**

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_LOW_PASS_H_
00002 #define M_ENG_LOW_PASS_H_
00003
00004 typedef struct
00005 {
00006     m_parameter cutoff_frequency;
00007
00008     float a0, a1, a2, a3, a4;
00009     float x1, x2, y1, y2;
00010 } m_eng_low_pass_filter_str;
00011
00012 int init_low_pass_filter_str(m_eng_low_pass_filter_str *str);
00013 int reconfigure_low_pass_filter(void *data_struct);
00014 int calc_low_pass_filter(void *data_struct, float *dest, float *src, int n_samples);
00015
00016 typedef struct
00017 {
00018     m_parameter cutoff_frequency;
00019
00020     float a0, a1, a2, a3, a4;
00021     float x1, x2, y1, y2;
00022 } m_eng_high_pass_filter_str;
00023
00024 int init_high_pass_filter_str(m_eng_high_pass_filter_str *str);
00025 int reconfigure_high_pass_filter(void *data_struct);
00026 int calc_high_pass_filter(void *data_struct, float *dest, float *src, int n_samples);
00027
00028 typedef struct
00029 {
00030     m_parameter center;
00031     m_parameter bandwidth;
00032
00033     float a0, a1, a2, a3, a4;
00034     float x1, x2, y1, y2;
00035 } m_eng_band_pass_filter_str;
00036
00037 int init_band_pass_filter_str(m_eng_band_pass_filter_str *str);
00038 int reconfigure_band_pass_filter(void *data_struct);
00039 int calc_band_pass_filter(void *data_struct, float *dest, float *src, int n_samples);
00040
00041 #endif
00042
```

## 4.57 m\_eng\_percussifier.h File Reference

### Data Structures

- struct [m\\_eng\\_percussifier\\_str](#)

### Macros

- #define [PERCUSSIFIER\\_MUTE](#) 0
- #define [PERCUSSIFIER\\_FADE\\_IN](#) 1
- #define [PERCUSSIFIER\\_HOLD](#) 2
- #define [PERCUSSIFIER\\_FADE\\_OUT](#) 3
- #define [PERCUSSIFIER\\_REFRACTORY](#) 4

### Functions

- int [init\\_percussifier\\_str](#) ([m\\_eng\\_percussifier\\_str](#) \*str)
- int [reconfigure\\_percussifier](#) (void \*data\_struct)
- int [calc\\_percussifier](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.57.1 Macro Definition Documentation

#### 4.57.1.1 PERCUSSIFIER\_FADE\_IN

```
#define PERCUSSIFIER_FADE_IN 1
```

#### 4.57.1.2 PERCUSSIFIER\_FADE\_OUT

```
#define PERCUSSIFIER_FADE_OUT 3
```

#### 4.57.1.3 PERCUSSIFIER\_HOLD

```
#define PERCUSSIFIER_HOLD 2
```

#### 4.57.1.4 PERCUSSIFIER\_MUTE

```
#define PERCUSSIFIER_MUTE 0
```

#### 4.57.1.5 PERCUSSIFIER\_REFRACTORY

```
#define PERCUSSIFIER_REFRACTORY 4
```

## 4.57.2 Function Documentation

### 4.57.2.1 calc\_percussifier()

```
int calc_percussifier (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

### 4.57.2.2 init\_percussifier\_str()

```
int init_percussifier_str (
    m_eng_percussifier_str * str)
```

### 4.57.2.3 reconfigure\_percussifier()

```
int reconfigure_percussifier (
    void * data_struct)
```

## 4.58 m\_eng\_percussifier.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_PERCUSSIFIER_H_
00002 #define M_ENG_PERCUSSIFIER_H_
00003
00004 #define PERCUSSIFIER_MUTE 0
00005 #define PERCUSSIFIER_FADE_IN 1
00006 #define PERCUSSIFIER_HOLD 2
00007 #define PERCUSSIFIER_FADE_OUT 3
00008 #define PERCUSSIFIER_REFRACTORY 4
00009
00010 typedef struct
00011 {
00012     m_parameter tempo;
00013     m_parameter note;
00014     m_parameter trigger_threshold;
00015     m_parameter arm_threshold;
00016     m_parameter refractory_period;
00017     m_parameter fade_in;
00018     m_parameter fade_out;
00019
00020     int state;
00021     int fade_in_samples;
00022     int hold_samples;
00023     int refractory_samples;
00024
00025     int timer;
00026
00027     float decay_rate;
00028     float rms_short;
00029     float rms_long;
00030
00031     float alpha_short;
00032     float alpha_long;
00033
00034     float gain;
00035     float fade_alpha;
00036
00037     int r;
00038 } m_eng_percussifier_str;
00039
00040 int init_percussifier_str(m_eng_percussifier_str *str);
00041 int reconfigure_percussifier(void *data_struct);
00042 int calc_percussifier(void *data_struct, float *dest, float *src, int n_samples);
00043
00044 #endif
```

## 4.59 m\_eng\_pipeline.h File Reference

```
#include "m_transformer.h"
#include "m_pipeline.h"
```

### Macros

- `#define INITIAL_TRANSFORMER_ARRAY_LENGTH 8`

### Functions

- `int init_pipeline (m_pipeline *pipeline)`
- `int compute_pipeline (m_pipeline *pipeline, float *dest, float *src)`
- `int pipeline_expand_transformer_array (m_pipeline *pipeline)`
- `int pipeline_expand_transformer_array_to (m_pipeline *pipeline, int n)`
- `int pipeline_append_transformer (m_pipeline *pipeline, m_transformer *trans)`
- `int pipeline_remove_transformer (m_pipeline *pipeline, uint16_t tid)`
- `int pipeline_insert_transformer (m_pipeline *pipeline, m_transformer *trans, int pos)`
- `int pipeline_prepend_transformer (m_pipeline *pipeline, m_transformer *trans)`
- `int pipeline_append_transformer_type (m_pipeline *pipeline, uint16_t type)`
- `int pipeline_insert_transformer_type (m_pipeline *pipeline, uint16_t type, uint16_t pos)`
- `int pipeline_prepend_transformer_type (m_pipeline *pipeline, uint16_t type)`
- `int pipeline_change_transformer_setting (m_pipeline *pipeline, uint16_t tid, uint16_t sid, int16_t new_val)`
- `int pipeline_get_transformer_position (m_pipeline *pipeline, uint16_t id)`
- `m_transformer * pipeline_get_transformer_by_id (m_pipeline *pipeline, uint16_t id)`
- `int pipeline_move_transformer (m_pipeline *pipeline, uint16_t id, int positio)`
- `int pipeline_update_transition_policy (m_pipeline *pipeline)`
- `int pipeline_swap_transformers (m_pipeline *pipeline, uint16_t id1, uint16_t id2)`
- `int pipeline_valid (m_pipeline *pipeline)`
- `int pipeline_compare (m_pipeline *pipeline_a, m_pipeline *pipeline_b)`
- `int gut_pipeline (m_pipeline *pipeline)`
- `int clone_pipeline (m_pipeline **dest, m_pipeline *src)`
- `int pipeline_clone_transformer_into_position (m_pipeline *pipeline, m_transformer *transformer, int pot)`

### 4.59.1 Macro Definition Documentation

#### 4.59.1.1 INITIAL\_TRANSFORMER\_ARRAY\_LENGTH

```
#define INITIAL_TRANSFORMER_ARRAY_LENGTH 8
```

### 4.59.2 Function Documentation

#### 4.59.2.1 clone\_pipeline()

```
int clone_pipeline (
    m_pipeline ** dest,
    m_pipeline * src)
```

#### 4.59.2.2 compute\_pipeline()

```
int compute_pipeline (  
    m_pipeline * pipeline,  
    float * dest,  
    float * src)
```

#### 4.59.2.3 gut\_pipeline()

```
int gut_pipeline (  
    m_pipeline * pipeline)
```

#### 4.59.2.4 init\_pipeline()

```
int init_pipeline (  
    m_pipeline * pipeline)
```

#### 4.59.2.5 pipeline\_append\_transformer()

```
int pipeline_append_transformer (  
    m_pipeline * pipeline,  
    m_transformer * trans)
```

#### 4.59.2.6 pipeline\_append\_transformer\_type()

```
int pipeline_append_transformer_type (  
    m_pipeline * pipeline,  
    uint16_t type)
```

#### 4.59.2.7 pipeline\_change\_transformer\_setting()

```
int pipeline_change_transformer_setting (  
    m_pipeline * pipeline,  
    uint16_t tid,  
    uint16_t sid,  
    int16_t new_val)
```

#### 4.59.2.8 pipeline\_clone\_transformer\_into\_position()

```
int pipeline_clone_transformer_into_position (  
    m_pipeline * pipeline,  
    m_transformer * transformer,  
    int pot)
```

#### 4.59.2.9 pipeline\_compare()

```
int pipeline_compare (
    m_pipeline * pipeline_a,
    m_pipeline * pipeline_b)
```

#### 4.59.2.10 pipeline\_expand\_transformer\_array()

```
int pipeline_expand_transformer_array (
    m_pipeline * pipeline)
```

#### 4.59.2.11 pipeline\_expand\_transformer\_array\_to()

```
int pipeline_expand_transformer_array_to (
    m_pipeline * pipeline,
    int n)
```

#### 4.59.2.12 pipeline\_get\_transformer\_by\_id()

```
m_transformer * pipeline_get_transformer_by_id (
    m_pipeline * pipeline,
    uint16_t id)
```

#### 4.59.2.13 pipeline\_get\_transformer\_position()

```
int pipeline_get_transformer_position (
    m_pipeline * pipeline,
    uint16_t id)
```

#### 4.59.2.14 pipeline\_insert\_transformer()

```
int pipeline_insert_transformer (
    m_pipeline * pipeline,
    m_transformer * trans,
    int pos)
```

#### 4.59.2.15 pipeline\_insert\_transformer\_type()

```
int pipeline_insert_transformer_type (
    m_pipeline * pipeline,
    uint16_t type,
    uint16_t pos)
```

#### 4.59.2.16 pipeline\_move\_transformer()

```
int pipeline_move_transformer (  
    m_pipeline * pipeline,  
    uint16_t id,  
    int positio)
```

#### 4.59.2.17 pipeline\_prepend\_transformer()

```
int pipeline_prepend_transformer (  
    m_pipeline * pipeline,  
    m_transformer * trans)
```

#### 4.59.2.18 pipeline\_prepend\_transformer\_type()

```
int pipeline_prepend_transformer_type (  
    m_pipeline * pipeline,  
    uint16_t type)
```

#### 4.59.2.19 pipeline\_remove\_transformer()

```
int pipeline_remove_transformer (  
    m_pipeline * pipeline,  
    uint16_t tid)
```

#### 4.59.2.20 pipeline\_swap\_transformers()

```
int pipeline_swap_transformers (  
    m_pipeline * pipeline,  
    uint16_t id1,  
    uint16_t id2)
```

#### 4.59.2.21 pipeline\_update\_transition\_policy()

```
int pipeline_update_transition_policy (  
    m_pipeline * pipeline)
```

#### 4.59.2.22 pipeline\_valid()

```
int pipeline_valid (  
    m_pipeline * pipeline)
```

## 4.60 m\_eng\_pipeline.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_PIPELINE_H_
00002 #define M_ENG_PIPELINE_H_
00003
00004 #include "m_transformer.h"
00005
00006 #define INITIAL_TRANSFORMER_ARRAY_LENGTH 8
00007
00008 #include "m_pipeline.h"
00009
00010 int init_pipeline(m_pipeline *pipeline);
00011 int compute_pipeline(m_pipeline *pipeline, float *dest, float *src);
00012
00013 int pipeline_expand_transformer_array(m_pipeline *pipeline);
00014 int pipeline_expand_transformer_array_to(m_pipeline *pipeline, int n);
00015 int pipeline_append_transformer(m_pipeline *pipeline, m_transformer *trans);
00016 int pipeline_remove_transformer(m_pipeline *pipeline, uint16_t tid);
00017 int pipeline_insert_transformer(m_pipeline *pipeline, m_transformer *trans, int pos);
00018 int pipeline_prepend_transformer(m_pipeline *pipeline, m_transformer *trans);
00019 int pipeline_append_transformer_type(m_pipeline *pipeline, uint16_t type);
00020 int pipeline_insert_transformer_type(m_pipeline *pipeline, uint16_t type, uint16_t pos);
00021 int pipeline_prepend_transformer_type(m_pipeline *pipeline, uint16_t type);
00022 int pipeline_change_transformer_setting(m_pipeline *pipeline, uint16_t tid, uint16_t sid, int16_t
new_val);
00023
00024 int pipeline_get_transformer_position(m_pipeline *pipeline, uint16_t id);
00025 m_transformer *pipeline_get_transformer_by_id(m_pipeline *pipeline, uint16_t id);
00026
00027 int pipeline_move_transformer(m_pipeline *pipeline, uint16_t id, int positio);
00028
00029 int pipeline_update_transition_policy(m_pipeline *pipeline);
00030
00031 int pipeline_swap_transformers(m_pipeline *pipeline, uint16_t id1, uint16_t id2);
00032
00033 int pipeline_valid (m_pipeline *pipeline);
00034 int pipeline_compare(m_pipeline *pipeline_a, m_pipeline *pipeline_b);
00035 int gut_pipeline (m_pipeline *pipeline);
00036 int clone_pipeline (m_pipeline **dest, m_pipeline *src);
00037
00038 int pipeline_clone_transformer_into_position(m_pipeline *pipeline, m_transformer *transformer, int
pot);
00039
00040 #endif
```

## 4.61 m\_eng\_pipeline\_mod.h File Reference

### Data Structures

- struct [m\\_pipeline\\_mod](#)

### Macros

- #define [PIPELINE\\_MOD\\_APPEND\\_TRANSFORMER](#) 0
- #define [PIPELINE\\_MOD\\_MOVE\\_TRANSFORMER](#) 1
- #define [PIPELINE\\_MOD\\_REMOVE\\_TRANSFORMER](#) 2
- #define [PIPELINE\\_MOD\\_CHANGE\\_TRANSFORMER\\_SETTING](#) 3

### Functions

- int [init\\_pipeline\\_mod](#) (m\_pipeline\_mod \*mod)
- m\_pipeline\_mod [create\\_pipeline\\_mod\\_append\\_transformer](#) (uint16\_t type)
- m\_pipeline\_mod [create\\_pipeline\\_mod\\_move\\_transformer](#) (uint16\_t tid, uint16\_t position)
- m\_pipeline\_mod [create\\_pipeline\\_mod\\_remove\\_transformer](#) (uint16\_t tid)
- m\_pipeline\_mod [create\\_pipeline\\_mod\\_change\\_transformer\\_setting](#) (uint16\_t tid, uint16\_t setting\_id, int16\_t new\_value)
- int [apply\\_pipeline\\_mod](#) (m\_pipeline \*pipeline, m\_pipeline\_mod mod, int \*err\_code)
- const char \* [pipeline\\_mod\\_type\\_string](#) (int type)
- [DECLARE\\_LINKED\\_LIST](#) (m\_pipeline\_mod)



## 4.61.1 Macro Definition Documentation

### 4.61.1.1 PIPELINE\_MOD\_APPEND\_TRANSFORMER

```
#define PIPELINE_MOD_APPEND_TRANSFORMER 0
```

### 4.61.1.2 PIPELINE\_MOD\_CHANGE\_TRANSFORMER\_SETTING

```
#define PIPELINE_MOD_CHANGE_TRANSFORMER_SETTING 3
```

### 4.61.1.3 PIPELINE\_MOD\_MOVE\_TRANSFORMER

```
#define PIPELINE_MOD_MOVE_TRANSFORMER 1
```

### 4.61.1.4 PIPELINE\_MOD\_REMOVE\_TRANSFORMER

```
#define PIPELINE_MOD_REMOVE_TRANSFORMER 2
```

## 4.61.2 Function Documentation

### 4.61.2.1 apply\_pipeline\_mod()

```
int apply_pipeline_mod (  
    m_pipeline * pipeline,  
    m_pipeline_mod mod,  
    int * err_code)
```

### 4.61.2.2 create\_pipeline\_mod\_append\_transformer()

```
m_pipeline_mod create_pipeline_mod_append_transformer (  
    uint16_t type)
```

### 4.61.2.3 create\_pipeline\_mod\_change\_transformer\_setting()

```
m_pipeline_mod create_pipeline_mod_change_transformer_setting (  
    uint16_t tid,  
    uint16_t setting_id,  
    int16_t new_value)
```

### 4.61.2.4 create\_pipeline\_mod\_move\_transformer()

```
m_pipeline_mod create_pipeline_mod_move_transformer (  
    uint16_t tid,  
    uint16_t position)
```

#### 4.61.2.5 create\_pipeline\_mod\_remove\_transformer()

```
m_pipeline_mod create_pipeline_mod_remove_transformer (
    uint16_t tid)
```

#### 4.61.2.6 DECLARE\_LINKED\_LIST()

```
DECLARE_LINKED_LIST (
    m_pipeline_mod )
```

#### 4.61.2.7 init\_pipeline\_mod()

```
int init_pipeline_mod (
    m_pipeline_mod * mod)
```

#### 4.61.2.8 pipeline\_mod\_type\_string()

```
const char * pipeline_mod_type_string (
    int type)
```

## 4.62 m\_eng\_pipeline\_mod.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_PIPELINE_MOD_H_
00002 #define M_ENG_PIPELINE_MOD_H_
00003
00004 #define PIPELINE_MOD_APPEND_TRANSFORMER      0
00005 #define PIPELINE_MOD_MOVE_TRANSFORMER        1
00006 #define PIPELINE_MOD_REMOVE_TRANSFORMER      2
00007 #define PIPELINE_MOD_CHANGE_TRANSFORMER_SETTING 3
00008
00009 typedef struct m_pipeline_mod
00010 {
00011     int type;
00012     uint16_t tid;
00013     uint16_t data;
00014     int16_t sdata;
00015 } m_pipeline_mod;
00016
00017 int init_pipeline_mod(m_pipeline_mod *mod);
00018
00019 m_pipeline_mod create_pipeline_mod_append_transformer      (uint16_t type);
00020 m_pipeline_mod create_pipeline_mod_move_transformer        (uint16_t tid, uint16_t position);
00021 m_pipeline_mod create_pipeline_mod_remove_transformer      (uint16_t tid);
00022 m_pipeline_mod create_pipeline_mod_change_transformer_setting (uint16_t tid, uint16_t setting_id,
    int16_t new_value);
00023
00024 int apply_pipeline_mod(m_pipeline *pipeline, m_pipeline_mod mod, int *err_code);
00025
00026 const char *pipeline_mod_type_string(int type);
00027
00028 DECLARE_LINKED_LIST(m_pipeline_mod);
00029
00030 #endif
```

## 4.63 m\_eng\_printf.h File Reference

### Functions

- void [m\\_voice\\_printf](#) (int who, const char \*fmt,...)
- void [m\\_mute\\_voice](#) (int who)
- void [m\\_unmute\\_voice](#) (int who)
- void [m\\_printf](#) (const char \*fmt,...)
- void [pretty\\_print\\_block](#) (int16\_t \*data, const char \*start)
- void [pretty\\_print\\_block\\_float](#) (float \*data, const char \*start)
- void [serial\\_print\\_blocks](#) (int n,...)

### 4.63.1 Function Documentation

#### 4.63.1.1 m\_mute\_voice()

```
void m_mute_voice (  
    int who)
```

#### 4.63.1.2 m\_printf()

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

#### 4.63.1.3 m\_unmute\_voice()

```
void m_unmute_voice (  
    int who)
```

#### 4.63.1.4 m\_voice\_printf()

```
void m_voice_printf (  
    int who,  
    const char * fmt,  
    ...)
```

#### 4.63.1.5 pretty\_print\_block()

```
void pretty_print_block (  
    int16_t * data,  
    const char * start)
```

#### 4.63.1.6 pretty\_print\_block\_float()

```
void pretty_print_block_float (  
    float * data,  
    const char * start)
```

#### 4.63.1.7 serial\_print\_blocks()

```
void serial_print_blocks (
    int n,
    ...)
```

### 4.64 m\_eng\_printf.h

[Go to the documentation of this file.](#)

```
00001 #ifndef MY_PRINTF_H_
00002 #define MY_PRINTF_H_
00003
00004 void m_voice_printf(int who, const char *fmt, ...);
00005
00006 void m_mute_voice(int who);
00007 void m_unmute_voice(int who);
00008
00009 void m_printf(const char *fmt, ...);
00010 void pretty_print_block(int16_t *data, const char *start);
00011 void pretty_print_block_float(float *data, const char *start);
00012
00013 void serial_print_blocks(int n, ...);
00014
00015 #endif
```

### 4.65 m\_eng\_profile.h File Reference

```
#include "m_eng_pipeline.h"
```

#### Data Structures

- struct [m\\_eng\\_profile](#)

#### Functions

- int [nullify\\_profile](#) ([m\\_eng\\_profile](#) \*profile)
- int [init\\_profile](#) ([m\\_eng\\_profile](#) \*profile)
- int [init\\_bypass\\_profile](#) ([m\\_eng\\_profile](#) \*profile)
- int [profile\\_process](#) ([m\\_eng\\_profile](#) \*profile, float \*dest, float \*src)
- int [profile\\_update](#) ([m\\_eng\\_profile](#) \*profile)
- int [profile\\_apply\\_pipeline\\_mod](#) ([m\\_eng\\_profile](#) \*profile, [m\\_pipeline\\_mod](#) mod)
  - Applies the given modification to the given profile.*
- int [profile\\_trigger\\_pipeline\\_swap](#) ([m\\_eng\\_profile](#) \*profile)
- int [profile\\_scheduled\\_maintenance](#) ([m\\_eng\\_profile](#) \*profile)

#### 4.65.1 Function Documentation

##### 4.65.1.1 init\_bypass\_profile()

```
int init_bypass_profile (
    m\_eng\_profile * profile)
```

#### 4.65.1.2 init\_profile()

```
int init_profile (  
    m_eng_profile * profile)
```

#### 4.65.1.3 nullify\_profile()

```
int nullify_profile (  
    m_eng_profile * profile)
```

#### 4.65.1.4 profile\_apply\_pipeline\_mod()

```
int profile_apply_pipeline_mod (  
    m_eng_profile * profile,  
    m_pipeline_mod mod)
```

Applies the given modification to the given profile.

If the current profile is active, to prevent audio artifacts, pipeline modifications ("mods", encoded by [m\\_pipeline\\_mod](#)) are applied in the background - to an inactive copy of the profile's pipeline (the "back pipeline"), and the outputs from the front and back pipeline are smoothly cross-faded between, after which it can be safely applied to the (now back, previously front) pipeline, keeping the two in sync.

To achieve this, [profile\\_apply\\_pipeline\\_mod](#) calls [apply\\_pipeline\\_mod](#) to apply the mod to the back pipeline, calls [profile\\_trigger\\_pipeline\\_swap](#) to trigger the pipeline swap, and saves the mod in a linked list ([m\\_eng\\_profile::jobs](#)). After the swap is complete, [profile\\_update](#) will retrieve the job from the list and apply it to the back (previously front) pipeline.

If, however, when [profile\\_apply\\_pipeline\\_mod](#) is called, a pipeline swap is already in progress, the mod is *not* applied, but rather stored in a second linked list, ([m\\_eng\\_profile::blocked\\_jobs](#)), whereupon [profile\\_update](#) will apply it (after the swap is finished and any jobs waiting in ([m\\_eng\\_profile::jobs](#)) have been applied) by calling [profile\\_apply\\_pipeline\\_mod](#).

##### Parameters

<i>profile</i>	The profile to be modified
<i>mod</i>	A struct describing the modification to apply

#### 4.65.1.5 profile\_process()

```
int profile_process (  
    m_eng_profile * profile,  
    float * dest,  
    float * src)
```

#### 4.65.1.6 profile\_scheduled\_maintenance()

```
int profile_scheduled_maintenance (  
    m_eng_profile * profile)
```

#### 4.65.1.7 profile\_trigger\_pipeline\_swap()

```
int profile_trigger_pipeline_swap (
    m_eng_profile * profile)
```

#### 4.65.1.8 profile\_update()

```
int profile_update (
    m_eng_profile * profile)
```

### 4.66 m\_eng\_profile.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_PROFILE_H_
00002 #define M_ENG_PROFILE_H_
00003
00004 #include "m_eng_pipeline.h"
00005
00006 typedef struct
00007 {
00008     m_pipeline *front_pipeline;
00009     m_pipeline *back_pipeline;
00010
00011     int pipelines_swapping;
00012     int pipeline_swap_progress;
00013     int pipeline_swap_samples;
00014     int pipeline_swap_type;
00015     int back_pipeline_warmed_up;
00016
00017     m_pipeline_mod_ll *jobs;
00018     m_pipeline_mod_ll *blocked_jobs;
00019
00020     int active;
00021     int transition_policy;
00022
00023     float *prev_block;
00024
00025     m_transformer output_amp;
00026
00027     int runs;
00028 } m_eng_profile;
00029
00030 int nullify_profile(m_eng_profile *profile);
00031 int init_profile(m_eng_profile *profile);
00032 int init_bypass_profile(m_eng_profile *profile);
00033
00034 int profile_process(m_eng_profile *profile, float *dest, float *src);
00035 int profile_update(m_eng_profile *profile);
00036
00037 int profile_apply_pipeline_mod(m_eng_profile *profile, m_pipeline_mod mod);
00038
00039 int profile_trigger_pipeline_swap(m_eng_profile *profile);
00040
00041 int profile_scheduled_maintenance(m_eng_profile *profile);
00042
00043 #endif
```

### 4.67 m\_eng\_sgtl5000.h File Reference

#### Functions

- int [sgtl5000\\_start](#) ()
- int [sgtl5000\\_soft\\_reboot](#) ()
- int [sgtl5000\\_healthy](#) ()
- int [sgtl5000\\_enable](#) ()

- void [sgtl5000\\_set\\_address](#) (uint8\_t level)
- int [sgtl5000\\_volume](#) (float n)
- int [sgtl5000\\_mute\\_headphone](#) ()
- int [sgtl5000\\_unmute\\_headphone](#) ()
- int [sgtl5000\\_mute\\_line\\_out](#) ()
- int [sgtl5000\\_unmute\\_line\\_out](#) ()
- int [sgtl5000\\_mic\\_gain](#) (unsigned int dB)
- int [sgtl5000\\_line\\_in\\_level](#) (uint8\_t n)
- unsigned short [sgtl5000\\_line\\_out\\_level](#) (uint8\_t n)
- unsigned short [sgtl5000\\_adc\\_high\\_pass\\_filter\\_enable](#) ()
- unsigned short [sgtl5000\\_adc\\_high\\_pass\\_filter\\_freeze](#) ()
- unsigned short [sgtl5000\\_adc\\_high\\_pass\\_filter\\_disable](#) ()
- void [sgtl5000\\_kill\\_automation](#) ()
- void [sgtl5000\\_set\\_master\\_mode](#) (uint32\_t freqMCLK\_in)
- int [sgtl5000\\_volum\\_eng\\_integer](#) (unsigned int n)
- unsigned int [sgtl5000\\_read\\_reg](#) (unsigned int reg)
- int [sgtl5000\\_write\\_reg](#) (unsigned int reg, unsigned int val)
- unsigned int [sgtl5000\\_modify\\_reg](#) (unsigned int reg, unsigned int val, unsigned int i\_mask)
- unsigned short [sgtl5000\\_dap\\_audio\\_eq\\_band](#) (uint8\_t band\_num, float n)
- void [sgtl5000\\_automate](#) (uint8\_t dap, uint8\_t eq)
- unsigned char [calc\\_vol](#) (float n, unsigned char range)

## 4.67.1 Function Documentation

### 4.67.1.1 [calc\\_vol\(\)](#)

```
unsigned char calc_vol (
    float n,
    unsigned char range)
```

### 4.67.1.2 [sgtl5000\\_adc\\_high\\_pass\\_filter\\_disable\(\)](#)

```
unsigned short sgtl5000_adc_high_pass_filter_disable ()
```

### 4.67.1.3 [sgtl5000\\_adc\\_high\\_pass\\_filter\\_enable\(\)](#)

```
unsigned short sgtl5000_adc_high_pass_filter_enable ()
```

### 4.67.1.4 [sgtl5000\\_adc\\_high\\_pass\\_filter\\_freeze\(\)](#)

```
unsigned short sgtl5000_adc_high_pass_filter_freeze ()
```

### 4.67.1.5 [sgtl5000\\_automate\(\)](#)

```
void sgtl5000_automate (
    uint8_t dap,
    uint8_t eq)
```

#### 4.67.1.6 sgtl5000\_dap\_audio\_eq\_band()

```
unsigned short sgtl5000_dap_audio_eq_band (  
    uint8_t band_num,  
    float n)
```

#### 4.67.1.7 sgtl5000\_enable()

```
int sgtl5000_enable ()
```

#### 4.67.1.8 sgtl5000\_healthy()

```
int sgtl5000_healthy ()
```

#### 4.67.1.9 sgtl5000\_kill\_automation()

```
void sgtl5000_kill_automation ()
```

#### 4.67.1.10 sgtl5000\_line\_in\_level()

```
int sgtl5000_line_in_level (  
    uint8_t n)
```

#### 4.67.1.11 sgtl5000\_line\_out\_level()

```
unsigned short sgtl5000_line_out_level (  
    uint8_t n)
```

#### 4.67.1.12 sgtl5000\_mic\_gain()

```
int sgtl5000_mic_gain (  
    unsigned int dB)
```

#### 4.67.1.13 sgtl5000\_modify\_reg()

```
unsigned int sgtl5000_modify_reg (  
    unsigned int reg,  
    unsigned int val,  
    unsigned int i_mask)
```

#### 4.67.1.14 sgtl5000\_mute\_headphone()

```
int sgtl5000_mute_headphone ()
```



**4.67.1.15 sgtl5000\_mute\_line\_out()**

```
int sgtl5000_mute_line_out ()
```

**4.67.1.16 sgtl5000\_read\_reg()**

```
unsigned int sgtl5000_read_reg (  
    unsigned int reg)
```

**4.67.1.17 sgtl5000\_set\_address()**

```
void sgtl5000_set_address (  
    uint8_t level)
```

**4.67.1.18 sgtl5000\_set\_master\_mode()**

```
void sgtl5000_set_master_mode (  
    uint32_t freqMCLK_in)
```

**4.67.1.19 sgtl5000\_soft\_reboot()**

```
int sgtl5000_soft_reboot ()
```

**4.67.1.20 sgtl5000\_start()**

```
int sgtl5000_start ()
```

**4.67.1.21 sgtl5000\_unmute\_headphone()**

```
int sgtl5000_unmute_headphone ()
```

**4.67.1.22 sgtl5000\_unmute\_line\_out()**

```
int sgtl5000_unmute_line_out ()
```

**4.67.1.23 sgtl5000\_volum\_eng\_integer()**

```
int sgtl5000_volum_eng_integer (  
    unsigned int n)
```

#### 4.67.1.24 sgtl5000\_volume()

```
int sgtl5000_volume (
    float n)
```

#### 4.67.1.25 sgtl5000\_write\_reg()

```
int sgtl5000_write_reg (
    unsigned int reg,
    unsigned int val)
```

### 4.68 m\_eng\_sgtl5000.h

[Go to the documentation of this file.](#)

```
00001 #ifndef control_sgtl5000_h_
00002 #define control_sgtl5000_h_
00003
00004 int sgtl5000_start();
00005 int sgtl5000_soft_reboot();
00006
00007 int sgtl5000_healthy();
00008
00009 int sgtl5000_enable();
00010
00011 void sgtl5000_set_address(uint8_t level);
00012
00013 int sgtl5000_volume(float n);
00014 int sgtl5000_mute_headphone();
00015 int sgtl5000_unmute_headphone();
00016 int sgtl5000_mute_line_out();
00017 int sgtl5000_unmute_line_out();
00018
00019 int sgtl5000_mic_gain(unsigned int dB);
00020 int sgtl5000_line_in_level(uint8_t n);
00021 unsigned short sgtl5000_line_out_level(uint8_t n);
00022 unsigned short sgtl5000_adc_high_pass_filter_enable();
00023 unsigned short sgtl5000_adc_high_pass_filter_freeze();
00024 unsigned short sgtl5000_adc_high_pass_filter_disable();
00025 void sgtl5000_kill_automation();
00026 void sgtl5000_set_master_mode(uint32_t freqMCLK_in);
00027
00028 int sgtl5000_volum_eng_integer (unsigned int n); // range: 0x00 to 0x80
00029 unsigned int sgtl5000_read_reg (unsigned int reg);
00030 int sgtl5000_write_reg (unsigned int reg, unsigned int val);
00031 unsigned int sgtl5000_modify_reg (unsigned int reg, unsigned int val, unsigned int i_mask);
00032 unsigned short sgtl5000_dap_audio_eq_band(uint8_t band_num, float n);
00033
00034 void sgtl5000_automate(uint8_t dap, uint8_t eq);
00035
00036 unsigned char calc_vol(float n, unsigned char range);
00037
00038 #endif
```

### 4.69 m\_eng\_sgtl5000\_defs.h File Reference

#### Macros

- #define `CHIP_ID` 0x0000
- #define `CHIP_DIG_POWER` 0x0002
- #define `CHIP_CLK_CTRL` 0x0004
- #define `CHIP_I2S_CTRL` 0x0006
- #define `CHIP_SSS_CTRL` 0x000A
- #define `CHIP_ADCDAC_CTRL` 0x000E

- #define [CHIP\\_DAC\\_VOL](#) 0x0010
- #define [CHIP\\_PAD\\_STRENGTH](#) 0x0014
- #define [CHIP\\_ANA\\_ADC\\_CTRL](#) 0x0020
- #define [CHIP\\_ANA\\_HP\\_CTRL](#) 0x0022
- #define [CHIP\\_ANA\\_CTRL](#) 0x0024
- #define [CHIP\\_LINREG\\_CTRL](#) 0x0026
- #define [CHIP\\_REF\\_CTRL](#) 0x0028
- #define [CHIP\\_MIC\\_CTRL](#) 0x002A
- #define [CHIP\\_LINE\\_OUT\\_CTRL](#) 0x002C
- #define [CHIP\\_LINE\\_OUT\\_VOL](#) 0x002E
- #define [CHIP\\_ANA\\_POWER](#) 0x0030
- #define [CHIP\\_PLL\\_CTRL](#) 0x0032
- #define [CHIP\\_CLK\\_TOP\\_CTRL](#) 0x0034
- #define [CHIP\\_ANA\\_STATUS](#) 0x0036
- #define [CHIP\\_ANA\\_TEST1](#) 0x0038
- #define [CHIP\\_ANA\\_TEST2](#) 0x003A
- #define [CHIP\\_SHORT\\_CTRL](#) 0x003C
- #define [DAP\\_CONTROL](#) 0x0100
- #define [DAP\\_PEQ](#) 0x0102
- #define [DAP\\_BASS\\_ENHANCE](#) 0x0104
- #define [DAP\\_BASS\\_ENHANCE\\_CTRL](#) 0x0106
- #define [DAP\\_AUDIO\\_EQ](#) 0x0108
- #define [DAP\\_SGTL\\_SURROUND](#) 0x010A
- #define [DAP\\_FILTER\\_COEF\\_ACCESS](#) 0x010C
- #define [DAP\\_COEF\\_WR\\_B0\\_MSB](#) 0x010E
- #define [DAP\\_COEF\\_WR\\_B0\\_LSB](#) 0x0110
- #define [DAP\\_AUDIO\\_EQ\\_BASS\\_BAND0](#) 0x0116
- #define [DAP\\_AUDIO\\_EQ\\_BAND1](#) 0x0118
- #define [DAP\\_AUDIO\\_EQ\\_BAND2](#) 0x011A
- #define [DAP\\_AUDIO\\_EQ\\_BAND3](#) 0x011C
- #define [DAP\\_AUDIO\\_EQ\\_TREBLE\\_BAND4](#) 0x011E
- #define [DAP\\_MAIN\\_CHAN](#) 0x0120
- #define [DAP\\_MIX\\_CHAN](#) 0x0122
- #define [DAP\\_AVC\\_CTRL](#) 0x0124
- #define [DAP\\_AVC\\_THRESHOLD](#) 0x0126
- #define [DAP\\_AVC\\_ATTACK](#) 0x0128
- #define [DAP\\_AVC\\_DECAY](#) 0x012A
- #define [DAP\\_COEF\\_WR\\_B1\\_MSB](#) 0x012C
- #define [DAP\\_COEF\\_WR\\_B1\\_LSB](#) 0x012E
- #define [DAP\\_COEF\\_WR\\_B2\\_MSB](#) 0x0130
- #define [DAP\\_COEF\\_WR\\_B2\\_LSB](#) 0x0132
- #define [DAP\\_COEF\\_WR\\_A1\\_MSB](#) 0x0134
- #define [DAP\\_COEF\\_WR\\_A1\\_LSB](#) 0x0136
- #define [DAP\\_COEF\\_WR\\_A2\\_MSB](#) 0x0138
- #define [DAP\\_COEF\\_WR\\_A2\\_LSB](#) 0x013A
- #define [SGTL5000\\_I2C\\_ADDR\\_CS\\_LOW](#) 0x0A
- #define [SGTL5000\\_I2C\\_ADDR\\_CS\\_HIGH](#) 0x2A
- #define [FILTER\\_LOPASS](#) 0
- #define [FILTER\\_HIPASS](#) 1
- #define [FILTER\\_BANDPASS](#) 2
- #define [FILTER\\_NOTCH](#) 3
- #define [FILTER\\_PARAEQ](#) 4
- #define [FILTER\\_LOSHELF](#) 5
- #define [FILTER\\_HISHELF](#) 6
- #define [FLAT\\_FREQUENCY](#) 0

- `#define` `PARAMETRIC_EQUALIZER` 1
- `#define` `TONE_CONTROLS` 2
- `#define` `GRAPHIC_EQUALIZER` 3
- `#define` `AUDIO_HEADPHONE_DAC` 0
- `#define` `AUDIO_HEADPHONE_LINEIN` 1

## 4.69.1 Macro Definition Documentation

### 4.69.1.1 `AUDIO_HEADPHONE_DAC`

```
#define AUDIO_HEADPHONE_DAC 0
```

### 4.69.1.2 `AUDIO_HEADPHONE_LINEIN`

```
#define AUDIO_HEADPHONE_LINEIN 1
```

### 4.69.1.3 `CHIP_ADCDAC_CTRL`

```
#define CHIP_ADCDAC_CTRL 0x000E
```

### 4.69.1.4 `CHIP_ANA_ADC_CTRL`

```
#define CHIP_ANA_ADC_CTRL 0x0020
```

### 4.69.1.5 `CHIP_ANA_CTRL`

```
#define CHIP_ANA_CTRL 0x0024
```

### 4.69.1.6 `CHIP_ANA_HP_CTRL`

```
#define CHIP_ANA_HP_CTRL 0x0022
```

### 4.69.1.7 `CHIP_ANA_POWER`

```
#define CHIP_ANA_POWER 0x0030
```

### 4.69.1.8 `CHIP_ANA_STATUS`

```
#define CHIP_ANA_STATUS 0x0036
```

#### 4.69.1.9 CHIP\_ANA\_TEST1

```
#define CHIP_ANA_TEST1 0x0038
```

#### 4.69.1.10 CHIP\_ANA\_TEST2

```
#define CHIP_ANA_TEST2 0x003A
```

#### 4.69.1.11 CHIP\_CLK\_CTRL

```
#define CHIP_CLK_CTRL 0x0004
```

#### 4.69.1.12 CHIP\_CLK\_TOP\_CTRL

```
#define CHIP_CLK_TOP_CTRL 0x0034
```

#### 4.69.1.13 CHIP\_DAC\_VOL

```
#define CHIP_DAC_VOL 0x0010
```

#### 4.69.1.14 CHIP\_DIG\_POWER

```
#define CHIP_DIG_POWER 0x0002
```

#### 4.69.1.15 CHIP\_I2S\_CTRL

```
#define CHIP_I2S_CTRL 0x0006
```

#### 4.69.1.16 CHIP\_ID

```
#define CHIP_ID 0x0000
```

#### 4.69.1.17 CHIP\_LINE\_OUT\_CTRL

```
#define CHIP_LINE_OUT_CTRL 0x002C
```

#### 4.69.1.18 CHIP\_LINE\_OUT\_VOL

```
#define CHIP_LINE_OUT_VOL 0x002E
```

#### 4.69.1.19 CHIP\_LINREG\_CTRL

```
#define CHIP_LINREG_CTRL 0x0026
```

#### 4.69.1.20 CHIP\_MIC\_CTRL

```
#define CHIP_MIC_CTRL 0x002A
```

#### 4.69.1.21 CHIP\_PAD\_STRENGTH

```
#define CHIP_PAD_STRENGTH 0x0014
```

#### 4.69.1.22 CHIP\_PLL\_CTRL

```
#define CHIP_PLL_CTRL 0x0032
```

#### 4.69.1.23 CHIP\_REF\_CTRL

```
#define CHIP_REF_CTRL 0x0028
```

#### 4.69.1.24 CHIP\_SHORT\_CTRL

```
#define CHIP_SHORT_CTRL 0x003C
```

#### 4.69.1.25 CHIP\_SSS\_CTRL

```
#define CHIP_SSS_CTRL 0x000A
```

#### 4.69.1.26 DAP\_AUDIO\_EQ

```
#define DAP_AUDIO_EQ 0x0108
```

#### 4.69.1.27 DAP\_AUDIO\_EQ\_BAND1

```
#define DAP_AUDIO_EQ_BAND1 0x0118
```

#### 4.69.1.28 DAP\_AUDIO\_EQ\_BAND2

```
#define DAP_AUDIO_EQ_BAND2 0x011A
```

**4.69.1.29 DAP\_AUDIO\_EQ\_BAND3**

```
#define DAP_AUDIO_EQ_BAND3 0x011C
```

**4.69.1.30 DAP\_AUDIO\_EQ\_BASS\_BAND0**

```
#define DAP_AUDIO_EQ_BASS_BAND0 0x0116
```

**4.69.1.31 DAP\_AUDIO\_EQ\_TREBLE\_BAND4**

```
#define DAP_AUDIO_EQ_TREBLE_BAND4 0x011E
```

**4.69.1.32 DAP\_AVC\_ATTACK**

```
#define DAP_AVC_ATTACK 0x0128
```

**4.69.1.33 DAP\_AVC\_CTRL**

```
#define DAP_AVC_CTRL 0x0124
```

**4.69.1.34 DAP\_AVC\_DECAY**

```
#define DAP_AVC_DECAY 0x012A
```

**4.69.1.35 DAP\_AVC\_THRESHOLD**

```
#define DAP_AVC_THRESHOLD 0x0126
```

**4.69.1.36 DAP\_BASS\_ENHANCE**

```
#define DAP_BASS_ENHANCE 0x0104
```

**4.69.1.37 DAP\_BASS\_ENHANCE\_CTRL**

```
#define DAP_BASS_ENHANCE_CTRL 0x0106
```

**4.69.1.38 DAP\_COEF\_WR\_A1\_LSB**

```
#define DAP_COEF_WR_A1_LSB 0x0136
```

**4.69.1.39 DAP\_COEF\_WR\_A1\_MSB**

```
#define DAP_COEF_WR_A1_MSB 0x0134
```

**4.69.1.40 DAP\_COEF\_WR\_A2\_LSB**

```
#define DAP_COEF_WR_A2_LSB 0x013A
```

**4.69.1.41 DAP\_COEF\_WR\_A2\_MSB**

```
#define DAP_COEF_WR_A2_MSB 0x0138
```

**4.69.1.42 DAP\_COEF\_WR\_B0\_LSB**

```
#define DAP_COEF_WR_B0_LSB 0x0110
```

**4.69.1.43 DAP\_COEF\_WR\_B0\_MSB**

```
#define DAP_COEF_WR_B0_MSB 0x010E
```

**4.69.1.44 DAP\_COEF\_WR\_B1\_LSB**

```
#define DAP_COEF_WR_B1_LSB 0x012E
```

**4.69.1.45 DAP\_COEF\_WR\_B1\_MSB**

```
#define DAP_COEF_WR_B1_MSB 0x012C
```

**4.69.1.46 DAP\_COEF\_WR\_B2\_LSB**

```
#define DAP_COEF_WR_B2_LSB 0x0132
```

**4.69.1.47 DAP\_COEF\_WR\_B2\_MSB**

```
#define DAP_COEF_WR_B2_MSB 0x0130
```

**4.69.1.48 DAP\_CONTROL**

```
#define DAP_CONTROL 0x0100
```



#### 4.69.1.49 DAP\_FILTER\_COEF\_ACCESS

```
#define DAP_FILTER_COEF_ACCESS 0x010C
```

#### 4.69.1.50 DAP\_MAIN\_CHAN

```
#define DAP_MAIN_CHAN 0x0120
```

#### 4.69.1.51 DAP\_MIX\_CHAN

```
#define DAP_MIX_CHAN 0x0122
```

#### 4.69.1.52 DAP\_PEQ

```
#define DAP_PEQ 0x0102
```

#### 4.69.1.53 DAP\_SGTL\_SURROUND

```
#define DAP_SGTL_SURROUND 0x010A
```

#### 4.69.1.54 FILTER\_BANDPASS

```
#define FILTER_BANDPASS 2
```

#### 4.69.1.55 FILTER\_HIPASS

```
#define FILTER_HIPASS 1
```

#### 4.69.1.56 FILTER\_HISHELF

```
#define FILTER_HISHELF 6
```

#### 4.69.1.57 FILTER\_LOPASS

```
#define FILTER_LOPASS 0
```

#### 4.69.1.58 FILTER\_LOSHELF

```
#define FILTER_LOSHELF 5
```

#### 4.69.1.59 FILTER\_NOTCH

```
#define FILTER_NOTCH 3
```

#### 4.69.1.60 FILTER\_PARAEQ

```
#define FILTER_PARAEQ 4
```

#### 4.69.1.61 FLAT\_FREQUENCY

```
#define FLAT_FREQUENCY 0
```

#### 4.69.1.62 GRAPHIC\_EQUALIZER

```
#define GRAPHIC_EQUALIZER 3
```

#### 4.69.1.63 PARAMETRIC\_EQUALIZER

```
#define PARAMETRIC_EQUALIZER 1
```

#### 4.69.1.64 SGTL5000\_I2C\_ADDR\_CS\_HIGH

```
#define SGTL5000_I2C_ADDR_CS_HIGH 0x2A
```

#### 4.69.1.65 SGTL5000\_I2C\_ADDR\_CS\_LOW

```
#define SGTL5000_I2C_ADDR_CS_LOW 0x0A
```

#### 4.69.1.66 TONE\_CONTROLS

```
#define TONE_CONTROLS 2
```

## 4.70 m\_eng\_sgtl5000\_defs.h

[Go to the documentation of this file.](#)

```

00001 #ifndef M_SGTL5000_DEFS_H_
00002 #define M_SGTL5000_DEFS_H_
00003
00004 #define CHIP_ID 0x0000
00005 // 15:8 PARTID 0xA0 - 8 bit identifier for SGTL5000
00006 // 7:0 REVID 0x00 - revision number for SGTL5000.
00007
00008 #define CHIP_DIG_POWER 0x0002
00009 // 6 ADC_POWERUP 1=Enable, 0=disable the ADC block, both digital & analog,
00010 // 5 DAC_POWERUP 1=Enable, 0=disable the DAC block, both analog and digital
00011 // 4 DAP_POWERUP 1=Enable, 0=disable the DAP block
00012 // 1 I2S_OUT_POWERUP 1=Enable, 0=disable the I2S data output
00013 // 0 I2S_IN_POWERUP 1=Enable, 0=disable the I2S data input
00014
00015 #define CHIP_CLK_CTRL 0x0004
00016 // 5:4 RATE_MODE Sets the sample rate mode. MCLK_FREQ is still specified
00017 // relative to the rate in SYS_FS
00018 // 0x0 = SYS_FS specifies the rate
00019 // 0x1 = Rate is 1/2 of the SYS_FS rate
00020 // 0x2 = Rate is 1/4 of the SYS_FS rate
00021 // 0x3 = Rate is 1/6 of the SYS_FS rate
00022 // 3:2 SYS_FS Sets the internal system sample rate (default=2)
00023 // 0x0 = 32 kHz
00024 // 0x1 = 44.1 kHz
00025 // 0x2 = 48 kHz
00026 // 0x3 = 96 kHz
00027 // 1:0 MCLK_FREQ Identifies incoming SYS_MCLK frequency and if the PLL should be used
00028 // 0x0 = 256*Fs
00029 // 0x1 = 384*Fs
00030 // 0x2 = 512*Fs
00031 // 0x3 = Use PLL
00032 // The 0x3 (Use PLL) setting must be used if the SYS_MCLK is not
00033 // a standard multiple of Fs (256, 384, or 512). This setting can
00034 // also be used if SYS_MCLK is a standard multiple of Fs.
00035 // Before this field is set to 0x3 (Use PLL), the PLL must be
00036 // powered up by setting CHIP_ANA_POWER->PLL_POWERUP and
00037 // CHIP_ANA_POWER->VCOAMP_POWERUP. Also, the PLL dividers must
00038 // be calculated based on the external MCLK rate and
00039 // CHIP_PLL_CTRL register must be set (see CHIP_PLL_CTRL register
00040 // description details on how to calculate the divisors).
00041
00042 #define CHIP_I2S_CTRL 0x0006
00043 // 8 SCLK_FREQ Sets frequency of I2S_SCLK when in master mode (MS=1). When in slave
00044 // mode (MS=0), this field must be set appropriately to match SCLK input
00045 // rate.
00046 // 0x0 = 64Fs
00047 // 0x1 = 32Fs - Not supported for RJ mode (I2S_MODE = 1)
00048 // 7 MS Configures master or slave of I2S_LRCLK and I2S_SCLK.
00049 // 0x0 = Slave: I2S_LRCLK and I2S_SCLK are inputs
00050 // 0x1 = Master: I2S_LRCLK and I2S_SCLK are outputs
00051 // NOTE: If the PLL is used (CHIP_CLK_CTRL->MCLK_FREQ==0x3),
00052 // the SGTL5000 must be a master of the I2S port (MS==1)
00053 // 6 SCLK_INV Sets the edge that data (input and output) is clocked in on for I2S_SCLK
00054 // 0x0 = data is valid on rising edge of I2S_SCLK
00055 // 0x1 = data is valid on falling edge of I2S_SCLK
00056 // 5:4 DLEN I2S data length (default=1)
00057 // 0x0 = 32 bits (only valid when SCLK_FREQ=0),
00058 // not valid for Right Justified Mode
00059 // 0x1 = 24 bits (only valid when SCLK_FREQ=0)
00060 // 0x2 = 20 bits
00061 // 0x3 = 16 bits
00062 // 3:2 I2S_MODE Sets the mode for the I2S port
00063 // 0x0 = I2S mode or Left Justified (Use LRALIGN to select)
00064 // 0x1 = Right Justified Mode
00065 // 0x2 = PCM Format A/B
00066 // 0x3 = RESERVED
00067 // 1 LRALIGN I2S_LRCLK Alignment to data word. Not used for Right Justified mode
00068 // 0x0 = Data word starts 1 I2S_SCLK delay after I2S_LRCLK
00069 // transition (I2S format, PCM format A)
00070 // 0x1 = Data word starts after I2S_LRCLK transition (left
00071 // justified format, PCM format B)
00072 // 0 LRPOL I2S_LRCLK Polarity when data is presented.
00073 // 0x0 = I2S_LRCLK = 0 - Left, 1 - Right
00074 // 1x0 = I2S_LRCLK = 0 - Right, 1 - Left
00075 // The left subframe should be presented first regardless of
00076 // the setting of LRPOL.
00077
00078 #define CHIP_SSS_CTRL 0x000A
00079 // 14 DAP_MIX_LRSWAP DAP Mixer Input Swap
00080 // 0x0 = Normal Operation
00081 // 0x1 = Left and Right channels for the DAP MIXER Input are swapped.
00082 // 13 DAP_LRSWAP DAP Mixer Input Swap

```

```

00083 //          0x0 = Normal Operation
00084 //          0x1 = Left and Right channels for the DAP Input are swapped
00085 // 12  DAC_LRSWAP  DAC Input Swap
00086 //          0x0 = Normal Operation
00087 //          0x1 = Left and Right channels for the DAC are swapped
00088 // 10  I2S_LRSWAP  I2S_DOUT Swap
00089 //          0x0 = Normal Operation
00090 //          0x1 = Left and Right channels for the I2S_DOUT are swapped
00091 // 9:8  DAP_MIX_SELECT  Select data source for DAP mixer
00092 //          0x0 = ADC
00093 //          0x1 = I2S_IN
00094 //          0x2 = Reserved
00095 //          0x3 = Reserved
00096 // 7:6  DAP_SELECT  Select data source for DAP
00097 //          0x0 = ADC
00098 //          0x1 = I2S_IN
00099 //          0x2 = Reserved
00100 //          0x3 = Reserved
00101 // 5:4  DAC_SELECT  Select data source for DAC (default=1)
00102 //          0x0 = ADC
00103 //          0x1 = I2S_IN
00104 //          0x2 = Reserved
00105 //          0x3 = DAP
00106 // 1:0  I2S_SELECT  Select data source for I2S_DOUT
00107 //          0x0 = ADC
00108 //          0x1 = I2S_IN
00109 //          0x2 = Reserved
00110 //          0x3 = DAP
00111
00112 #define CHIP_ADCDAC_CTRL          0x000E
00113 // 13  VOL_BUSY_DAC_RIGHT  Volume Busy DAC Right
00114 //          0x0 = Ready
00115 //          0x1 = Busy - This indicates the channel has not reached its
00116 //                programmed volume/mute level
00117 // 12  VOL_BUSY_DAC_LEFT  Volume Busy DAC Left
00118 //          0x0 = Ready
00119 //          0x1 = Busy - This indicates the channel has not reached its
00120 //                programmed volume/mute level
00121 // 9  VOL_RAMP_EN  Volume Ramp Enable (default=1)
00122 //          0x0 = Disables volume ramp. New volume settings take immediate
00123 //                effect without a ramp
00124 //          0x1 = Enables volume ramp
00125 //          This field affects DAC_VOL. The volume ramp effects both
00126 //          volume settings and mute. When set to 1 a soft mute is enabled.
00127 // 8  VOL_EXPO_RAMP  Exponential Volume Ramp Enable
00128 //          0x0 = Linear ramp over top 4 volume octaves
00129 //          0x1 = Exponential ramp over full volume range
00130 //          This bit only takes effect if VOL_RAMP_EN is 1.
00131 // 3  DAC_MUTE_RIGHT  DAC Right Mute (default=1)
00132 //          0x0 = Unmute
00133 //          0x1 = Muted
00134 //          If VOL_RAMP_EN = 1, this is a soft mute.
00135 // 2  DAC_MUTE_LEFT  DAC Left Mute (default=1)
00136 //          0x0 = Unmute
00137 //          0x1 = Muted
00138 //          If VOL_RAMP_EN = 1, this is a soft mute.
00139 // 1  ADC_HPF_FREEZE  ADC High Pass Filter Freeze
00140 //          0x0 = Normal operation
00141 //          0x1 = Freeze the ADC high-pass filter offset register. The
00142 //                offset continues to be subtracted from the ADC data stream.
00143 // 0  ADC_HPF_BYPASS  ADC High Pass Filter Bypass
00144 //          0x0 = Normal operation
00145 //          0x1 = Bypassed and offset not updated
00146
00147 #define CHIP_DAC_VOL          0x0010
00148 // 15:8  DAC_VOL_RIGHT  DAC Right Channel Volume. Set the Right channel DAC volume
00149 //          with 0.5017 dB steps from 0 to -90 dB
00150 //          0x3B and less = Reserved
00151 //          0x3C = 0 dB
00152 //          0x3D = -0.5 dB
00153 //          0xF0 = -90 dB
00154 //          0xFC and greater = Muted
00155 //          If VOL_RAMP_EN = 1, there is an automatic ramp to the
00156 //          new volume setting.
00157 // 7:0  DAC_VOL_LEFT  DAC Left Channel Volume. Set the Left channel DAC volume
00158 //          with 0.5017 dB steps from 0 to -90 dB
00159 //          0x3B and less = Reserved
00160 //          0x3C = 0 dB
00161 //          0x3D = -0.5 dB
00162 //          0xF0 = -90 dB
00163 //          0xFC and greater = Muted
00164 //          If VOL_RAMP_EN = 1, there is an automatic ramp to the
00165 //          new volume setting.
00166
00167 #define CHIP_PAD_STRENGTH          0x0014
00168 // 9:8  I2S_LRCLK  I2S LRCLK Pad Drive Strength (default=1)
00169 //          Sets drive strength for output pads per the table below.

```

```

00170 //          VDDIO 1.8 V    2.5 V    3.3 V
00171 //          0x0 = Disable
00172 //          0x1 =    1.66 mA    2.87 mA    4.02 mA
00173 //          0x2 =    3.33 mA    5.74 mA    8.03 mA
00174 //          0x3 =    4.99 mA    8.61 mA   12.05 mA
00175 // 7:6 I2S_SCLK    I2S SCLK Pad Drive Strength (default=1)
00176 // 5:4 I2S_DOUT    I2S DOUT Pad Drive Strength (default=1)
00177 // 3:2 CTRL_DATA    I2C DATA Pad Drive Strength (default=3)
00178 // 1:0 CTRL_CLK    I2C CLK Pad Drive Strength (default=3)
00179 //          (all use same table as I2S_LRCLK)
00180
00181 #define CHIP_ANA_ADC_CTRL    0x0020
00182 // 8    ADC_VOL_M6DB    ADC Volume Range Reduction
00183 //          This bit shifts both right and left analog ADC volume
00184 //          range down by 6.0 dB.
00185 //          0x0 = No change in ADC range
00186 //          0x1 = ADC range reduced by 6.0 dB
00187 // 7:4    ADC_VOL_RIGHT    ADC Right Channel Volume
00188 //          Right channel analog ADC volume control in 1.5 dB steps.
00189 //          0x0 = 0 dB
00190 //          0x1 = +1.5 dB
00191 //          ...
00192 //          0xF = +22.5 dB
00193 //          This range is -6.0 dB to +16.5 dB if ADC_VOL_M6DB is set to 1.
00194 // 3:0    ADC_VOL_LEFT    ADC Left Channel Volume
00195 //          (same scale as ADC_VOL_RIGHT)
00196
00197 #define CHIP_ANA_HP_CTRL    0x0022
00198 // 14:8 HP_VOL_RIGHT    Headphone Right Channel Volume (default 0x18)
00199 //          Right channel headphone volume control with 0.5 dB steps.
00200 //          0x00 = +12 dB
00201 //          0x01 = +11.5 dB
00202 //          0x18 = 0 dB
00203 //          ...
00204 //          0x7F = -51.5 dB
00205 // 6:0    HP_VOL_LEFT    Headphone Left Channel Volume (default 0x18)
00206 //          (same scale as HP_VOL_RIGHT)
00207
00208 #define CHIP_ANA_CTRL    0x0024
00209 // 8    MUTE_LO    LINEOUT Mute, 0 = Unmute, 1 = Mute (default 1)
00210 // 6    SELECT_HP    Select the headphone input, 0 = DAC, 1 = LINEIN
00211 // 5    EN_ZCD_HP    Enable the headphone zero cross detector (ZCD)
00212 //          0x0 = HP ZCD disabled
00213 //          0x1 = HP ZCD enabled
00214 // 4    MUTE_HP    Mute the headphone outputs, 0 = Unmute, 1 = Mute (default)
00215 // 2    SELECT_ADC    Select the ADC input, 0 = Microphone, 1 = LINEIN
00216 // 1    EN_ZCD_ADC    Enable the ADC analog zero cross detector (ZCD)
00217 //          0x0 = ADC ZCD disabled
00218 //          0x1 = ADC ZCD enabled
00219 // 0    MUTE_ADC    Mute the ADC analog volume, 0 = Unmute, 1 = Mute (default)
00220
00221 #define CHIP_LINREG_CTRL    0x0026
00222 // 6    VDDC_MAN_ASSN    Determines chargepump source when VDDC_ASSN_OVRD is set.
00223 //          0x0 = VDDA
00224 //          0x1 = VDDIO
00225 // 5    VDDC_ASSN_OVRD    Charge pump Source Assignment Override
00226 //          0x0 = Charge pump source is automatically assigned based
00227 //          on higher of VDDA and VDDIO
00228 //          0x1 = the source of charge pump is manually assigned by
00229 //          VDDC_MAN_ASSN If VDDIO and VDDA are both the same
00230 //          and greater than 3.1 V, VDDC_ASSN_OVRD and
00231 //          VDDC_MAN_ASSN should be used to manually assign
00232 //          VDDIO as the source for charge pump.
00233 // 3:0 D_PROGRAMMING    Sets the VDDD linear regulator output voltage in 50 mV steps.
00234 //          Must clear the LINREG_SIMPLE_POWERUP and STARTUP_POWERUP bits
00235 //          in the 0x0030 (CHIP_ANA_POWER) register after power-up, for
00236 //          this setting to produce the proper VDDD voltage.
00237 //          0x0 = 1.60
00238 //          0xF = 0.85
00239
00240 #define CHIP_REF_CTRL    0x0028 // bandgap reference bias voltage and currents
00241 // 8:4 VAG_VAL    Analog Ground Voltage Control
00242 //          These bits control the analog ground voltage in 25 mV steps.
00243 //          This should usually be set to VDDA/2 or lower for best
00244 //          performance (maximum output swing at minimum THD). This VAG
00245 //          reference is also used for the DAC and ADC voltage reference.
00246 //          So changing this voltage scales the output swing of the DAC
00247 //          and the output signal of the ADC.
00248 //          0x00 = 0.800 V
00249 //          0x1F = 1.575 V
00250 // 3:1 BIAS_CTRL    Bias control
00251 //          These bits adjust the bias currents for all of the analog
00252 //          blocks. By lowering the bias current a lower quiescent power
00253 //          is achieved. It should be noted that this mode can affect
00254 //          performance by 3-4 dB.
00255 //          0x0 = Nominal
00256 //          0x1-0x3=+12.5%

```

```

00257 //          0x4=-12.5%
00258 //          0x5=-25%
00259 //          0x6=-37.5%
00260 //          0x7=-50%
00261 // 0      SMALL_POP      VAG Ramp Control
00262 //          Setting this bit slows down the VAG ramp from ~200 to ~400 ms
00263 //          to reduce the startup pop, but increases the turn on/off time.
00264 //          0x0 = Normal VAG ramp
00265 //          0x1 = Slow down VAG ramp
00266
00267 #define CHIP_MIC_CTRL      0x002A // microphone gain & internal microphone bias
00268 // 9:8      BIAS_RESISTOR      MIC Bias Output Impedance Adjustment
00269 //          Controls an adjustable output impedance for the microphone bias.
00270 //          If this is set to zero the micbias block is powered off and
00271 //          the output is highZ.
00272 //          0x0 = Powered off
00273 //          0x1 = 2.0 kohm
00274 //          0x2 = 4.0 kohm
00275 //          0x3 = 8.0 kohm
00276 // 6:4      BIAS_VOLT      MIC Bias Voltage Adjustment
00277 //          Controls an adjustable bias voltage for the microphone bias
00278 //          amp in 250 mV steps. This bias voltage setting should be no
00279 //          more than VDDA-200 mV for adequate power supply rejection.
00280 //          0x0 = 1.25 V
00281 //          ...
00282 //          0x7 = 3.00 V
00283 // 1:0      GAIN          MIC Amplifier Gain
00284 //          Sets the microphone amplifier gain. At 0 dB setting the THD
00285 //          can be slightly higher than other paths- typically around
00286 //          ~65 dB. At other gain settings the THD are better.
00287 //          0x0 = 0 dB
00288 //          0x1 = +20 dB
00289 //          0x2 = +30 dB
00290 //          0x3 = +40 dB
00291
00292 #define CHIP_LINE_OUT_CTRL      0x002C
00293 // 11:8      OUT_CURRENT      Controls the output bias current for the LINEOUT amplifiers. The
00294 //          nominal recommended setting for a 10 kohm load with 1.0 nF load cap
00295 //          is 0x3. There are only 5 valid settings.
00296 //          0x0=0.18 mA
00297 //          0x1=0.27 mA
00298 //          0x3=0.36 mA
00299 //          0x7=0.45 mA
00300 //          0xF=0.54 mA
00301 // 5:0      LO_VAGCNTRL      LINEOUT Amplifier Analog Ground Voltage
00302 //          Controls the analog ground voltage for the LINEOUT amplifiers
00303 //          in 25 mV steps. This should usually be set to VDDIO/2.
00304 //          0x00 = 0.800 V
00305 //          ...
00306 //          0x1F = 1.575 V
00307 //          ...
00308 //          0x23 = 1.675 V
00309 //          0x24-0x3F are invalid
00310
00311 #define CHIP_LINE_OUT_VOL      0x002E
00312 // 12:8      LO_VOL_RIGHT      LINEOUT Right Channel Volume (default=4)
00313 //          Controls the right channel LINEOUT volume in 0.5 dB steps.
00314 //          Higher codes have more attenuation.
00315 // 4:0      LO_VOL_LEFT      LINEOUT Left Channel Output Level (default=4)
00316 //          Used to normalize the output level of the left line output
00317 //          to full scale based on the values used to set
00318 //          LINE_OUT_CTRL->LO_VAGCNTRL and CHIP_REF_CTRL->VAG_VAL.
00319 //          In general this field should be set to:
00320 //          40*log((VAG_VAL)/(LO_VAGCNTRL)) + 15
00321 //          Suggested values based on typical VDDIO and VDDA voltages.
00322 //          VDDA  VAG_VAL  VDDIO  LO_VAGCNTRL  LO_VOL_*
00323 //          1.8 V  0.9    3.3 V  1.55    0x06
00324 //          1.8 V  0.9    1.8 V  0.9     0x0F
00325 //          3.3 V  1.55   1.8 V  0.9     0x19
00326 //          3.3 V  1.55   3.3 V  1.55    0x0F
00327 //          After setting to the nominal voltage, this field can be used
00328 //          to adjust the output level in +/-0.5 dB increments by using
00329 //          values higher or lower than the nominal setting.
00330
00331 #define CHIP_ANA_POWER      0x0030 // power down controls for the analog blocks.
00332 //          The only other power-down controls are BIAS_RESISTOR in the MIC_CTRL register
00333 //          and the EN_ZCD control bits in ANA_CTRL.
00334 // 14      DAC_MONO      While DAC_POWERUP is set, this allows the DAC to be put into left only
00335 //          mono operation for power savings. 0=mono, 1=stereo (default)
00336 // 13      LINREG_SIMPLE_POWERUP      Power up the simple (low power) digital supply regulator.
00337 //          After reset, this bit can be cleared IF VDDD is driven
00338 //          externally OR the primary digital linreg is enabled with
00339 //          LINREG_D_POWERUP
00340 // 12      STARTUP_POWERUP      Power up the circuitry needed during the power up ramp and reset.
00341 //          After reset this bit can be cleared if VDDD is coming from
00342 //          an external source.
00343 // 11      VDDC_CHRGPMPOWERUP      Power up the VDDC charge pump block. If neither VDDA or VDDIO

```

```

00344 //          is 3.0 V or larger this bit should be cleared before analog
00345 //          blocks are powered up.
00346 // 10  PLL_POWERUP PLL Power Up, 0 = Power down, 1 = Power up
00347 //          When cleared, the PLL is turned off. This must be set before
00348 //          CHIP_CLK_CTRL->MCLK_FREQ is programmed to 0x3. The
00349 //          CHIP_PLL_CTRL register must be configured correctly before
00350 //          setting this bit.
00351 // 9  LINREG_D_POWERUP Power up the primary VDDD linear regulator, 0 = Power down, 1 = Power up
00352 // 8  VCOAMP_POWERUP Power up the PLL VCO amplifier, 0 = Power down, 1 = Power up
00353 // 7  VAG_POWERUP Power up the VAG reference buffer.
00354 //          Setting this bit starts the power up ramp for the headphone
00355 //          and LINEOUT. The headphone (and/or LINEOUT) powerup should
00356 //          be set BEFORE clearing this bit. When this bit is cleared
00357 //          the power-down ramp is started. The headphone (and/or LINEOUT)
00358 //          powerup should stay set until the VAG is fully ramped down
00359 //          (200 to 400 ms after clearing this bit).
00360 //          0x0 = Power down, 0x1 = Power up
00361 // 6  ADC_MONO While ADC_POWERUP is set, this allows the ADC to be put into left only
00362 //          mono operation for power savings. This mode is useful when
00363 //          only using the microphone input.
00364 //          0x0 = Mono (left only), 0x1 = Stereo
00365 // 5  REFTOP_POWERUP Power up the reference bias currents
00366 //          0x0 = Power down, 0x1 = Power up
00367 //          This bit can be cleared when the part is a sleep state
00368 //          to minimize analog power.
00369 // 4  HEADPHONE_POWERUP Power up the headphone amplifiers
00370 //          0x0 = Power down, 0x1 = Power up
00371 // 3  DAC_POWERUP Power up the DACs
00372 //          0x0 = Power down, 0x1 = Power up
00373 // 2  CAPLESS_HEADPHONE_POWERUP Power up the capless headphone mode
00374 //          0x0 = Power down, 0x1 = Power up
00375 // 1  ADC_POWERUP Power up the ADCs
00376 //          0x0 = Power down, 0x1 = Power up
00377 // 0  LINEOUT_POWERUP Power up the LINEOUT amplifiers
00378 //          0x0 = Power down, 0x1 = Power up
00379
00380 #define CHIP_PLL_CTRL          0x0032
00381 // 15:11 INT_DIVISOR
00382 // 10:0 FRAC_DIVISOR
00383
00384 #define CHIP_CLK_TOP_CTRL      0x0034
00385 // 11  ENABLE_INT_OSC Setting this bit enables an internal oscillator to be used for the
00386 //          zero cross detectors, the short detect recovery, and the
00387 //          charge pump. This allows the I2S clock to be shut off while
00388 //          still operating an analog signal path. This bit can be kept
00389 //          on when the I2S clock is enabled, but the I2S clock is more
00390 //          accurate so it is preferred to clear this bit when I2S is present.
00391 // 3  INPUT_FREQ_DIV2_SYS_MCLK divider before PLL input
00392 //          0x0 = pass through
00393 //          0x1 = SYS_MCLK is divided by 2 before entering PLL
00394 //          This must be set when the input clock is above 17 Mhz. This
00395 //          has no effect when the PLL is powered down.
00396
00397 #define CHIP_ANA_STATUS        0x0036
00398 // 9  LRSHORT_STS This bit is high whenever a short is detected on the left or right
00399 //          channel headphone drivers.
00400 // 8  CSHORT_STS This bit is high whenever a short is detected on the capless headphone
00401 //          common/center channel driver.
00402 // 4  PLL_IS_LOCKED This bit goes high after the PLL is locked.
00403
00404 #define CHIP_ANA_TEST1         0x0038 // intended only for debug.
00405 #define CHIP_ANA_TEST2         0x003A // intended only for debug.
00406
00407 #define CHIP_SHORT_CTRL        0x003C
00408 // 14:12 LVLADJR Right channel headphone short detector in 25 mA steps.
00409 //          0x3=25 mA
00410 //          0x2=50 mA
00411 //          0x1=75 mA
00412 //          0x0=100 mA
00413 //          0x4=125 mA
00414 //          0x5=150 mA
00415 //          0x6=175 mA
00416 //          0x7=200 mA
00417 //          This trip point can vary by ~30% over process so leave plenty
00418 //          of guard band to avoid false trips. This short detect trip
00419 //          point is also effected by the bias current adjustments made
00420 //          by CHIP_REF_CTRL->BIAS_CTRL and by CHIP_ANA_TEST1->HP_IALL_ADJ.
00421 // 10:8 LVLADJL Left channel headphone short detector in 25 mA steps.
00422 //          (same scale as LVLADJR)
00423 // 6:4 LVLADJC Capless headphone center channel short detector in 50 mA steps.
00424 //          0x3=50 mA
00425 //          0x2=100 mA
00426 //          0x1=150 mA
00427 //          0x0=200 mA
00428 //          0x4=250 mA
00429 //          0x5=300 mA
00430 //          0x6=350 mA

```

```

00431 //          0x7=400 mA
00432 // 3:2  MODE_LR      Behavior of left/right short detection
00433 //          0x0 = Disable short detector, reset short detect latch,
00434 //          software view non-latched short signal
00435 //          0x1 = Enable short detector and reset the latch at timeout
00436 //          (every ~50 ms)
00437 //          0x2 = This mode is not used/invalid
00438 //          0x3 = Enable short detector with only manual reset (have
00439 //          to return to 0x0 to reset the latch)
00440 // 1:0  MODE_CM      Behavior of capless headphone central short detection
00441 //          (same settings as MODE_LR)
00442
00443 #define DAP_CONTROL      0x0100
00444 #define DAP_PEQ          0x0102
00445 #define DAP_BASS_ENHANCE 0x0104
00446 #define DAP_BASS_ENHANCE_CTRL 0x0106
00447 #define DAP_AUDIO_EQ     0x0108
00448 #define DAP_SGTL_SURROUND 0x010A
00449 #define DAP_FILTER_COEF_ACCESS 0x010C
00450 #define DAP_COEF_WR_B0_MSB 0x010E
00451 #define DAP_COEF_WR_B0_LSB 0x0110
00452 #define DAP_AUDIO_EQ_BASS_BAND0 0x0116 // 115 Hz
00453 #define DAP_AUDIO_EQ_BAND1 0x0118 // 330 Hz
00454 #define DAP_AUDIO_EQ_BAND2 0x011A // 990 Hz
00455 #define DAP_AUDIO_EQ_BAND3 0x011C // 3000 Hz
00456 #define DAP_AUDIO_EQ_TREBLE_BAND4 0x011E // 9900 Hz
00457 #define DAP_MAIN_CHAN    0x0120
00458 #define DAP_MIX_CHAN     0x0122
00459 #define DAP_AVC_CTRL     0x0124
00460 #define DAP_AVC_THRESHOLD 0x0126
00461 #define DAP_AVC_ATTACK   0x0128
00462 #define DAP_AVC_DECAY    0x012A
00463 #define DAP_COEF_WR_B1_MSB 0x012C
00464 #define DAP_COEF_WR_B1_LSB 0x012E
00465 #define DAP_COEF_WR_B2_MSB 0x0130
00466 #define DAP_COEF_WR_B2_LSB 0x0132
00467 #define DAP_COEF_WR_A1_MSB 0x0134
00468 #define DAP_COEF_WR_A1_LSB 0x0136
00469 #define DAP_COEF_WR_A2_MSB 0x0138
00470 #define DAP_COEF_WR_A2_LSB 0x013A
00471
00472 #define SGTL5000_I2C_ADDR_CS_LOW 0x0A // CTRL_ADR0_CS pin low (normal configuration)
00473 #define SGTL5000_I2C_ADDR_CS_HIGH 0x2A // CTRL_ADR0_CS pin highz
00474
00475 //For Filter Type: 0 = LPF, 1 = HPF, 2 = BPF, 3 = NOTCH, 4 = PeakingEQ, 5 = LowShelf, 6 = HighShelf
00476 #define FILTER_LOPASS 0
00477 #define FILTER_HIPASS 1
00478 #define FILTER_BANDPASS 2
00479 #define FILTER_NOTCH 3
00480 #define FILTER_PARAEQ 4
00481 #define FILTER_LOSHELF 5
00482 #define FILTER_HISHELF 6
00483
00484 //For frequency adjustment
00485 #define FLAT_FREQUENCY 0
00486 #define PARAMETRIC_EQUALIZER 1
00487 #define TONE_CONTROLS 2
00488 #define GRAPHIC_EQUALIZER 3
00489
00490 // SGTL5000-specific defines for headphones
00491 #define AUDIO_HEADPHONE_DAC 0
00492 #define AUDIO_HEADPHONE_LINEIN 1
00493
00494 #endif

```

## 4.71 m\_eng\_simple\_distortion.h File Reference

### Data Structures

- struct [m\\_eng\\_simple\\_distortion\\_str](#)

### Functions

- int [init\\_simple\\_distortion\\_str](#) ([m\\_eng\\_simple\\_distortion\\_str](#) \*str)
- int [reconfigure\\_simple\\_distortion](#) (void \*data\_struct)
- int [calc\\_simple\\_distortion](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)



## 4.71.1 Function Documentation

### 4.71.1.1 calc\_simple\_distortion()

```
int calc_simple_distortion (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

### 4.71.1.2 init\_simple\_distortion\_str()

```
int init_simple_distortion_str (
    m_eng_simple_distortion_str * str)
```

### 4.71.1.3 reconfigure\_simple\_distortion()

```
int reconfigure_simple_distortion (
    void * data_struct)
```

## 4.72 m\_eng\_simple\_distortion.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_SIMPLE_DISTORTION_H_
00002 #define M_ENG_SIMPLE_DISTORTION_H_
00003
00004 typedef struct
00005 {
00006     m_parameter pregain;
00007     m_parameter postgain;
00008 } m_eng_simple_distortion_str;
00009
00010
00011 int init_simple_distortion_str(m_eng_simple_distortion_str *str);
00012 int reconfigure_simple_distortion(void *data_struct);
00013 int calc_simple_distortion(void *data_struct, float *dest, float *src, int n_samples);
00014
00015 #endif
```

## 4.73 m\_eng\_transformer.h File Reference

```
#include "m_transformer_enum.h"
#include "m_parameter.h"
#include "m_eng_linkowitz_riley.h"
#include "m_transformer.h"
```

## Macros

- `#define TRANSFORMER_MAX_INPUTS 4`
- `#define TRANSFORMER_MAX_OUTPUTS 4`
- `#define FADER_FADE_IN 0`
- `#define FADER_FADE_OUT 1`
- `#define PRINT_TRANSFORMER_INFO`
- `#define TRANSFORMER_SWITCH_ACTION_BYPASS 0`
- `#define TRANSFORMER_TRANSITION_INSTANT 0`
- `#define TRANSFORMER_TRANSITION_MONOBLOCK_LINEAR 1`
- `#define TRANSFORMER_TRANSITION_BIBLOCK_LINEAR 2`
- `#define TRANSFORMER_TRANSITION_QUADBLOCK_LINEAR 3`
- `#define TRANSFORMER_TRANSITION_TAIL 4`
- `#define N_NATIVE_PARAMETERS 3`
- `#define N_NATIVE_SETTINGS 1`

## Functions

- `int transformer_add_setting (m_transformer *trans, m_setting *setting)`
- `int transformer_add_parameter (m_transformer *trans, m_parameter *param)`
- `m_parameter * transformer_get_parameter (m_transformer *trans, uint16_t ppid)`
- `m_setting * transformer_get_setting (m_transformer *trans, uint16_t sid)`
- `int transformer_init_parameter_array (m_transformer *trans, int n)`
- `int transformer_init_setting_array (m_transformer *trans, int n)`
- `int run_transformer (m_transformer *trans, float *dest, float *src)`
- `void free_transformer (m_transformer *trans)`
- `int clone_transformer (m_transformer **dest_ptr, m_transformer *src)`
- `const char * transformer_type_to_string (uint16_t type)`
- `int transformer_init_controls (m_transformer *trans)`

## 4.73.1 Macro Definition Documentation

### 4.73.1.1 FADER\_FADE\_IN

```
#define FADER_FADE_IN 0
```

### 4.73.1.2 FADER\_FADE\_OUT

```
#define FADER_FADE_OUT 1
```

### 4.73.1.3 N\_NATIVE\_PARAMETERS

```
#define N_NATIVE_PARAMETERS 3
```

### 4.73.1.4 N\_NATIVE\_SETTINGS

```
#define N_NATIVE_SETTINGS 1
```

#### 4.73.1.5 PRINT\_TRANSFORMER\_INFO

```
#define PRINT_TRANSFORMER_INFO
```

#### 4.73.1.6 TRANSFORMER\_MAX\_INPUTS

```
#define TRANSFORMER_MAX_INPUTS 4
```

#### 4.73.1.7 TRANSFORMER\_MAX\_OUTPUTS

```
#define TRANSFORMER_MAX_OUTPUTS 4
```

#### 4.73.1.8 TRANSFORMER\_SWITCH\_ACTION\_BYPASS

```
#define TRANSFORMER_SWITCH_ACTION_BYPASS 0
```

#### 4.73.1.9 TRANSFORMER\_TRANSITION\_BIBLOCK\_LINEAR

```
#define TRANSFORMER_TRANSITION_BIBLOCK_LINEAR 2
```

#### 4.73.1.10 TRANSFORMER\_TRANSITION\_INSTANT

```
#define TRANSFORMER_TRANSITION_INSTANT 0
```

#### 4.73.1.11 TRANSFORMER\_TRANSITION\_MONOBLOCK\_LINEAR

```
#define TRANSFORMER_TRANSITION_MONOBLOCK_LINEAR 1
```

#### 4.73.1.12 TRANSFORMER\_TRANSITION\_QUADBLOCK\_LINEAR

```
#define TRANSFORMER_TRANSITION_QUADBLOCK_LINEAR 3
```

#### 4.73.1.13 TRANSFORMER\_TRANSITION\_TAIL

```
#define TRANSFORMER_TRANSITION_TAIL 4
```

### 4.73.2 Function Documentation

#### 4.73.2.1 clone\_transformer()

```
int clone_transformer (  
    m_transformer ** dest_ptr,  
    m_transformer * src)
```

#### 4.73.2.2 free\_transformer()

```
void free_transformer (  
    m_transformer * trans)
```

#### 4.73.2.3 run\_transformer()

```
int run_transformer (  
    m_transformer * trans,  
    float * dest,  
    float * src)
```

#### 4.73.2.4 transformer\_add\_parameter()

```
int transformer_add_parameter (  
    m_transformer * trans,  
    m_parameter * param)
```

#### 4.73.2.5 transformer\_add\_setting()

```
int transformer_add_setting (  
    m_transformer * trans,  
    m_setting * setting)
```

#### 4.73.2.6 transformer\_get\_parameter()

```
m_parameter * transformer_get_parameter (  
    m_transformer * trans,  
    uint16_t ppid)
```

#### 4.73.2.7 transformer\_get\_setting()

```
m_setting * transformer_get_setting (  
    m_transformer * trans,  
    uint16_t sid)
```

#### 4.73.2.8 transformer\_init\_controls()

```
int transformer_init_controls (  
    m_transformer * trans)
```

#### 4.73.2.9 transformer\_init\_parameter\_array()

```
int transformer_init_parameter_array (  
    m_transformer * trans,  
    int n)
```

**4.73.2.10 transformer\_init\_setting\_array()**

```
int transformer_init_setting_array (
    m_transformer * trans,
    int n)
```

**4.73.2.11 transformer\_type\_to\_string()**

```
const char * transformer_type_to_string (
    uint16_t type)
```

**4.74 m\_eng\_transformer.h**

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_TRANSFORMER_H_
00002 #define M_ENG_TRANSFORMER_H_
00003
00004 #include "m_transformer_enum.h"
00005
00006 #include "m_parameter.h"
00007
00008 #define TRANSFORMER_MAX_INPUTS      4
00009 #define TRANSFORMER_MAX_OUTPUTS     4
00010
00011 #define FADER_FADE_IN    0
00012 #define FADER_FADE_OUT  1
00013
00014 #define PRINT_TRANSFORMER_INFO
00015
00016 #define TRANSFORMER_SWITCH_ACTION_BYPASS 0
00017
00018 #define TRANSFORMER_TRANSITION_INSTANT      0
00019 #define TRANSFORMER_TRANSITION_MONOBLOCK_LINEAR 1
00020 #define TRANSFORMER_TRANSITION_BIBLOCK_LINEAR 2
00021 #define TRANSFORMER_TRANSITION_QUADBLOCK_LINEAR 3
00022 #define TRANSFORMER_TRANSITION_TAIL      4
00023
00024 #define N_NATIVE_PARAMETERS 3
00025 #define N_NATIVE_SETTINGS 1
00026
00027 #include "m_eng_linkowitz_riley.h"
00028
00029 #include "m_transformer.h"
00030
00031 int transformer_add_setting(m_transformer *trans, m_setting *setting);
00032 int transformer_add_parameter(m_transformer *trans, m_parameter *param);
00033
00034 m_parameter *transformer_get_parameter(m_transformer *trans, uint16_t ppid);
00035 m_setting *transformer_get_setting (m_transformer *trans, uint16_t sid );
00036
00037 int transformer_init_parameter_array(m_transformer *trans, int n);
00038 int transformer_init_setting_array(m_transformer *trans, int n);
00039
00040 int run_transformer(m_transformer *trans, float *dest, float *src);
00041
00042 void free_transformer(m_transformer *trans);
00043
00044 int clone_transformer(m_transformer **dest_ptr, m_transformer *src);
00045
00046 const char *transformer_type_to_string(uint16_t type);
00047
00048 int transformer_init_controls(m_transformer *trans);
00049
00050 #endif
```

**4.75 m\_eng\_transformer\_init.h File Reference****Functions**

- int `init_transformer` (m\_transformer \*trans, uint16\_t type)

## 4.75.1 Function Documentation

### 4.75.1.1 init\_transformer()

```
int init_transformer (
    m_transformer * trans,
    uint16_t type)
```

## 4.76 m\_eng\_transformer\_init.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_TRANSFORMER_INIT_H_
00002 #define M_ENG_TRANSFORMER_INIT_H_
00003
00004 int init_transformer(m_transformer *trans, uint16_t type);
00005
00006 #endif
```

## 4.77 m\_eng\_transformer\_template.h File Reference

### Data Structures

- struct [m\\_transformer\\_str](#)

### Functions

- int [init\\_transformer\\_str](#) ([m\\_transformer\\_str](#) \*str)
- int [reconfigure\\_transformer](#) (void \*data\_struct)
- int [calc\\_transformer](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

## 4.77.1 Function Documentation

### 4.77.1.1 calc\_transformer()

```
int calc_transformer (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

### 4.77.1.2 init\_transformer\_str()

```
int init_transformer_str (
    m_transformer_str * str)
```

### 4.77.1.3 reconfigure\_transformer()

```
int reconfigure_transformer (  
    void * data_struct)
```

## 4.78 m\_eng\_transformer\_template.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_TRANSFORMER_H_  
00002 #define M_ENG_TRANSFORMER_H_  
00003  
00004 typedef struct  
00005 {  
00006     m_parameter param;  
00007  
00008 } m_transformer_str;  
00009  
00010 int init_transformer_str(m_transformer_str *str);  
00011 int reconfigure_transformer(void *data_struct);  
00012 int calc_transformer(void *data_struct, float *dest, float *src, int n_samples);  
00013  
00014 #endif
```

## 4.79 m\_eng\_transition.h File Reference

### Macros

- `#define` [TRANSITION\\_MONOBLOCK\\_COS2](#) 0
- `#define` [TRANSITION\\_QUADBLOCK\\_COS2](#) 1
- `#define` [TRANSITION\\_OCTOBLOCK\\_COS2](#) 2
- `#define` [TRANSITION\\_TAIL](#) 3
- `#define` [TAIL\\_NEW\\_FADE\\_IN\\_SAMPLES](#) 256
- `#define` [TAIL\\_INPUT\\_FADE\\_SAMPLES](#) 256

### 4.79.1 Macro Definition Documentation

#### 4.79.1.1 TAIL\_INPUT\_FADE\_SAMPLES

```
#define TAIL_INPUT_FADE_SAMPLES 256
```

#### 4.79.1.2 TAIL\_NEW\_FADE\_IN\_SAMPLES

```
#define TAIL_NEW_FADE_IN_SAMPLES 256
```

#### 4.79.1.3 TRANSITION\_MONOBLOCK\_COS2

```
#define TRANSITION_MONOBLOCK_COS2 0
```

#### 4.79.1.4 TRANSITION\_OCTOBLOCK\_COS2

```
#define TRANSITION_OCTOBLOCK_COS2 2
```

#### 4.79.1.5 TRANSITION\_QUADBLOCK\_COS2

```
#define TRANSITION_QUADBLOCK_COS2 1
```

#### 4.79.1.6 TRANSITION\_TAIL

```
#define TRANSITION_TAIL 3
```

### 4.80 m\_eng\_transition.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_TRANSITION_H_
00002 #define M_ENG_TRANSITION_H_
00003
00004 #define TRANSITION_MONOBLOCK_COS2 0
00005 #define TRANSITION_QUADBLOCK_COS2 1
00006 #define TRANSITION_OCTOBLOCK_COS2 2
00007 #define TRANSITION_TAIL 3
00008
00009 #define TAIL_NEW_FADE_IN_SAMPLES 256
00010 #define TAIL_INPUT_FADE_SAMPLES 256
00011
00012 #endif
```

### 4.81 m\_eng\_update.h File Reference

#### Macros

- `#define m_eng_disable_software_interrupts()`
- `#define m_eng_enable_software_interrupts()`

#### Functions

- void `update_all()`
- int `update_setup()`
- void `update_stop()`
- void `m_eng_software_isr()`

#### 4.81.1 Macro Definition Documentation

##### 4.81.1.1 m\_eng\_disable\_software\_interrupts

```
#define m_eng_disable_software_interrupts()
```

#### Value:

```
(NVIC_DISABLE_IRQ(IRQ_SOFTWARE))
```



### 4.81.1.2 m\_eng\_enable\_software\_interrupts

```
#define m_eng_enable_software_interrupts()
```

**Value:**

```
(NVIC_ENABLE_IRQ (IRQ_SOFTWARE))
```

## 4.81.2 Function Documentation

### 4.81.2.1 m\_eng\_software\_isr()

```
void m_eng_software_isr ()
```

### 4.81.2.2 update\_all()

```
void update_all ()
```

### 4.81.2.3 update\_setup()

```
int update_setup ()
```

### 4.81.2.4 update\_stop()

```
void update_stop ()
```

## 4.82 m\_eng\_update.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_UPDATE_H_
00002 #define M_UPDATE_H_
00003
00004 #ifndef M_SIMULATED
00005 #define m_eng_disable_software_interrupts() (NVIC_DISABLE_IRQ (IRQ_SOFTWARE))
00006 #define m_eng_enable_software_interrupts() (NVIC_ENABLE_IRQ (IRQ_SOFTWARE))
00007 #else
00008 #define m_eng_disable_software_interrupts()
00009 #define m_eng_enable_software_interrupts()
00010 #endif
00011
00012 void update_all();
00013
00014 int update_setup();
00015 void update_stop();
00016
00017 void m_eng_software_isr();
00018
00019 #endif
```

## 4.83 m\_eng\_useful\_functions.h File Reference

### Functions

- float [identity\\_function](#) (float x)
- float [normalised\\_arctan](#) (float x)
- float [hard\\_clip](#) (float x)
- float [soft\\_fold](#) (float x)
- float [trig\\_transition\\_function](#) (float x)
- int [convert\\_block\\_int\\_to\\_float](#) (float \*dest, int16\_t \*src)
- int [convert\\_block\\_float\\_to\\_int](#) (int16\_t \*src, float \*dest)

### 4.83.1 Function Documentation

#### 4.83.1.1 convert\_block\_float\_to\_int()

```
int convert_block_float_to_int (  
    int16_t * src,  
    float * dest)
```

#### 4.83.1.2 convert\_block\_int\_to\_float()

```
int convert_block_int_to_float (  
    float * dest,  
    int16_t * src)
```

#### 4.83.1.3 hard\_clip()

```
float hard_clip (  
    float x)
```

#### 4.83.1.4 identity\_function()

```
float identity_function (  
    float x)
```

#### 4.83.1.5 normalised\_arctan()

```
float normalised_arctan (  
    float x)
```

#### 4.83.1.6 soft\_fold()

```
float soft_fold (  
    float x)
```

## 4.83.1.7 trig\_transition\_function()

```
float trig_transition_function (
    float x)
```

## 4.84 m\_eng\_useful\_functions.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_USEFUL_FUNCTIONS_H_
00002 #define M_ENG_USEFUL_FUNCTIONS_H_
00003
00004 float identity_function(float x);
00005 float normalised_arctan(float x);
00006 float hard_clip(float x);
00007 float soft_fold(float x);
00008 float trig_transition_function(float x);
00009
00010 int convert_block_int_to_float(float *dest, int16_t *src);
00011 int convert_block_float_to_int(int16_t *src, float *dest);
00012
00013 static inline int positive_mod(int x, int m)
00014 {
00015     int r = x % m;
00016     return r < 0 ? r + m : r;
00017 }
00018
00019 static inline int floor_divide(int x, int d)
00020 {
00021     // d > 0
00022     if (x >= 0) return x / d;
00023     return -((-x + d - 1) / d);
00024 }
00025
00026 static inline int ring_index(int i, int n)
00027 {
00028     int r = i % n;
00029     return r < 0 ? r + n : r;
00030 }
00031
00032 #endif
```

## 4.85 m\_eng\_warbler.h File Reference

## Data Structures

- struct [m\\_eng\\_warbler\\_str](#)

## Functions

- int [init\\_warbler\\_str](#) ([m\\_eng\\_warbler\\_str](#) \*str)
- int [reconfigure\\_warbler](#) (void \*data\_struct)
- int [calc\\_warbler](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

## 4.85.1 Function Documentation

## 4.85.1.1 calc\_warbler()

```
int calc_warbler (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

#### 4.85.1.2 init\_warbler\_str()

```
int init_warbler_str (  
    m_eng_warbler_str * str)
```

#### 4.85.1.3 reconfigure\_warbler()

```
int reconfigure_warbler (  
    void * data_struct)
```

### 4.86 m\_eng\_warbler.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ENG_WARBLER_H_  
00002 #define M_ENG_WARBLER_H_  
00003  
00004 typedef struct  
00005 {  
00006     m_parameter center;  
00007     m_parameter width;  
00008     m_parameter reactivity;  
00009     m_parameter sensitivity;  
00010     m_parameter max_rate;  
00011     m_parameter min_rate;  
00012  
00013     float alpha;  
00014     float e;  
00015  
00016     float t;  
00017     float rate;  
00018  
00019     m_eng_band_pass_filter_str filter;  
00020 } m_eng_warbler_str;  
00021  
00022 int init_warbler_str(m_eng_warbler_str *str);  
00023 int reconfigure_warbler(void *data_struct);  
00024 int calc_warbler(void *data_struct, float *dest, float *src, int n_samples);  
00025  
00026 #endif
```

### 4.87 m\_eng\_waveshaper.h File Reference

#### Data Structures

- struct [m\\_eng\\_waveshaper\\_str](#)

#### Macros

- #define [WAVESHAPER\\_ENVELOPE\\_ATTACK](#) expf(-7.0/8.0)
- #define [WAVESHAPER\\_ENVELOPE\\_RELEASE](#) expf(-7.0/2048.0)

#### Functions

- int [init\\_waveshaper\\_str](#) ([m\\_eng\\_waveshaper\\_str](#) \*str)
- int [calc\\_waveshaper](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

## 4.87.1 Macro Definition Documentation

### 4.87.1.1 WAVESHAPER\_ENVELOPE\_ATTACK

```
#define WAVESHAPER_ENVELOPE_ATTACK expf(-7.0/8.0)
```

### 4.87.1.2 WAVESHAPER\_ENVELOPE\_RELEASE

```
#define WAVESHAPER_ENVELOPE_RELEASE expf(-7.0/2048.0)
```

## 4.87.2 Function Documentation

### 4.87.2.1 calc\_waveshaper()

```
int calc_waveshaper (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

### 4.87.2.2 init\_waveshaper\_str()

```
int init_waveshaper_str (
    m_eng_waveshaper_str * str)
```

## 4.88 m\_eng\_waveshaper.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_WAVESHAPER_H_
00002 #define M_WAVESHAPER_H_
00003
00004 #define WAVESHAPER_ENVELOPE_ATTACK expf(-7.0/8.0)
00005 #define WAVESHAPER_ENVELOPE_RELEASE expf(-7.0/2048.0)
00006
00007 typedef struct
00008 {
00009     m_parameter coefficient;
00010
00011     float (*shape)(float x);
00012 } m_eng_waveshaper_str;
00013
00014 int init_waveshaper_str(m_eng_waveshaper_str *str);
00015 int calc_waveshaper(void *data_struct, float *dest, float *src, int n_samples);
00016
00017
00018 #endif
```

## 4.89 m\_eng.cpp File Reference

```
#include <Wire.h>
#include "m_eng.h"
```

## Macros

- `#define LED_BLINK_MILLIS 50`
- `#define DEBUG_PRINT_MILLIS 1000`
- `#define PRINT_LOG`
- `#define LOG_PRINT_MILLIS 10`
- `#define PROFILER_PRINT_MILLIS 10000`
- `#define MEM_REPORT_MILLIS 1000`
- `#define SCHEDULED_MAINTAINANCE_MILLIS 100`
- `#define SGT5000_CHECK_PERIOD 100`
- `#define SCHEDULED_MAINTAINANCE`

## Functions

- `int main ()`

## 4.89.1 Macro Definition Documentation

### 4.89.1.1 DEBUG\_PRINT\_MILLIS

```
#define DEBUG_PRINT_MILLIS 1000
```

### 4.89.1.2 LED\_BLINK\_MILLIS

```
#define LED_BLINK_MILLIS 50
```

### 4.89.1.3 LOG\_PRINT\_MILLIS

```
#define LOG_PRINT_MILLIS 10
```

### 4.89.1.4 MEM\_REPORT\_MILLIS

```
#define MEM_REPORT_MILLIS 1000
```

### 4.89.1.5 PRINT\_LOG

```
#define PRINT_LOG
```

### 4.89.1.6 PROFILER\_PRINT\_MILLIS

```
#define PROFILER_PRINT_MILLIS 10000
```

#### 4.89.1.7 SCHEDULED\_MAINTAINANCE

```
#define SCHEDULED_MAINTAINANCE
```

#### 4.89.1.8 SCHEDULED\_MAINTAINANCE\_MILLIS

```
#define SCHEDULED_MAINTAINANCE_MILLIS 100
```

#### 4.89.1.9 SGTL5000\_CHECK\_PERIOD

```
#define SGTL5000_CHECK_PERIOD 100
```

### 4.89.2 Function Documentation

#### 4.89.2.1 main()

```
int main ()
```

## 4.90 m\_eng\_adaptive\_waveshaper.c File Reference

```
#include "m_eng.h"
```

### Functions

- [int init\\_adaptive\\_waveshaper\\_str](#) ([m\\_eng\\_adaptive\\_waveshaper\\_str](#) \*str)
- [int calc\\_adaptive\\_waveshaper](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.90.1 Function Documentation

#### 4.90.1.1 calc\_adaptive\_waveshaper()

```
int calc_adaptive_waveshaper (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.90.1.2 init\_adaptive\_waveshaper\_str()

```
int init_adaptive_waveshaper_str (  
    m\_eng\_adaptive\_waveshaper\_str * str)
```

## 4.91 m\_eng\_audio\_block.c File Reference

```
#include "m_eng.h"
```

## 4.92 m\_eng\_band\_splitter.c File Reference

## 4.93 m\_eng\_biquad.c File Reference

```
#include "m_eng.h"
```

### Functions

- int [calc\\_biquad](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- int [reconfigure\\_biquad](#) (void \*data\_struct)
- int [init\\_biquad\\_str](#) (m\_eng\_biquad\_str \*str)

### 4.93.1 Function Documentation

#### 4.93.1.1 calc\_biquad()

```
int calc_biquad (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.93.1.2 init\_biquad\_str()

```
int init_biquad_str (  
    m_eng_biquad_str * str)
```

#### 4.93.1.3 reconfigure\_biquad()

```
int reconfigure_biquad (  
    void * data_struct)
```

## 4.94 m\_eng\_buffer\_mixer\_amp.c File Reference

```
#include "m_eng.h"
```



## Functions

- int [calc\\_buffer](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- int [init\\_amplifier\\_str](#) ([m\\_eng\\_amplifier\\_str](#) \*str)
- int [reconfigure\\_amplifier](#) (void \*data\_struct)
- int [calc\\_amplifier](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

## 4.94.1 Function Documentation

### 4.94.1.1 [calc\\_amplifier\(\)](#)

```
int calc_amplifier (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

### 4.94.1.2 [calc\\_buffer\(\)](#)

```
int calc_buffer (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

### 4.94.1.3 [init\\_amplifier\\_str\(\)](#)

```
int init_amplifier_str (  
    m\_eng\_amplifier\_str * str)
```

### 4.94.1.4 [reconfigure\\_amplifier\(\)](#)

```
int reconfigure_amplifier (  
    void * data_struct)
```

## 4.95 m\_eng\_comms.cpp File Reference

```
#include <cstdint>  
#include "m_eng.h"  
#include "signal.h"  
#include <Wire.h>  
#include "utility/imxrt_hw.h"
```

## Macros

- #define [PRINT\\_RESPONSE\\_BYTES](#)

## Enumerations

- enum `comms_fsm_eng_state_t` { `IDLE`, `SENDING_STRING`, `RECIEVING_NEW_PARAM_NAM_ENG_LONG` }

## Functions

- int `m_message_sanity_check` (`m_message` msg, int len)
- void `handle_esp32_message` (`m_message` msg)
- void `i2c_receive_isr` (int n)
- void `i2c_request_isr` ()
- int `init_esp32_link` ()
- void `esp32_message_check_handle` ()

## Variables

- `m_response` response
- `m_response` prev\_response
- `m_message` received
- volatile uint8\_t `receive_buffer` [M\_MESSAGE\_MAX\_TRANSFER\_LEN]
- uint8\_t `response_buffer` [M\_MESSAGE\_MAX\_TRANSFER\_LEN]
- uint8\_t `wait_message` [M\_MESSAGE\_MAX\_TRANSFER\_LEN]
- volatile unsigned int `received_length`
- unsigned int `response_length`
- volatile sig\_atomic\_t `message_pending` = 0
- volatile sig\_atomic\_t `response_ready` = 0
- char \* `string_out` = NULL
- int `string_out_pos`
- char \* `string_in` = NULL
- int `string_in_pos`
- `comms_fsm_eng_state_t` `comms_fsm_eng_state` = `IDLE`

## 4.95.1 Macro Definition Documentation

### 4.95.1.1 PRINT\_RESPONSE\_BYTES

```
#define PRINT_RESPONSE_BYTES
```

## 4.95.2 Enumeration Type Documentation

### 4.95.2.1 comms\_fsm\_eng\_state\_t

```
enum comms_fsm_eng_state_t
```

#### Enumerator

IDLE	
SENDING_STRING	
RECIEVING_NEW_PARAM_NAM_ENG_LONG	

### 4.95.3 Function Documentation

#### 4.95.3.1 esp32\_message\_check\_handle()

```
void esp32_message_check_handle ()
```

#### 4.95.3.2 handle\_esp32\_message()

```
void handle_esp32_message (  
    m_message msg)
```

#### 4.95.3.3 i2c\_receive\_isr()

```
void i2c_receive_isr (  
    int n)
```

#### 4.95.3.4 i2c\_request\_isr()

```
void i2c_request_isr ()
```

#### 4.95.3.5 init\_esp32\_link()

```
int init_esp32_link ()
```

#### 4.95.3.6 m\_message\_sanity\_check()

```
int m_message_sanity_check (  
    m_message msg,  
    int len)
```

### 4.95.4 Variable Documentation

#### 4.95.4.1 comms\_fsm\_eng\_state

```
comms_fsm_eng_state_t comms_fsm_eng_state = IDLE
```

#### 4.95.4.2 message\_pending

```
volatile sig_atomic_t message_pending = 0
```

#### 4.95.4.3 prev\_response

```
m_response prev_response
```

#### 4.95.4.4 receive\_buffer

```
volatile uint8_t receive_buffer[M_MESSAGE_MAX_TRANSFER_LEN]
```

#### 4.95.4.5 received

```
m_message received
```

#### 4.95.4.6 received\_length

```
volatile unsigned int received_length
```

#### 4.95.4.7 response

```
m_response response
```

#### 4.95.4.8 response\_buffer

```
uint8_t response_buffer[M_MESSAGE_MAX_TRANSFER_LEN]
```

#### 4.95.4.9 response\_length

```
unsigned int response_length
```

#### 4.95.4.10 response\_ready

```
volatile sig_atomic_t response_ready = 0
```

#### 4.95.4.11 string\_in

```
char* string_in = NULL
```

#### 4.95.4.12 string\_in\_pos

```
int string_in_pos
```

#### 4.95.4.13 string\_out

```
char* string_out = NULL
```

#### 4.95.4.14 string\_out\_pos

```
int string_out_pos
```

#### 4.95.4.15 wait\_message

```
uint8_t wait_message[M_MESSAGE_MAX_TRANSFER_LEN]
```

## 4.96 m\_eng\_compressor.c File Reference

```
#include "m_eng.h"  
#include "m_eng_compressor.h"
```

### Macros

- #define [EPSILON](#) 0.00000001

### Functions

- int [init\\_compressor\\_str](#) ([m\\_eng\\_compressor\\_str](#) \*str)
- int [reconfigure\\_compressor](#) (void \*data\_struct)
- int [calc\\_compressor](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.96.1 Macro Definition Documentation

#### 4.96.1.1 EPSILON

```
#define EPSILON 0.00000001
```

### 4.96.2 Function Documentation

#### 4.96.2.1 calc\_compressor()

```
int calc_compressor (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.96.2.2 init\_compressor\_str()

```
int init_compressor_str (  
    m\_eng\_compressor\_str * str)
```

### 4.96.2.3 reconfigure\_compressor()

```
int reconfigure_compressor (
    void * data_struct)
```

## 4.97 m\_eng\_context.c File Reference

```
#include "m_eng.h"
```

### Macros

- #define `AVG_DURATION_UPDATE_COEF` 0.99f

### Functions

- int `init_m_eng_context` (m\_eng\_context \*cxt)
- int `m_eng_context_new_profile` (m\_eng\_context \*cxt)
- int `cxt_profile_id_valid` (m\_eng\_context \*cxt, uint16\_t pid)
- int `cxt_transformer_id_valid` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid)
- int `pcxt_profile_id_valid` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t ppid)
- m\_parameter \* `cxt_get_parameter_by_id` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t ppid)
- m\_parameter \* `cxt_get_front_parameter_by_id` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t ppid)
- m\_parameter \* `cxt_get_back_parameter_by_id` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t ppid)
- int `cxt_update_parameter_value_by_id` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t ppid, float new\_value)
- m\_setting \* `cxt_get_setting_by_id` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t sid)
- int `cxt_update_setting_value_by_id` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid, uint16\_t sid, uint16\_t new\_value)
- void `m_eng_safe_reboot` (m\_eng\_context \*cxt)
- int `reset_context` (m\_eng\_context \*cxt)
- int `cxt_append_transformer_to_profile` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t type)
- int `cxt_remove_transformer_from_profile` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid)
- int `cxt_move_transformer` (m\_eng\_context \*cxt, uint16\_t tid, uint16\_t new\_pos)
- int `cxt_insert_transformer_to_profile` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t type, uint16\_t pos)
- int `cxt_prepend_transformer_to_profile` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t type)
- int `cxt_get_n_profile_transformers` (m\_eng\_context \*cxt, uint16\_t pid)
- int `cxt_get_n_transformer_params` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid)
- int `cxt_get_n_transformer_settings` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid)
- int `cxt_get_transformer_type` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid)
- int `cxt_get_tid_by_pos` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t pos)
- m\_transformer \* `cxt_get_transformer_by_id` (m\_eng\_context \*cxt, uint16\_t pid, uint16\_t tid)
- int `cxt_set_active_profile` (m\_eng\_context \*cxt, uint16\_t pid)
- int `cxt_switch_to_profile` (m\_eng\_context \*cxt, uint16\_t pid)
- int `cxt_process` (m\_eng\_context \*cxt)
- int `cxt_run_scheduled_maintenance` (m\_eng\_context \*cxt)

## 4.97.1 Macro Definition Documentation

### 4.97.1.1 AVG\_DURATION\_UPDATE\_COEF

```
#define AVG_DURATION_UPDATE_COEF 0.99f
```

## 4.97.2 Function Documentation

### 4.97.2.1 cxt\_append\_transformer\_to\_profile()

```
int cxt_append_transformer_to_profile (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t type)
```

### 4.97.2.2 cxt\_get\_back\_parameter\_by\_id()

```
m_parameter * cxt_get_back_parameter_by_id (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid,  
    uint16_t ppid)
```

### 4.97.2.3 cxt\_get\_front\_parameter\_by\_id()

```
m_parameter * cxt_get_front_parameter_by_id (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid,  
    uint16_t ppid)
```

### 4.97.2.4 cxt\_get\_n\_profile\_transformers()

```
int cxt_get_n_profile_transformers (  
    m_eng_context * cxt,  
    uint16_t pid)
```

### 4.97.2.5 cxt\_get\_n\_transformer\_params()

```
int cxt_get_n_transformer_params (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid)
```

#### 4.97.2.6 cxt\_get\_n\_transformer\_settings()

```
int cxt_get_n_transformer_settings (
    m_eng_context * cxt,
    uint16_t pid,
    uint16_t tid)
```

#### 4.97.2.7 cxt\_get\_parameter\_by\_id()

```
m_parameter * cxt_get_parameter_by_id (
    m_eng_context * cxt,
    uint16_t pid,
    uint16_t tid,
    uint16_t ppid)
```

#### 4.97.2.8 cxt\_get\_setting\_by\_id()

```
m_setting * cxt_get_setting_by_id (
    m_eng_context * cxt,
    uint16_t pid,
    uint16_t tid,
    uint16_t sid)
```

#### 4.97.2.9 cxt\_get\_tid\_by\_pos()

```
int cxt_get_tid_by_pos (
    m_eng_context * cxt,
    uint16_t pid,
    uint16_t pos)
```

#### 4.97.2.10 cxt\_get\_transformer\_by\_id()

```
m_transformer * cxt_get_transformer_by_id (
    m_eng_context * cxt,
    uint16_t pid,
    uint16_t tid)
```

#### 4.97.2.11 cxt\_get\_transformer\_type()

```
int cxt_get_transformer_type (
    m_eng_context * cxt,
    uint16_t pid,
    uint16_t tid)
```



**4.97.2.12 cxt\_insert\_transformer\_to\_profile()**

```
int cxt_insert_transformer_to_profile (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t type,  
    uint16_t pos)
```

**4.97.2.13 cxt\_move\_transformer()**

```
int cxt_move_transformer (  
    m_eng_context * cxt,  
    uint16_t tid,  
    uint16_t new_pos)
```

**4.97.2.14 cxt\_prepend\_transformer\_to\_profile()**

```
int cxt_prepend_transformer_to_profile (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t type)
```

**4.97.2.15 cxt\_process()**

```
int cxt_process (  
    m_eng_context * cxt)
```

**4.97.2.16 cxt\_profile\_id\_valid()**

```
int cxt_profile_id_valid (  
    m_eng_context * cxt,  
    uint16_t pid)
```

**4.97.2.17 cxt\_remove\_transformer\_from\_profile()**

```
int cxt_remove_transformer_from_profile (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid)
```

**4.97.2.18 cxt\_run\_scheduled\_maintenance()**

```
int cxt_run_scheduled_maintenance (  
    m_eng_context * cxt)
```

#### 4.97.2.19 cxt\_set\_active\_profile()

```
int cxt_set_active_profile (  
    m_eng_context * cxt,  
    uint16_t pid)
```

#### 4.97.2.20 cxt\_switch\_to\_profile()

```
int cxt_switch_to_profile (  
    m_eng_context * cxt,  
    uint16_t pid)
```

#### 4.97.2.21 cxt\_transformer\_id\_valid()

```
int cxt_transformer_id_valid (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid)
```

#### 4.97.2.22 cxt\_update\_parameter\_value\_by\_id()

```
int cxt_update_parameter_value_by_id (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid,  
    uint16_t ppid,  
    float new_value)
```

#### 4.97.2.23 cxt\_update\_setting\_value\_by\_id()

```
int cxt_update_setting_value_by_id (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid,  
    uint16_t sid,  
    uint16_t new_value)
```

#### 4.97.2.24 init\_m\_eng\_context()

```
int init_m_eng_context (  
    m_eng_context * cxt)
```

#### 4.97.2.25 m\_eng\_context\_new\_profile()

```
int m_eng_context_new_profile (  
    m_eng_context * cxt)
```

#### 4.97.2.26 m\_eng\_safe\_reboot()

```
void m_eng_safe_reboot (  
    m_eng_context * cxt)
```

#### 4.97.2.27 pcxt\_profile\_id\_valid()

```
int pcxt_profile_id_valid (  
    m_eng_context * cxt,  
    uint16_t pid,  
    uint16_t tid,  
    uint16_t ppid)
```

#### 4.97.2.28 reset\_context()

```
int reset_context (  
    m_eng_context * cxt)
```

## 4.98 m\_eng\_debugging.c File Reference

```
#include "m_eng.h"
```

### Functions

- void [print\\_binary](#) (int v, int bits)
- void [full\\_debug\\_print](#) (m\_eng\_context \*cxt)
- void [print\\_context\\_info](#) (m\_eng\_context \*cxt, int depth)
- void [print\\_profile\\_info](#) (m\_eng\_profile \*profile, int depth)
- void [print\\_pipeline\\_info](#) (m\_pipeline \*pipeline, int depth)
- void [print\\_transformer\\_info](#) (m\_transformer \*trans, int depth)

### 4.98.1 Function Documentation

#### 4.98.1.1 full\_debug\_print()

```
void full_debug_print (  
    m_eng_context * cxt)
```

#### 4.98.1.2 print\_binary()

```
void print_binary (  
    int v,  
    int bits)
```

#### 4.98.1.3 print\_context\_info()

```
void print_context_info (
    m_eng_context * cxt,
    int depth)
```

#### 4.98.1.4 print\_pipeline\_info()

```
void print_pipeline_info (
    m_pipeline * pipeline,
    int depth)
```

#### 4.98.1.5 print\_profile\_info()

```
void print_profile_info (
    m_eng_profile * profile,
    int depth)
```

#### 4.98.1.6 print\_transformer\_info()

```
void print_transformer_info (
    m_transformer * trans,
    int depth)
```

## 4.99 m\_eng\_delay.c File Reference

```
#include "m_eng.h"
```

### Functions

- int [init\\_delay\\_str](#) ([m\\_eng\\_delay\\_str](#) \*str)
- int [reconfigure\\_delay](#) (void \*data\_struct)
- int [calc\\_delay](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.99.1 Function Documentation

#### 4.99.1.1 calc\_delay()

```
int calc_delay (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

#### 4.99.1.2 init\_delay\_str()

```
int init_delay_str (  
    m_eng_delay_str * str)
```

#### 4.99.1.3 reconfigure\_delay()

```
int reconfigure_delay (  
    void * data_struct)
```

## 4.100 m\_eng\_delay\_buffer.c File Reference

```
#include "m_eng.h"
```

### Functions

- int [init\\_delay\\_buffer](#) ([m\\_delay\\_buffer](#) \*buf)
- int [m\\_delay\\_buffer\\_resize\\_samples](#) ([m\\_delay\\_buffer](#) \*buf, int n\_samples)
- int [m\\_delay\\_buffer\\_resize\\_seconds](#) ([m\\_delay\\_buffer](#) \*buf, float seconds)
- int [m\\_delay\\_buffer\\_resize\\_milliseconds](#) ([m\\_delay\\_buffer](#) \*buf, float ms)
- int [m\\_delay\\_buffer\\_tick](#) ([m\\_delay\\_buffer](#) \*buf, float new\_sample)
- int [m\\_delay\\_buffer\\_advance](#) ([m\\_delay\\_buffer](#) \*buf, float \*new\_samples, unsigned int n)
- int [m\\_delay\\_buffer\\_get\\_delayed\\_sample](#) ([m\\_delay\\_buffer](#) \*buf, float \*dest, unsigned int delay)
- float \* [m\\_delay\\_buffer\\_get\\_delayed\\_sample\\_ptr](#) ([m\\_delay\\_buffer](#) \*buf, unsigned int delay)
- int [m\\_delay\\_buffer\\_get\\_fractional\\_delayed\\_sample](#) ([m\\_delay\\_buffer](#) \*buf, float \*dest, float delay)

### 4.100.1 Function Documentation

#### 4.100.1.1 init\_delay\_buffer()

```
int init_delay_buffer (  
    m_delay_buffer * buf)
```

#### 4.100.1.2 m\_delay\_buffer\_advance()

```
int m_delay_buffer_advance (  
    m_delay_buffer * buf,  
    float * new_samples,  
    unsigned int n)
```

#### 4.100.1.3 m\_delay\_buffer\_get\_delayed\_sample()

```
int m_delay_buffer_get_delayed_sample (  
    m_delay_buffer * buf,  
    float * dest,  
    unsigned int delay)
```

#### 4.100.1.4 `m_delay_buffer_get_delayed_sample_ptr()`

```
float * m_delay_buffer_get_delayed_sample_ptr (
    m_delay_buffer * buf,
    unsigned int delay)
```

#### 4.100.1.5 `m_delay_buffer_get_fractional_delayed_sample()`

```
int m_delay_buffer_get_fractional_delayed_sample (
    m_delay_buffer * buf,
    float * dest,
    float delay)
```

#### 4.100.1.6 `m_delay_buffer_resize_milliseconds()`

```
int m_delay_buffer_resize_milliseconds (
    m_delay_buffer * buf,
    float ms)
```

#### 4.100.1.7 `m_delay_buffer_resize_samples()`

```
int m_delay_buffer_resize_samples (
    m_delay_buffer * buf,
    int n_samples)
```

#### 4.100.1.8 `m_delay_buffer_resize_seconds()`

```
int m_delay_buffer_resize_seconds (
    m_delay_buffer * buf,
    float seconds)
```

#### 4.100.1.9 `m_delay_buffer_tick()`

```
int m_delay_buffer_tick (
    m_delay_buffer * buf,
    float new_sample)
```

## 4.101 `m_eng_dirty_octave.c` File Reference

```
#include "m_eng.h"
```

### Functions

- int [reconfigure\\_dirty\\_octave](#) (void \*data\_struct)
- int [calc\\_dirty\\_octave](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- int [init\\_dirty\\_octave\\_str](#) ([m\\_eng\\_dirty\\_octave\\_str](#) \*str)

## 4.101.1 Function Documentation

### 4.101.1.1 calc\_dirty\_octave()

```
int calc_dirty_octave (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

### 4.101.1.2 init\_dirty\_octave\_str()

```
int init_dirty_octave_str (  
    m_eng_dirty_octave_str * str)
```

### 4.101.1.3 reconfigure\_dirty\_octave()

```
int reconfigure_dirty_octave (  
    void * data_struct)
```

## 4.102 m\_eng\_distortion.c File Reference

```
#include "m_eng.h"
```

### Functions

- int [init\\_distortion\\_str](#) ([m\\_eng\\_distortion\\_str](#) \*str)
- int [reconfigure\\_distortion](#) (void \*data\_struct)
- int [calc\\_distortion](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

## 4.102.1 Function Documentation

### 4.102.1.1 calc\_distortion()

```
int calc_distortion (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

### 4.102.1.2 init\_distortion\_str()

```
int init_distortion_str (  
    m_eng_distortion_str * str)
```

#### 4.102.1.3 reconfigure\_distortion()

```
int reconfigure_distortion (  
    void * data_struct)
```

### 4.103 m\_eng\_envelope.c File Reference

```
#include "m_eng.h"
```

#### Functions

- int [init\\_envelope\\_str](#) ([m\\_eng\\_envelope\\_str](#) \*str)
- int [reconfigure\\_envelope](#) (void \*data\_struct)
- int [calc\\_envelope](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

#### 4.103.1 Function Documentation

##### 4.103.1.1 calc\_envelope()

```
int calc_envelope (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

##### 4.103.1.2 init\_envelope\_str()

```
int init_envelope_str (  
    m\_eng\_envelope\_str * str)
```

##### 4.103.1.3 reconfigure\_envelope()

```
int reconfigure_envelope (  
    void * data_struct)
```

### 4.104 m\_eng\_equaliser.c File Reference

```
#include "m_eng.h"
```



## Functions

- int [init\\_3\\_band\\_eq\\_str](#) ([m\\_eng\\_3\\_band\\_eq\\_str](#) \*str)
- int [reconfigure\\_3\\_band\\_eq](#) (void \*data\_struct)
- int [calc\\_3\\_band\\_eq](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.104.1 Function Documentation

#### 4.104.1.1 [calc\\_3\\_band\\_eq\(\)](#)

```
int calc_3_band_eq (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.104.1.2 [init\\_3\\_band\\_eq\\_str\(\)](#)

```
int init_3_band_eq_str (  
    m\_eng\_3\_band\_eq\_str * str)
```

#### 4.104.1.3 [reconfigure\\_3\\_band\\_eq\(\)](#)

```
int reconfigure_3_band_eq (  
    void * data_struct)
```

## 4.105 m\_eng\_flanger.c File Reference

```
#include "m_eng.h"
```

## Functions

- int [init\\_flanger\\_str](#) ([m\\_eng\\_flanger\\_str](#) \*str)
- int [reconfigure\\_flanger](#) (void \*data\_struct)
- int [calc\\_flanger](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- int [free\\_flanger\\_struct](#) (void \*data\_struct)

### 4.105.1 Function Documentation

#### 4.105.1.1 [calc\\_flanger\(\)](#)

```
int calc_flanger (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.105.1.2 free\_flanger\_struct()

```
int free_flanger_struct (  
    void * data_struct)
```

#### 4.105.1.3 init\_flanger\_str()

```
int init_flanger_str (  
    m_eng_flanger_str * str)
```

#### 4.105.1.4 reconfigure\_flanger()

```
int reconfigure_flanger (  
    void * data_struct)
```

### 4.106 m\_eng\_globals.c File Reference

```
#include "m_eng.h"
```

#### Functions

- double [cycles\\_to\\_seconds](#) (uint64\_t cycles)
- uint64\_t [current\\_cycle](#) ()

#### Variables

- [m\\_eng\\_context](#) global\_cxt
- uint16\_t [cpu\\_cycles\\_total](#) = 0
- uint16\_t [cpu\\_cycles\\_total\\_max](#) = 0
- uint16\_t [memory\\_used](#) = 0
- uint16\_t [memory\\_used\\_max](#) = 0
- int [update\\_scheduled](#) = 0
- uint32\_t [trace\\_depth](#) = 0
- uint32\_t [cycles\\_upper](#) = 0

### 4.106.1 Function Documentation

#### 4.106.1.1 current\_cycle()

```
uint64_t current_cycle () [inline]
```

#### 4.106.1.2 cycles\_to\_seconds()

```
double cycles_to_seconds (  
    uint64_t cycles)
```

## 4.106.2 Variable Documentation

### 4.106.2.1 cpu\_cycles\_total

```
uint16_t cpu_cycles_total = 0
```

### 4.106.2.2 cpu\_cycles\_total\_max

```
uint16_t cpu_cycles_total_max = 0
```

### 4.106.2.3 cycles\_upper

```
uint32_t cycles_upper = 0
```

### 4.106.2.4 global\_cxt

```
m_eng_context global_cxt
```

### 4.106.2.5 memory\_used

```
uint16_t memory_used = 0
```

### 4.106.2.6 memory\_used\_max

```
uint16_t memory_used_max = 0
```

### 4.106.2.7 trace\_depth

```
uint32_t trace_depth = 0
```

### 4.106.2.8 update\_scheduled

```
int update_scheduled = 0
```

## 4.107 m\_eng\_i2s\_dma.cpp File Reference

```
#include <DMAChannel.h>
#include "utility/imxrt_hw.h"
#include "m_eng.h"
```

## Functions

- DMAMEM `__attribute__ ((aligned(32)))` static uint32\_t i2s\_rx\_buffer[AUDIO\_BLOCK\_SAMPLES]
- DMAChannel `i2s_in_dma` (false)
- DMAChannel `i2s_out_dma` (false)
- void `configure_i2s_dma` ()
- void `init_i2s_dma` ()
- void `m_eng_i2s_input_isr` ()
- void `m_eng_i2s_output_isr` ()
- void `i2s_in_transmit` (raw\_sample\_t \*block, unsigned char index)
- void `i2s_input_update` ()
- void `i2s_output_update` ()
- void `i2s_output_transmit_mono_int` (raw\_sample\_t \*block)
- void `i2s_output_transmit_mono_float` (float \*block)

## Variables

- raw\_sample\_t \* `i2s_in_block_left` = NULL
- raw\_sample\_t \* `i2s_in_block_right` = NULL
- uint16\_t `i2s_in_block_offset` = 0
- bool `i2s_in_update_responsibility` = false
- raw\_sample\_t \* `i2s_out_block_left_1st` = NULL
- raw\_sample\_t \* `i2s_out_block_right_1st` = NULL
- raw\_sample\_t \* `i2s_out_block_left_2nd` = NULL
- raw\_sample\_t \* `i2s_out_block_right_2nd` = NULL
- uint16\_t `i2s_out_block_left_offset` = 0
- uint16\_t `i2s_out_block_right_offset` = 0
- bool `i2s_out_update_responsibility` = false
- raw\_sample\_t `i2s_input_blocks` [2][AUDIO\_BLOCK\_SAMPLES]
- raw\_sample\_t `i2s_output_blocks` [2][AUDIO\_BLOCK\_SAMPLES]

## 4.107.1 Function Documentation

### 4.107.1.1 `__attribute__()`

```
DMAMEM __attribute__ (
    (aligned(32)) )
```

### 4.107.1.2 `configure_i2s_dma()`

```
void configure_i2s_dma ()
```

### 4.107.1.3 `i2s_in_dma()`

```
DMAChannel i2s_in_dma (
    false )
```

#### 4.107.1.4 i2s\_in\_transmit()

```
void i2s_in_transmit (
    raw_sample_t * block,
    unsigned char index)
```

#### 4.107.1.5 i2s\_input\_update()

```
void i2s_input_update ()
```

#### 4.107.1.6 i2s\_out\_dma()

```
DMAChannel i2s_out_dma (
    false )
```

#### 4.107.1.7 i2s\_output\_transmit\_mono\_float()

```
void i2s_output_transmit_mono_float (
    float * block)
```

#### 4.107.1.8 i2s\_output\_transmit\_mono\_int()

```
void i2s_output_transmit_mono_int (
    raw_sample_t * block)
```

#### 4.107.1.9 i2s\_output\_update()

```
void i2s_output_update ()
```

#### 4.107.1.10 init\_i2s\_dma()

```
void init_i2s_dma ()
```

#### 4.107.1.11 m\_eng\_i2s\_input\_isr()

```
void m_eng_i2s_input_isr ()
```

#### 4.107.1.12 m\_eng\_i2s\_output\_isr()

```
void m_eng_i2s_output_isr ()
```

## 4.107.2 Variable Documentation

### 4.107.2.1 i2s\_in\_block\_left

```
raw_sample_t* i2s_in_block_left = NULL
```

### 4.107.2.2 i2s\_in\_block\_offset

```
uint16_t i2s_in_block_offset = 0
```

### 4.107.2.3 i2s\_in\_block\_right

```
raw_sample_t* i2s_in_block_right = NULL
```

### 4.107.2.4 i2s\_in\_update\_responsibility

```
bool i2s_in_update_responsibility = false
```

### 4.107.2.5 i2s\_input\_blocks

```
raw_sample_t i2s_input_blocks[2][AUDIO_BLOCK_SAMPLES]
```

### 4.107.2.6 i2s\_out\_block\_left\_1st

```
raw_sample_t* i2s_out_block_left_1st = NULL
```

### 4.107.2.7 i2s\_out\_block\_left\_2nd

```
raw_sample_t* i2s_out_block_left_2nd = NULL
```

### 4.107.2.8 i2s\_out\_block\_left\_offset

```
uint16_t i2s_out_block_left_offset = 0
```

### 4.107.2.9 i2s\_out\_block\_right\_1st

```
raw_sample_t* i2s_out_block_right_1st = NULL
```

### 4.107.2.10 i2s\_out\_block\_right\_2nd

```
raw_sample_t* i2s_out_block_right_2nd = NULL
```

#### 4.107.2.11 i2s\_out\_block\_right\_offset

```
uint16_t i2s_out_block_right_offset = 0
```

#### 4.107.2.12 i2s\_out\_update\_responsibility

```
bool i2s_out_update_responsibility = false
```

#### 4.107.2.13 i2s\_output\_blocks

```
raw_sample_t i2s_output_blocks[2][AUDIO_BLOCK_SAMPLES]
```

## 4.108 m\_eng\_linkowitz\_riley.c File Reference

```
#include "m_eng.h"
```

### Functions

- [int init\\_lr\\_low\\_pass\\_filter\\_str](#) ([m\\_lr\\_low\\_pass\\_filter\\_str](#) \*str)
- [int reconfigure\\_lr\\_low\\_pass\\_filter](#) (void \*data\_struct)
- [int calc\\_lr\\_low\\_pass\\_filter](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- [int init\\_lr\\_high\\_pass\\_filter\\_str](#) ([m\\_lr\\_high\\_pass\\_filter\\_str](#) \*str)
- [int reconfigure\\_lr\\_high\\_pass\\_filter](#) (void \*data\_struct)
- [int calc\\_lr\\_high\\_pass\\_filter](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.108.1 Function Documentation

#### 4.108.1.1 calc\_lr\_high\_pass\_filter()

```
int calc_lr_high_pass_filter (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.108.1.2 calc\_lr\_low\_pass\_filter()

```
int calc_lr_low_pass_filter (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.108.1.3 init\_lr\_high\_pass\_filter\_str()

```
int init_lr_high_pass_filter_str (
    m_lr_high_pass_filter_str * str)
```

#### 4.108.1.4 init\_lr\_low\_pass\_filter\_str()

```
int init_lr_low_pass_filter_str (
    m_lr_low_pass_filter_str * str)
```

#### 4.108.1.5 reconfigure\_lr\_high\_pass\_filter()

```
int reconfigure_lr_high_pass_filter (
    void * data_struct)
```

#### 4.108.1.6 reconfigure\_lr\_low\_pass\_filter()

```
int reconfigure_lr_low_pass_filter (
    void * data_struct)
```

### 4.109 m\_eng\_logging.cpp File Reference

```
#include <Arduino.h>
#include "m_eng.h"
```

#### Macros

- #define M\_ENG\_LOG\_ENTRIES\_N 64
- #define MESSAGE\_BEGIN\_COL 43
- #define LOG\_ENTRIES\_PRINT\_BUF\_LEN 4096
- #define M\_ENG\_PROFILER\_ARRAY\_N 256
- #define M\_ENG\_PROFILER\_RA\_CYCLES\_ALPHA 0.9



## Functions

- int [format\\_log\\_entry](#) (int index, char \*buf, int max\_len, int max\_indent)
- void [m\\_eng\\_trace\\_log\\_begin](#) (const char \*fname, const char \*line, const char \*function, int local\_trace\_↵ depth, uint64\_t cycle)
- void [m\\_eng\\_trace\\_log\\_return](#) (const char \*fname, const char \*line, const char \*function, int local\_trace\_↵ depth, uint64\_t cycle)
- void [m\\_eng\\_log\\_error\\_code](#) (const char \*fname, const char \*line, const char \*function, int error\_code, uint64\_t cycle)
- void [m\\_eng\\_log\\_return\\_err](#) (const char \*fname, const char \*line, const char \*function, int error\_code, uint64\_t cycle)
- void [m\\_eng\\_log\\_return\\_ptr](#) (const char \*fname, const char \*line, const char \*function, void \*ptr, uint64\_t cycle)
- void [m\\_eng\\_log\\_return\\_int](#) (const char \*fname, const char \*line, const char \*function, int val, uint64\_t cycle)
- void [m\\_eng\\_log\\_return\\_](#) (const char \*fname, const char \*line, const char \*function, uint64\_t cycle)
- void [m\\_eng\\_log\\_message](#) (const char \*fname, const char \*line, const char \*function, uint64\_t cycle, const char \*fmt,...)
- void [m\\_eng\\_print\\_flush\\_log](#) ()
- void [m\\_eng\\_init\\_profiler](#) ()
- void [m\\_eng\\_profiler\\_log\\_entry](#) (const char \*function\_name)
- void [m\\_eng\\_profiler\\_log\\_return](#) (const char \*function\_name, uint64\_t cycle)
- void [m\\_eng\\_profiler\\_sort](#) ()
- void [m\\_eng\\_profiler\\_print](#) ()

## 4.109.1 Macro Definition Documentation

### 4.109.1.1 LOG\_ENTRIES\_PRINT\_BUF\_LEN

```
#define LOG_ENTRIES_PRINT_BUF_LEN 4096
```

### 4.109.1.2 M\_ENG\_LOG\_ENTRIES\_N

```
#define M_ENG_LOG_ENTRIES_N 64
```

### 4.109.1.3 M\_ENG\_PROFILER\_ARRAY\_N

```
#define M_ENG_PROFILER_ARRAY_N 256
```

### 4.109.1.4 M\_ENG\_PROFILER\_RA\_CYCLES\_ALPHA

```
#define M_ENG_PROFILER_RA_CYCLES_ALPHA 0.9
```

### 4.109.1.5 MESSAGE\_BEGIN\_COL

```
#define MESSAGE_BEGIN_COL 43
```

## 4.109.2 Function Documentation

### 4.109.2.1 `format_log_entry()`

```
int format_log_entry (  
    int index,  
    char * buf,  
    int max_len,  
    int max_indent)
```

### 4.109.2.2 `m_eng_init_profiler()`

```
void m_eng_init_profiler ()
```

### 4.109.2.3 `m_eng_log_error_code()`

```
void m_eng_log_error_code (  
    const char * fname,  
    const char * line,  
    const char * function,  
    int error_code,  
    uint64_t cycle)
```

### 4.109.2.4 `m_eng_log_message()`

```
void m_eng_log_message (  
    const char * fname,  
    const char * line,  
    const char * function,  
    uint64_t cycle,  
    const char * fmt,  
    ...)
```

### 4.109.2.5 `m_eng_log_return_()`

```
void m_eng_log_return_ (  
    const char * fname,  
    const char * line,  
    const char * function,  
    uint64_t cycle)
```

### 4.109.2.6 `m_eng_log_return_err()`

```
void m_eng_log_return_err (  
    const char * fname,  
    const char * line,  
    const char * function,  
    int error_code,  
    uint64_t cycle)
```

#### 4.109.2.7 m\_eng\_log\_return\_int()

```
void m_eng_log_return_int (
    const char * fname,
    const char * line,
    const char * function,
    int val,
    uint64_t cycle)
```

#### 4.109.2.8 m\_eng\_log\_return\_ptr()

```
void m_eng_log_return_ptr (
    const char * fname,
    const char * line,
    const char * function,
    void * ptr,
    uint64_t cycle)
```

#### 4.109.2.9 m\_eng\_print\_flush\_log()

```
void m_eng_print_flush_log ()
```

#### 4.109.2.10 m\_eng\_profiler\_log\_entry()

```
void m_eng_profiler_log_entry (
    const char * function_name)
```

#### 4.109.2.11 m\_eng\_profiler\_log\_return()

```
void m_eng_profiler_log_return (
    const char * function_name,
    uint64_t cycle)
```

#### 4.109.2.12 m\_eng\_profiler\_print()

```
void m_eng_profiler_print ()
```

#### 4.109.2.13 m\_eng\_profiler\_sort()

```
void m_eng_profiler_sort ()
```

#### 4.109.2.14 m\_eng\_trace\_log\_begin()

```
void m_eng_trace_log_begin (
    const char * fname,
    const char * line,
    const char * function,
    int local_trace_depth,
    uint64_t cycle)
```

#### 4.109.2.15 m\_eng\_trace\_log\_return()

```
void m_eng_trace_log_return (
    const char * fname,
    const char * line,
    const char * function,
    int local_trace_depth,
    uint64_t cycle)
```

### 4.110 m\_eng\_low\_end\_compressor.c File Reference

```
#include "m_eng.h"
```

#### Functions

- int [init\\_low\\_end\\_compressor\\_str](#) ([m\\_eng\\_low\\_end\\_compressor\\_str](#) \*str)
- int [reconfigure\\_low\\_end\\_compressor](#) (void \*data\_struct)
- int [calc\\_low\\_end\\_compressor](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

#### 4.110.1 Function Documentation

##### 4.110.1.1 calc\_low\_end\_compressor()

```
int calc_low_end_compressor (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

##### 4.110.1.2 init\_low\_end\_compressor\_str()

```
int init_low_end_compressor_str (
    m\_eng\_low\_end\_compressor\_str * str)
```

##### 4.110.1.3 reconfigure\_low\_end\_compressor()

```
int reconfigure_low_end_compressor (
    void * data_struct)
```

### 4.111 m\_eng\_mempool.c File Reference

```
#include "m_eng.h"
```

## Macros

- #define `BUFFER_QUEUE_STATIC`
- #define `MEMPOOL_MALLOC_TRIES` 6

## Functions

- FLASHMEM void `init_mem_pools` ()
- float \* `allocate_buffer` ()
- void `release_buffer` (float \*buffer)
- void `print_mempool_info` ()

## Variables

- int `head` = 0
- int `tail` = `MEM_SIZE` - 1
- int `mem_pools_initialised` = 0
- float `buffer_pool` [`M_BUFFER_POOL_SIZE`][`AUDIO_BLOCK_SAMPLES`]
- float \* `buffer_buffer` [`M_BUFFER_POOL_SIZE`]
- int `buffer_head` = 0
- int `buffer_tail` = `M_BUFFER_POOL_SIZE` - 1
- float `zero_buffer` [`AUDIO_BLOCK_SAMPLES`]
- float `sink_buffer` [`AUDIO_BLOCK_SAMPLES`]

## 4.111.1 Macro Definition Documentation

### 4.111.1.1 BUFFER\_QUEUE\_STATIC

```
#define BUFFER_QUEUE_STATIC
```

### 4.111.1.2 MEMPOOL\_MALLOC\_TRIES

```
#define MEMPOOL_MALLOC_TRIES 6
```

## 4.111.2 Function Documentation

### 4.111.2.1 allocate\_buffer()

```
float * allocate_buffer ()
```

### 4.111.2.2 init\_mem\_pools()

```
FLASHMEM void init_mem_pools ()
```

#### 4.111.2.3 print\_mempool\_info()

```
void print_mempool_info ()
```

#### 4.111.2.4 release\_buffer()

```
void release_buffer (  
    float * buffer)
```

### 4.111.3 Variable Documentation

#### 4.111.3.1 buffer\_buffer

```
float* buffer_buffer[M_BUFFER_POOL_SIZE]
```

#### 4.111.3.2 buffer\_head

```
int buffer_head = 0
```

#### 4.111.3.3 buffer\_pool

```
float buffer_pool[M_BUFFER_POOL_SIZE][AUDIO_BLOCK_SAMPLES]
```

#### 4.111.3.4 buffer\_tail

```
int buffer_tail = M_BUFFER_POOL_SIZE - 1
```

#### 4.111.3.5 head

```
int head = 0
```

#### 4.111.3.6 mem\_pools\_initialised

```
int mem_pools_initialised = 0
```

#### 4.111.3.7 sink\_buffer

```
float sink_buffer[AUDIO_BLOCK_SAMPLES]
```

#### 4.111.3.8 tail

```
int tail = MEM_SIZE - 1
```

#### 4.111.3.9 zero\_buffer

```
float zero_buffer[AUDIO_BLOCK_SAMPLES]
```

## 4.112 m\_eng\_noise\_suppressor.c File Reference

```
#include "m_eng.h"
```

### Functions

- int [reconfigure\\_noise\\_suppressor](#) (void \*data\_struct)
- int [calc\\_noise\\_suppressor](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- int [init\\_noise\\_suppressor\\_str](#) ([m\\_eng\\_noise\\_suppressor\\_str](#) \*str)

### 4.112.1 Function Documentation

#### 4.112.1.1 calc\_noise\_suppressor()

```
int calc_noise_suppressor (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.112.1.2 init\_noise\_suppressor\_str()

```
int init_noise_suppressor_str (  
    m\_eng\_noise\_suppressor\_str * str)
```

#### 4.112.1.3 reconfigure\_noise\_suppressor()

```
int reconfigure_noise_suppressor (  
    void * data_struct)
```

## 4.113 m\_eng\_parameter.c File Reference

```
#include "m_eng.h"
```

### Functions

- void [init\\_parameter](#) ([m\\_parameter](#) \*param, float initial, float min, float max, float max\_jump, int scale)
- int [init\\_setting](#) ([m\\_setting](#) \*setting, int16\_t initial)
- int [update\\_setting](#) ([m\\_setting](#) \*setting, uint16\_t new\_value)

### 4.113.1 Function Documentation

#### 4.113.1.1 `init_parameter()`

```
void init_parameter (
    m_parameter * param,
    float initial,
    float min,
    float max,
    float max_jump,
    int scale)
```

#### 4.113.1.2 `init_setting()`

```
int init_setting (
    m_setting * setting,
    int16_t initial)
```

#### 4.113.1.3 `update_setting()`

```
int update_setting (
    m_setting * setting,
    uint16_t new_value)
```

## 4.114 `m_eng_pass_filter.c` File Reference

```
#include "m_eng.h"
```

### Functions

- `int reconfigure_low_pass_filter` (void \*data\_struct)
- `int calc_low_pass_filter` (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- `int init_low_pass_filter_str` (m\_eng\_low\_pass\_filter\_str \*str)
- `int reconfigure_high_pass_filter` (void \*data\_struct)
- `int calc_high_pass_filter` (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- `int init_high_pass_filter_str` (m\_eng\_high\_pass\_filter\_str \*str)
- `int reconfigure_band_pass_filter` (void \*data\_struct)
- `int calc_band_pass_filter` (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- `int init_band_pass_filter_str` (m\_eng\_band\_pass\_filter\_str \*str)

### 4.114.1 Function Documentation

#### 4.114.1.1 `calc_band_pass_filter()`

```
int calc_band_pass_filter (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```



#### 4.114.1.2 calc\_high\_pass\_filter()

```
int calc_high_pass_filter (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.114.1.3 calc\_low\_pass\_filter()

```
int calc_low_pass_filter (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.114.1.4 init\_band\_pass\_filter\_str()

```
int init_band_pass_filter_str (  
    m_eng_band_pass_filter_str * str)
```

#### 4.114.1.5 init\_high\_pass\_filter\_str()

```
int init_high_pass_filter_str (  
    m_eng_high_pass_filter_str * str)
```

#### 4.114.1.6 init\_low\_pass\_filter\_str()

```
int init_low_pass_filter_str (  
    m_eng_low_pass_filter_str * str)
```

#### 4.114.1.7 reconfigure\_band\_pass\_filter()

```
int reconfigure_band_pass_filter (  
    void * data_struct)
```

#### 4.114.1.8 reconfigure\_high\_pass\_filter()

```
int reconfigure_high_pass_filter (  
    void * data_struct)
```

#### 4.114.1.9 reconfigure\_low\_pass\_filter()

```
int reconfigure_low_pass_filter (  
    void * data_struct)
```

## 4.115 m\_eng\_percussifier.c File Reference

```
#include "m_eng.h"
```

### Functions

- int [reconfigure\\_percussifier](#) (void \*data\_struct)
- int [calc\\_percussifier](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)
- int [init\\_percussifier\\_str](#) ([m\\_eng\\_percussifier\\_str](#) \*str)

### 4.115.1 Function Documentation

#### 4.115.1.1 calc\_percussifier()

```
int calc_percussifier (  
    void * data_struct,  
    float * dest,  
    float * src,  
    int n_samples)
```

#### 4.115.1.2 init\_percussifier\_str()

```
int init_percussifier_str (  
    m\_eng\_percussifier\_str * str)
```

#### 4.115.1.3 reconfigure\_percussifier()

```
int reconfigure_percussifier (  
    void * data_struct)
```

## 4.116 m\_eng\_pipeline.c File Reference

```
#include "m_eng.h"
```

### Macros

- [#define TRANSFORMERS\\_MALLOC\\_CHUNK\\_SIZE](#) 8

## Functions

- int [init\\_pipeline](#) ([m\\_pipeline](#) \*pipeline)
- int [compute\\_pipeline](#) ([m\\_pipeline](#) \*pipeline, float \*dest, float \*src)
- int [pipeline\\_expand\\_transformer\\_array](#) ([m\\_pipeline](#) \*pipeline)
- int [pipeline\\_expand\\_transformer\\_array\\_to](#) ([m\\_pipeline](#) \*pipeline, int n)
- int [pipeline\\_update\\_transition\\_policy](#) ([m\\_pipeline](#) \*pipeline)
- int [pipeline\\_print\\_transformer\\_array](#) ([m\\_pipeline](#) \*pipeline)
- int [pipeline\\_append\\_transformer](#) ([m\\_pipeline](#) \*pipeline, [m\\_transformer](#) \*trans)
- int [pipeline\\_remove\\_transformer](#) ([m\\_pipeline](#) \*pipeline, uint16\_t tid)
- int [pipeline\\_insert\\_transformer](#) ([m\\_pipeline](#) \*pipeline, [m\\_transformer](#) \*trans, int pos)
- int [pipeline\\_prepend\\_transformer](#) ([m\\_pipeline](#) \*pipeline, [m\\_transformer](#) \*trans)
- int [pipeline\\_append\\_transformer\\_type](#) ([m\\_pipeline](#) \*pipeline, uint16\_t type)
- int [pipeline\\_insert\\_transformer\\_type](#) ([m\\_pipeline](#) \*pipeline, uint16\_t type, uint16\_t pos)
- int [pipeline\\_prepend\\_transformer\\_type](#) ([m\\_pipeline](#) \*pipeline, uint16\_t type)
- int [pipeline\\_get\\_transformer\\_position](#) ([m\\_pipeline](#) \*pipeline, uint16\_t id)
- int [pipeline\\_move\\_transformer](#) ([m\\_pipeline](#) \*pipeline, uint16\_t id, int new\_pos)
- [m\\_transformer](#) \* [pipeline\\_get\\_transformer\\_by\\_id](#) ([m\\_pipeline](#) \*pipeline, uint16\_t id)
- int [pipeline\\_swap\\_transformers](#) ([m\\_pipeline](#) \*pipeline, uint16\_t id1, uint16\_t id2)
- int [pipeline\\_valid](#) ([m\\_pipeline](#) \*pipeline)
- int [pipeline\\_compare](#) ([m\\_pipeline](#) \*pipeline\_a, [m\\_pipeline](#) \*pipeline\_b)
- int [gut\\_pipeline](#) ([m\\_pipeline](#) \*pipeline)
- int [clone\\_pipeline](#) ([m\\_pipeline](#) \*\*dest\_ptr, [m\\_pipeline](#) \*src)
- int [pipeline\\_clone\\_transformer\\_into\\_position](#) ([m\\_pipeline](#) \*pipeline, [m\\_transformer](#) \*trans, int pos)
- int [pipeline\\_change\\_transformer\\_setting](#) ([m\\_pipeline](#) \*pipeline, uint16\_t tid, uint16\_t sid, int16\_t new\_val)

## 4.116.1 Macro Definition Documentation

### 4.116.1.1 TRANSFORMERS\_MALLOC\_CHUNK\_SIZE

```
#define TRANSFORMERS_MALLOC_CHUNK_SIZE 8
```

## 4.116.2 Function Documentation

### 4.116.2.1 clone\_pipeline()

```
int clone_pipeline (
    m\_pipeline ** dest_ptr,
    m\_pipeline * src)
```

### 4.116.2.2 compute\_pipeline()

```
int compute_pipeline (
    m\_pipeline * pipeline,
    float * dest,
    float * src)
```

#### 4.116.2.3 `gut_pipeline()`

```
int gut_pipeline (  
    m_pipeline * pipeline)
```

#### 4.116.2.4 `init_pipeline()`

```
int init_pipeline (  
    m_pipeline * pipeline)
```

#### 4.116.2.5 `pipeline_append_transformer()`

```
int pipeline_append_transformer (  
    m_pipeline * pipeline,  
    m_transformer * trans)
```

#### 4.116.2.6 `pipeline_append_transformer_type()`

```
int pipeline_append_transformer_type (  
    m_pipeline * pipeline,  
    uint16_t type)
```

#### 4.116.2.7 `pipeline_change_transformer_setting()`

```
int pipeline_change_transformer_setting (  
    m_pipeline * pipeline,  
    uint16_t tid,  
    uint16_t sid,  
    int16_t new_val)
```

#### 4.116.2.8 `pipeline_clone_transformer_into_position()`

```
int pipeline_clone_transformer_into_position (  
    m_pipeline * pipeline,  
    m_transformer * trans,  
    int pos)
```

#### 4.116.2.9 `pipeline_compare()`

```
int pipeline_compare (  
    m_pipeline * pipeline_a,  
    m_pipeline * pipeline_b)
```

#### 4.116.2.10 `pipeline_expand_transformer_array()`

```
int pipeline_expand_transformer_array (  
    m_pipeline * pipeline)
```

**4.116.2.11 pipeline\_expand\_transformer\_array\_to()**

```
int pipeline_expand_transformer_array_to (  
    m_pipeline * pipeline,  
    int n)
```

**4.116.2.12 pipeline\_get\_transformer\_by\_id()**

```
m_transformer * pipeline_get_transformer_by_id (  
    m_pipeline * pipeline,  
    uint16_t id)
```

**4.116.2.13 pipeline\_get\_transformer\_position()**

```
int pipeline_get_transformer_position (  
    m_pipeline * pipeline,  
    uint16_t id)
```

**4.116.2.14 pipeline\_insert\_transformer()**

```
int pipeline_insert_transformer (  
    m_pipeline * pipeline,  
    m_transformer * trans,  
    int pos)
```

**4.116.2.15 pipeline\_insert\_transformer\_type()**

```
int pipeline_insert_transformer_type (  
    m_pipeline * pipeline,  
    uint16_t type,  
    uint16_t pos)
```

**4.116.2.16 pipeline\_move\_transformer()**

```
int pipeline_move_transformer (  
    m_pipeline * pipeline,  
    uint16_t id,  
    int new_pos)
```

**4.116.2.17 pipeline\_prepend\_transformer()**

```
int pipeline_prepend_transformer (  
    m_pipeline * pipeline,  
    m_transformer * trans)
```

**4.116.2.18 pipeline\_prepend\_transformer\_type()**

```
int pipeline_prepend_transformer_type (
    m_pipeline * pipeline,
    uint16_t type)
```

**4.116.2.19 pipeline\_print\_transformer\_array()**

```
int pipeline_print_transformer_array (
    m_pipeline * pipeline)
```

**4.116.2.20 pipeline\_remove\_transformer()**

```
int pipeline_remove_transformer (
    m_pipeline * pipeline,
    uint16_t tid)
```

**4.116.2.21 pipeline\_swap\_transformers()**

```
int pipeline_swap_transformers (
    m_pipeline * pipeline,
    uint16_t id1,
    uint16_t id2)
```

**4.116.2.22 pipeline\_update\_transition\_policy()**

```
int pipeline_update_transition_policy (
    m_pipeline * pipeline)
```

**4.116.2.23 pipeline\_valid()**

```
int pipeline_valid (
    m_pipeline * pipeline)
```

**4.117 m\_eng\_pipeline\_mod.c File Reference**

```
#include "m_eng.h"
```

**Functions**

- [IMPLEMENT\\_LINKED\\_LIST](#) (m\_pipeline\_mod)
- [m\\_pipeline\\_mod\\_create\\_pipeline\\_mod\\_append\\_transformer](#) (uint16\_t type)
- [m\\_pipeline\\_mod\\_create\\_pipeline\\_mod\\_move\\_transformer](#) (uint16\_t tid, uint16\_t position)
- [m\\_pipeline\\_mod\\_create\\_pipeline\\_mod\\_remove\\_transformer](#) (uint16\_t tid)
- [m\\_pipeline\\_mod\\_create\\_pipeline\\_mod\\_change\\_transformer\\_setting](#) (uint16\_t tid, uint16\_t setting\_id, int16\_t new\_value)
- const char \* [pipeline\\_mod\\_type\\_string](#) (int type)
- int [apply\\_pipeline\\_mod](#) (m\_pipeline \*pipeline, m\_pipeline\_mod mod, int \*err\_code)

## 4.117.1 Function Documentation

### 4.117.1.1 apply\_pipeline\_mod()

```
int apply_pipeline_mod (  
    m_pipeline * pipeline,  
    m_pipeline_mod mod,  
    int * err_code)
```

### 4.117.1.2 create\_pipeline\_mod\_append\_transformer()

```
m_pipeline_mod create_pipeline_mod_append_transformer (  
    uint16_t type)
```

### 4.117.1.3 create\_pipeline\_mod\_change\_transformer\_setting()

```
m_pipeline_mod create_pipeline_mod_change_transformer_setting (  
    uint16_t tid,  
    uint16_t setting_id,  
    int16_t new_value)
```

### 4.117.1.4 create\_pipeline\_mod\_move\_transformer()

```
m_pipeline_mod create_pipeline_mod_move_transformer (  
    uint16_t tid,  
    uint16_t position)
```

### 4.117.1.5 create\_pipeline\_mod\_remove\_transformer()

```
m_pipeline_mod create_pipeline_mod_remove_transformer (  
    uint16_t tid)
```

### 4.117.1.6 IMPLEMENT\_LINKED\_LIST()

```
IMPLEMENT_LINKED_LIST (  
    m_pipeline_mod )
```

### 4.117.1.7 pipeline\_mod\_type\_string()

```
const char * pipeline_mod_type_string (  
    int type)
```

## 4.118 m\_eng\_printf.cpp File Reference

```
#include "m_eng.h"
```

## Macros

- `#define SPACING 7`
- `#define SPACING 7`

## Functions

- void `m_printf` (const char \*fmt,...)
- void `m_voice_printf` (int who, const char \*fmt,...)
- void `m_mute_voice` (int who)
- void `m_unmute_voice` (int who)
- void `serial_print_blocks` (int n,...)
- void `pretty_print_block` (int16\_t \*data, const char \*start)
- void `pretty_print_block_float` (float \*data, const char \*start)

## Variables

- const char \* `voice_colour_table` [64]

### 4.118.1 Macro Definition Documentation

#### 4.118.1.1 SPACING [1/2]

```
#define SPACING 7
```

#### 4.118.1.2 SPACING [2/2]

```
#define SPACING 7
```

### 4.118.2 Function Documentation

#### 4.118.2.1 m\_mute\_voice()

```
void m_mute_voice (  
    int who)
```

#### 4.118.2.2 m\_printf()

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

#### 4.118.2.3 m\_unmute\_voice()

```
void m_unmute_voice (  
    int who)
```



#### 4.118.2.4 m\_voice\_printf()

```
void m_voice_printf (
    int who,
    const char * fmt,
    ...)
```

#### 4.118.2.5 pretty\_print\_block()

```
void pretty_print_block (
    int16_t * data,
    const char * start)
```

#### 4.118.2.6 pretty\_print\_block\_float()

```
void pretty_print_block_float (
    float * data,
    const char * start)
```

#### 4.118.2.7 serial\_print\_blocks()

```
void serial_print_blocks (
    int n,
    ...)
```

### 4.118.3 Variable Documentation

#### 4.118.3.1 voice colour table

```
const char* voice_colour_table[64]
```

**Initial value:**

[illegible]

#### 4.119 m\_eng\_profile.c File Reference

```
#include "m_eng.h"
```

## Functions

- `int nullify_profile (m_eng_profile *profile)`
- `int init_profile (m_eng_profile *profile)`
- `int profile_print_job_list (m_eng_profile *profile)`
- `int profile_print_ujob_list (m_eng_profile *profile)`
- `int profile_apply_pipeline_mod (m_eng_profile *profile, m_pipeline_mod mod)`  
*Applies the given modification to the given profile.*
- `int profile_update (m_eng_profile *profile)`
- `int profile_process (m_eng_profile *profile, float *dest, float *src)`
- `int profile_trigger_pipeline_swap (m_eng_profile *profile)`
- `int profile_regenerate_back_pipeline (m_eng_profile *profile)`
- `int profile_scheduled_maintenance (m_eng_profile *profile)`

## 4.119.1 Function Documentation

### 4.119.1.1 init\_profile()

```
int init_profile (
    m_eng_profile * profile)
```

### 4.119.1.2 nullify\_profile()

```
int nullify_profile (
    m_eng_profile * profile)
```

### 4.119.1.3 profile\_apply\_pipeline\_mod()

```
int profile_apply_pipeline_mod (
    m_eng_profile * profile,
    m_pipeline_mod mod)
```

Applies the given modification to the given profile.

If the current profile is active, to prevent audio artifacts, pipeline modifications ("mods", encoded by `m_pipeline_mod`) are applied in the background - to an inactive copy of the profile's pipeline (the "back pipeline"), and the outputs from the front and back pipeline are smoothly cross-faded between, after which it can be safely applied to the (now back, previously front) pipeline, keeping the two in sync.

To achieve this, `profile_apply_pipeline_mod` calls `apply_pipeline_mod` to apply the mod to the back pipeline, calls `profile_trigger_pipeline_swap` to trigger the pipeline swap, and saves the mod in a linked list (`m_eng_profile::jobs`). After the swap is complete, `profile_update` will retrieve the job from the list and apply it to the back (previously front) pipeline.

If, however, when `profile_apply_pipeline_mod` is called, a pipeline swap is already in progress, the mod is *not* applied, but rather stored in a second linked list, (`m_eng_profile::blocked_jobs`), whereupon `profile_update` will apply it (after the swap is finished and any jobs waiting in (`m_eng_profile::jobs`) have been applied) by calling `profile_apply_pipeline_mod`.

## Parameters

<i>profile</i>	The profile to be modified
<i>mod</i>	A struct describing the modification to apply

**4.119.1.4 profile\_print\_job\_list()**

```
int profile_print_job_list (  
    m_eng_profile * profile)
```

**4.119.1.5 profile\_print\_ujob\_list()**

```
int profile_print_ujob_list (  
    m_eng_profile * profile)
```

**4.119.1.6 profile\_process()**

```
int profile_process (  
    m_eng_profile * profile,  
    float * dest,  
    float * src)
```

**4.119.1.7 profile\_regenerate\_back\_pipeline()**

```
int profile_regenerate_back_pipeline (  
    m_eng_profile * profile)
```

**4.119.1.8 profile\_scheduled\_maintenance()**

```
int profile_scheduled_maintenance (  
    m_eng_profile * profile)
```

**4.119.1.9 profile\_trigger\_pipeline\_swap()**

```
int profile_trigger_pipeline_swap (  
    m_eng_profile * profile)
```

**4.119.1.10 profile\_update()**

```
int profile_update (  
    m_eng_profile * profile)
```

## 4.120 m\_eng\_sgtl5000.cpp File Reference

```
#include <Arduino.h>
#include <Wire.h>
#include "m_eng.h"
#include "m_eng_sgtl5000_defs.h"
```

### Functions

- int [sgtl5000\\_volume](#) (float n)
- int [sgtl5000\\_mute\\_headphone](#) ()
- int [sgtl5000\\_unmute\\_headphone](#) ()
- int [sgtl5000\\_mute\\_line\\_out](#) ()
- int [sgtl5000\\_unmute\\_line\\_out](#) ()
- void [sgtl5000\\_set\\_address](#) (uint8\_t level)
- int [sgtl5000\\_start](#) ()
- int [sgtl5000\\_enable](#) ()
- unsigned int [sgtl5000\\_read\\_reg](#) (unsigned int reg)
- int [sgtl5000\\_write\\_reg](#) (unsigned int reg, unsigned int val)
- unsigned int [sgtl5000\\_modify\\_reg](#) (unsigned int reg, unsigned int val, unsigned int i\_mask)
- int [sgtl5000\\_volum\\_eng\\_integer](#) (unsigned int n)
- int [sgtl5000\\_line\\_in\\_level](#) (uint8\_t n)
- unsigned short [sgtl5000\\_line\\_out\\_level](#) (uint8\_t n)
- unsigned short [sgtl5000\\_adc\\_high\\_pass\\_filter\\_enable](#) ()
- unsigned short [sgtl5000\\_adc\\_high\\_pass\\_filter\\_freeze](#) ()
- unsigned short [sgtl5000\\_adc\\_high\\_pass\\_filter\\_disable](#) ()
- void [sgtl5000\\_kill\\_automation](#) ()
- unsigned char [calc\\_vol](#) (float n, unsigned char range)
- unsigned short [sgtl5000\\_dap\\_audio\\_eq\\_band](#) (uint8\_t band\_num, float n)
- unsigned short [sgtl5000\\_eq\\_select](#) (uint8\_t n)
- void [sgtl5000\\_automate](#) (uint8\_t dap, uint8\_t eq)

### 4.120.1 Function Documentation

#### 4.120.1.1 [calc\\_vol\(\)](#)

```
unsigned char calc_vol (
    float n,
    unsigned char range)
```

#### 4.120.1.2 [sgtl5000\\_adc\\_high\\_pass\\_filter\\_disable\(\)](#)

```
unsigned short sgtl5000_adc_high_pass_filter_disable ()
```

#### 4.120.1.3 [sgtl5000\\_adc\\_high\\_pass\\_filter\\_enable\(\)](#)

```
unsigned short sgtl5000_adc_high_pass_filter_enable ()
```

**4.120.1.4 sgtl5000\_adc\_high\_pass\_filter\_freeze()**

```
unsigned short sgtl5000_adc_high_pass_filter_freeze ()
```

**4.120.1.5 sgtl5000\_automate()**

```
void sgtl5000_automate (  
    uint8_t dap,  
    uint8_t eq)
```

**4.120.1.6 sgtl5000\_dap\_audio\_eq\_band()**

```
unsigned short sgtl5000_dap_audio_eq_band (  
    uint8_t band_num,  
    float n)
```

**4.120.1.7 sgtl5000\_enable()**

```
int sgtl5000_enable ()
```

**4.120.1.8 sgtl5000\_eq\_select()**

```
unsigned short sgtl5000_eq_select (  
    uint8_t n)
```

**4.120.1.9 sgtl5000\_kill\_automation()**

```
void sgtl5000_kill_automation ()
```

**4.120.1.10 sgtl5000\_line\_in\_level()**

```
int sgtl5000_line_in_level (  
    uint8_t n)
```

**4.120.1.11 sgtl5000\_line\_out\_level()**

```
unsigned short sgtl5000_line_out_level (  
    uint8_t n)
```

**4.120.1.12 sgtl5000\_modify\_reg()**

```
unsigned int sgtl5000_modify_reg (  
    unsigned int reg,  
    unsigned int val,  
    unsigned int i_mask)
```

**4.120.1.13 sgtl5000\_mute\_headphone()**

```
int sgtl5000_mute_headphone ()
```

**4.120.1.14 sgtl5000\_mute\_line\_out()**

```
int sgtl5000_mute_line_out ()
```

**4.120.1.15 sgtl5000\_read\_reg()**

```
unsigned int sgtl5000_read_reg (  
    unsigned int reg)
```

**4.120.1.16 sgtl5000\_set\_address()**

```
void sgtl5000_set_address (  
    uint8_t level)
```

**4.120.1.17 sgtl5000\_start()**

```
int sgtl5000_start ()
```

**4.120.1.18 sgtl5000\_unmute\_headphone()**

```
int sgtl5000_unmute_headphone ()
```

**4.120.1.19 sgtl5000\_unmute\_line\_out()**

```
int sgtl5000_unmute_line_out ()
```

**4.120.1.20 sgtl5000\_volum\_eng\_integer()**

```
int sgtl5000_volum_eng_integer (  
    unsigned int n)
```

**4.120.1.21 sgtl5000\_volume()**

```
int sgtl5000_volume (  
    float n)
```

#### 4.120.1.22 sgtl5000\_write\_reg()

```
int sgtl5000_write_reg (
    unsigned int reg,
    unsigned int val)
```

## 4.121 m\_eng\_simple\_distortion.c File Reference

```
#include "m_eng.h"
```

### Functions

- int [init\\_simple\\_distortion\\_str](#) ([m\\_eng\\_simple\\_distortion\\_str](#) \*str)
- int [reconfigure\\_simple\\_distortion](#) (void \*data\_struct)
- int [calc\\_simple\\_distortion](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

### 4.121.1 Function Documentation

#### 4.121.1.1 calc\_simple\_distortion()

```
int calc_simple_distortion (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

#### 4.121.1.2 init\_simple\_distortion\_str()

```
int init_simple_distortion_str (
    m\_eng\_simple\_distortion\_str * str)
```

#### 4.121.1.3 reconfigure\_simple\_distortion()

```
int reconfigure_simple_distortion (
    void * data_struct)
```

## 4.122 m\_eng\_transformer.c File Reference

```
#include "m_eng.h"
```

### Macros

- [#define MAX\\_BLOCK\\_DIVIDER](#) 8

## Functions

- int [transformer\\_init\\_parameter\\_array](#) ([m\\_transformer](#) \*trans, int n)
- int [transformer\\_init\\_setting\\_array](#) ([m\\_transformer](#) \*trans, int n)
- int [transformer\\_init\\_controls](#) ([m\\_transformer](#) \*trans)
- void [run\\_bypass](#) (float \*\*dest, float \*\*src, int n\_inputs, int n\_valid\_inputs, int n\_outputs)
- int [run\\_transformer](#) ([m\\_transformer](#) \*trans, float \*dest, float \*src)
- int [transformer\\_add\\_setting](#) ([m\\_transformer](#) \*trans, [m\\_setting](#) \*setting)
- int [transformer\\_add\\_parameter](#) ([m\\_transformer](#) \*trans, [m\\_parameter](#) \*param)
- [m\\_parameter](#) \* [transformer\\_get\\_parameter](#) ([m\\_transformer](#) \*trans, uint16\_t ppid)
- [m\\_setting](#) \* [transformer\\_get\\_setting](#) ([m\\_transformer](#) \*trans, uint16\_t sid)
- void [free\\_transformer](#) ([m\\_transformer](#) \*trans)
- int [clone\\_transformer](#) ([m\\_transformer](#) \*\*dest\_ptr, [m\\_transformer](#) \*src)

## 4.122.1 Macro Definition Documentation

### 4.122.1.1 MAX\_BLOCK\_DIVIDER

```
#define MAX_BLOCK_DIVIDER 8
```

## 4.122.2 Function Documentation

### 4.122.2.1 clone\_transformer()

```
int clone_transformer (
    m\_transformer ** dest_ptr,
    m\_transformer * src)
```

### 4.122.2.2 free\_transformer()

```
void free_transformer (
    m\_transformer * trans)
```

### 4.122.2.3 run\_bypass()

```
void run_bypass (
    float ** dest,
    float ** src,
    int n_inputs,
    int n_valid_inputs,
    int n_outputs)
```

### 4.122.2.4 run\_transformer()

```
int run_transformer (
    m\_transformer * trans,
    float * dest,
    float * src)
```



#### 4.122.2.5 transformer\_add\_parameter()

```
int transformer_add_parameter (
    m_transformer * trans,
    m_parameter * param)
```

#### 4.122.2.6 transformer\_add\_setting()

```
int transformer_add_setting (
    m_transformer * trans,
    m_setting * setting)
```

#### 4.122.2.7 transformer\_get\_parameter()

```
m_parameter * transformer_get_parameter (
    m_transformer * trans,
    uint16_t ppid)
```

#### 4.122.2.8 transformer\_get\_setting()

```
m_setting * transformer_get_setting (
    m_transformer * trans,
    uint16_t sid)
```

#### 4.122.2.9 transformer\_init\_controls()

```
int transformer_init_controls (
    m_transformer * trans)
```

#### 4.122.2.10 transformer\_init\_parameter\_array()

```
int transformer_init_parameter_array (
    m_transformer * trans,
    int n)
```

#### 4.122.2.11 transformer\_init\_setting\_array()

```
int transformer_init_setting_array (
    m_transformer * trans,
    int n)
```

## 4.123 m\_eng\_transformer\_init.c File Reference

```
#include "m_eng.h"
```

## Functions

- `int init_3_band_eq (m_transformer *trans)`
- `int init_amplifier (m_transformer *trans)`
- `int init_band_pass_filter (m_transformer *trans)`
- `int init_compressor (m_transformer *trans)`
- `int init_delay (m_transformer *trans)`
- `int init_dirty_octave (m_transformer *trans)`
- `int init_distortion (m_transformer *trans)`
- `int init_envelope (m_transformer *trans)`
- `int init_flanger (m_transformer *trans)`
- `int init_high_pass_filter (m_transformer *trans)`
- `int init_low_end_compressor (m_transformer *trans)`
- `int init_low_pass_filter (m_transformer *trans)`
- `int init_noise_suppressor (m_transformer *trans)`
- `int init_percussifier (m_transformer *trans)`
- `int init_warbler (m_transformer *trans)`
- `int init_transformer (m_transformer *trans, uint16_t type)`

## 4.123.1 Function Documentation

### 4.123.1.1 `init_3_band_eq()`

```
int init_3_band_eq (  
    m_transformer * trans)
```

### 4.123.1.2 `init_amplifier()`

```
int init_amplifier (  
    m_transformer * trans)
```

### 4.123.1.3 `init_band_pass_filter()`

```
int init_band_pass_filter (  
    m_transformer * trans)
```

### 4.123.1.4 `init_compressor()`

```
int init_compressor (  
    m_transformer * trans)
```

### 4.123.1.5 `init_delay()`

```
int init_delay (  
    m_transformer * trans)
```

#### 4.123.1.6 init\_dirty\_octave()

```
int init_dirty_octave (  
    m_transformer * trans)
```

#### 4.123.1.7 init\_distortion()

```
int init_distortion (  
    m_transformer * trans)
```

#### 4.123.1.8 init\_envelope()

```
int init_envelope (  
    m_transformer * trans)
```

#### 4.123.1.9 init\_flanger()

```
int init_flanger (  
    m_transformer * trans)
```

#### 4.123.1.10 init\_high\_pass\_filter()

```
int init_high_pass_filter (  
    m_transformer * trans)
```

#### 4.123.1.11 init\_low\_end\_compressor()

```
int init_low_end_compressor (  
    m_transformer * trans)
```

#### 4.123.1.12 init\_low\_pass\_filter()

```
int init_low_pass_filter (  
    m_transformer * trans)
```

#### 4.123.1.13 init\_noise\_suppressor()

```
int init_noise_suppressor (  
    m_transformer * trans)
```

#### 4.123.1.14 init\_percussifier()

```
int init_percussifier (  
    m_transformer * trans)
```

#### 4.123.1.15 init\_transformer()

```
int init_transformer (  
    m_transformer * trans,  
    uint16_t type)
```

#### 4.123.1.16 init\_warbler()

```
int init_warbler (  
    m_transformer * trans)
```

### 4.124 m\_eng\_transformer\_template.c File Reference

### 4.125 m\_eng\_update.c File Reference

```
#include "m_eng.h"
```

#### Functions

- void [update\\_all](#) ()
- int [update\\_setup](#) ()
- void [update\\_stop](#) ()
- void [m\\_eng\\_software\\_isr](#) ()

#### 4.125.1 Function Documentation

##### 4.125.1.1 m\_eng\_software\_isr()

```
void m_eng_software_isr ()
```

##### 4.125.1.2 update\_all()

```
void update_all ()
```

##### 4.125.1.3 update\_setup()

```
int update_setup ()
```

##### 4.125.1.4 update\_stop()

```
void update_stop ()
```

## 4.126 m\_eng\_useful\_functions.c File Reference

```
#include "m_eng.h"
```

### Macros

- #define [DENORMAL\\_THRESHOLD](#) 1e-30f
- #define [FLOAT\\_TO\\_INT16\\_MAX](#) (32767.0f / 32768.0f)
- #define [SCALE\\_FACTOR](#) 32768.0f
- #define [MAX\\_INT](#) 32768.0

### Functions

- float [identity\\_function](#) (float x)
- float [normalised\\_arctan](#) (float x)
- float [hard\\_clip](#) (float x)
- float [soft\\_fold](#) (float x)
- float [trig\\_transition\\_function](#) (float x)
- int [convert\\_block\\_int\\_to\\_float](#) (float \*dest, int16\_t \*src)
- int [convert\\_block\\_float\\_to\\_int](#) (int16\_t \*dest, float \*src)

### 4.126.1 Macro Definition Documentation

#### 4.126.1.1 DENORMAL\_THRESHOLD

```
#define DENORMAL_THRESHOLD 1e-30f
```

#### 4.126.1.2 FLOAT\_TO\_INT16\_MAX

```
#define FLOAT_TO_INT16_MAX (32767.0f / 32768.0f)
```

#### 4.126.1.3 MAX\_INT

```
#define MAX_INT 32768.0
```

#### 4.126.1.4 SCALE\_FACTOR

```
#define SCALE_FACTOR 32768.0f
```

### 4.126.2 Function Documentation

#### 4.126.2.1 convert\_block\_float\_to\_int()

```
int convert_block_float_to_int (  
    int16_t * dest,  
    float * src)
```

#### 4.126.2.2 `convert_block_int_to_float()`

```
int convert_block_int_to_float (
    float * dest,
    int16_t * src)
```

#### 4.126.2.3 `hard_clip()`

```
float hard_clip (
    float x)
```

#### 4.126.2.4 `identity_function()`

```
float identity_function (
    float x)
```

#### 4.126.2.5 `normalised_arctan()`

```
float normalised_arctan (
    float x)
```

#### 4.126.2.6 `soft_fold()`

```
float soft_fold (
    float x)
```

#### 4.126.2.7 `trig_transition_function()`

```
float trig_transition_function (
    float x)
```

## 4.127 `m_eng_warbler.c` File Reference

```
#include "m_eng.h"
```

### Functions

- int [init\\_warbler\\_str](#) ([m\\_eng\\_warbler\\_str](#) \*str)
- int [reconfigure\\_warbler](#) (void \*data\_struct)
- int [calc\\_warbler](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

## 4.127.1 Function Documentation

### 4.127.1.1 calc\_warbler()

```
int calc_warbler (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

### 4.127.1.2 init\_warbler\_str()

```
int init_warbler_str (
    m_eng_warbler_str * str)
```

### 4.127.1.3 reconfigure\_warbler()

```
int reconfigure_warbler (
    void * data_struct)
```

## 4.128 m\_eng\_waveshaper.c File Reference

```
#include "m_eng.h"
```

### Functions

- int [init\\_waveshaper\\_str](#) ([m\\_eng\\_waveshaper\\_str](#) \*str)
- int [calc\\_waveshaper](#) (void \*data\_struct, float \*dest, float \*src, int n\_samples)

## 4.128.1 Function Documentation

### 4.128.1.1 calc\_waveshaper()

```
int calc_waveshaper (
    void * data_struct,
    float * dest,
    float * src,
    int n_samples)
```

### 4.128.1.2 init\_waveshaper\_str()

```
int init_waveshaper_str (
    m_eng_waveshaper_str * str)
```

## 4.129 m\_alloc.c File Reference

### Functions

- void \* [m\\_alloc](#) (size\_t size)
- void \* [m\\_realloc](#) (void \*ptr, size\_t size)
- char \* [m\\_strndup](#) (const char \*str, size\_t n)
- void [m\\_free](#) (void \*ptr)
- void [print\\_memory\\_report](#) ()

### 4.129.1 Function Documentation

#### 4.129.1.1 m\_alloc()

```
void * m_alloc (  
    size_t size)
```

#### 4.129.1.2 m\_free()

```
void m_free (  
    void * ptr)
```

#### 4.129.1.3 m\_realloc()

```
void * m_realloc (  
    void * ptr,  
    size_t size)
```

#### 4.129.1.4 m\_strndup()

```
char * m_strndup (  
    const char * str,  
    size_t n)
```

#### 4.129.1.5 print\_memory\_report()

```
void print_memory_report ()
```

## 4.130 m\_alloc.h File Reference

### Functions

- void \* [m\\_alloc](#) (size\_t size)
- void \* [m\\_realloc](#) (void \*ptr, size\_t size)
- char \* [m\\_strndup](#) (const char \*str, size\_t n)
- void [m\\_free](#) (void \*ptr)
- void \* [m\\_int\\_lv\\_malloc](#) (size\_t size)
- void [m\\_int\\_lv\\_free](#) (void \*ptr)
- void [print\\_memory\\_report](#) ()



## 4.130.1 Function Documentation

### 4.130.1.1 m\_alloc()

```
void * m_alloc (  
    size_t size)
```

### 4.130.1.2 m\_free()

```
void m_free (  
    void * ptr)
```

### 4.130.1.3 m\_int\_lv\_free()

```
void m_int_lv_free (  
    void * ptr)
```

### 4.130.1.4 m\_int\_lv\_malloc()

```
void * m_int_lv_malloc (  
    size_t size)
```

### 4.130.1.5 m\_realloc()

```
void * m_realloc (  
    void * ptr,  
    size_t size)
```

### 4.130.1.6 m\_strndup()

```
char * m_strndup (  
    const char * str,  
    size_t n)
```

### 4.130.1.7 print\_memory\_report()

```
void print_memory_report ()
```

## 4.131 m\_alloc.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_MALLOC_WRAPPER_H_
00002 #define M_MALLOC_WRAPPER_H_
00003
00004 void *m_alloc(size_t size);
00005 void *m_realloc(void *ptr, size_t size);
00006 char *m_strndup(const char *str, size_t n);
00007 void m_free(void *ptr);
00008
00009 void *m_int_lv_malloc(size_t size);
00010 void m_int_lv_free(void *ptr);
00011
00012 void print_memory_report();
00013
00014 #endif
```

## 4.132 m\_comms.c File Reference

```
#include "m_eng.h"
#include "m_comms.h"
#include "m_error_codes.h"
```

### Functions

- uint8\_t [crc\\_8](#) (const uint8\_t \*data, int len)
- int [et\\_message\\_data\\_length](#) (m\_message msg)
- int [et\\_message\\_default\\_retries](#) (uint16\_t type)
- int [encode\\_m\\_message](#) (uint8\_t \*buf, m\_message msg)
- m\_message [decode\\_m\\_message](#) (uint8\_t \*bytes, unsigned int len)
- m\_message [create\\_m\\_message\\_nodata](#) (uint16\_t type)
- m\_message [create\\_m\\_message](#) (uint16\_t type, const char \*fmt,...)
- int [valid\\_m\\_message\\_type](#) (uint8\_t type)
- const char \* [m\\_message\\_code\\_to\\_string](#) (uint16\_t code)
- const char \* [m\\_response\\_code\\_to\\_string](#) (uint16\_t code)
- int [te\\_message\\_data\\_length](#) (m\_response msg)
- int [encode\\_m\\_response](#) (uint8\_t \*buf, m\_response msg)
- m\_response [decode\\_m\\_response](#) (uint8\_t \*bytes, unsigned int len)
- int [valid\\_m\\_response\\_type](#) (uint8\_t type)
- m\_response [create\\_m\\_response](#) (uint16\_t type, const char \*fmt,...)
- m\_response [create\\_m\\_response\\_nodata](#) (uint16\_t type)
- m\_response [create\\_m\\_response\\_ok](#) ()
- m\_response [create\\_m\\_response\\_parameter\\_value](#) (uint16\_t pid, uint16\_t tid, uint16\_t ppid, float val)
- m\_response [create\\_m\\_response\\_error](#) (uint16\_t error\_code)
- m\_response [create\\_m\\_response\\_profile\\_id](#) (uint16\_t pid)
- m\_response [create\\_m\\_response\\_transformer\\_id](#) (uint16\_t pid, uint16\_t tid)

### 4.132.1 Function Documentation

#### 4.132.1.1 [crc\\_8\(\)](#)

```
uint8_t crc\_8 (
    const uint8_t * data,
    int len)
```

#### 4.132.1.2 create\_m\_message()

```
m_message create_m_message (
    uint16_t type,
    const char * fmt,
    ...)
```

#### 4.132.1.3 create\_m\_message\_nodata()

```
m_message create_m_message_nodata (
    uint16_t type)
```

#### 4.132.1.4 create\_m\_response()

```
m_response create_m_response (
    uint16_t type,
    const char * fmt,
    ...)
```

#### 4.132.1.5 create\_m\_response\_error()

```
m_response create_m_response_error (
    uint16_t error_code)
```

#### 4.132.1.6 create\_m\_response\_nodata()

```
m_response create_m_response_nodata (
    uint16_t type)
```

#### 4.132.1.7 create\_m\_response\_ok()

```
m_response create_m_response_ok ()
```

#### 4.132.1.8 create\_m\_response\_parameter\_value()

```
m_response create_m_response_parameter_value (
    uint16_t pid,
    uint16_t tid,
    uint16_t ppid,
    float val)
```

#### 4.132.1.9 create\_m\_response\_profile\_id()

```
m_response create_m_response_profile_id (
    uint16_t pid)
```

**4.132.1.10 create\_m\_response\_transformer\_id()**

```
m_response create_m_response_transformer_id (  
    uint16_t pid,  
    uint16_t tid)
```

**4.132.1.11 decode\_m\_message()**

```
m_message decode_m_message (  
    uint8_t * bytes,  
    unsigned int len)
```

**4.132.1.12 decode\_m\_response()**

```
m_response decode_m_response (  
    uint8_t * bytes,  
    unsigned int len)
```

**4.132.1.13 encode\_m\_message()**

```
int encode_m_message (  
    uint8_t * buf,  
    m_message msg)
```

**4.132.1.14 encode\_m\_response()**

```
int encode_m_response (  
    uint8_t * buf,  
    m_response msg)
```

**4.132.1.15 et\_message\_data\_length()**

```
int et_message_data_length (  
    m_message msg)
```

**4.132.1.16 et\_message\_default\_retries()**

```
int et_message_default_retries (  
    uint16_t type)
```

**4.132.1.17 m\_message\_code\_to\_string()**

```
const char * m_message_code_to_string (  
    uint16_t code)
```

#### 4.132.1.18 m\_response\_code\_to\_string()

```
const char * m_response_code_to_string (  
    uint16_t code)
```

#### 4.132.1.19 te\_message\_data\_length()

```
int te_message_data_length (  
    m_response msg)
```

#### 4.132.1.20 valid\_m\_message\_type()

```
int valid_m_message_type (  
    uint8_t type)
```

#### 4.132.1.21 valid\_m\_response\_type()

```
int valid_m_response_type (  
    uint8_t type)
```

## 4.133 m\_comms.h File Reference

### Data Structures

- struct [m\\_response](#)
- struct [m\\_message](#)

### Macros

- #define [TEENSY\\_ADDR](#) 0x08
- #define [M\\_MESSAGE\\_NO\\_MESSAGE](#) 255
- #define [M\\_MESSAGE\\_CRC\\_FAIL](#) 254
- #define [M\\_MESSAGE\\_INVALID](#) 0
- #define [M\\_MESSAGE\\_HI](#) 1
- #define [M\\_MESSAGE\\_RESET](#) 2
- #define [M\\_MESSAGE\\_REBOOT](#) 3
- #define [M\\_MESSAGE\\_CREATE\\_PROFILE](#) 4
- #define [M\\_MESSAGE\\_APPEND\\_TRANSFORMER](#) 5
- #define [M\\_MESSAGE\\_MOVE\\_TRANSFORMER](#) 6
- #define [M\\_MESSAGE\\_REMOVE\\_TRANSFORMER](#) 7
- #define [M\\_MESSAGE\\_GET\\_N\\_PROFILES](#) 8
- #define [M\\_MESSAGE\\_GET\\_N\\_TRANSFORMERS](#) 9
- #define [M\\_MESSAGE\\_GET\\_TRANSFORMER\\_ID](#) 10
- #define [M\\_MESSAGE\\_GET\\_TRANSFORMER\\_TYPE](#) 11
- #define [M\\_MESSAGE\\_GET\\_N\\_PARAMETERS](#) 12
- #define [M\\_MESSAGE\\_GET\\_PARAM\\_VALUE](#) 13
- #define [M\\_MESSAGE\\_SET\\_PARAM\\_VALUE](#) 14

- `#define M_MESSAGE_GET_N_SETTINGS 15`
- `#define M_MESSAGE_GET_SETTING_VALUE 16`
- `#define M_MESSAGE_SET_SETTING_VALUE 17`
- `#define M_MESSAGE_STRING_CONTINUE 18`
- `#define M_MESSAGE_STRING_CONTINUING 19`
- `#define M_MESSAGE_SWITCH_PROFILE 20`
- `#define M_MESSAGE_DELETE_PROFILE 21`
- `#define M_MESSAGE_REPEAT_MESSAGE 22`
- `#define M_MESSAGE_ENTER_TUNER_MODE 23`
- `#define M_MESSAGE_EXIT_TUNER_MODE 24`
- `#define M_MESSAGE_TYPE_MAX M_MESSAGE_EXIT_TUNER_MODE`
- `#define MESSAGE_LEN_VARIABLE -2`
- `#define M_MESSAGE_MAX_DATA_LEN 16`
- `#define M_MESSAGE_MAX_TRANSFER_LEN (M_RESPONSE_MAX_DATA_LEN + 2)`
- `#define M_RESPONSE_MAX_DATA_LEN 16`
- `#define M_RESPONSE_MAX_TRANSFER_LEN (M_RESPONSE_MAX_DATA_LEN + 2)`
- `#define M_RESPONSE_NO_MESSAGE 255`
- `#define M_RESPONSE_CRC_FAIL 254`
- `#define M_RESPONSE_INVALID 0`
- `#define M_RESPONSE_WAIT 1`
- `#define M_RESPONSE_HI 2`
- `#define M_RESPONSE_BAD_MESSAGE 3`
- `#define M_RESPONSE_BAD_REQUEST 4`
- `#define M_RESPONSE_TRY_AGAIN 5`
- `#define M_RESPONSE_OK 6`
- `#define M_RESPONSE_ERROR 7`
- `#define M_RESPONSE_PROFILE_ID 8`
- `#define M_RESPONSE_TRANSFORMER_ID 10`
- `#define M_RESPONSE_N_PROFILES 11`
- `#define M_RESPONSE_N_TRANSFORMERS 12`
- `#define M_RESPONSE_TRANSFORMER_TYPE 13`
- `#define M_RESPONSE_N_PARAMETERS 14`
- `#define M_RESPONSE_PARAM_VALUE 15`
- `#define M_RESPONSE_N_SETTINGS 16`
- `#define M_RESPONSE_SETTING_VALUE 17`
- `#define M_RESPONSE_STRING_CONTINUING 18`
- `#define M_RESPONSE_START_OVER 19`
- `#define M_RESPONSE_SWITCHING_PROFILE 20`
- `#define M_RESPONSE_DELETED_PROFILE 21`
- `#define M_RESPONSE_REPEAT_MESSAGE 22`
- `#define M_RESPONSE_TYPE_MAX M_RESPONSE_REPEAT_MESSAGE`

## Functions

- `m_message create_m_message_nodata (uint16_t type)`
- `m_message create_m_message (uint16_t type, const char *fmt,...)`
- `m_response create_m_response_nodata (uint16_t type)`
- `m_response create_m_response (uint16_t type, const char *fmt,...)`
- `m_response create_m_response_ok ()`
- `m_response create_m_response_error (uint16_t error_code)`
- `m_response create_m_response_profile_id (uint16_t pid)`
- `m_response create_m_response_transformer_id (uint16_t pid, uint16_t tid)`
- `m_response create_m_response_parameter_value (uint16_t pid, uint16_t tid, uint16_t ppid, float value)`
- `int et_message_data_length (m_message msg)`

- int [valid\\_m\\_message\\_type](#) (uint8\_t type)
- int [encode\\_m\\_message](#) (uint8\_t \*buf, [m\\_message](#) msg)
- [m\\_message](#) [decode\\_m\\_message](#) (uint8\_t \*bytes, unsigned int len)
- int [te\\_message\\_data\\_length](#) ([m\\_response](#) msg)
- int [valid\\_m\\_response\\_type](#) (uint8\_t type)
- int [encode\\_m\\_response](#) (uint8\_t \*buf, [m\\_response](#) msg)
- [m\\_response](#) [decode\\_m\\_response](#) (uint8\_t \*bytes, unsigned int len)
- const char \* [m\\_message\\_code\\_to\\_string](#) (uint16\_t code)
- const char \* [m\\_response\\_code\\_to\\_string](#) (uint16\_t code)

## 4.133.1 Macro Definition Documentation

### 4.133.1.1 M\_MESSAGE\_APPEND\_TRANSFORMER

```
#define M_MESSAGE_APPEND_TRANSFORMER 5
```

### 4.133.1.2 M\_MESSAGE\_CRC\_FAIL

```
#define M_MESSAGE_CRC_FAIL 254
```

### 4.133.1.3 M\_MESSAGE\_CREATE\_PROFILE

```
#define M_MESSAGE_CREATE_PROFILE 4
```

### 4.133.1.4 M\_MESSAGE\_DELETE\_PROFILE

```
#define M_MESSAGE_DELETE_PROFILE 21
```

### 4.133.1.5 M\_MESSAGE\_ENTER\_TUNER\_MODE

```
#define M_MESSAGE_ENTER_TUNER_MODE 23
```

### 4.133.1.6 M\_MESSAGE\_EXIT\_TUNER\_MODE

```
#define M_MESSAGE_EXIT_TUNER_MODE 24
```

### 4.133.1.7 M\_MESSAGE\_GET\_N\_PARAMETERS

```
#define M_MESSAGE_GET_N_PARAMETERS 12
```

### 4.133.1.8 M\_MESSAGE\_GET\_N\_PROFILES

```
#define M_MESSAGE_GET_N_PROFILES 8
```

**4.133.1.9 M\_MESSAGE\_GET\_N\_SETTINGS**

```
#define M_MESSAGE_GET_N_SETTINGS 15
```

**4.133.1.10 M\_MESSAGE\_GET\_N\_TRANSFORMERS**

```
#define M_MESSAGE_GET_N_TRANSFORMERS 9
```

**4.133.1.11 M\_MESSAGE\_GET\_PARAM\_VALUE**

```
#define M_MESSAGE_GET_PARAM_VALUE 13
```

**4.133.1.12 M\_MESSAGE\_GET\_SETTING\_VALUE**

```
#define M_MESSAGE_GET_SETTING_VALUE 16
```

**4.133.1.13 M\_MESSAGE\_GET\_TRANSFORMER\_ID**

```
#define M_MESSAGE_GET_TRANSFORMER_ID 10
```

**4.133.1.14 M\_MESSAGE\_GET\_TRANSFORMER\_TYPE**

```
#define M_MESSAGE_GET_TRANSFORMER_TYPE 11
```

**4.133.1.15 M\_MESSAGE\_HI**

```
#define M_MESSAGE_HI 1
```

**4.133.1.16 M\_MESSAGE\_INVALID**

```
#define M_MESSAGE_INVALID 0
```

**4.133.1.17 M\_MESSAGE\_MAX\_DATA\_LEN**

```
#define M_MESSAGE_MAX_DATA_LEN 16
```

**4.133.1.18 M\_MESSAGE\_MAX\_TRANSFER\_LEN**

```
#define M_MESSAGE_MAX_TRANSFER_LEN (M_RESPONSE_MAX_DATA_LEN + 2)
```



**4.133.1.19 M\_MESSAGE\_MOVE\_TRANSFORMER**

```
#define M_MESSAGE_MOVE_TRANSFORMER 6
```

**4.133.1.20 M\_MESSAGE\_NO\_MESSAGE**

```
#define M_MESSAGE_NO_MESSAGE 255
```

**4.133.1.21 M\_MESSAGE\_REBOOT**

```
#define M_MESSAGE_REBOOT 3
```

**4.133.1.22 M\_MESSAGE\_REMOVE\_TRANSFORMER**

```
#define M_MESSAGE_REMOVE_TRANSFORMER 7
```

**4.133.1.23 M\_MESSAGE\_REPEAT\_MESSAGE**

```
#define M_MESSAGE_REPEAT_MESSAGE 22
```

**4.133.1.24 M\_MESSAGE\_RESET**

```
#define M_MESSAGE_RESET 2
```

**4.133.1.25 M\_MESSAGE\_SET\_PARAM\_VALUE**

```
#define M_MESSAGE_SET_PARAM_VALUE 14
```

**4.133.1.26 M\_MESSAGE\_SET\_SETTING\_VALUE**

```
#define M_MESSAGE_SET_SETTING_VALUE 17
```

**4.133.1.27 M\_MESSAGE\_STRING\_CONTINUE**

```
#define M_MESSAGE_STRING_CONTINUE 18
```

**4.133.1.28 M\_MESSAGE\_STRING\_CONTINUEING**

```
#define M_MESSAGE_STRING_CONTINUEING 19
```

**4.133.1.29 M\_MESSAGE\_SWITCH\_PROFILE**

```
#define M_MESSAGE_SWITCH_PROFILE 20
```

**4.133.1.30 M\_MESSAGE\_TYPE\_MAX**

```
#define M_MESSAGE_TYPE_MAX M_MESSAGE_EXIT_TUNER_MODE
```

**4.133.1.31 M\_RESPONSE\_BAD\_MESSAGE**

```
#define M_RESPONSE_BAD_MESSAGE 3
```

**4.133.1.32 M\_RESPONSE\_BAD\_REQUEST**

```
#define M_RESPONSE_BAD_REQUEST 4
```

**4.133.1.33 M\_RESPONSE\_CRC\_FAIL**

```
#define M_RESPONSE_CRC_FAIL 254
```

**4.133.1.34 M\_RESPONSE\_DELETED\_PROFILE**

```
#define M_RESPONSE_DELETED_PROFILE 21
```

**4.133.1.35 M\_RESPONSE\_ERROR**

```
#define M_RESPONSE_ERROR 7
```

**4.133.1.36 M\_RESPONSE\_HI**

```
#define M_RESPONSE_HI 2
```

**4.133.1.37 M\_RESPONSE\_INVALID**

```
#define M_RESPONSE_INVALID 0
```

**4.133.1.38 M\_RESPONSE\_MAX\_DATA\_LEN**

```
#define M_RESPONSE_MAX_DATA_LEN 16
```

**4.133.1.39 M\_RESPONSE\_MAX\_TRANSFER\_LEN**

```
#define M_RESPONSE_MAX_TRANSFER_LEN (M_RESPONSE_MAX_DATA_LEN + 2)
```

**4.133.1.40 M\_RESPONSE\_N\_PARAMETERS**

```
#define M_RESPONSE_N_PARAMETERS 14
```

**4.133.1.41 M\_RESPONSE\_N\_PROFILES**

```
#define M_RESPONSE_N_PROFILES 11
```

**4.133.1.42 M\_RESPONSE\_N\_SETTINGS**

```
#define M_RESPONSE_N_SETTINGS 16
```

**4.133.1.43 M\_RESPONSE\_N\_TRANSFORMERS**

```
#define M_RESPONSE_N_TRANSFORMERS 12
```

**4.133.1.44 M\_RESPONSE\_NO\_MESSAGE**

```
#define M_RESPONSE_NO_MESSAGE 255
```

**4.133.1.45 M\_RESPONSE\_OK**

```
#define M_RESPONSE_OK 6
```

**4.133.1.46 M\_RESPONSE\_PARAM\_VALUE**

```
#define M_RESPONSE_PARAM_VALUE 15
```

**4.133.1.47 M\_RESPONSE\_PROFILE\_ID**

```
#define M_RESPONSE_PROFILE_ID 8
```

**4.133.1.48 M\_RESPONSE\_REPEAT\_MESSAGE**

```
#define M_RESPONSE_REPEAT_MESSAGE 22
```

**4.133.1.49 M\_RESPONSE\_SETTING\_VALUE**

```
#define M_RESPONSE_SETTING_VALUE 17
```

**4.133.1.50 M\_RESPONSE\_START\_OVER**

```
#define M_RESPONSE_START_OVER 19
```

**4.133.1.51 M\_RESPONSE\_STRING\_CONTINUING**

```
#define M_RESPONSE_STRING_CONTINUING 18
```

**4.133.1.52 M\_RESPONSE\_SWITCHING\_PROFILE**

```
#define M_RESPONSE_SWITCHING_PROFILE 20
```

**4.133.1.53 M\_RESPONSE\_TRANSFORMER\_ID**

```
#define M_RESPONSE_TRANSFORMER_ID 10
```

**4.133.1.54 M\_RESPONSE\_TRANSFORMER\_TYPE**

```
#define M_RESPONSE_TRANSFORMER_TYPE 13
```

**4.133.1.55 M\_RESPONSE\_TRY\_AGAIN**

```
#define M_RESPONSE_TRY_AGAIN 5
```

**4.133.1.56 M\_RESPONSE\_TYPE\_MAX**

```
#define M_RESPONSE_TYPE_MAX M\_RESPONSE\_REPEAT\_MESSAGE
```

**4.133.1.57 M\_RESPONSE\_WAIT**

```
#define M_RESPONSE_WAIT 1
```

**4.133.1.58 MESSAGE\_LEN\_VARIABLE**

```
#define MESSAGE_LEN_VARIABLE -2
```

#### 4.133.1.59 TEENSY\_ADDR

```
#define TEENSY_ADDR 0x08
```

### 4.133.2 Function Documentation

#### 4.133.2.1 create\_m\_message()

```
m_message create_m_message (  
    uint16_t type,  
    const char * fmt,  
    ...)
```

#### 4.133.2.2 create\_m\_message\_nodata()

```
m_message create_m_message_nodata (  
    uint16_t type)
```

#### 4.133.2.3 create\_m\_response()

```
m_response create_m_response (  
    uint16_t type,  
    const char * fmt,  
    ...)
```

#### 4.133.2.4 create\_m\_response\_error()

```
m_response create_m_response_error (  
    uint16_t error_code)
```

#### 4.133.2.5 create\_m\_response\_nodata()

```
m_response create_m_response_nodata(  
    uint16_t type)
```

#### 4.133.2.6 create\_m\_response\_ok()

```
m_response create_m_response_ok ()
```

#### 4.133.2.7 create\_m\_response\_parameter\_value()

```
m_response create_m_response_parameter_value (  
    uint16_t pid,  
    uint16_t tid,  
    uint16_t ppid,  
    float value)
```

#### 4.133.2.8 create\_m\_response\_profile\_id()

```
m_response create_m_response_profile_id (  
    uint16_t pid)
```

#### 4.133.2.9 create\_m\_response\_transformer\_id()

```
m_response create_m_response_transformer_id (  
    uint16_t pid,  
    uint16_t tid)
```

#### 4.133.2.10 decode\_m\_message()

```
m_message decode_m_message (  
    uint8_t * bytes,  
    unsigned int len)
```

#### 4.133.2.11 decode\_m\_response()

```
m_response decode_m_response (  
    uint8_t * bytes,  
    unsigned int len)
```

#### 4.133.2.12 encode\_m\_message()

```
int encode_m_message (  
    uint8_t * buf,  
    m_message msg)
```

#### 4.133.2.13 encode\_m\_response()

```
int encode_m_response (  
    uint8_t * buf,  
    m_response msg)
```

#### 4.133.2.14 et\_message\_data\_length()

```
int et_message_data_length (  
    m_message msg)
```

#### 4.133.2.15 m\_message\_code\_to\_string()

```
const char * m_message_code_to_string (  
    uint16_t code)
```

**4.133.2.16 m\_response\_code\_to\_string()**

```
const char * m_response_code_to_string (
    uint16_t code)
```

**4.133.2.17 te\_message\_data\_length()**

```
int te_message_data_length (
    m_response msg)
```

**4.133.2.18 valid\_m\_message\_type()**

```
int valid_m_message_type (
    uint8_t type)
```

**4.133.2.19 valid\_m\_response\_type()**

```
int valid_m_response_type (
    uint8_t type)
```

**4.134 m\_comms.h**

[Go to the documentation of this file.](#)

```
00001 #ifndef M_COMMS_H_
00002 #define M_COMMS_H_
00003
00004 #ifndef TEENSY_ADDR
00005 #define TEENSY_ADDR 0x08
00006 #endif
00007
00008 /* Messages from ESP32 to Teensy */
00009
00010 #define M_MESSAGE_NO_MESSAGE 255
00011 #define M_MESSAGE_CRC_FAIL 254
00012 #define M_MESSAGE_INVALID 0
00013 #define M_MESSAGE_HI 1
00014 #define M_MESSAGE_RESET 2
00015 #define M_MESSAGE_REBOOT 3
00016 #define M_MESSAGE_CREATE_PROFILE 4
00017 #define M_MESSAGE_APPEND_TRANSFORMER 5
00018 #define M_MESSAGE_MOVE_TRANSFORMER 6
00019 #define M_MESSAGE_REMOVE_TRANSFORMER 7
00020 #define M_MESSAGE_GET_N_PROFILES 8
00021 #define M_MESSAGE_GET_N_TRANSFORMERS 9
00022 #define M_MESSAGE_GET_TRANSFORMER_ID 10
00023 #define M_MESSAGE_GET_TRANSFORMER_TYPE 11
00024 #define M_MESSAGE_GET_N_PARAMETERS 12
00025 #define M_MESSAGE_GET_PARAM_VALUE 13
00026 #define M_MESSAGE_SET_PARAM_VALUE 14
00027 #define M_MESSAGE_GET_N_SETTINGS 15
00028 #define M_MESSAGE_GET_SETTING_VALUE 16
00029 #define M_MESSAGE_SET_SETTING_VALUE 17
00030 #define M_MESSAGE_STRING_CONTINUE 18
00031 #define M_MESSAGE_STRING_CONTINUING 19
00032 #define M_MESSAGE_SWITCH_PROFILE 20
00033 #define M_MESSAGE_DELETE_PROFILE 21
00034 #define M_MESSAGE_REPEAT_MESSAGE 22
00035 #define M_MESSAGE_ENTER_TUNER_MODE 23
00036 #define M_MESSAGE_EXIT_TUNER_MODE 24
00037
00038 #define M_MESSAGE_TYPE_MAX M_MESSAGE_EXIT_TUNER_MODE
00039
00040 #define MESSAGE_LEN_VARIABLE -2
```

```

00041 #define M_MESSAGE_MAX_DATA_LEN 16
00042 #define M_MESSAGE_MAX_TRANSFER_LEN (M_RESPONSE_MAX_DATA_LEN + 2)
00043
00044 #define M_RESPONSE_MAX_DATA_LEN 16
00045 #define M_RESPONSE_MAX_TRANSFER_LEN (M_RESPONSE_MAX_DATA_LEN + 2)
00046
00047 typedef struct
00048 {
00049     uint8_t type;
00050     uint8_t data[M_RESPONSE_MAX_DATA_LEN];
00051     void *extra;
00052 } m_response;
00053
00054 typedef struct m_message
00055 {
00056     uint8_t type;
00057     uint8_t data[M_RESPONSE_MAX_DATA_LEN];
00058     void (*callback)(struct m_message msg, m_response response);
00059     void *cb_arg;
00060     int retries;
00061 } m_message;
00062
00063 m_message create_m_message_nodata(uint16_t type);
00064 m_message create_m_message(uint16_t type, const char *fmt, ...);
00065
00066 /* Messages from Teensy to ESP32 */
00067
00068 #define M_RESPONSE_NO_MESSAGE 255
00069 #define M_RESPONSE_CRC_FAIL 254
00070 #define M_RESPONSE_INVALID 0
00071 #define M_RESPONSE_WAIT 1
00072 #define M_RESPONSE_HI 2
00073 #define M_RESPONSE_BAD_MESSAGE 3
00074 #define M_RESPONSE_BAD_REQUEST 4
00075 #define M_RESPONSE_TRY_AGAIN 5
00076 #define M_RESPONSE_OK 6
00077 #define M_RESPONSE_ERROR 7
00078 #define M_RESPONSE_PROFILE_ID 8
00079 #define M_RESPONSE_TRANSFORMER_ID 10
00080 #define M_RESPONSE_N_PROFILES 11
00081 #define M_RESPONSE_N_TRANSFORMERS 12
00082 #define M_RESPONSE_TRANSFORMER_TYPE 13
00083 #define M_RESPONSE_N_PARAMETERS 14
00084 #define M_RESPONSE_PARAM_VALUE 15
00085 #define M_RESPONSE_N_SETTINGS 16
00086 #define M_RESPONSE_SETTING_VALUE 17
00087 #define M_RESPONSE_STRING_CONTINUING 18
00088 #define M_RESPONSE_START_OVER 19
00089 #define M_RESPONSE_SWITCHING_PROFILE 20
00090 #define M_RESPONSE_DELETED_PROFILE 21
00091 #define M_RESPONSE_REPEAT_MESSAGE 22
00092
00093 #define M_RESPONSE_TYPE_MAX M_RESPONSE_REPEAT_MESSAGE
00094
00095 m_response create_m_response_nodata(uint16_t type);
00096 m_response create_m_response(uint16_t type, const char *fmt, ...);
00097 m_response create_m_response_ok();
00098
00099 m_response create_m_response_error(uint16_t error_code);
00100 m_response create_m_response_profile_id(uint16_t pid);
00101 m_response create_m_response_transformer_id(uint16_t pid, uint16_t tid);
00102 m_response create_m_response_parameter_value(uint16_t pid, uint16_t tid, uint16_t ppid, float value);
00103
00104 int et_message_data_length(m_message msg);
00105 int valid_m_message_type(uint8_t type);
00106
00107 int encode_m_message(uint8_t *buf, m_message msg);
00108 m_message decode_m_message(uint8_t *bytes, unsigned int len);
00109
00110 int te_message_data_length(m_response msg);
00111
00112 int valid_m_response_type(uint8_t type);
00113 int encode_m_response(uint8_t *buf, m_response msg);
00114 m_response decode_m_response(uint8_t *bytes, unsigned int len);
00115
00116 const char *m_message_code_to_string(uint16_t code);
00117 const char *m_response_code_to_string(uint16_t code);
00118
00119 #endif

```



## 4.135 m\_error\_codes.c File Reference

```
#include "m_error_codes.h"
```

### Functions

- const char \* [m\\_error\\_code\\_to\\_string](#) (int error\_code)

### 4.135.1 Function Documentation

#### 4.135.1.1 m\_error\_code\_to\_string()

```
const char * m_error_code_to_string (  
    int error_code)
```

## 4.136 m\_error\_codes.h File Reference

### Macros

- #define [NO\\_ERROR](#) 0
- #define [ERR\\_NULL\\_PTR](#) 1
- #define [ERR\\_BAD\\_ARGS](#) 2
- #define [ERR\\_SGTL5000\\_WRITE\\_FAIL](#) 3
- #define [ERR\\_ALLOC\\_FAIL](#) 5
- #define [ERR\\_PIPELINE\\_NULL](#) 4
- #define [ERR\\_PIPELINE\\_FULL](#) 6
- #define [ERR\\_POSITION\\_ILLEGAL](#) 7
- #define [ERR\\_POSITION\\_OCCUPIED](#) 8
- #define [ERR\\_TRANSFORMER\\_MALFORMED](#) 9
- #define [ERR\\_ARRAY\\_MALFORMED](#) 10
- #define [ERR\\_POT\\_LINK\\_MALFORMED](#) 11
- #define [ERR\\_SWITCH\\_LINK\\_MALFORMED](#) 12
- #define [ERR\\_MUTEX\\_UNAVAILABLE](#) 13
- #define [ERR\\_FIXED\\_ARRAY\\_FULL](#) 14
- #define [ERR\\_BUSTED\\_MSG](#) 15
- #define [ERR\\_BAD\\_REQUEST](#) 16
- #define [ERR\\_QUEUE\\_SEND\\_FAILED](#) 17
- #define [ERR\\_QUEUE\\_FULL](#) 18
- #define [ERR\\_LOOP\\_DETECTED](#) 19
- #define [ERR\\_NODE\\_PRIVATE](#) 20
- #define [ERR\\_PIPELINE\\_BUSTED](#) 21
- #define [ERR\\_INVALID\\_MESSAGE](#) 22
- #define [ERR\\_VALUE\\_OUT\\_OF\\_BOUNDS](#) 23
- #define [ERR\\_INVALID\\_PARAMETER\\_ID](#) 24
- #define [ERR\\_INVALID\\_SETTING\\_ID](#) 25
- #define [ERR\\_INVALID\\_TRANSFORMER\\_ID](#) 26
- #define [ERR\\_INVALID\\_PROFILE\\_ID](#) 27
- #define [ERR\\_INCONSISTENT\\_BACK\\_PIPELINE](#) 28

- `#define ERR_SPI_INIT_FAIL` 29
- `#define ERR_SD_INIT_FAIL` 30
- `#define ERR_SD_MOUNT_FAIL` 31
- `#define ERR_FOPEN_FAIL` 32
- `#define ERR_UNFINISHED_WRITE` 33
- `#define ERR_MANGLED_FILE` 34
- `#define ERR_I2C_FAIL` 35
- `#define ERR_NO_RESPONSE` 36
- `#define ERR_COMMS_FAIL` 37
- `#define ERR_UNKNOWN_ERR` 4999
- `#define ERR_UNIMPLEMENTED` 5000

## Functions

- `const char * m_error_code_to_string` (int error\_code)

## 4.136.1 Macro Definition Documentation

### 4.136.1.1 ERR\_ALLOC\_FAIL

```
#define ERR_ALLOC_FAIL 5
```

### 4.136.1.2 ERR\_ARRAY\_MALFORMED

```
#define ERR_ARRAY_MALFORMED 10
```

### 4.136.1.3 ERR\_BAD\_ARGS

```
#define ERR_BAD_ARGS 2
```

### 4.136.1.4 ERR\_BAD\_REQUEST

```
#define ERR_BAD_REQUEST 16
```

### 4.136.1.5 ERR\_BUSTED\_MSG

```
#define ERR_BUSTED_MSG 15
```

### 4.136.1.6 ERR\_COMMS\_FAIL

```
#define ERR_COMMS_FAIL 37
```

**4.136.1.7 ERR\_FIXED\_ARRAY\_FULL**

```
#define ERR_FIXED_ARRAY_FULL 14
```

**4.136.1.8 ERR\_FOPEN\_FAIL**

```
#define ERR_FOPEN_FAIL 32
```

**4.136.1.9 ERR\_I2C\_FAIL**

```
#define ERR_I2C_FAIL 35
```

**4.136.1.10 ERR\_INCONSISTENT\_BACK\_PIPELINE**

```
#define ERR_INCONSISTENT_BACK_PIPELINE 28
```

**4.136.1.11 ERR\_INVALID\_MESSAGE**

```
#define ERR_INVALID_MESSAGE 22
```

**4.136.1.12 ERR\_INVALID\_PARAMETER\_ID**

```
#define ERR_INVALID_PARAMETER_ID 24
```

**4.136.1.13 ERR\_INVALID\_PROFILE\_ID**

```
#define ERR_INVALID_PROFILE_ID 27
```

**4.136.1.14 ERR\_INVALID\_SETTING\_ID**

```
#define ERR_INVALID_SETTING_ID 25
```

**4.136.1.15 ERR\_INVALID\_TRANSFORMER\_ID**

```
#define ERR_INVALID_TRANSFORMER_ID 26
```

**4.136.1.16 ERR\_LOOP\_DETECTED**

```
#define ERR_LOOP_DETECTED 19
```

**4.136.1.17 ERR\_MANGLED\_FILE**

```
#define ERR_MANGLED_FILE 34
```

**4.136.1.18 ERR\_MUTEX\_UNAVAILABLE**

```
#define ERR_MUTEX_UNAVAILABLE 13
```

**4.136.1.19 ERR\_NO\_RESPONSE**

```
#define ERR_NO_RESPONSE 36
```

**4.136.1.20 ERR\_NODE\_PRIVATE**

```
#define ERR_NODE_PRIVATE 20
```

**4.136.1.21 ERR\_NULL\_PTR**

```
#define ERR_NULL_PTR 1
```

**4.136.1.22 ERR\_PIPELINE\_BUSTED**

```
#define ERR_PIPELINE_BUSTED 21
```

**4.136.1.23 ERR\_PIPELINE\_FULL**

```
#define ERR_PIPELINE_FULL 6
```

**4.136.1.24 ERR\_PIPELINE\_NULL**

```
#define ERR_PIPELINE_NULL 4
```

**4.136.1.25 ERR\_POSITION\_ILLEGAL**

```
#define ERR_POSITION_ILLEGAL 7
```

**4.136.1.26 ERR\_POSITION\_OCCUPIED**

```
#define ERR_POSITION_OCCUPIED 8
```

**4.136.1.27 ERR\_POT\_LINK\_MALFORMED**

```
#define ERR_POT_LINK_MALFORMED 11
```

**4.136.1.28 ERR\_QUEUE\_FULL**

```
#define ERR_QUEUE_FULL 18
```

**4.136.1.29 ERR\_QUEUE\_SEND\_FAILED**

```
#define ERR_QUEUE_SEND_FAILED 17
```

**4.136.1.30 ERR\_SD\_INIT\_FAIL**

```
#define ERR_SD_INIT_FAIL 30
```

**4.136.1.31 ERR\_SD\_MOUNT\_FAIL**

```
#define ERR_SD_MOUNT_FAIL 31
```

**4.136.1.32 ERR\_SGTL5000\_WRITE\_FAIL**

```
#define ERR_SGTL5000_WRITE_FAIL 3
```

**4.136.1.33 ERR\_SPI\_INIT\_FAIL**

```
#define ERR_SPI_INIT_FAIL 29
```

**4.136.1.34 ERR\_SWITCH\_LINK\_MALFORMED**

```
#define ERR_SWITCH_LINK_MALFORMED 12
```

**4.136.1.35 ERR\_TRANSFORMER\_MALFORMED**

```
#define ERR_TRANSFORMER_MALFORMED 9
```

**4.136.1.36 ERR\_UNFINISHED\_WRITE**

```
#define ERR_UNFINISHED_WRITE 33
```

**4.136.1.37 ERR\_UNIMPLEMENTED**

```
#define ERR_UNIMPLEMENTED 5000
```

**4.136.1.38 ERR\_UNKNOWN\_ERR**

```
#define ERR_UNKNOWN_ERR 4999
```

**4.136.1.39 ERR\_VALUE\_OUT\_OF\_BOUNDS**

```
#define ERR_VALUE_OUT_OF_BOUNDS 23
```

**4.136.1.40 NO\_ERROR**

```
#define NO_ERROR 0
```

**4.136.2 Function Documentation****4.136.2.1 m\_error\_code\_to\_string()**

```
const char * m_error_code_to_string (
    int error_code)
```

**4.137 m\_error\_codes.h**

[Go to the documentation of this file.](#)

```
00001 #ifndef M_ERROR_CODES_H_
00002 #define M_ERROR_CODES_H_
00003
00004 #define NO_ERROR 0
00005 #define ERR_NULL_PTR 1
00006 #define ERR_BAD_ARGS 2
00007 #define ERR_SGTL5000_WRITE_FAIL 3
00008
00009 #define ERR_ALLOC_FAIL 5
00010
00011 #define ERR_PIPELINE_NULL 4
00012 #define ERR_PIPELINE_FULL 6
00013 #define ERR_POSITION_ILLEGAL 7
00014 #define ERR_POSITION_OCCUPIED 8
00015
00016 #define ERR_TRANSFORMER_MALFORMED 9
00017 #define ERR_ARRAY_MALFORMED 10
00018
00019 #define ERR_POT_LINK_MALFORMED 11
00020 #define ERR_SWITCH_LINK_MALFORMED 12
00021
00022 #define ERR_MUTEX_UNAVAILABLE 13
00023
00024 #define ERR_FIXED_ARRAY_FULL 14
00025
00026 #define ERR_BUSTED_MSG 15
00027 #define ERR_BAD_REQUEST 16
00028
00029 #define ERR_QUEUE_SEND_FAILED 17
00030 #define ERR_QUEUE_FULL 18
00031 #define ERR_LOOP_DETECTED 19
00032
```

```

00033 #define ERR_NODE_PRIVATE                20
00034 #define ERR_PIPELINE_BUSTED              21
00035
00036 #define ERR_INVALID_MESSAGE              22
00037
00038 #define ERR_VALUE_OUT_OF_BOUNDS          23
00039
00040 #define ERR_INVALID_PARAMETER_ID          24
00041 #define ERR_INVALID_SETTING_ID           25
00042 #define ERR_INVALID_TRANSFORMER_ID       26
00043 #define ERR_INVALID_PROFILE_ID           27
00044
00045 #define ERR_INCONSISTENT_BACK_PIPELINE    28
00046
00047 #define ERR_SPI_INIT_FAIL                 29
00048 #define ERR_SD_INIT_FAIL                 30
00049 #define ERR_SD_MOUNT_FAIL                31
00050 #define ERR_FOPEN_FAIL                   32
00051 #define ERR_UNFINISHED_WRITE             33
00052 #define ERR_MANGLED_FILE                  34
00053
00054 #define ERR_I2C_FAIL                     35
00055 #define ERR_NO_RESPONSE                  36
00056 #define ERR_COMMS_FAIL                   37
00057
00058 #define ERR_UNKNOWN_ERR                   4999
00059 #define ERR_UNIMPLEMENTED                 5000
00060
00061 const char *m_error_code_to_string(int error_code);
00062
00063 #endif

```

## 4.138 m\_linked\_list.h File Reference

### Macros

- #define [LL\\_FREE](#) free
- #define [LL\\_MALLOC](#) malloc
- #define [DECLARE\\_LINKED\\_LIST\(X\)](#)
- #define [IMPLEMENT\\_LINKED\\_LIST\(X\)](#)
- #define [DECLARE\\_LINKED\\_PTR\\_LIST\(X\)](#)
- #define [IMPLEMENT\\_LINKED\\_PTR\\_LIST\(X\)](#)

### 4.138.1 Macro Definition Documentation

#### 4.138.1.1 DECLARE\_LINKED\_LIST

```

#define DECLARE_LINKED_LIST(
    X)

```

#### Value:

```

struct X##_ll;
\
typedef struct X##_ll {
    \
    X data;
    \
    struct X##_ll *next;
    \
} X##_ll;
\

X##_ll *X##_ll_new(X x);
\
void free_##X##_ll(X##_ll *list);
\
X##_ll *X##_ll_tail(X##_ll *list);
\

```

```

X##_ll *X##_ll_append(X##_ll *list, X x);
X##_ll \
X##_ll *X##_ll_append_return_tail(X##_ll **list, X x);
X##_ll \
X##_ll *X##_ll_remove_next(X##_ll *list);
void destructor_free_##X##_ll(X##_ll *list, void (*destructor)(X x));
void X##_ll_map(X##_ll *list, X (*fmap)(X x));
X##_ll \
X##_ll *X##_ll_cmp_search(X##_ll *list, int (*cmp_function)(X, X), X x);
X##_ll \
X##_ll *X##_ll_destructor_free_and_remove_matching(X##_ll *list, int (*cmp_function)(X, X), X x, void
(*destructor)(X));

```

#### 4.138.1.2 DECLARE\_LINKED\_PTR\_LIST

```

#define DECLARE_LINKED_PTR_LIST(
    X)

```

##### Value:

```

struct X##_pll;
typedef struct X##_pll {
    X *data;
    struct X##_pll *next;
} X##_pll;
\

X##_pll *X##_pll_new(X *value);
void free_##X##_pll(X##_pll *list);
X##_pll *X##_pll_tail(X##_pll *list);
X##_pll *X##_pll_append(X##_pll *list, X *value);
X##_pll *X##_pll_append_return_tail(X##_pll **list, X *x);
X##_pll *X##_pll_remove_next(X##_pll *list);
void X##_pll_free_all(X##_pll *list);
void destructor_free_##X##_pll(X##_pll *list, void (*destructor)(X *x));
X##_pll *X##_pll_cmp_search(X##_pll *list, int (*cmp_function)(const X*, const X*), const X *x);
X##_pll *X##_pll_destructor_free_and_remove_matching(X##_pll *list, int (*cmp_function)(X*, X*), X *x, void
(*destructor)(X*));

```

#### 4.138.1.3 IMPLEMENT\_LINKED\_LIST

```

#define IMPLEMENT_LINKED_LIST(
    X)

```

#### 4.138.1.4 IMPLEMENT\_LINKED\_PTR\_LIST

```

#define IMPLEMENT_LINKED_PTR_LIST(
    X)

```

#### 4.138.1.5 LL\_FREE

```

#define LL_FREE free

```



## 4.138.1.6 LL\_MALLOC

```
#define LL_MALLOC malloc
```

## 4.139 m\_linked\_list.h

[Go to the documentation of this file.](#)

```
00001 #ifndef LINKED_LIST_H
00002 #define LINKED_LIST_H
00003
00004 #ifndef LL_FREE
00005 #define LL_FREE free
00006 #endif
00007
00008 #ifndef LL_MALLOC
00009 #define LL_MALLOC malloc
00010 #endif
00011
00012 #define DECLARE_LINKED_LIST(X) struct X##_ll;
00013 \
00014 typedef struct X##_ll {
00015 \
00016     X data;
00017 \
00018     struct X##_ll *next;
00019 \
00020 } X##_ll;
00021 \
00022 X##_ll *X##_ll_new(X x);
00023 \
00024 void free_##X##_ll(X##_ll *list);
00025 \
00026 X##_ll *X##_ll_tail(X##_ll *list);
00027 \
00028 X##_ll *X##_ll_append(X##_ll *list, X x);
00029 \
00030 X##_ll *X##_ll_append_return_tail(X##_ll **list, X x);
00031 \
00032 X##_ll *X##_ll_remove_next(X##_ll *list);
00033 \
00034 void destructor_free_##X##_ll(X##_ll *list, void (*destructor)(X x));
00035 \
00036 void X##_ll_map(X##_ll *list, X (*fmap)(X x));
00037 \
00038 X##_ll *X##_ll_cmp_search(X##_ll *list, int (*cmp_function)(X, X), X x);
00039 \
00040 X##_ll *X##_ll_destructor_free_and_remove_matching(X##_ll *list, int (*cmp_function)(X, X), X x, void (*destructor)(X));
00041 \
00042 #define IMPLEMENT_LINKED_LIST(X)
00043 \
00044 X##_ll *X##_ll_new(X x)
00045 \
00046 {
00047 \
00048     X##_ll *result = (X##_ll*)LL_MALLOC(sizeof(X##_ll));
00049 \
00050     if (result == NULL)
00051 \
00052         return NULL;
00053 \
00054     result->data = x;
00055 \
00056     result->next = NULL;
00057 \
00058     return result;
00059 \
00060 }
```

```
00043 X##_ll *X##_ll_tail(X##_ll *list)
00044 {
00045     if (list == NULL)
00046         return NULL;
00047
00048     X##_ll *current = list;
00049
00050     while (current->next != NULL)
00051         current = current->next;
00052
00053     return current;
00054 }
00055
00056 void free_##X##_ll(X##_ll *list)
00057 {
00058     if (list == NULL)
00059         return;
00060
00061     X##_ll *current = list;
00062     X##_ll *next = NULL;
00063     while (current != NULL) {
00064         next = current->next;
00065         LL_FREE(current);
00066         current = next;
00067     }
00068 }
00069 void destructor_free_##X##_ll(X##_ll *list, void (*destructor)(X x))
00070 {
00071     if (list == NULL)
00072         return;
00073
00074     X##_ll *current = list;
00075     X##_ll *next = NULL;
00076
00077     while (current != NULL) {
00078         next = current->next;
00079         destructor(current->data);
00080         LL_FREE(current);
00081         current = next;
00082     }
00083 }
00084
00085 X##_ll *X##_ll_append(X##_ll *list, X x)
00086 {
```

```
00087 \
X##_ll *next = X##_ll_new(x);
00088 \
if (list == NULL) return next;
00089 \
00090 \
X##_ll *current = list;
00091 \
00092 \
while (current->next != NULL)
00093 \
current = current->next;
00094 \
00095 \
current->next = next;
00096 \
00097 \
return list;
00098 }
00099 \
00100 X##_ll *X##_ll_append_return_tail(X##_ll **list, X x)
00101 {
00102 \
if (list == NULL)
00103 \
return NULL;
00104 \
00105 \
X##_ll *next = X##_ll_new(x);
00106 \
if (*list == NULL) {
00107 \
*list = next;
00108 \
return next;
00109 \
}
00110 \
00111 \
X##_ll *current = *list;
00112 \
00113 \
while (current->next != NULL)
00114 \
current = current->next;
00115 \
00116 \
current->next = next;
00117 \
00118 \
return next;
00119 }
00120 \
00121 X##_ll *X##_ll_remove_next(X##_ll *list)
00122 {
00123 \
if (list == NULL)
00124 \
return NULL;
00125 \
00126 \
X##_ll *next = list->next;
00127 \
00128 \
if (next == NULL)
00129 \
return NULL;
```

```
00130 \
00131 \     list->next = list->next->next;
00132 \
00133 \     return next;
00134 }
00135 \
00136 void X##_ll_map(X##_ll *list, X (*fmap)(X x))
00137 {
00138 \     if (list == NULL)
00139 \         return;
00140 \
00141 \     list->data = fmap(list->data);
00142 \     X##_ll_map(list->next, fmap);
00143 }
00144 \
00145 X##_ll *X##_ll_cmp_search(X##_ll *list, int (*cmp_function)(X, X), X x)
00146 {
00147 \     if (list == NULL)
00148 \         return NULL;
00149 \
00150 \     X##_ll *current = list;
00151 \
00152 \     while (current) {
00153 \         if (cmp_function(current->data, x) == 0)
00154 \             return current;
00155 \
00156 \         current = current->next;
00157 \     }
00158 \
00159 \     return NULL;
00160 }
00161 \
00162 X##_ll *X##_ll_destructor_free_and_remove_matching(X##_ll *list, int (*cmp_function)(X, X), X x, void
(*destructor)(X)) \
00163 {
00164 \     if (list == NULL)
00165 \         return NULL;
00166 \
00167 \     X##_ll *current = list;
00168 \     X##_ll *prev = NULL;
00169 \     X##_ll *next = NULL;
00170 \
00171 \     while (current) {
00172 \         next = current->next;
00173 \         if (cmp_function(current->data, x) == 0) {
```

```

00174 \         if (current == list)
00175 \             list = next;
00176 \         destructor(current->data);
00177 \         LL_FREE(current);
00178 \         if (prev)
00179 \             prev->next = next;
00180 \     } else {
00181 \         prev = current;
00182 \     }
00183 \     current = next;
00184 \ }
00185 \
00186 \     return list;
00187 \ }
00188 \
00189 \ #define DECLARE_LINKED_PTR_LIST(X) struct X##_pll;
00190 \ typedef struct X##_pll {
00191 \     X *data;
00192 \     struct X##_pll *next;
00193 \ } X##_pll;
00194 \
00195 \ X##_pll *X##_pll_new(X *value);
00196 \ void free_##X##_pll(X##_pll *list);
00197 \ X##_pll *X##_pll_tail(X##_pll *list);
00198 \ X##_pll *X##_pll_append(X##_pll *list, X *value);
00199 \ X##_pll *X##_pll_append_return_tail(X##_pll **list, X *x);
00200 \ X##_pll *X##_pll_remove_next(X##_pll *list);
00201 \ void X##_pll_free_all(X##_pll *list);
00202 \ void destructor_free_##X##_pll(X##_pll *list, void (*destructor)(X *x));
00203 \ X##_pll *X##_pll_cmp_search(X##_pll *list, int (*cmp_function)(const X*, const X*), const X *x);
00204 \ X##_pll *X##_pll_destructor_free_and_remove_matching(X##_pll *list, int (*cmp_function)(X*, X*), X *x,
00205 \     void (*destructor)(X*));
00206 \ #define IMPLEMENT_LINKED_PTR_LIST(X)
00207 \
00208 \ X##_pll *X##_pll_new(X *value)
00209 \ {
00210 \     X##_pll *result = (X##_pll*)LL_MALLOC(sizeof(X##_pll));
00211 \
00212 \     if (result == NULL)
00213 \         return NULL;
00214 \
00215 \     result->data = value;
00216 \     result->next = NULL;
00217 \
00218 \     return result;

```

```
\
00219 }
\
00220
\
00221 X##_pll *X##_pll_tail(X##_pll *list)
\
00222 {
\
00223     if (list == NULL)
\
00224         return NULL;
\
00225
\
00226     X##_pll *current = list;
\
00227
\
00228     while (current->next != NULL)
\
00229         current = current->next;
\
00230
\
00231     return current;
\
00232 }
\
00233
\
00234 X##_pll *X##_pll_append(X##_pll *list, X *x)
\
00235 {
\
00236     X##_pll *next = X##_pll_new(x);
\
00237     if (list == NULL) return next;
\
00238
\
00239     X##_pll *current = list;
\
00240
\
00241     while (current->next != NULL)
\
00242         current = current->next;
\
00243
\
00244     current->next = next;
\
00245
\
00246     return list;
\
00247 }
\
00248
\
00249 X##_pll *X##_pll_append_return_tail(X##_pll **list, X *x)
\
00250 {
\
00251     if (list == NULL)
\
00252         return NULL;
\
00253
\
00254     X##_pll *next = X##_pll_new(x);
\
00255     if (*list == NULL) {
\
00256         *list = next;
\
00257         return next;
\
00258     }
\
00259
\
00260     X##_pll *current = *list;
\
00261
```

```
00262     while (current->next != NULL)
00263     \
00264     \
00265         current = current->next;
00266     \
00267     return next;
00268 }
00269 \
00270 \
00271 void free_##X##_pll(X##_pll *list)
00272 {
00273     \
00274     if (list == NULL)
00275     \
00276         return;
00277     X##_pll *current = list;
00278     X##_pll *next = NULL;
00279     while (current != NULL) {
00280         \
00281         next = current->next;
00282         \
00283         LL_FREE(current->data);
00284         \
00285         LL_FREE(current);
00286         \
00287         current = next;
00288     }
00289 }
00290 \
00291 void destructor_free_##X##_pll(X##_pll *list, void (*destructor)(X *x))
00292 {
00293     \
00294     if (list == NULL)
00295     \
00296         return;
00297     X##_pll *current = list;
00298     X##_pll *next = NULL;
00299     while (current != NULL) {
00300         \
00301         next = current->next;
00302         \
00303         destructor(current->data);
00304         \
00305         LL_FREE(current);
00306         \
00307         current = next;
00308     }
00309 }
00310 \
00311 X##_pll *X##_pll_remove_next(X##_pll *list)
00312 {
00313     \
00314     if (list == NULL)
00315     \
00316         return NULL;
00317     \
00318     X##_pll *next = list->next;
```

```
00306 \
00307 \   if (next == NULL)
00308 \       return NULL;
00309 \
00310 \   list->next = next->next;
00311 \
00312 \   return next;
00313 }
00314 \
00315 void X##_pll_map(X##_pll *list, X *(*fmap)(X *x))
00316 {
00317 \   if (list == NULL)
00318 \       return;
00319 \
00320 \   list->data = fmap(list->data);
00321 \   X##_pll_map(list->next, fmap);
00322 }
00323 \
00324 X##_pll *X##_pll_cmp_search(X##_pll *list, int (*cmp_function)(const X*, const X*), const X *x)
00325 {
00326 \   if (list == NULL)
00327 \       return NULL;
00328 \
00329 \   X##_pll *current = list;
00330 \
00331 \   while (current) {
00332 \       if (cmp_function(current->data, x) == 0)
00333 \           return current;
00334 \
00335 \       current = current->next;
00336 \   }
00337 \
00338 \   return NULL;
00339 }
00340 \
00341 \
00342 X##_pll *X##_pll_destructor_free_and_remove_matching(X##_pll *list, int (*cmp_function)(X*, X*), X *x,
00343 void (*destructor)(X*))\
00344 {
00345 \   if (list == NULL)
00346 \       return NULL;
00347 \
00348 \   X##_pll *current = list;
00349 \   X##_pll *next = NULL;
```



```
00349     X##_pll *prev = NULL;
00350 \
00351     while (current) {
00352 \         next = current->next;
00353 \         if (cmp_function(current->data, x) == 0) {
00354 \             if (current == list)
00355 \                 list = next;
00356 \             destructor(current->data);
00357 \             LL_FREE(current);
00358 \             if (prev)
00359 \                 prev->next = next;
00360 \         } else {
00361 \             prev = current;
00362 \         }
00363 \         current = next;
00364 \     }
00365 \
00366     return list;
00367 }
00368
00369 #endif
```

## 4.140 m\_parameter.h File Reference

### Data Structures

- struct [m\\_parameter\\_id](#)
- struct [m\\_parameter](#)
- struct [m\\_setting\\_id](#)
- struct [m\\_setting](#)

### Macros

- #define [PARAMETER\\_SCALE\\_LINEAR](#) 0
- #define [PARAMETER\\_SCALE\\_LOGARITHMIC](#) 1
- #define [TRANSFORMER\\_SETTING\\_ENUM](#) 0
- #define [TRANSFORMER\\_SETTING\\_BOOL](#) 1
- #define [TRANSFORMER\\_SETTING\\_INT](#) 2
- #define [TRANSFORMER\\_SETTING\\_PAGE\\_SETTINGS](#) 0
- #define [TRANSFORMER\\_SETTING\\_PAGE\\_MAIN](#) 1

### Functions

- [DECLARE\\_LINKED\\_PTR\\_LIST](#) ([m\\_parameter](#))
- [DECLARE\\_LINKED\\_PTR\\_LIST](#) ([m\\_setting](#))

## 4.140.1 Macro Definition Documentation

### 4.140.1.1 PARAMETER\_SCALE\_LINEAR

```
#define PARAMETER_SCALE_LINEAR 0
```

### 4.140.1.2 PARAMETER\_SCALE\_LOGARITHMIC

```
#define PARAMETER_SCALE_LOGARITHMIC 1
```

### 4.140.1.3 TRANSFORMER\_SETTING\_BOOL

```
#define TRANSFORMER_SETTING_BOOL 1
```

### 4.140.1.4 TRANSFORMER\_SETTING\_ENUM

```
#define TRANSFORMER_SETTING_ENUM 0
```

### 4.140.1.5 TRANSFORMER\_SETTING\_INT

```
#define TRANSFORMER_SETTING_INT 2
```

### 4.140.1.6 TRANSFORMER\_SETTING\_PAGE\_MAIN

```
#define TRANSFORMER_SETTING_PAGE_MAIN 1
```

### 4.140.1.7 TRANSFORMER\_SETTING\_PAGE\_SETTINGS

```
#define TRANSFORMER_SETTING_PAGE_SETTINGS 0
```

## 4.140.2 Function Documentation

### 4.140.2.1 DECLARE\_LINKED\_PTR\_LIST() [1/2]

```
DECLARE_LINKED_PTR_LIST (
    m_parameter )
```

### 4.140.2.2 DECLARE\_LINKED\_PTR\_LIST() [2/2]

```
DECLARE_LINKED_PTR_LIST (
    m_setting )
```

## 4.141 m\_parameter.h

[Go to the documentation of this file.](#)

```

00001 #ifndef M_PARAMETER_H_
00002 #define M_PARAMETER_H_
00003
00004 #define PARAMETER_SCALE_LINEAR 0
00005 #define PARAMETER_SCALE_LOGARITHMIC 1
00006
00007 typedef struct m_parameter_id
00008 {
00009     uint16_t profile_id;
00010     uint16_t transformer_id;
00011     uint16_t parameter_id;
00012 } m_parameter_id;
00013
00014 typedef struct m_parameter
00015 {
00016     float value;
00017     float min;
00018     float max;
00019
00020     int scale;
00021
00022     int updated;
00023     float old_value;
00024     float new_value;
00025
00026     #if defined(M_ENGINE)
00027
00028     float max_jump;
00029
00030     #elif defined(M_INTERFACE)
00031
00032     m_parameter_id id;
00033
00034     float factor;
00035
00036     int widget_type;
00037     const char *name;
00038     const char *units;
00039
00040     int group;
00041
00042     #endif
00043 } m_parameter;
00044
00045 #if defined(M_INTERFACE)
00046
00047 typedef struct m_setting_option
00048 {
00049     uint16_t value;
00050     const char *name;
00051 } m_setting_option;
00052
00053 #endif
00054
00055 typedef struct m_setting_id
00056 {
00057     uint16_t profile_id;
00058     uint16_t transformer_id;
00059     uint16_t setting_id;
00060 } m_setting_id;
00061
00062 #define TRANSFORMER_SETTING_ENUM 0
00063 #define TRANSFORMER_SETTING_BOOL 1
00064 #define TRANSFORMER_SETTING_INT 2
00065
00066 #define TRANSFORMER_SETTING_PAGE_SETTINGS 0
00067 #define TRANSFORMER_SETTING_PAGE_MAIN 1
00068
00069 typedef struct m_setting
00070 {
00071     int16_t value;
00072
00073     int updated;
00074     int16_t old_value;
00075     int16_t new_value;
00076
00077     #if defined(M_ENGINE)
00078
00079     #elif defined(M_INTERFACE)
00080
00081     int type;

```

```

00083     int page;
00084
00085     m_setting_id id;
00086
00087     uint16_t min;
00088     uint16_t max;
00089
00090     int n_options;
00091     m_setting_option *options;
00092
00093     int widget_type;
00094     const char *name;
00095     const char *units;
00096
00097     int group;
00098
00099     #endif
00100 } m_setting;
00101
00102 DECLARE_LINKED_PTR_LIST(m_parameter);
00103 DECLARE_LINKED_PTR_LIST(m_setting);
00104
00105 #endif

```

## 4.142 m\_pipeline.h File Reference

### Data Structures

- struct [m\\_pipeline](#)

## 4.143 m\_pipeline.h

[Go to the documentation of this file.](#)

```

00001 #ifndef M_PIPELINE_H_
00002 #define M_PIPELINE_H_
00003
00004 typedef struct
00005 {
00006     #if defined(M_ENGINE)
00007
00008     int n_transformers;
00009     int transformer_array_length;
00010     m_transformer **transformers;
00011
00012     uint16_t next_id;
00013
00014     int transition_policy;
00015
00016     #elif defined(M_INTERFACE)
00017
00018     m_transformer_pll *transformers;
00019
00020     #endif
00021 } m_pipeline;
00022
00023 #endif

```

## 4.144 m\_profile.h File Reference

### Data Structures

- struct [m\\_profile](#)

## 4.145 m\_profile.h

[Go to the documentation of this file.](#)

```

00001 #ifndef M_PROFILE_H_
00002 #define M_PROFILE_H_
00003
00004 #ifdef M_INTERFACE
00005     struct m_int_glide_button_pll;
00006     struct m_int_menu_item_pll;
00007 #endif
00008
00009 #ifdef M_ENGINE
00010     struct m_pipeline_mod_ll;
00011 #endif
00012
00013 typedef struct m_profile
00014 {
00015     int active;
00016
00017     #if defined(M_ENGINE)
00018     m_pipeline *front_pipeline;
00019     m_pipeline *back_pipeline;
00020
00021     int pipelines_swapping;
00022     int pipeline_swap_progress;
00023     int pipeline_swap_samples;
00024     int pipeline_swap_type;
00025     int back_pipeline_warmed_up;
00026
00027     struct m_pipeline_mod_ll *jobs;
00028     struct m_pipeline_mod_ll *ujobs;
00029
00030     int transition_policy;
00031
00032     float *prev_block;
00033
00034     m_transformer output_amp;
00035
00036     #elif defined(M_INTERFACE)
00037
00038     char *name;
00039     uint16_t id;
00040     m_pipeline pipeline;
00041
00042     struct m_ui_page *view_page;
00043
00044     struct m_int_menu_item_pll *listings;
00045     struct m_int_glide_button_pll *gbs;
00046
00047     m_parameter volume;
00048
00049     char *fname;
00050
00051     int default_profile;
00052     int unsaved_changes;
00053
00054     #endif
00055 } m_profile;
00056
00057 #endif

```

## 4.146 m\_status.h File Reference

### Macros

- `#define M_STATUS_OK 0`
- `#define M_STATUS_BOOTING 0b0001`
- `#define M_STATUS_FRESH_BOOT 0b0010`

### 4.146.1 Macro Definition Documentation

#### 4.146.1.1 M\_STATUS\_BOOTING

```
#define M_STATUS_BOOTING 0b0001
```

#### 4.146.1.2 M\_STATUS\_FRESH\_BOOT

```
#define M_STATUS_FRESH_BOOT 0b0010
```

#### 4.146.1.3 M\_STATUS\_OK

```
#define M_STATUS_OK 0
```

### 4.147 m\_status.h

[Go to the documentation of this file.](#)

```
00001 #ifndef M_STATUS_H_
00002 #define M_STATUS_H_
00003
00004 #define M_STATUS_OK 0
00005 #define M_STATUS_BOOTING 0b0001
00006 #define M_STATUS_FRESH_BOOT 0b0010
00007
00008 #endif
```

## 4.148 m\_transformer.h File Reference

### Data Structures

- struct [m\\_transformer](#)

### Functions

- [DECLARE\\_LINKED\\_PTR\\_LIST](#) ([m\\_transformer](#))

### 4.148.1 Function Documentation

#### 4.148.1.1 DECLARE\_LINKED\_PTR\_LIST()

```
DECLARE_LINKED_PTR_LIST (
    m\_transformer )
```

## 4.149 m\_transformer.h

[Go to the documentation of this file.](#)

```

00001 #ifndef M_TRANSFORMER_H_
00002 #define M_TRANSFORMER_H_
00003
00004 #ifdef M_INTERFACE
00005     struct m_ui_page;
00006 #endif
00007
00008 #ifdef M_ENGINE
00009     #include "m_eng_linkowitz_riley.h"
00010 #endif
00011
00012 typedef struct m_transformer
00013 {
00014     uint16_t type;
00015     uint16_t id;
00016
00017     m_parameter wet_mix;
00018
00019     m_setting band_mode;
00020     m_parameter band_lp_cutoff;
00021     m_parameter band_hp_cutoff;
00022     m_parameter band_center;
00023     m_parameter band_width;
00024
00025     #if defined(M_ENGINE)
00026
00027     int n_settings;
00028     int setting_array_size;
00029
00030     m_setting **settings;
00031
00032     int n_parameters;
00033     int parameter_array_size;
00034     m_parameter **parameters;
00035
00036     struct m_lr_low_pass_filter_str input_lpf;
00037     struct m_lr_low_pass_filter_str input_lpf_complement;
00038     struct m_lr_high_pass_filter_str input_hpf;
00039     struct m_lr_high_pass_filter_str input_hpf_band;
00040     struct m_lr_high_pass_filter_str input_hpf_complement;
00041
00042     void *data_struct;
00043     int (*compute_transformer)(void *data_struct, float *dest, float *src, int n_samples);
00044     int (*compute_transformer_nl)(void *data_struct, float **dest, float **src, int n_samples);
00045
00046     int (*reconfigure)(void *data_struct);
00047     int (*clone_struct)(void *dest, void *src);
00048     int (*free_struct)(void *data_struct);
00049
00050     size_t struct_size;
00051     int transition_policy;
00052
00053     #elif defined(M_INTERFACE)
00054
00055     int position;
00056
00057     m_parameter_pll *parameters;
00058     m_setting_pll *settings;
00059
00060     struct m_profile *profile;
00061     struct m_ui_page *view_page;
00062
00063     #endif
00064 } m_transformer;
00065
00066 DECLARE_LINKED_PTR_LIST(m_transformer);
00067
00068 #endif

```

## 4.150 m\_transformer\_enum.c File Reference

```

#include <stdint.h>
#include "m_transformer_enum.h"

```

## Functions

- const char \* [transformer\\_type\\_to\\_string](#) (uint16\_t type)
- int [transformer\\_type\\_valid](#) (uint16\_t type)

### 4.150.1 Function Documentation

#### 4.150.1.1 [transformer\\_type\\_to\\_string\(\)](#)

```
const char * transformer_type_to_string (  
    uint16_t type)
```

#### 4.150.1.2 [transformer\\_type\\_valid\(\)](#)

```
int transformer_type_valid (  
    uint16_t type)
```

## 4.151 [m\\_transformer\\_enum.h](#) File Reference

### Macros

- #define [TRANSFORMER\\_MODE\\_FULL\\_SPECTRUM](#) 0
- #define [TRANSFORMER\\_MODE\\_UPPER\\_SPECTRUM](#) 1
- #define [TRANSFORMER\\_MODE\\_LOWER\\_SPECTRUM](#) 2
- #define [TRANSFORMER\\_MODE\\_BAND](#) 3
- #define [TRANSFORMER\\_WET\\_MIX\\_PID](#) 0xFFFF
- #define [TRANSFORMER\\_BAND\\_LP\\_CUTOFF\\_PID](#) 0xFFFE
- #define [TRANSFORMER\\_BAND\\_HP\\_CUTOFF\\_PID](#) 0xFFFD
- #define [TRANSFORMER\\_BAND\\_MODE\\_SID](#) 0xFFFF
- #define [TRANSFORMER\\_3\\_BAND\\_EQ](#) 0
- #define [TRANSFORMER\\_AMPLIFIER](#) 1
- #define [TRANSFORMER\\_BAND\\_PASS\\_FILTER](#) 2
- #define [TRANSFORMER\\_COMPRESSOR](#) 3
- #define [TRANSFORMER\\_DELAY](#) 4
- #define [TRANSFORMER\\_DIRTY\\_OCTAVE](#) 5
- #define [TRANSFORMER\\_DISTORTION](#) 6
- #define [TRANSFORMER\\_ENVELOPE](#) 7
- #define [TRANSFORMER\\_FLANGER](#) 8
- #define [TRANSFORMER\\_HIGH\\_PASS\\_FILTER](#) 9
- #define [TRANSFORMER\\_LOW\\_END\\_COMPRESSOR](#) 10
- #define [TRANSFORMER\\_LOW\\_PASS\\_FILTER](#) 11
- #define [TRANSFORMER\\_NOISE\\_SUPPRESSOR](#) 12
- #define [TRANSFORMER\\_PERCUSSIFIER](#) 13
- #define [TRANSFORMER\\_WARBLER](#) 14
- #define [DISTORTION\\_SOFT\\_FOLD](#) 0
- #define [DISTORTION\\_ARCTAN](#) 1
- #define [DISTORTION\\_TANH](#) 2
- #define [DISTORTION\\_CLIP](#) 3



## Enumerations

- enum [biquad\\_type](#) {  
    [low\\_pass](#) = 0 , [high\\_pass](#) = 1 , [band\\_pass](#) = 2 , [notch](#) = 3 ,  
    [peaking\\_band\\_eq](#) = 4 , [low\\_shelf](#) = 5 , [high\\_shelf](#) = 6 }

## Functions

- const char \* [transformer\\_type\\_to\\_string](#) (uint16\_t type)
- int [transformer\\_type\\_valid](#) (uint16\_t type)

## 4.151.1 Macro Definition Documentation

### 4.151.1.1 DISTORTION\_ARCTAN

```
#define DISTORTION_ARCTAN 1
```

### 4.151.1.2 DISTORTION\_CLIP

```
#define DISTORTION_CLIP 3
```

### 4.151.1.3 DISTORTION\_SOFT\_FOLD

```
#define DISTORTION_SOFT_FOLD 0
```

### 4.151.1.4 DISTORTION\_TANH

```
#define DISTORTION_TANH 2
```

### 4.151.1.5 TRANSFORMER\_3\_BAND\_EQ

```
#define TRANSFORMER_3_BAND_EQ 0
```

### 4.151.1.6 TRANSFORMER\_AMPLIFIER

```
#define TRANSFORMER_AMPLIFIER 1
```

### 4.151.1.7 TRANSFORMER\_BAND\_HP\_CUTOFF\_PID

```
#define TRANSFORMER_BAND_HP_CUTOFF_PID 0xFFFFD
```

**4.151.1.8 TRANSFORMER\_BAND\_LP\_CUTOFF\_PID**

```
#define TRANSFORMER_BAND_LP_CUTOFF_PID 0xFFFE
```

**4.151.1.9 TRANSFORMER\_BAND\_MODE\_SID**

```
#define TRANSFORMER_BAND_MODE_SID 0xFFFF
```

**4.151.1.10 TRANSFORMER\_BAND\_PASS\_FILTER**

```
#define TRANSFORMER_BAND_PASS_FILTER 2
```

**4.151.1.11 TRANSFORMER\_COMPRESSOR**

```
#define TRANSFORMER_COMPRESSOR 3
```

**4.151.1.12 TRANSFORMER\_DELAY**

```
#define TRANSFORMER_DELAY 4
```

**4.151.1.13 TRANSFORMER\_DIRTY\_OCTAVE**

```
#define TRANSFORMER_DIRTY_OCTAVE 5
```

**4.151.1.14 TRANSFORMER\_DISTORTION**

```
#define TRANSFORMER_DISTORTION 6
```

**4.151.1.15 TRANSFORMER\_ENVELOPE**

```
#define TRANSFORMER_ENVELOPE 7
```

**4.151.1.16 TRANSFORMER\_FLANGER**

```
#define TRANSFORMER_FLANGER 8
```

**4.151.1.17 TRANSFORMER\_HIGH\_PASS\_FILTER**

```
#define TRANSFORMER_HIGH_PASS_FILTER 9
```

#### 4.151.1.18 TRANSFORMER\_LOW\_END\_COMPRESSOR

```
#define TRANSFORMER_LOW_END_COMPRESSOR 10
```

#### 4.151.1.19 TRANSFORMER\_LOW\_PASS\_FILTER

```
#define TRANSFORMER_LOW_PASS_FILTER 11
```

#### 4.151.1.20 TRANSFORMER\_MODE\_BAND

```
#define TRANSFORMER_MODE_BAND 3
```

#### 4.151.1.21 TRANSFORMER\_MODE\_FULL\_SPECTRUM

```
#define TRANSFORMER_MODE_FULL_SPECTRUM 0
```

#### 4.151.1.22 TRANSFORMER\_MODE\_LOWER\_SPECTRUM

```
#define TRANSFORMER_MODE_LOWER_SPECTRUM 2
```

#### 4.151.1.23 TRANSFORMER\_MODE\_UPPER\_SPECTRUM

```
#define TRANSFORMER_MODE_UPPER_SPECTRUM 1
```

#### 4.151.1.24 TRANSFORMER\_NOISE\_SUPPRESSOR

```
#define TRANSFORMER_NOISE_SUPPRESSOR 12
```

#### 4.151.1.25 TRANSFORMER\_PERCUSSIFIER

```
#define TRANSFORMER_PERCUSSIFIER 13
```

#### 4.151.1.26 TRANSFORMER\_WARBLER

```
#define TRANSFORMER_WARBLER 14
```

#### 4.151.1.27 TRANSFORMER\_WET\_MIX\_PID

```
#define TRANSFORMER_WET_MIX_PID 0xFFFF
```

### 4.151.2 Enumeration Type Documentation

#### 4.151.2.1 biquad\_type

```
enum biquad\_type
```

## Enumerator

low_pass	
high_pass	
band_pass	
notch	
peaking_band_eq	
low_shelf	
high_shelf	

### 4.151.3 Function Documentation

#### 4.151.3.1 transformer\_type\_to\_string()

```
const char * transformer_type_to_string (
    uint16_t type)
```

#### 4.151.3.2 transformer\_type\_valid()

```
int transformer_type_valid (
    uint16_t type)
```

## 4.152 m\_transformer\_enum.h

[Go to the documentation of this file.](#)

```
00001 // Code generated from config/transformer/*.yaml by codegen.py
00002 #ifndef M_TRANSFORMER_ENUM_H_
00003 #define M_TRANSFORMER_ENUM_H_
00004
00005 #define TRANSFORMER_MODE_FULL_SPECTRUM 0
00006 #define TRANSFORMER_MODE_UPPER_SPECTRUM 1
00007 #define TRANSFORMER_MODE_LOWER_SPECTRUM 2
00008 #define TRANSFORMER_MODE_BAND 3
00009
00010 #define TRANSFORMER_WET_MIX_PID 0xFFFF
00011
00012 #define TRANSFORMER_BAND_LP_CUTOFF_PID 0xFFFE
00013 #define TRANSFORMER_BAND_HP_CUTOFF_PID 0xFFFD
00014
00015 #define TRANSFORMER_BAND_MODE_SID 0xFFFF
00016
00017 #define TRANSFORMER_3_BAND_EQ 0
00018 #define TRANSFORMER_AMPLIFIER 1
00019 #define TRANSFORMER_BAND_PASS_FILTER 2
00020 #define TRANSFORMER_COMPRESSOR 3
00021 #define TRANSFORMER_DELAY 4
00022 #define TRANSFORMER_DIRTY_OCTAVE 5
00023 #define TRANSFORMER_DISTORTION 6
00024 #define TRANSFORMER_ENVELOPE 7
00025 #define TRANSFORMER_FLANGER 8
00026 #define TRANSFORMER_HIGH_PASS_FILTER 9
00027 #define TRANSFORMER_LOW_END_COMPRESSOR 10
00028 #define TRANSFORMER_LOW_PASS_FILTER 11
00029 #define TRANSFORMER_NOISE_SUPPRESSOR 12
00030 #define TRANSFORMER_PERCUSSIONIFIER 13
00031 #define TRANSFORMER_WARBLER 14
00032
00033 typedef enum
00034 {
00035     low_pass = 0,
00036     high_pass = 1,
```

```
00037     band_pass      = 2,
00038     notch          = 3,
00039     peaking_band_eq = 4,
00040     low_shelf       = 5,
00041     high_shelf      = 6
00042 } biquad_type;
00043
00044 #define DISTORTION_SOFT_FOLD    0
00045 #define DISTORTION_ARCTAN      1
00046 #define DISTORTION_TANH        2
00047 #define DISTORTION_CLIP        3
00048
00049 const char *transformer_type_to_string(uint16_t type);
00050 int transformer_type_valid(uint16_t type);
00051
00052 #endif
```



# Index

- `__attribute__`
  - `m_eng_i2s_dma.cpp`, 178
- `a0`
  - `m_eng_band_pass_filter_str`, 11
  - `m_eng_biquad_str`, 13
  - `m_eng_high_pass_filter_str`, 24
  - `m_eng_low_pass_filter_str`, 28
  - `m_lr_high_pass_filter_str`, 40
  - `m_lr_low_pass_filter_str`, 41
- `a1`
  - `m_eng_band_pass_filter_str`, 11
  - `m_eng_biquad_str`, 13
  - `m_eng_high_pass_filter_str`, 24
  - `m_eng_low_pass_filter_str`, 28
  - `m_lr_high_pass_filter_str`, 40
  - `m_lr_low_pass_filter_str`, 41
- `a2`
  - `m_eng_band_pass_filter_str`, 11
  - `m_eng_biquad_str`, 13
  - `m_eng_high_pass_filter_str`, 24
  - `m_eng_low_pass_filter_str`, 28
  - `m_lr_high_pass_filter_str`, 40
  - `m_lr_low_pass_filter_str`, 42
- `a3`
  - `m_eng_band_pass_filter_str`, 11
  - `m_eng_biquad_str`, 13
  - `m_eng_high_pass_filter_str`, 25
  - `m_eng_low_pass_filter_str`, 28
  - `m_lr_high_pass_filter_str`, 40
  - `m_lr_low_pass_filter_str`, 42
- `a4`
  - `m_eng_band_pass_filter_str`, 11
  - `m_eng_biquad_str`, 13
  - `m_eng_high_pass_filter_str`, 25
  - `m_eng_low_pass_filter_str`, 28
  - `m_lr_high_pass_filter_str`, 40
  - `m_lr_low_pass_filter_str`, 42
- `active`
  - `m_eng_profile`, 34
  - `m_profile`, 47
- `active_profile`
  - `m_eng_context`, 16
- `ADAPTIVE_WAVESHAPER_ENVELOPE_ATTACK`
  - `m_eng_adaptive_waveshaper.h`, 58
- `ADAPTIVE_WAVESHAPER_ENVELOPE_RELEASE`
  - `m_eng_adaptive_waveshaper.h`, 58
- `allocate_buffer`
  - `m_eng_mempool.c`, 187
  - `m_eng_mempool.h`, 106
- `ALLOW_PRINTLINES`
  - `m_eng.h`, 53
- `alpha`
  - `m_eng_compressor_str`, 15
  - `m_eng_envelope_str`, 21
  - `m_eng_warbler_str`, 37
- `alpha_long`
  - `m_eng_percussifier_str`, 31
- `alpha_short`
  - `m_eng_percussifier_str`, 31
- `apply_pipeline_mod`
  - `m_eng_pipeline_mod.c`, 197
  - `m_eng_pipeline_mod.h`, 119
- `arm_threshold`
  - `m_eng_percussifier_str`, 31
- `attack`
  - `m_eng_compressor_str`, 15
- `AUDIO_BLOCK_MS`
  - `m_eng.h`, 53
- `AUDIO_BLOCK_SAMPLES`
  - `m_eng.h`, 53
  - `m_eng_audio_block.h`, 59
- `AUDIO_HEADPHONE_DAC`
  - `m_eng_sgtl5000_defs.h`, 130
- `AUDIO_HEADPHONE_LINEIN`
  - `m_eng_sgtl5000_defs.h`, 130
- `AUDIO_SAMPLE_RATE`
  - `m_eng.h`, 53
- `AUDIO_SAMPLE_RATE_EXACT`
  - `m_eng.h`, 53
- `AVG_DURATION_UPDATE_COEF`
  - `m_eng_context.c`, 165
- `back_pipeline`
  - `m_eng_profile`, 34
- `back_pipeline_warmed_up`
  - `m_eng_profile`, 34
- `band_center`
  - `m_transformer`, 49
- `band_hp_cutoff`
  - `m_transformer`, 49
- `band_lp_cutoff`
  - `m_transformer`, 49
- `band_mode`
  - `m_transformer`, 50
- `band_pass`
  - `m_transformer_enum.h`, 258
- `band_width`
  - `m_transformer`, 50
- `bandwidth`

- m\_eng\_band\_pass\_filter\_str, 12
  - m\_eng\_biquad\_str, 13
- bass\_comp
  - m\_eng\_low\_end\_compressor\_str, 27
- bass\_cutoff
  - m\_eng\_distortion\_str, 20
- bass\_mix
  - m\_eng\_distortion\_str, 20
- binary\_max
  - m\_eng.h, 53
- binary\_min
  - m\_eng.h, 53
- biquad\_type
  - m\_transformer\_enum.h, 257
- blocked\_jobs
  - m\_eng\_profile, 34
- buf
  - m\_eng\_delay\_str, 18
  - m\_eng\_flanger\_str, 23
- buffer\_array\_size
  - m\_delay\_buffer, 7
- buffer\_buffer
  - m\_eng\_mempool.c, 188
- buffer\_head
  - m\_eng\_mempool.c, 188
- buffer\_pool
  - m\_eng\_mempool.c, 188
- BUFFER\_QUEUE\_STATIC
  - m\_eng\_mempool.c, 187
- buffer\_tail
  - m\_eng\_mempool.c, 188
- buffers
  - m\_delay\_buffer, 7
- calc\_3\_band\_eq
  - m\_eng\_equaliser.c, 175
  - m\_eng\_equaliser.h, 82
- calc\_3\_band\_splitter
  - m\_eng\_band\_splitter.h, 60
- calc\_adaptive\_waveshaper
  - m\_eng\_adaptive\_waveshaper.c, 157
  - m\_eng\_adaptive\_waveshaper.h, 59
- calc\_amplifier
  - m\_eng\_buffer\_mixer\_amp.c, 159
  - m\_eng\_buffer\_mixer\_amp.h, 63
- calc\_band\_pass\_filter
  - m\_eng\_pass\_filter.c, 190
  - m\_eng\_pass\_filter.h, 110
- calc\_biquad
  - m\_eng\_biquad.c, 158
  - m\_eng\_biquad.h, 62
- calc\_buffer
  - m\_eng\_buffer\_mixer\_amp.c, 159
  - m\_eng\_buffer\_mixer\_amp.h, 63
- calc\_compressor
  - m\_eng\_compressor.c, 163
  - m\_eng\_compressor.h, 66
- calc\_delay
  - m\_eng\_delay.c, 170
  - m\_eng\_delay.h, 75
- calc\_dirty\_octave
  - m\_eng\_dirty\_octave.c, 173
  - m\_eng\_dirty\_octave.h, 78
- calc\_distortion
  - m\_eng\_distortion.c, 173
  - m\_eng\_distortion.h, 80
- calc\_envelope
  - m\_eng\_envelope.c, 174
  - m\_eng\_envelope.h, 81
- calc\_flanger
  - m\_eng\_flanger.c, 175
  - m\_eng\_flanger.h, 83
- calc\_high\_pass\_filter
  - m\_eng\_pass\_filter.c, 190
  - m\_eng\_pass\_filter.h, 110
- calc\_low\_end\_compressor
  - m\_eng\_low\_end\_compressor.c, 186
  - m\_eng\_low\_end\_compressor.h, 103
- calc\_low\_pass\_filter
  - m\_eng\_pass\_filter.c, 191
  - m\_eng\_pass\_filter.h, 110
- calc\_lr\_high\_pass\_filter
  - m\_eng\_linkowitz\_riley.c, 181
  - m\_eng\_linkowitz\_riley.h, 91
- calc\_lr\_low\_pass\_filter
  - m\_eng\_linkowitz\_riley.c, 181
  - m\_eng\_linkowitz\_riley.h, 91
- calc\_n\_band\_splitter
  - m\_eng\_band\_splitter.h, 60
- calc\_noise\_suppressor
  - m\_eng\_noise\_suppressor.c, 189
  - m\_eng\_noise\_suppressor.h, 107
- calc\_percussifier
  - m\_eng\_percussifier.c, 192
  - m\_eng\_percussifier.h, 113
- calc\_simple\_distortion
  - m\_eng\_simple\_distortion.c, 205
  - m\_eng\_simple\_distortion.h, 143
- calc\_transformer
  - m\_eng\_transformer\_template.h, 148
- calc\_vol
  - m\_eng\_sgtl5000.cpp, 202
  - m\_eng\_sgtl5000.h, 125
- calc\_warbler
  - m\_eng\_warbler.c, 213
  - m\_eng\_warbler.h, 153
- calc\_waveshaper
  - m\_eng\_waveshaper.c, 213
  - m\_eng\_waveshaper.h, 155
- callback
  - m\_message, 43
- calls
  - m\_eng\_profiler\_entry, 36
- cb\_arg
  - m\_message, 43
- center
  - m\_eng\_band\_pass\_filter\_str, 12



- [m\\_eng\\_warbler\\_str](#), [37](#)
- CHIP\_ADCDAC\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [130](#)
- CHIP\_ANA\_ADC\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [130](#)
- CHIP\_ANA\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [130](#)
- CHIP\_ANA\_HP\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [130](#)
- CHIP\_ANA\_POWER
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [130](#)
- CHIP\_ANA\_STATUS
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [130](#)
- CHIP\_ANA\_TEST1
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [130](#)
- CHIP\_ANA\_TEST2
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [131](#)
- CHIP\_CLK\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [131](#)
- CHIP\_CLK\_TOP\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [131](#)
- CHIP\_DAC\_VOL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [131](#)
- CHIP\_DIG\_POWER
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [131](#)
- CHIP\_I2S\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [131](#)
- CHIP\_ID
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [131](#)
- CHIP\_LINE\_OUT\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [131](#)
- CHIP\_LINE\_OUT\_VOL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [131](#)
- CHIP\_LINREG\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [131](#)
- CHIP\_MIC\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [132](#)
- CHIP\_PAD\_STRENGTH
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [132](#)
- CHIP\_PLL\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [132](#)
- CHIP\_REF\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [132](#)
- CHIP\_SHORT\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [132](#)
- CHIP\_SSS\_CTRL
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [132](#)
- chunk\_size
  - [m\\_eng\\_envelope\\_str](#), [21](#)
- CLICK\_SLOPE\_THRESHOLD
  - [m\\_eng\\_context.h](#), [68](#)
- clone\_pipeline
  - [m\\_eng\\_pipeline.c](#), [193](#)
  - [m\\_eng\\_pipeline.h](#), [114](#)
- clone\_transformer
  - [m\\_eng\\_transformer.c](#), [206](#)
  - [m\\_eng\\_transformer.h](#), [145](#)
- coefficient
  - [m\\_eng\\_adaptive\\_waveshaper\\_str](#), [10](#)
  - [m\\_eng\\_waveshaper\\_str](#), [39](#)
- coefs
  - [m\\_eng\\_3\\_band\\_eq\\_str](#), [8](#)
- comms\_fsm\_eng\_state
  - [m\\_eng\\_comms.cpp](#), [161](#)
- comms\_fsm\_eng\_state\_t
  - [m\\_eng\\_comms.cpp](#), [160](#)
- compute\_pipeline
  - [m\\_eng\\_pipeline.c](#), [193](#)
  - [m\\_eng\\_pipeline.h](#), [114](#)
- configure\_i2s\_dma
  - [m\\_eng\\_i2s\\_dma.cpp](#), [178](#)
- control\_mode
  - [m\\_eng\\_3\\_band\\_eq\\_str](#), [8](#)
- convert\_block\_float\_to\_int
  - [m\\_eng\\_useful\\_functions.c](#), [211](#)
  - [m\\_eng\\_useful\\_functions.h](#), [152](#)
- convert\_block\_int\_to\_float
  - [m\\_eng\\_useful\\_functions.c](#), [211](#)
  - [m\\_eng\\_useful\\_functions.h](#), [152](#)
- cpu\_cycles\_total
  - [m\\_eng\\_globals.c](#), [177](#)
  - [m\\_eng\\_globals.h](#), [88](#)
- cpu\_cycles\_total\_max
  - [m\\_eng\\_globals.c](#), [177](#)
  - [m\\_eng\\_globals.h](#), [88](#)
- crc\_8
  - [m\\_comms.c](#), [216](#)
- create\_m\_message
  - [m\\_comms.c](#), [216](#)
  - [m\\_comms.h](#), [227](#)
- create\_m\_message\_nodata
  - [m\\_comms.c](#), [217](#)
  - [m\\_comms.h](#), [227](#)
- create\_m\_response
  - [m\\_comms.c](#), [217](#)
  - [m\\_comms.h](#), [227](#)
- create\_m\_response\_error
  - [m\\_comms.c](#), [217](#)
  - [m\\_comms.h](#), [227](#)
- create\_m\_response\_nodata
  - [m\\_comms.c](#), [217](#)
  - [m\\_comms.h](#), [227](#)
- create\_m\_response\_ok
  - [m\\_comms.c](#), [217](#)
  - [m\\_comms.h](#), [227](#)
- create\_m\_response\_parameter\_value
  - [m\\_comms.c](#), [217](#)
  - [m\\_comms.h](#), [227](#)
- create\_m\_response\_profile\_id
  - [m\\_comms.c](#), [217](#)
  - [m\\_comms.h](#), [227](#)
- create\_m\_response\_transformer\_id
  - [m\\_comms.c](#), [217](#)
  - [m\\_comms.h](#), [228](#)
- create\_pipeline\_mod\_append\_transformer
  - [m\\_eng\\_pipeline\\_mod.c](#), [197](#)

- m\_eng\_pipeline\_mod.h, 119
- create\_pipeline\_mod\_change\_transformer\_setting
  - m\_eng\_pipeline\_mod.c, 197
  - m\_eng\_pipeline\_mod.h, 119
- create\_pipeline\_mod\_move\_transformer
  - m\_eng\_pipeline\_mod.c, 197
  - m\_eng\_pipeline\_mod.h, 119
- create\_pipeline\_mod\_remove\_transformer
  - m\_eng\_pipeline\_mod.c, 197
  - m\_eng\_pipeline\_mod.h, 119
- current\_cycle
  - m\_eng\_globals.c, 176
  - m\_eng\_globals.h, 88
- cutoff
  - m\_eng\_biquad\_str, 13
- cutoff\_frequency
  - m\_eng\_high\_pass\_filter\_str, 25
  - m\_eng\_low\_pass\_filter\_str, 28
  - m\_lr\_high\_pass\_filter\_str, 40
  - m\_lr\_low\_pass\_filter\_str, 42
- cxt\_append\_transformer\_to\_profile
  - m\_eng\_context.c, 165
  - m\_eng\_context.h, 68
- cxt\_get\_back\_parameter\_by\_id
  - m\_eng\_context.c, 165
- cxt\_get\_front\_parameter\_by\_id
  - m\_eng\_context.c, 165
- cxt\_get\_n\_profile\_transformers
  - m\_eng\_context.c, 165
  - m\_eng\_context.h, 68
- cxt\_get\_n\_transformer\_params
  - m\_eng\_context.c, 165
  - m\_eng\_context.h, 69
- cxt\_get\_n\_transformer\_settings
  - m\_eng\_context.c, 165
  - m\_eng\_context.h, 69
- cxt\_get\_parameter\_by\_id
  - m\_eng\_context.c, 166
  - m\_eng\_context.h, 69
- cxt\_get\_setting\_by\_id
  - m\_eng\_context.c, 166
  - m\_eng\_context.h, 69
- cxt\_get\_tid\_by\_pos
  - m\_eng\_context.c, 166
  - m\_eng\_context.h, 69
- cxt\_get\_transformer\_by\_id
  - m\_eng\_context.c, 166
  - m\_eng\_context.h, 69
- cxt\_get\_transformer\_type
  - m\_eng\_context.c, 166
  - m\_eng\_context.h, 69
- cxt\_insert\_transformer\_to\_profile
  - m\_eng\_context.c, 166
  - m\_eng\_context.h, 70
- cxt\_move\_transformer
  - m\_eng\_context.c, 167
  - m\_eng\_context.h, 70
- cxt\_parameter\_id\_valid
  - m\_eng\_context.h, 70
- cxt\_prepend\_transformer\_to\_profile
  - m\_eng\_context.c, 167
  - m\_eng\_context.h, 70
- cxt\_process
  - m\_eng\_context.c, 167
  - m\_eng\_context.h, 70
- cxt\_profile\_id\_valid
  - m\_eng\_context.c, 167
  - m\_eng\_context.h, 70
- cxt\_remove\_transformer\_from\_profile
  - m\_eng\_context.c, 167
  - m\_eng\_context.h, 70
- cxt\_run\_scheduled\_maintenance
  - m\_eng\_context.c, 167
  - m\_eng\_context.h, 71
- cxt\_set\_active\_profile
  - m\_eng\_context.c, 167
  - m\_eng\_context.h, 71
- cxt\_switch\_to\_profile
  - m\_eng\_context.c, 168
  - m\_eng\_context.h, 71
- cxt\_transformer\_id\_valid
  - m\_eng\_context.c, 168
  - m\_eng\_context.h, 71
- cxt\_update\_parameter\_value\_by\_id
  - m\_eng\_context.c, 168
  - m\_eng\_context.h, 71
- cxt\_update\_setting\_value\_by\_id
  - m\_eng\_context.c, 168
  - m\_eng\_context.h, 71
- cycle
  - m\_eng\_log\_entry, 26
- CYCLES\_TO\_SECONDS
  - m\_eng\_logging.h, 94
- cycles\_to\_seconds
  - m\_eng\_globals.c, 176
  - m\_eng\_globals.h, 88
- cycles\_upper
  - m\_eng\_globals.c, 177
- d
  - m\_eng\_flanger\_str, 23
- DAP\_AUDIO\_EQ
  - m\_eng\_sgtl5000\_defs.h, 132
- DAP\_AUDIO\_EQ\_BAND1
  - m\_eng\_sgtl5000\_defs.h, 132
- DAP\_AUDIO\_EQ\_BAND2
  - m\_eng\_sgtl5000\_defs.h, 132
- DAP\_AUDIO\_EQ\_BAND3
  - m\_eng\_sgtl5000\_defs.h, 132
- DAP\_AUDIO\_EQ\_BASS\_BAND0
  - m\_eng\_sgtl5000\_defs.h, 133
- DAP\_AUDIO\_EQ\_TREBLE\_BAND4
  - m\_eng\_sgtl5000\_defs.h, 133
- DAP\_AVC\_ATTACK
  - m\_eng\_sgtl5000\_defs.h, 133
- DAP\_AVC\_CTRL
  - m\_eng\_sgtl5000\_defs.h, 133

- DAP\_AVC\_DECAY
  - m\_eng\_sgtl5000\_defs.h, [133](#)
- DAP\_AVC\_THRESHOLD
  - m\_eng\_sgtl5000\_defs.h, [133](#)
- DAP\_BASS\_ENHANCE
  - m\_eng\_sgtl5000\_defs.h, [133](#)
- DAP\_BASS\_ENHANCE\_CTRL
  - m\_eng\_sgtl5000\_defs.h, [133](#)
- DAP\_COEF\_WR\_A1\_LSB
  - m\_eng\_sgtl5000\_defs.h, [133](#)
- DAP\_COEF\_WR\_A1\_MSB
  - m\_eng\_sgtl5000\_defs.h, [133](#)
- DAP\_COEF\_WR\_A2\_LSB
  - m\_eng\_sgtl5000\_defs.h, [134](#)
- DAP\_COEF\_WR\_A2\_MSB
  - m\_eng\_sgtl5000\_defs.h, [134](#)
- DAP\_COEF\_WR\_B0\_LSB
  - m\_eng\_sgtl5000\_defs.h, [134](#)
- DAP\_COEF\_WR\_B0\_MSB
  - m\_eng\_sgtl5000\_defs.h, [134](#)
- DAP\_COEF\_WR\_B1\_LSB
  - m\_eng\_sgtl5000\_defs.h, [134](#)
- DAP\_COEF\_WR\_B1\_MSB
  - m\_eng\_sgtl5000\_defs.h, [134](#)
- DAP\_COEF\_WR\_B2\_LSB
  - m\_eng\_sgtl5000\_defs.h, [134](#)
- DAP\_COEF\_WR\_B2\_MSB
  - m\_eng\_sgtl5000\_defs.h, [134](#)
- DAP\_CONTROL
  - m\_eng\_sgtl5000\_defs.h, [134](#)
- DAP\_FILTER\_COEF\_ACCESS
  - m\_eng\_sgtl5000\_defs.h, [134](#)
- DAP\_MAIN\_CHAN
  - m\_eng\_sgtl5000\_defs.h, [135](#)
- DAP\_MIX\_CHAN
  - m\_eng\_sgtl5000\_defs.h, [135](#)
- DAP\_PEQ
  - m\_eng\_sgtl5000\_defs.h, [135](#)
- DAP\_SGTL\_SURROUND
  - m\_eng\_sgtl5000\_defs.h, [135](#)
- data
  - m\_eng\_log\_entry, [26](#)
  - m\_message, [43](#)
  - m\_pipeline\_mod, [46](#)
  - m\_response, [47](#)
- data\_type
  - m\_eng\_log\_entry, [26](#)
- db
  - m\_eng\_amplifier\_str, [10](#)
- db\_gain
  - m\_eng\_biquad\_str, [13](#)
- dc\_average
  - m\_eng\_dirty\_octave\_str, [19](#)
- DC\_BLOCKER\_ALPHA
  - m\_eng\_context.h, [68](#)
- dc\_blocker\_avg
  - m\_eng\_context, [16](#)
- DEBUG\_PRINT\_MILLIS
  - m\_eng.cpp, [156](#)
- decay\_rate
  - m\_eng\_percussifier\_str, [32](#)
- DECLARE\_LINKED\_LIST
  - m\_eng\_pipeline\_mod.h, [120](#)
  - m\_linked\_list.h, [237](#)
- DECLARE\_LINKED\_PTR\_LIST
  - m\_linked\_list.h, [238](#)
  - m\_parameter.h, [248](#)
  - m\_transformer.h, [252](#)
- declick\_buffer
  - m\_eng\_context, [16](#)
- DECLICK\_BUFSIZE
  - m\_eng\_context.h, [68](#)
- decode\_m\_message
  - m\_comms.c, [218](#)
  - m\_comms.h, [228](#)
- decode\_m\_response
  - m\_comms.c, [218](#)
  - m\_comms.h, [228](#)
- DEFAULT\_MAX\_JUMP
  - m\_eng\_parameter.h, [108](#)
- delay\_gain
  - m\_eng\_delay\_str, [18](#)
- delay\_samples
  - m\_eng\_delay\_str, [18](#)
- DENORMAL\_THRESHOLD
  - m\_eng\_useful\_functions.c, [211](#)
- depth
  - m\_eng\_flanger\_str, [23](#)
- dist
  - m\_eng\_distortion\_str, [20](#)
- DISTORTION\_ARCTAN
  - m\_transformer\_enum.h, [255](#)
- DISTORTION\_CLIP
  - m\_transformer\_enum.h, [255](#)
- DISTORTION\_SOFT\_FOLD
  - m\_transformer\_enum.h, [255](#)
- DISTORTION\_TANH
  - m\_transformer\_enum.h, [255](#)
- dry\_mix
  - m\_eng\_flanger\_str, [23](#)
- e
  - m\_eng\_envelope\_str, [21](#)
  - m\_eng\_warbler\_str, [37](#)
- e\_final
  - m\_eng\_compressor\_str, [15](#)
  - m\_eng\_noise\_suppressor\_str, [30](#)
- encode\_m\_message
  - m\_comms.c, [218](#)
  - m\_comms.h, [228](#)
- encode\_m\_response
  - m\_comms.c, [218](#)
  - m\_comms.h, [228](#)
- EPSILON
  - m\_eng\_compressor.c, [163](#)
- ERR\_ALLOC\_FAIL
  - m\_error\_codes.h, [232](#)

ERR\_ARRAY\_MALFORMED  
     m\_error\_codes.h, [232](#)  
 ERR\_BAD\_ARGS  
     m\_error\_codes.h, [232](#)  
 ERR\_BAD\_REQUEST  
     m\_error\_codes.h, [232](#)  
 ERR\_BUSTED\_MSG  
     m\_error\_codes.h, [232](#)  
 ERR\_COMMS\_FAIL  
     m\_error\_codes.h, [232](#)  
 ERR\_FIXED\_ARRAY\_FULL  
     m\_error\_codes.h, [232](#)  
 ERR\_FOPEN\_FAIL  
     m\_error\_codes.h, [233](#)  
 ERR\_I2C\_FAIL  
     m\_error\_codes.h, [233](#)  
 ERR\_INCONSISTENT\_BACK\_PIPELINE  
     m\_error\_codes.h, [233](#)  
 ERR\_INVALID\_MESSAGE  
     m\_error\_codes.h, [233](#)  
 ERR\_INVALID\_PARAMETER\_ID  
     m\_error\_codes.h, [233](#)  
 ERR\_INVALID\_PROFILE\_ID  
     m\_error\_codes.h, [233](#)  
 ERR\_INVALID\_SETTING\_ID  
     m\_error\_codes.h, [233](#)  
 ERR\_INVALID\_TRANSFORMER\_ID  
     m\_error\_codes.h, [233](#)  
 ERR\_LOOP\_DETECTED  
     m\_error\_codes.h, [233](#)  
 ERR\_MANGLED\_FILE  
     m\_error\_codes.h, [233](#)  
 ERR\_MUTEX\_UNAVAILABLE  
     m\_error\_codes.h, [234](#)  
 ERR\_NO\_RESPONSE  
     m\_error\_codes.h, [234](#)  
 ERR\_NODE\_PRIVATE  
     m\_error\_codes.h, [234](#)  
 ERR\_NULL\_PTR  
     m\_error\_codes.h, [234](#)  
 ERR\_PIPELINE\_BUSTED  
     m\_error\_codes.h, [234](#)  
 ERR\_PIPELINE\_FULL  
     m\_error\_codes.h, [234](#)  
 ERR\_PIPELINE\_NULL  
     m\_error\_codes.h, [234](#)  
 ERR\_POSITION\_ILLEGAL  
     m\_error\_codes.h, [234](#)  
 ERR\_POSITION\_OCCUPIED  
     m\_error\_codes.h, [234](#)  
 ERR\_POT\_LINK\_MALFORMED  
     m\_error\_codes.h, [234](#)  
 ERR\_QUEUE\_FULL  
     m\_error\_codes.h, [235](#)  
 ERR\_QUEUE\_SEND\_FAILED  
     m\_error\_codes.h, [235](#)  
 ERR\_SD\_INIT\_FAIL  
     m\_error\_codes.h, [235](#)  
 ERR\_SD\_MOUNT\_FAIL  
     m\_error\_codes.h, [235](#)  
 ERR\_SGTL5000\_WRITE\_FAIL  
     m\_error\_codes.h, [235](#)  
 ERR\_SPI\_INIT\_FAIL  
     m\_error\_codes.h, [235](#)  
 ERR\_SWITCH\_LINK\_MALFORMED  
     m\_error\_codes.h, [235](#)  
 ERR\_TRANSFORMER\_MALFORMED  
     m\_error\_codes.h, [235](#)  
 ERR\_UNFINISHED\_WRITE  
     m\_error\_codes.h, [235](#)  
 ERR\_UNIMPLEMENTED  
     m\_error\_codes.h, [235](#)  
 ERR\_UNKNOWN\_ERR  
     m\_error\_codes.h, [236](#)  
 ERR\_VALUE\_OUT\_OF\_BOUNDS  
     m\_error\_codes.h, [236](#)  
 esp32\_message\_check\_handle  
     m\_eng\_comms.cpp, [161](#)  
     m\_eng\_comms.h, [65](#)  
 et\_message\_data\_length  
     m\_comms.c, [218](#)  
     m\_comms.h, [228](#)  
 et\_message\_default\_retries  
     m\_comms.c, [218](#)  
 extra  
     m\_response, [47](#)  
 fade\_alpha  
     m\_eng\_percussifier\_str, [32](#)  
 fade\_in  
     m\_eng\_percussifier\_str, [32](#)  
 fade\_in\_samples  
     m\_eng\_percussifier\_str, [32](#)  
 fade\_out  
     m\_eng\_percussifier\_str, [32](#)  
 FADER\_FADE\_IN  
     m\_eng\_transformer.h, [144](#)  
 FADER\_FADE\_OUT  
     m\_eng\_transformer.h, [144](#)  
 file\_name  
     m\_eng\_log\_entry, [26](#)  
 filter  
     m\_eng\_envelope\_str, [21](#)  
     m\_eng\_warbler\_str, [38](#)  
 FILTER\_BANDPASS  
     m\_eng\_sgtl5000\_defs.h, [135](#)  
 FILTER\_HIPASS  
     m\_eng\_sgtl5000\_defs.h, [135](#)  
 FILTER\_HISHELF  
     m\_eng\_sgtl5000\_defs.h, [135](#)  
 FILTER\_LOPASS  
     m\_eng\_sgtl5000\_defs.h, [135](#)  
 FILTER\_LOSHELF  
     m\_eng\_sgtl5000\_defs.h, [135](#)  
 FILTER\_NOTCH  
     m\_eng\_sgtl5000\_defs.h, [135](#)  
 FILTER\_PARAEQ

- [m\\_eng\\_sgtl5000\\_defs.h](#), [136](#)
- [filters](#)
  - [m\\_eng\\_3\\_band\\_eq\\_str](#), [8](#)
  - [m\\_eng\\_3\\_band\\_splitter\\_str](#), [9](#)
  - [m\\_eng\\_n\\_band\\_splitter\\_str](#), [30](#)
- [FLAT\\_FREQUENCY](#)
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [136](#)
- [FLOAT\\_TO\\_INT16\\_MAX](#)
  - [m\\_eng\\_useful\\_functions.c](#), [211](#)
- [format\\_log\\_entry](#)
  - [m\\_eng\\_logging.cpp](#), [184](#)
- [free\\_flanger\\_struct](#)
  - [m\\_eng\\_flanger.c](#), [175](#)
- [free\\_transformer](#)
  - [m\\_eng\\_transformer.c](#), [206](#)
  - [m\\_eng\\_transformer.h](#), [145](#)
- [front\\_pipeline](#)
  - [m\\_eng\\_profile](#), [34](#)
- [full\\_debug\\_print](#)
  - [m\\_eng\\_debugging.c](#), [169](#)
  - [m\\_eng\\_debugging.h](#), [74](#)
- [function](#)
  - [m\\_eng\\_distortion\\_str](#), [20](#)
  - [m\\_eng\\_log\\_entry](#), [26](#)
- [function\\_name](#)
  - [m\\_eng\\_profiler\\_entry](#), [36](#)
- [FUNCTION\\_START](#)
  - [m\\_eng\\_logging.h](#), [94](#)
- [fuzz](#)
  - [m\\_eng\\_dirty\\_octave\\_str](#), [19](#)
- [g](#)
  - [m\\_eng\\_amplifier\\_str](#), [10](#)
  - [m\\_eng\\_delay\\_str](#), [18](#)
- [gain](#)
  - [m\\_eng\\_amplifier\\_str](#), [10](#)
  - [m\\_eng\\_percussifier\\_str](#), [32](#)
- [global\\_cxt](#)
  - [m\\_eng\\_context.h](#), [72](#)
  - [m\\_eng\\_globals.c](#), [177](#)
- [GRAPHIC\\_EQUALIZER](#)
  - [m\\_eng\\_sgtl5000\\_defs.h](#), [136](#)
- [gut\\_pipeline](#)
  - [m\\_eng\\_pipeline.c](#), [193](#)
  - [m\\_eng\\_pipeline.h](#), [115](#)
- [handle\\_esp32\\_message](#)
  - [m\\_eng\\_comms.cpp](#), [161](#)
- [hard\\_clip](#)
  - [m\\_eng\\_useful\\_functions.c](#), [212](#)
  - [m\\_eng\\_useful\\_functions.h](#), [152](#)
- [head](#)
  - [m\\_eng\\_mempool.c](#), [188](#)
- [high](#)
  - [m\\_eng\\_3\\_band\\_eq\\_str](#), [8](#)
- [high\\_pass](#)
  - [m\\_transformer\\_enum.h](#), [258](#)
- [high\\_shelf](#)
  - [m\\_transformer\\_enum.h](#), [258](#)
- [hold\\_samples](#)
  - [m\\_eng\\_percussifier\\_str](#), [32](#)
- [i2c\\_receive\\_isr](#)
  - [m\\_eng\\_comms.cpp](#), [161](#)
  - [m\\_eng\\_comms.h](#), [65](#)
- [i2c\\_request\\_isr](#)
  - [m\\_eng\\_comms.cpp](#), [161](#)
  - [m\\_eng\\_comms.h](#), [65](#)
- [i2s\\_in\\_block\\_left](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [180](#)
- [i2s\\_in\\_block\\_offset](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [180](#)
- [i2s\\_in\\_block\\_right](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [180](#)
- [i2s\\_in\\_dma](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [178](#)
- [i2s\\_in\\_transmit](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [178](#)
- [i2s\\_in\\_update\\_responsibility](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [180](#)
- [i2s\\_input\\_blocks](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [180](#)
  - [m\\_eng\\_i2s\\_dma.h](#), [90](#)
- [i2s\\_input\\_update](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [179](#)
  - [m\\_eng\\_i2s\\_dma.h](#), [90](#)
- [i2s\\_out\\_block\\_left\\_1st](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [180](#)
- [i2s\\_out\\_block\\_left\\_2nd](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [180](#)
- [i2s\\_out\\_block\\_left\\_offset](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [180](#)
- [i2s\\_out\\_block\\_right\\_1st](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [180](#)
- [i2s\\_out\\_block\\_right\\_2nd](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [180](#)
- [i2s\\_out\\_block\\_right\\_offset](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [180](#)
- [i2s\\_out\\_dma](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [179](#)
- [i2s\\_out\\_update\\_responsibility](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [181](#)
- [i2s\\_output\\_blocks](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [181](#)
- [i2s\\_output\\_transmit\\_mono\\_float](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [179](#)
  - [m\\_eng\\_i2s\\_dma.h](#), [90](#)
- [i2s\\_output\\_transmit\\_mono\\_int](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [179](#)
  - [m\\_eng\\_i2s\\_dma.h](#), [90](#)
- [i2s\\_output\\_update](#)
  - [m\\_eng\\_i2s\\_dma.cpp](#), [179](#)
  - [m\\_eng\\_i2s\\_dma.h](#), [90](#)
- [id](#)
  - [m\\_transformer](#), [50](#)
- [identity\\_function](#)
  - [m\\_eng\\_useful\\_functions.c](#), [212](#)
  - [m\\_eng\\_useful\\_functions.h](#), [152](#)

- IDLE
  - m\_eng\_comms.cpp, 160
- IMPLEMENT\_LINKED\_LIST
  - m\_eng\_pipeline\_mod.c, 197
  - m\_linked\_list.h, 238
- IMPLEMENT\_LINKED\_PTR\_LIST
  - m\_linked\_list.h, 238
- index
  - m\_delay\_buffer, 7
- init\_3\_band\_eq
  - m\_eng\_transformer\_init.c, 208
- init\_3\_band\_eq\_str
  - m\_eng\_equaliser.c, 175
  - m\_eng\_equaliser.h, 82
- init\_3\_band\_splitter\_str
  - m\_eng\_band\_splitter.h, 60
- init\_adaptive\_waveshaper\_str
  - m\_eng\_adaptive\_waveshaper.c, 157
  - m\_eng\_adaptive\_waveshaper.h, 59
- init\_amplifier
  - m\_eng\_transformer\_init.c, 208
- init\_amplifier\_str
  - m\_eng\_buffer\_mixer\_amp.c, 159
  - m\_eng\_buffer\_mixer\_amp.h, 63
- init\_band\_pass\_filter
  - m\_eng\_transformer\_init.c, 208
- init\_band\_pass\_filter\_str
  - m\_eng\_pass\_filter.c, 191
  - m\_eng\_pass\_filter.h, 110
- init\_biquad\_str
  - m\_eng\_biquad.c, 158
  - m\_eng\_biquad.h, 62
- init\_bypass\_profile
  - m\_eng\_profile.h, 122
- init\_compressor
  - m\_eng\_transformer\_init.c, 208
- init\_compressor\_str
  - m\_eng\_compressor.c, 163
  - m\_eng\_compressor.h, 66
- init\_delay
  - m\_eng\_transformer\_init.c, 208
- init\_delay\_buffer
  - m\_eng\_delay\_buffer.c, 171
  - m\_eng\_delay\_buffer.h, 76
- init\_delay\_str
  - m\_eng\_delay.c, 170
  - m\_eng\_delay.h, 75
- init\_dirty\_octave
  - m\_eng\_transformer\_init.c, 208
- init\_dirty\_octave\_str
  - m\_eng\_dirty\_octave.c, 173
  - m\_eng\_dirty\_octave.h, 78
- init\_distortion
  - m\_eng\_transformer\_init.c, 209
- init\_distortion\_str
  - m\_eng\_distortion.c, 173
  - m\_eng\_distortion.h, 80
- init\_envelope
  - m\_eng\_transformer\_init.c, 209
- init\_envelope\_str
  - m\_eng\_envelope.c, 174
  - m\_eng\_envelope.h, 81
- init\_esp32\_link
  - m\_eng\_comms.cpp, 161
  - m\_eng\_comms.h, 65
- init\_flanger
  - m\_eng\_transformer\_init.c, 209
- init\_flanger\_str
  - m\_eng\_flanger.c, 176
  - m\_eng\_flanger.h, 83
- init\_high\_pass\_filter
  - m\_eng\_transformer\_init.c, 209
- init\_high\_pass\_filter\_str
  - m\_eng\_pass\_filter.c, 191
  - m\_eng\_pass\_filter.h, 110
- init\_i2s\_dma
  - m\_eng\_i2s\_dma.cpp, 179
  - m\_eng\_i2s\_dma.h, 90
- init\_low\_end\_compressor
  - m\_eng\_transformer\_init.c, 209
- init\_low\_end\_compressor\_str
  - m\_eng\_low\_end\_compressor.c, 186
  - m\_eng\_low\_end\_compressor.h, 103
- init\_low\_pass\_filter
  - m\_eng\_transformer\_init.c, 209
- init\_low\_pass\_filter\_str
  - m\_eng\_pass\_filter.c, 191
  - m\_eng\_pass\_filter.h, 110
- init\_lr\_high\_pass\_filter\_str
  - m\_eng\_linkowitz\_riley.c, 181
  - m\_eng\_linkowitz\_riley.h, 91
- init\_lr\_low\_pass\_filter\_str
  - m\_eng\_linkowitz\_riley.c, 182
  - m\_eng\_linkowitz\_riley.h, 92
- init\_m\_eng\_context
  - m\_eng\_context.c, 168
  - m\_eng\_context.h, 71
- init\_mem\_pools
  - m\_eng\_mempool.c, 187
  - m\_eng\_mempool.h, 106
- init\_n\_band\_splitter\_str
  - m\_eng\_band\_splitter.h, 61
- init\_noise\_suppressor
  - m\_eng\_transformer\_init.c, 209
- init\_noise\_suppressor\_str
  - m\_eng\_noise\_suppressor.c, 189
  - m\_eng\_noise\_suppressor.h, 107
- init\_parameter
  - m\_eng\_parameter.c, 190
  - m\_eng\_parameter.h, 109
- init\_percussifier
  - m\_eng\_transformer\_init.c, 209
- init\_percussifier\_str
  - m\_eng\_percussifier.c, 192
  - m\_eng\_percussifier.h, 113
- init\_pipeline

- m\_eng\_pipeline.c, 194
  - m\_eng\_pipeline.h, 115
- init\_pipeline\_mod
  - m\_eng\_pipeline\_mod.h, 120
- init\_profile
  - m\_eng\_profile.c, 200
  - m\_eng\_profile.h, 122
- init\_setting
  - m\_eng\_parameter.c, 190
  - m\_eng\_parameter.h, 109
- init\_simple\_distortion\_str
  - m\_eng\_simple\_distortion.c, 205
  - m\_eng\_simple\_distortion.h, 143
- init\_transformer
  - m\_eng\_transformer\_init.c, 209
  - m\_eng\_transformer\_init.h, 148
- init\_transformer\_str
  - m\_eng\_transformer\_template.h, 148
- init\_warbler
  - m\_eng\_transformer\_init.c, 210
- init\_warbler\_str
  - m\_eng\_warbler.c, 213
  - m\_eng\_warbler.h, 153
- init\_waveshaper\_str
  - m\_eng\_waveshaper.c, 213
  - m\_eng\_waveshaper.h, 155
- INITIAL\_TRANSFORMER\_ARRAY\_LENGTH
  - m\_eng\_pipeline.h, 114
- input\_lpf
  - m\_eng\_context, 16
- jobs
  - m\_eng\_profile, 34
- last\_out\_sample
  - m\_eng\_dirty\_octave\_str, 19
- LED\_BLINK\_MILLIS
  - m\_eng.cpp, 156
- line
  - m\_eng\_log\_entry, 26
- LL\_FREE
  - m\_eng.h, 53
  - m\_linked\_list.h, 238
- LL\_MALLOC
  - m\_eng.h, 53
  - m\_linked\_list.h, 238
- LN\_2
  - m\_eng.h, 54
- local\_amplitude
  - m\_eng\_adaptive\_waveshaper\_str, 10
- LOG\_ENTRIES\_PRINT\_BUF\_LEN
  - m\_eng\_logging.cpp, 183
- LOG\_PRINT\_MILLIS
  - m\_eng.cpp, 156
- low
  - m\_eng\_3\_band\_eq\_str, 8
- low\_pass
  - m\_eng\_distortion\_str, 20
  - m\_eng\_low\_end\_compressor\_str, 27
- m\_transformer\_enum.h, 258
- low\_shelf
  - m\_transformer\_enum.h, 258
- lpf\_alpha
  - m\_eng\_dirty\_octave\_str, 19
- m\_alloc
  - m\_alloc.c, 214
  - m\_alloc.h, 215
- m\_alloc.c, 214
  - m\_alloc, 214
  - m\_free, 214
  - m\_realloc, 214
  - m\_strndup, 214
  - print\_memory\_report, 214
- m\_alloc.h, 214, 216
  - m\_alloc, 215
  - m\_free, 215
  - m\_int\_lv\_free, 215
  - m\_int\_lv\_malloc, 215
  - m\_realloc, 215
  - m\_strndup, 215
  - print\_memory\_report, 215
- M\_BUFFER\_POOL\_SIZE
  - m\_eng\_mempool.h, 105
- m\_comms.c, 216
  - crc\_8, 216
  - create\_m\_message, 216
  - create\_m\_message\_nodata, 217
  - create\_m\_response, 217
  - create\_m\_response\_error, 217
  - create\_m\_response\_nodata, 217
  - create\_m\_response\_ok, 217
  - create\_m\_response\_parameter\_value, 217
  - create\_m\_response\_profile\_id, 217
  - create\_m\_response\_transformer\_id, 217
  - decode\_m\_message, 218
  - decode\_m\_response, 218
  - encode\_m\_message, 218
  - encode\_m\_response, 218
  - et\_message\_data\_length, 218
  - et\_message\_default\_retries, 218
  - m\_message\_code\_to\_string, 218
  - m\_response\_code\_to\_string, 218
  - te\_message\_data\_length, 219
  - valid\_m\_message\_type, 219
  - valid\_m\_response\_type, 219
- m\_comms.h, 219, 229
  - create\_m\_message, 227
  - create\_m\_message\_nodata, 227
  - create\_m\_response, 227
  - create\_m\_response\_error, 227
  - create\_m\_response\_nodata, 227
  - create\_m\_response\_ok, 227
  - create\_m\_response\_parameter\_value, 227
  - create\_m\_response\_profile\_id, 227
  - create\_m\_response\_transformer\_id, 228
  - decode\_m\_message, 228
  - decode\_m\_response, 228



- encode\_m\_message, 228
- encode\_m\_response, 228
- et\_message\_data\_length, 228
- M\_MESSAGE\_APPEND\_TRANSFORMER, 221
- m\_message\_code\_to\_string, 228
- M\_MESSAGE\_CRC\_FAIL, 221
- M\_MESSAGE\_CREATE\_PROFILE, 221
- M\_MESSAGE\_DELETE\_PROFILE, 221
- M\_MESSAGE\_ENTER\_TUNER\_MODE, 221
- M\_MESSAGE\_EXIT\_TUNER\_MODE, 221
- M\_MESSAGE\_GET\_N\_PARAMETERS, 221
- M\_MESSAGE\_GET\_N\_PROFILES, 221
- M\_MESSAGE\_GET\_N\_SETTINGS, 221
- M\_MESSAGE\_GET\_N\_TRANSFORMERS, 222
- M\_MESSAGE\_GET\_PARAM\_VALUE, 222
- M\_MESSAGE\_GET\_SETTING\_VALUE, 222
- M\_MESSAGE\_GET\_TRANSFORMER\_ID, 222
- M\_MESSAGE\_GET\_TRANSFORMER\_TYPE, 222
- M\_MESSAGE\_HI, 222
- M\_MESSAGE\_INVALID, 222
- M\_MESSAGE\_MAX\_DATA\_LEN, 222
- M\_MESSAGE\_MAX\_TRANSFER\_LEN, 222
- M\_MESSAGE\_MOVE\_TRANSFORMER, 222
- M\_MESSAGE\_NO\_MESSAGE, 223
- M\_MESSAGE\_REBOOT, 223
- M\_MESSAGE\_REMOVE\_TRANSFORMER, 223
- M\_MESSAGE\_REPEAT\_MESSAGE, 223
- M\_MESSAGE\_RESET, 223
- M\_MESSAGE\_SET\_PARAM\_VALUE, 223
- M\_MESSAGE\_SET\_SETTING\_VALUE, 223
- M\_MESSAGE\_STRING\_CONTINUE, 223
- M\_MESSAGE\_STRING\_CONTINUING, 223
- M\_MESSAGE\_SWITCH\_PROFILE, 223
- M\_MESSAGE\_TYPE\_MAX, 224
- M\_RESPONSE\_BAD\_MESSAGE, 224
- M\_RESPONSE\_BAD\_REQUEST, 224
- m\_response\_code\_to\_string, 228
- M\_RESPONSE\_CRC\_FAIL, 224
- M\_RESPONSE\_DELETED\_PROFILE, 224
- M\_RESPONSE\_ERROR, 224
- M\_RESPONSE\_HI, 224
- M\_RESPONSE\_INVALID, 224
- M\_RESPONSE\_MAX\_DATA\_LEN, 224
- M\_RESPONSE\_MAX\_TRANSFER\_LEN, 224
- M\_RESPONSE\_N\_PARAMETERS, 225
- M\_RESPONSE\_N\_PROFILES, 225
- M\_RESPONSE\_N\_SETTINGS, 225
- M\_RESPONSE\_N\_TRANSFORMERS, 225
- M\_RESPONSE\_NO\_MESSAGE, 225
- M\_RESPONSE\_OK, 225
- M\_RESPONSE\_PARAM\_VALUE, 225
- M\_RESPONSE\_PROFILE\_ID, 225
- M\_RESPONSE\_REPEAT\_MESSAGE, 225
- M\_RESPONSE\_SETTING\_VALUE, 225
- M\_RESPONSE\_START\_OVER, 226
- M\_RESPONSE\_STRING\_CONTINUING, 226
- M\_RESPONSE\_SWITCHING\_PROFILE, 226
- M\_RESPONSE\_TRANSFORMER\_ID, 226
- M\_RESPONSE\_TRANSFORMER\_TYPE, 226
- M\_RESPONSE\_TRY\_AGAIN, 226
- M\_RESPONSE\_TYPE\_MAX, 226
- M\_RESPONSE\_WAIT, 226
- MESSAGE\_LEN\_VARIABLE, 226
- te\_message\_data\_length, 229
- TEENSY\_ADDR, 226
- valid\_m\_message\_type, 229
- valid\_m\_response\_type, 229
- m\_delay\_buffer, 7
  - buffer\_array\_size, 7
  - buffers, 7
  - index, 7
  - n\_buffers, 7
  - pos, 7
  - valid, 8
- m\_delay\_buffer\_advance
  - m\_eng\_delay\_buffer.c, 171
  - m\_eng\_delay\_buffer.h, 76
- m\_delay\_buffer\_get\_delayed\_sample
  - m\_eng\_delay\_buffer.c, 171
  - m\_eng\_delay\_buffer.h, 76
- m\_delay\_buffer\_get\_delayed\_sample\_ptr
  - m\_eng\_delay\_buffer.c, 171
  - m\_eng\_delay\_buffer.h, 76
- m\_delay\_buffer\_get\_fractional\_delayed\_sample
  - m\_eng\_delay\_buffer.c, 172
  - m\_eng\_delay\_buffer.h, 76
- m\_delay\_buffer\_resize\_milliseconds
  - m\_eng\_delay\_buffer.c, 172
  - m\_eng\_delay\_buffer.h, 77
- m\_delay\_buffer\_resize\_samples
  - m\_eng\_delay\_buffer.c, 172
  - m\_eng\_delay\_buffer.h, 77
- m\_delay\_buffer\_resize\_seconds
  - m\_eng\_delay\_buffer.c, 172
  - m\_eng\_delay\_buffer.h, 77
- m\_delay\_buffer\_tick
  - m\_eng\_delay\_buffer.c, 172
  - m\_eng\_delay\_buffer.h, 77
- M\_DISTORTION\_ARCTAN
  - m\_eng\_distortion.h, 79
- M\_DISTORTION\_CLIP
  - m\_eng\_distortion.h, 79
- M\_DISTORTION\_FOLD
  - m\_eng\_distortion.h, 79
- M\_DISTORTION\_TANH
  - m\_eng\_distortion.h, 79
- m\_eng.cpp, 155
  - DEBUG\_PRINT\_MILLIS, 156
  - LED\_BLINK\_MILLIS, 156
  - LOG\_PRINT\_MILLIS, 156
  - main, 157
  - MEM\_REPORT\_MILLIS, 156
  - PRINT\_LOG, 156
  - PROFILER\_PRINT\_MILLIS, 156
  - SCHEDULED\_MAINTAINANCE, 156



- SCHEDULED\_MAINTAINANCE\_MILLIS, 157
- SGTL5000\_CHECK\_PERIOD, 157
- m\_eng.h, 51, 56
  - ALLOW\_PRINTLINES, 53
  - AUDIO\_BLOCK\_MS, 53
  - AUDIO\_BLOCK\_SAMPLES, 53
  - AUDIO\_SAMPLE\_RATE, 53
  - AUDIO\_SAMPLE\_RATE\_EXACT, 53
  - binary\_max, 53
  - binary\_min, 53
  - LL\_FREE, 53
  - LL\_MALLOC, 53
  - LN\_2, 54
  - M\_ENGINE, 54
  - M\_VOICE\_COMMS, 54
  - M\_VOICE\_CXT, 54
  - M\_VOICE\_ERR, 54
  - M\_VOICE\_LOG, 54
  - M\_VOICE\_PL, 54
  - M\_VOICE\_PR, 54
  - M\_VOICE\_PRF, 54
  - M\_VOICE\_TR, 54
  - MS\_TO\_SAMPLES, 55
  - NUM\_MASKS, 55
  - S\_TO\_SAMPLES, 55
  - SAMPLE\_FREQUENCY, 55
  - SAMPLES\_TO\_MS, 55
  - SAMPLES\_TO\_S, 55
  - sqr, 55
  - trig\_transition\_function, 56
- m\_eng\_3\_band\_eq\_str, 8
  - coefs, 8
  - control\_mode, 8
  - filters, 8
  - high, 8
  - low, 8
  - mid, 9
- m\_eng\_3\_band\_splitter\_str, 9
  - filters, 9
- m\_eng\_adaptive\_waveshaper.c, 157
  - calc\_adaptive\_waveshaper, 157
  - init\_adaptive\_waveshaper\_str, 157
- m\_eng\_adaptive\_waveshaper.h, 58, 59
  - ADAPTIVE\_WAVESHAPER\_ENVELOPE\_ATTACK, 58
  - ADAPTIVE\_WAVESHAPER\_ENVELOPE\_RELEASE, 58
  - calc\_adaptive\_waveshaper, 59
  - init\_adaptive\_waveshaper\_str, 59
- m\_eng\_adaptive\_waveshaper\_str, 9
  - coefficient, 10
  - local\_amplitude, 10
  - shape, 10
- M\_ENG\_AMPLIFIER\_DB
  - m\_eng\_buffer\_mixer\_amp.h, 63
- M\_ENG\_AMPLIFIER\_LINEAR
  - m\_eng\_buffer\_mixer\_amp.h, 63
- m\_eng\_amplifier\_str, 10
  - db, 10
  - g, 10
  - gain, 10
  - mode, 10
- m\_eng\_audio\_block.c, 158
- m\_eng\_audio\_block.h, 59, 60
  - AUDIO\_BLOCK\_SAMPLES, 59
- m\_eng\_band\_pass\_filter\_str, 11
  - a0, 11
  - a1, 11
  - a2, 11
  - a3, 11
  - a4, 11
  - bandwidth, 12
  - center, 12
  - x1, 12
  - x2, 12
  - y1, 12
  - y2, 12
- m\_eng\_band\_splitter.c, 158
- m\_eng\_band\_splitter.h, 60, 61
  - calc\_3\_band\_splitter, 60
  - calc\_n\_band\_splitter, 60
  - init\_3\_band\_splitter\_str, 60
  - init\_n\_band\_splitter\_str, 61
  - reconfigure\_3\_band\_splitter, 61
  - reconfigure\_n\_band\_splitter, 61
- m\_eng\_biquad.c, 158
  - calc\_biquad, 158
  - init\_biquad\_str, 158
  - reconfigure\_biquad, 158
- m\_eng\_biquad.h, 61, 62
  - calc\_biquad, 62
  - init\_biquad\_str, 62
  - reconfigure\_biquad, 62
- m\_eng\_biquad\_str, 12
  - a0, 13
  - a1, 13
  - a2, 13
  - a3, 13
  - a4, 13
  - bandwidth, 13
  - cutoff, 13
  - db\_gain, 13
  - type, 14
  - x1, 14
  - x2, 14
  - y1, 14
  - y2, 14
- m\_eng\_buffer\_mixer\_amp.c, 158
  - calc\_amplifier, 159
  - calc\_buffer, 159
  - init\_amplifier\_str, 159
  - reconfigure\_amplifier, 159
- m\_eng\_buffer\_mixer\_amp.h, 62, 64
  - calc\_amplifier, 63
  - calc\_buffer, 63
  - init\_amplifier\_str, 63

- M\_ENG\_AMPLIFIER\_DB, 63
- M\_ENG\_AMPLIFIER\_LINEAR, 63
- reconfigure\_amplifier, 63
- m\_eng\_comms.cpp, 159
  - comms\_fsm\_eng\_state, 161
  - comms\_fsm\_eng\_state\_t, 160
  - esp32\_message\_check\_handle, 161
  - handle\_esp32\_message, 161
  - i2c\_receive\_isr, 161
  - i2c\_request\_isr, 161
  - IDLE, 160
  - init\_esp32\_link, 161
  - m\_message\_sanity\_check, 161
  - message\_pending, 161
  - prev\_response, 161
  - PRINT\_RESPONSE\_BYTES, 160
  - receive\_buffer, 161
  - received, 162
  - received\_length, 162
  - RECIEVING\_NEW\_PARAM\_NAM\_ENG\_LONG, 160
  - response, 162
  - response\_buffer, 162
  - response\_length, 162
  - response\_ready, 162
  - SENDING\_STRING, 160
  - string\_in, 162
  - string\_in\_pos, 162
  - string\_out, 162
  - string\_out\_pos, 162
  - wait\_message, 163
- m\_eng\_comms.h, 64, 65
  - esp32\_message\_check\_handle, 65
  - i2c\_receive\_isr, 65
  - i2c\_request\_isr, 65
  - init\_esp32\_link, 65
  - TEENSY\_I2C\_SLAVE\_ADDR, 64
- m\_eng\_compressor.c, 163
  - calc\_compressor, 163
  - EPSILON, 163
  - init\_compressor\_str, 163
  - reconfigure\_compressor, 163
- m\_eng\_compressor.h, 65, 66
  - calc\_compressor, 66
  - init\_compressor\_str, 66
  - reconfigure\_compressor, 66
- m\_eng\_compressor\_str, 14
  - alpha, 15
  - attack, 15
  - e\_final, 15
  - ratio, 15
  - release, 15
  - rho, 15
  - threshold, 15
- m\_eng\_context, 15
  - active\_profile, 16
  - dc\_blocker\_avg, 16
  - declick\_buffer, 16
  - input\_lpf, 16
  - n\_profiles, 16
  - new\_profile, 16
  - output\_amp, 16
  - output\_hpf, 17
  - prev\_block, 17
  - profile\_array\_size, 17
  - profile\_maintainance\_index, 17
  - profile\_switch\_progress, 17
  - profile\_switch\_samples, 17
  - profile\_switch\_triggered, 17
  - profile\_switch\_type, 17
  - profiles, 17
  - profiles\_switching, 17
  - runs, 18
  - status\_flags, 18
- m\_eng\_context.c, 164
  - AVG\_DURATION\_UPDATE\_COEF, 165
  - cxt\_append\_transformer\_to\_profile, 165
  - cxt\_get\_back\_parameter\_by\_id, 165
  - cxt\_get\_front\_parameter\_by\_id, 165
  - cxt\_get\_n\_profile\_transformers, 165
  - cxt\_get\_n\_transformer\_params, 165
  - cxt\_get\_n\_transformer\_settings, 165
  - cxt\_get\_parameter\_by\_id, 166
  - cxt\_get\_setting\_by\_id, 166
  - cxt\_get\_tid\_by\_pos, 166
  - cxt\_get\_transformer\_by\_id, 166
  - cxt\_get\_transformer\_type, 166
  - cxt\_insert\_transformer\_to\_profile, 166
  - cxt\_move\_transformer, 167
  - cxt\_prepend\_transformer\_to\_profile, 167
  - cxt\_process, 167
  - cxt\_profile\_id\_valid, 167
  - cxt\_remove\_transformer\_from\_profile, 167
  - cxt\_run\_scheduled\_maintainance, 167
  - cxt\_set\_active\_profile, 167
  - cxt\_switch\_to\_profile, 168
  - cxt\_transformer\_id\_valid, 168
  - cxt\_update\_parameter\_value\_by\_id, 168
  - cxt\_update\_setting\_value\_by\_id, 168
  - init\_m\_eng\_context, 168
  - m\_eng\_context\_new\_profile, 168
  - m\_eng\_safe\_reboot, 168
  - pcxt\_profile\_id\_valid, 169
  - reset\_context, 169
- m\_eng\_context.h, 66, 72
  - CLICK\_SLOPE\_THRESHOLD, 68
  - cxt\_append\_transformer\_to\_profile, 68
  - cxt\_get\_n\_profile\_transformers, 68
  - cxt\_get\_n\_transformer\_params, 69
  - cxt\_get\_n\_transformer\_settings, 69
  - cxt\_get\_parameter\_by\_id, 69
  - cxt\_get\_setting\_by\_id, 69
  - cxt\_get\_tid\_by\_pos, 69
  - cxt\_get\_transformer\_by\_id, 69
  - cxt\_get\_transformer\_type, 69
  - cxt\_insert\_transformer\_to\_profile, 70

- cxt\_move\_transformer, 70
- cxt\_parameter\_id\_valid, 70
- cxt\_prepend\_transformer\_to\_profile, 70
- cxt\_process, 70
- cxt\_profile\_id\_valid, 70
- cxt\_remove\_transformer\_from\_profile, 70
- cxt\_run\_scheduled\_maintenance, 71
- cxt\_set\_active\_profile, 71
- cxt\_switch\_to\_profile, 71
- cxt\_transformer\_id\_valid, 71
- cxt\_update\_parameter\_value\_by\_id, 71
- cxt\_update\_setting\_value\_by\_id, 71
- DC\_BLOCKER\_ALPHA, 68
- DECLICK\_BUFSIZE, 68
- global\_cxt, 72
- init\_m\_eng\_context, 71
- m\_eng\_context\_new\_profile, 72
- m\_eng\_safe\_reboot, 72
- M\_PROFILE\_SWITCH\_SAMPLES, 68
- PROFILE\_ARRAY\_INITIAL\_SIZE, 68
- PROFILES\_MALLOC\_CHUNK\_SIZE, 68
- reset\_context, 72
- SILENCE\_BLOCKS\_THRESHOLD, 68
- SILENCE\_ENERGY\_THRESHOLD, 68
- m\_eng\_context\_new\_profile
  - m\_eng\_context.c, 168
  - m\_eng\_context.h, 72
- m\_eng\_debugging.c, 169
  - full\_debug\_print, 169
  - print\_binary, 169
  - print\_context\_info, 169
  - print\_pipeline\_info, 170
  - print\_profile\_info, 170
  - print\_transformer\_info, 170
- m\_eng\_debugging.h, 73, 74
  - full\_debug\_print, 74
  - print\_context\_info, 74
  - print\_pipeline\_info, 74
  - print\_profile\_info, 74
  - print\_transformer\_info, 74
- m\_eng\_delay.c, 170
  - calc\_delay, 170
  - init\_delay\_str, 170
  - reconfigure\_delay, 171
- m\_eng\_delay.h, 75
  - calc\_delay, 75
  - init\_delay\_str, 75
  - reconfigure\_delay, 75
- m\_eng\_delay\_buffer.c, 171
  - init\_delay\_buffer, 171
  - m\_delay\_buffer\_advance, 171
  - m\_delay\_buffer\_get\_delayed\_sample, 171
  - m\_delay\_buffer\_get\_delayed\_sample\_ptr, 171
  - m\_delay\_buffer\_get\_fractional\_delayed\_sample, 172
  - m\_delay\_buffer\_resize\_milliseconds, 172
  - m\_delay\_buffer\_resize\_samples, 172
  - m\_delay\_buffer\_resize\_seconds, 172
  - m\_delay\_buffer\_tick, 172
- m\_eng\_delay\_buffer.h, 76, 77
  - init\_delay\_buffer, 76
  - m\_delay\_buffer\_advance, 76
  - m\_delay\_buffer\_get\_delayed\_sample, 76
  - m\_delay\_buffer\_get\_delayed\_sample\_ptr, 76
  - m\_delay\_buffer\_get\_fractional\_delayed\_sample, 76
  - m\_delay\_buffer\_resize\_milliseconds, 77
  - m\_delay\_buffer\_resize\_samples, 77
  - m\_delay\_buffer\_resize\_seconds, 77
  - m\_delay\_buffer\_tick, 77
- m\_eng\_delay\_str, 18
  - buf, 18
  - delay\_gain, 18
  - delay\_samples, 18
  - g, 18
  - note, 19
  - tempo, 19
- m\_eng\_dirty\_octave.c, 172
  - calc\_dirty\_octave, 173
  - init\_dirty\_octave\_str, 173
  - reconfigure\_dirty\_octave, 173
- m\_eng\_dirty\_octave.h, 78
  - calc\_dirty\_octave, 78
  - init\_dirty\_octave\_str, 78
  - reconfigure\_dirty\_octave, 78
- m\_eng\_dirty\_octave\_str, 19
  - dc\_average, 19
  - fuzz, 19
  - last\_out\_sample, 19
  - lpf\_alpha, 19
- m\_eng\_disable\_software\_interrupts
  - m\_eng\_update.h, 150
- m\_eng\_distortion.c, 173
  - calc\_distortion, 173
  - init\_distortion\_str, 173
  - reconfigure\_distortion, 173
- m\_eng\_distortion.h, 79, 80
  - calc\_distortion, 80
  - init\_distortion\_str, 80
  - M\_DISTORTION\_ARCTAN, 79
  - M\_DISTORTION\_CLIP, 79
  - M\_DISTORTION\_FOLD, 79
  - M\_DISTORTION\_TANH, 79
  - reconfigure\_distortion, 80
  - USE\_GLOBAL\_TEMP\_BUFFERS, 79
- m\_eng\_distortion\_str, 20
  - bass\_cutoff, 20
  - bass\_mix, 20
  - dist, 20
  - function, 20
  - low\_pass, 20
  - wet\_mix, 20
- m\_eng\_enable\_software\_interrupts
  - m\_eng\_update.h, 150
- m\_eng\_envelope.c, 174
  - calc\_envelope, 174

- init\_envelope\_str, 174
- reconfigure\_envelope, 174
- m\_eng\_envelope.h, 80, 81
  - calc\_envelope, 81
  - init\_envelope\_str, 81
  - reconfigure\_envelope, 81
- m\_eng\_envelope\_str, 21
  - alpha, 21
  - chunk\_size, 21
  - e, 21
  - filter, 21
  - max\_center, 21
  - min\_center, 22
  - sensitivity, 22
  - smoothness, 22
  - speed, 22
  - width, 22
- M\_ENG\_EQ\_CONTROL\_DB\_GAIN
  - m\_eng\_equaliser.h, 82
- M\_ENG\_EQ\_CONTROL\_DIRECT
  - m\_eng\_equaliser.h, 82
- m\_eng\_equaliser.c, 174
  - calc\_3\_band\_eq, 175
  - init\_3\_band\_eq\_str, 175
  - reconfigure\_3\_band\_eq, 175
- m\_eng\_equaliser.h, 82, 83
  - calc\_3\_band\_eq, 82
  - init\_3\_band\_eq\_str, 82
  - M\_ENG\_EQ\_CONTROL\_DB\_GAIN, 82
  - M\_ENG\_EQ\_CONTROL\_DIRECT, 82
  - reconfigure\_3\_band\_eq, 82
- m\_eng\_flanger.c, 175
  - calc\_flanger, 175
  - free\_flanger\_struct, 175
  - init\_flanger\_str, 176
  - reconfigure\_flanger, 176
- m\_eng\_flanger.h, 83, 84
  - calc\_flanger, 83
  - init\_flanger\_str, 83
  - reconfigure\_flanger, 83
- m\_eng\_flanger\_str, 22
  - buf, 23
  - d, 23
  - depth, 23
  - dry\_mix, 23
  - mix, 23
  - note, 23
  - period, 23
  - r, 23
  - range, 23
  - s, 23
  - t, 23
  - tempo, 24
  - wet\_mix, 24
- m\_eng\_flops.h, 84, 85
  - RESTRICT, 84
- m\_eng\_globals.c, 176
  - cpu\_cycles\_total, 177
  - cpu\_cycles\_total\_max, 177
  - current\_cycle, 176
  - cycles\_to\_seconds, 176
  - cycles\_upper, 177
  - global\_cxt, 177
  - memory\_used, 177
  - memory\_used\_max, 177
  - trace\_depth, 177
  - update\_scheduled, 177
- m\_eng\_globals.h, 87, 89
  - cpu\_cycles\_total, 88
  - cpu\_cycles\_total\_max, 88
  - current\_cycle, 88
  - cycles\_to\_seconds, 88
  - memory\_used, 88
  - memory\_used\_max, 88
  - trace\_depth, 88
  - update\_scheduled, 89
  - update\_upper\_cycles, 88
- m\_eng\_high\_pass\_filter\_str, 24
  - a0, 24
  - a1, 24
  - a2, 24
  - a3, 25
  - a4, 25
  - cutoff\_frequency, 25
  - x1, 25
  - x2, 25
  - y1, 25
  - y2, 25
- m\_eng\_i2s\_dma.cpp, 177
  - \_\_attribute\_\_, 178
  - configure\_i2s\_dma, 178
  - i2s\_in\_block\_left, 180
  - i2s\_in\_block\_offset, 180
  - i2s\_in\_block\_right, 180
  - i2s\_in\_dma, 178
  - i2s\_in\_transmit, 178
  - i2s\_in\_update\_responsibility, 180
  - i2s\_input\_blocks, 180
  - i2s\_input\_update, 179
  - i2s\_out\_block\_left\_1st, 180
  - i2s\_out\_block\_left\_2nd, 180
  - i2s\_out\_block\_left\_offset, 180
  - i2s\_out\_block\_right\_1st, 180
  - i2s\_out\_block\_right\_2nd, 180
  - i2s\_out\_block\_right\_offset, 180
  - i2s\_out\_dma, 179
  - i2s\_out\_update\_responsibility, 181
  - i2s\_output\_blocks, 181
  - i2s\_output\_transmit\_mono\_float, 179
  - i2s\_output\_transmit\_mono\_int, 179
  - i2s\_output\_update, 179
  - init\_i2s\_dma, 179
  - m\_eng\_i2s\_input\_isr, 179
  - m\_eng\_i2s\_output\_isr, 179
- m\_eng\_i2s\_dma.h, 89, 91
  - i2s\_input\_blocks, 90

- i2s\_input\_update, 90
- i2s\_output\_transmit\_mono\_float, 90
- i2s\_output\_transmit\_mono\_int, 90
- i2s\_output\_update, 90
- init\_i2s\_dma, 90
- m\_eng\_i2s\_input\_isr, 90
- m\_eng\_i2s\_output\_isr, 90
- raw\_sample\_t, 90
- m\_eng\_i2s\_input\_isr
  - m\_eng\_i2s\_dma.cpp, 179
  - m\_eng\_i2s\_dma.h, 90
- m\_eng\_i2s\_output\_isr
  - m\_eng\_i2s\_dma.cpp, 179
  - m\_eng\_i2s\_dma.h, 90
- m\_eng\_init\_profiler
  - m\_eng\_logging.cpp, 184
- m\_eng\_linkowitz\_riley.c, 181
  - calc\_lr\_high\_pass\_filter, 181
  - calc\_lr\_low\_pass\_filter, 181
  - init\_lr\_high\_pass\_filter\_str, 181
  - init\_lr\_low\_pass\_filter\_str, 182
  - reconfigure\_lr\_high\_pass\_filter, 182
  - reconfigure\_lr\_low\_pass\_filter, 182
- m\_eng\_linkowitz\_riley.h, 91, 92
  - calc\_lr\_high\_pass\_filter, 91
  - calc\_lr\_low\_pass\_filter, 91
  - init\_lr\_high\_pass\_filter\_str, 91
  - init\_lr\_low\_pass\_filter\_str, 92
  - reconfigure\_lr\_high\_pass\_filter, 92
  - reconfigure\_lr\_low\_pass\_filter, 92
- m\_eng\_log
  - m\_eng\_logging.h, 99
- M\_ENG\_LOG\_ENTRIES\_N
  - m\_eng\_logging.cpp, 183
- m\_eng\_log\_entry, 25
  - cycle, 26
  - data, 26
  - data\_type, 26
  - file\_name, 26
  - function, 26
  - line, 26
  - message, 26
  - trace\_depth, 26
  - type, 27
- M\_ENG\_LOG\_ENTRY\_ERROR
  - m\_eng\_logging.h, 94
- M\_ENG\_LOG\_ENTRY\_MESSAGE
  - m\_eng\_logging.h, 94
- M\_ENG\_LOG\_ENTRY\_RETURN
  - m\_eng\_logging.h, 94
- M\_ENG\_LOG\_ENTRY\_RETURN\_ERR
  - m\_eng\_logging.h, 94
- M\_ENG\_LOG\_ENTRY\_RETURN\_INT
  - m\_eng\_logging.h, 95
- M\_ENG\_LOG\_ENTRY\_RETURN\_PTR
  - m\_eng\_logging.h, 95
- M\_ENG\_LOG\_ERROR
  - m\_eng\_logging.h, 95
- m\_eng\_log\_error\_code
  - m\_eng\_logging.cpp, 184
  - m\_eng\_logging.h, 99
- M\_ENG\_LOG\_ERRORS
  - m\_eng\_logging.h, 95
- M\_ENG\_LOG EVERYTHING
  - m\_eng\_logging.h, 95
- M\_ENG\_LOG\_INDENT\_TRACE
  - m\_eng\_logging.h, 95
- M\_ENG\_LOG\_LEVEL
  - m\_eng\_logging.h, 95
- m\_eng\_log\_message
  - m\_eng\_logging.cpp, 184
  - m\_eng\_logging.h, 99
- M\_ENG\_LOG\_RETURN
  - m\_eng\_logging.h, 95
- m\_eng\_log\_return\_
  - m\_eng\_logging.cpp, 184
  - m\_eng\_logging.h, 99
- m\_eng\_log\_return\_err
  - m\_eng\_logging.cpp, 184
  - m\_eng\_logging.h, 99
- M\_ENG\_LOG\_RETURN\_ERR\_CODE
  - m\_eng\_logging.h, 95
- M\_ENG\_LOG\_RETURN\_INT
  - m\_eng\_logging.h, 96
- m\_eng\_log\_return\_int
  - m\_eng\_logging.cpp, 184
  - m\_eng\_logging.h, 99
- M\_ENG\_LOG\_RETURN\_PTR
  - m\_eng\_logging.h, 96
- m\_eng\_log\_return\_ptr
  - m\_eng\_logging.cpp, 185
  - m\_eng\_logging.h, 99
- M\_ENG\_LOG\_RETURNS
  - m\_eng\_logging.h, 96
- M\_ENG\_LOG SPIKES
  - m\_eng\_logging.h, 96
- M\_ENG\_LOG\_TRACE
  - m\_eng\_logging.h, 96
- m\_eng\_logging.cpp, 182
  - format\_log\_entry, 184
  - LOG\_ENTRIES\_PRINT\_BUF\_LEN, 183
  - m\_eng\_init\_profiler, 184
  - M\_ENG\_LOG\_ENTRIES\_N, 183
  - m\_eng\_log\_error\_code, 184
  - m\_eng\_log\_message, 184
  - m\_eng\_log\_return\_, 184
  - m\_eng\_log\_return\_err, 184
  - m\_eng\_log\_return\_int, 184
  - m\_eng\_log\_return\_ptr, 185
  - m\_eng\_print\_flush\_log, 185
  - M\_ENG\_PROFILER\_ARRAY\_N, 183
  - m\_eng\_profiler\_log\_entry, 185
  - m\_eng\_profiler\_log\_return, 185
  - m\_eng\_profiler\_print, 185
  - M\_ENG\_PROFILER\_RA\_CYCLES\_ALPHA, 183
  - m\_eng\_profiler\_sort, 185

- m\_eng\_trace\_log\_begin, 185
  - m\_eng\_trace\_log\_return, 185
  - MESSAGE\_BEGIN\_COL, 183
- m\_eng\_logging.h, 93, 100
  - CYCLES\_TO\_SECONDS, 94
  - FUNCTION\_START, 94
  - m\_eng\_log, 99
  - M\_ENG\_LOG\_ENTRY\_ERROR, 94
  - M\_ENG\_LOG\_ENTRY\_MESSAGE, 94
  - M\_ENG\_LOG\_ENTRY\_RETURN, 94
  - M\_ENG\_LOG\_ENTRY\_RETURN\_ERR, 94
  - M\_ENG\_LOG\_ENTRY\_RETURN\_INT, 95
  - M\_ENG\_LOG\_ENTRY\_RETURN\_PTR, 95
  - M\_ENG\_LOG\_ERROR, 95
  - m\_eng\_log\_error\_code, 99
  - M\_ENG\_LOG\_ERRORS, 95
  - M\_ENG\_LOG\_EVERYTHING, 95
  - M\_ENG\_LOG\_INDENT\_TRACE, 95
  - M\_ENG\_LOG\_LEVEL, 95
  - m\_eng\_log\_message, 99
  - M\_ENG\_LOG\_RETURN, 95
  - m\_eng\_log\_return\_, 99
  - m\_eng\_log\_return\_err, 99
  - M\_ENG\_LOG\_RETURN\_ERR\_CODE, 95
  - M\_ENG\_LOG\_RETURN\_INT, 96
  - m\_eng\_log\_return\_int, 99
  - M\_ENG\_LOG\_RETURN\_PTR, 96
  - m\_eng\_log\_return\_ptr, 99
  - M\_ENG\_LOG\_RETURNS, 96
  - M\_ENG\_LOG\_SPIKES, 96
  - M\_ENG\_LOG\_TRACE, 96
  - m\_eng\_print\_flush\_log, 100
  - m\_eng\_print\_log, 100
  - M\_ENG\_PROFILER\_LOG\_ENTRY, 96
  - M\_ENG\_PROFILER\_LOG\_RETURN, 96
  - M\_ENG\_TRACE\_FUNCTION\_ENTER, 96
  - M\_ENG\_TRACE\_FUNCTION\_RETURN, 96
  - m\_eng\_trace\_log\_begin, 100
  - M\_ENG\_TRACE\_LOG\_ENTRY, 96
  - M\_ENG\_TRACE\_LOG\_RETURN, 97
  - m\_eng\_trace\_log\_return, 100
  - M\_LOG, 97
  - M\_LOG\_ERROR, 97
  - RETURN, 97
  - RETURN\_ERR\_CODE, 97
  - RETURN\_INT, 97
  - RETURN\_NEG\_ERR\_CODE, 97
  - RETURN\_PTR, 98
  - RETURN\_VOID, 98
  - SECONDS\_TO\_CYCLES, 98
  - STR, 98
  - XSTR, 98
- m\_eng\_low\_end\_compressor.c, 186
  - calc\_low\_end\_compressor, 186
  - init\_low\_end\_compressor\_str, 186
  - reconfigure\_low\_end\_compressor, 186
- m\_eng\_low\_end\_compressor.h, 102, 103
  - calc\_low\_end\_compressor, 103
  - init\_low\_end\_compressor\_str, 103
  - reconfigure\_low\_end\_compressor, 103
- m\_eng\_low\_end\_compressor\_str, 27
  - bass\_comp, 27
  - low\_pass, 27
  - mid\_pass, 27
  - mids\_comp, 27
- m\_eng\_low\_pass\_filter\_str, 28
  - a0, 28
  - a1, 28
  - a2, 28
  - a3, 28
  - a4, 28
  - cutoff\_frequency, 28
  - x1, 28
  - x2, 29
  - y1, 29
  - y2, 29
- m\_eng\_memcpy\_audio.h, 103, 104
  - memcpy\_tointerleaveL, 104
  - memcpy\_tointerleaveLR, 104
  - memcpy\_tointerleaveQuad, 104
  - memcpy\_tointerleaveR, 104
- m\_eng\_mempool.c, 186
  - allocate\_buffer, 187
  - buffer\_buffer, 188
  - buffer\_head, 188
  - buffer\_pool, 188
  - BUFFER\_QUEUE\_STATIC, 187
  - buffer\_tail, 188
  - head, 188
  - init\_mem\_pools, 187
  - mem\_pools\_initialised, 188
  - MEMPOOL\_MALLOC\_TRIES, 187
  - print\_mempool\_info, 187
  - release\_buffer, 188
  - sink\_buffer, 188
  - tail, 188
  - zero\_buffer, 188
- m\_eng\_mempool.h, 105, 106
  - allocate\_buffer, 106
  - init\_mem\_pools, 106
  - M\_BUFFER\_POOL\_SIZE, 105
  - MAX\_AUDIO\_MEMORY, 105
  - MEM\_SIZE, 105
  - print\_mempool\_info, 106
  - release\_buffer, 106
  - sink\_buffer, 106
  - zero\_buffer, 106
- m\_eng\_mixer\_str, 29
  - ratio, 29
- m\_eng\_n\_band\_splitter\_str, 29
  - filters, 30
- m\_eng\_noise\_suppressor.c, 189
  - calc\_noise\_suppressor, 189
  - init\_noise\_suppressor\_str, 189
  - reconfigure\_noise\_suppressor, 189
- m\_eng\_noise\_suppressor.h, 107

- calc\_noise\_suppressor, 107
- init\_noise\_suppressor\_str, 107
- reconfigure\_noise\_suppressor, 107
- m\_eng\_noise\_suppressor\_str, 30
  - e\_final, 30
  - max\_reduction, 30
  - r, 30
  - ratio, 30
  - threshold, 30
- m\_eng\_parameter.c, 189
  - init\_parameter, 190
  - init\_setting, 190
  - update\_setting, 190
- m\_eng\_parameter.h, 108, 109
  - DEFAULT\_MAX\_JUMP, 108
  - init\_parameter, 109
  - init\_setting, 109
  - PARAM\_NAM\_ENG\_MAX\_LEN, 108
  - PARAMETER\_UPDATE\_BIBLOCK\_LINEAR, 108
  - PARAMETER\_UPDATE\_INSTANT, 108
  - PARAMETER\_UPDATE\_MONOBLOCK\_LINEAR, 108
  - PARAMETER\_UPDATE\_QUADBLOCK\_LINEAR, 108
- m\_eng\_pass\_filter.c, 190
  - calc\_band\_pass\_filter, 190
  - calc\_high\_pass\_filter, 190
  - calc\_low\_pass\_filter, 191
  - init\_band\_pass\_filter\_str, 191
  - init\_high\_pass\_filter\_str, 191
  - init\_low\_pass\_filter\_str, 191
  - reconfigure\_band\_pass\_filter, 191
  - reconfigure\_high\_pass\_filter, 191
  - reconfigure\_low\_pass\_filter, 191
- m\_eng\_pass\_filter.h, 109, 111
  - calc\_band\_pass\_filter, 110
  - calc\_high\_pass\_filter, 110
  - calc\_low\_pass\_filter, 110
  - init\_band\_pass\_filter\_str, 110
  - init\_high\_pass\_filter\_str, 110
  - init\_low\_pass\_filter\_str, 110
  - reconfigure\_band\_pass\_filter, 111
  - reconfigure\_high\_pass\_filter, 111
  - reconfigure\_low\_pass\_filter, 111
- m\_eng\_percussifier.c, 192
  - calc\_percussifier, 192
  - init\_percussifier\_str, 192
  - reconfigure\_percussifier, 192
- m\_eng\_percussifier.h, 112, 113
  - calc\_percussifier, 113
  - init\_percussifier\_str, 113
  - PERCUSSIFIER\_FADE\_IN, 112
  - PERCUSSIFIER\_FADE\_OUT, 112
  - PERCUSSIFIER\_HOLD, 112
  - PERCUSSIFIER\_MUTE, 112
  - PERCUSSIFIER\_REFRACTORY, 112
  - reconfigure\_percussifier, 113
- m\_eng\_percussifier\_str, 31
- alpha\_long, 31
- alpha\_short, 31
- arm\_threshold, 31
- decay\_rate, 32
- fade\_alpha, 32
- fade\_in, 32
- fade\_in\_samples, 32
- fade\_out, 32
- gain, 32
- hold\_samples, 32
- note, 32
- r, 32
- refractory\_period, 32
- refractory\_samples, 33
- rms\_long, 33
- rms\_short, 33
- state, 33
- tempo, 33
- timer, 33
- trigger\_threshold, 33
- m\_eng\_pipeline.c, 192
  - clone\_pipeline, 193
  - compute\_pipeline, 193
  - gut\_pipeline, 193
  - init\_pipeline, 194
  - pipeline\_append\_transformer, 194
  - pipeline\_append\_transformer\_type, 194
  - pipeline\_change\_transformer\_setting, 194
  - pipeline\_clone\_transformer\_into\_position, 194
  - pipeline\_compare, 194
  - pipeline\_expand\_transformer\_array, 194
  - pipeline\_expand\_transformer\_array\_to, 194
  - pipeline\_get\_transformer\_by\_id, 195
  - pipeline\_get\_transformer\_position, 195
  - pipeline\_insert\_transformer, 195
  - pipeline\_insert\_transformer\_type, 195
  - pipeline\_move\_transformer, 195
  - pipeline\_prepend\_transformer, 195
  - pipeline\_prepend\_transformer\_type, 195
  - pipeline\_print\_transformer\_array, 196
  - pipeline\_remove\_transformer, 196
  - pipeline\_swap\_transformers, 196
  - pipeline\_update\_transition\_policy, 196
  - pipeline\_valid, 196
  - TRANSFORMERS\_MALLOC\_CHUNK\_SIZE, 193
- m\_eng\_pipeline.h, 114, 118
  - clone\_pipeline, 114
  - compute\_pipeline, 114
  - gut\_pipeline, 115
  - init\_pipeline, 115
  - INITIAL\_TRANSFORMER\_ARRAY\_LENGTH, 114
  - pipeline\_append\_transformer, 115
  - pipeline\_append\_transformer\_type, 115
  - pipeline\_change\_transformer\_setting, 115
  - pipeline\_clone\_transformer\_into\_position, 115
  - pipeline\_compare, 115
  - pipeline\_expand\_transformer\_array, 116
  - pipeline\_expand\_transformer\_array\_to, 116



- pipeline\_get\_transformer\_by\_id, 116
- pipeline\_get\_transformer\_position, 116
- pipeline\_insert\_transformer, 116
- pipeline\_insert\_transformer\_type, 116
- pipeline\_move\_transformer, 116
- pipeline\_prepend\_transformer, 117
- pipeline\_prepend\_transformer\_type, 117
- pipeline\_remove\_transformer, 117
- pipeline\_swap\_transformers, 117
- pipeline\_update\_transition\_policy, 117
- pipeline\_valid, 117
- m\_eng\_pipeline\_mod.c, 196
  - apply\_pipeline\_mod, 197
  - create\_pipeline\_mod\_append\_transformer, 197
  - create\_pipeline\_mod\_change\_transformer\_setting, 197
  - create\_pipeline\_mod\_move\_transformer, 197
  - create\_pipeline\_mod\_remove\_transformer, 197
  - IMPLEMENT\_LINKED\_LIST, 197
  - pipeline\_mod\_type\_string, 197
- m\_eng\_pipeline\_mod.h, 118, 120
  - apply\_pipeline\_mod, 119
  - create\_pipeline\_mod\_append\_transformer, 119
  - create\_pipeline\_mod\_change\_transformer\_setting, 119
  - create\_pipeline\_mod\_move\_transformer, 119
  - create\_pipeline\_mod\_remove\_transformer, 119
  - DECLARE\_LINKED\_LIST, 120
  - init\_pipeline\_mod, 120
  - PIPELINE\_MOD\_APPEND\_TRANSFORMER, 119
  - PIPELINE\_MOD\_CHANGE\_TRANSFORMER\_SETTING, 119
  - PIPELINE\_MOD\_MOVE\_TRANSFORMER, 119
  - PIPELINE\_MOD\_REMOVE\_TRANSFORMER, 119
  - pipeline\_mod\_type\_string, 120
- m\_eng\_print\_flush\_log
  - m\_eng\_logging.cpp, 185
  - m\_eng\_logging.h, 100
- m\_eng\_print\_log
  - m\_eng\_logging.h, 100
- m\_eng\_printf.cpp, 197
  - m\_mute\_voice, 198
  - m\_printf, 198
  - m\_unmute\_voice, 198
  - m\_voice\_printf, 198
  - pretty\_print\_block, 199
  - pretty\_print\_block\_float, 199
  - serial\_print\_blocks, 199
  - SPACING, 198
  - voice\_colour\_table, 199
- m\_eng\_printf.h, 121, 122
  - m\_mute\_voice, 121
  - m\_printf, 121
  - m\_unmute\_voice, 121
  - m\_voice\_printf, 121
  - pretty\_print\_block, 121
  - pretty\_print\_block\_float, 121
  - serial\_print\_blocks, 121
- m\_eng\_profile, 33
  - active, 34
  - back\_pipeline, 34
  - back\_pipeline\_warmed\_up, 34
  - blocked\_jobs, 34
  - front\_pipeline, 34
  - jobs, 34
  - output\_amp, 34
  - pipeline\_swap\_progress, 35
  - pipeline\_swap\_samples, 35
  - pipeline\_swap\_type, 35
  - pipelines\_swapping, 35
  - prev\_block, 35
  - runs, 35
  - transition\_policy, 35
- m\_eng\_profile.c, 199
  - init\_profile, 200
  - nullify\_profile, 200
  - profile\_apply\_pipeline\_mod, 200
  - profile\_print\_job\_list, 201
  - profile\_print\_ujob\_list, 201
  - profile\_process, 201
  - profile\_regenerate\_back\_pipeline, 201
  - profile\_scheduled\_maintenance, 201
  - profile\_trigger\_pipeline\_swap, 201
  - profile\_update, 201
- m\_eng\_profile.h, 122, 124
  - init\_bypass\_profile, 122
  - init\_profile, 122
  - nullify\_profile, 123
  - profile\_apply\_pipeline\_mod, 123
  - profile\_process, 123
  - profile\_scheduled\_maintenance, 123
  - profile\_trigger\_pipeline\_swap, 123
  - profile\_update, 124
- M\_ENG\_PROFILER\_ARRAY\_N
  - m\_eng\_logging.cpp, 183
- m\_eng\_profiler\_entry, 35
  - calls, 36
  - function\_name, 36
  - open\_cycle, 36
  - peak\_cycles, 36
  - ra\_cycles, 36
  - total\_cycles, 36
- M\_ENG\_PROFILER\_LOG\_ENTRY
  - m\_eng\_logging.h, 96
- m\_eng\_profiler\_log\_entry
  - m\_eng\_logging.cpp, 185
- M\_ENG\_PROFILER\_LOG\_RETURN
  - m\_eng\_logging.h, 96
- m\_eng\_profiler\_log\_return
  - m\_eng\_logging.cpp, 185
- m\_eng\_profiler\_print
  - m\_eng\_logging.cpp, 185
- M\_ENG\_PROFILER\_RA\_CYCLES\_ALPHA
  - m\_eng\_logging.cpp, 183
- m\_eng\_profiler\_sort



- m\_eng\_logging.cpp, 185
- m\_eng\_safe\_reboot
  - m\_eng\_context.c, 168
  - m\_eng\_context.h, 72
- m\_eng\_sgtl5000.cpp, 202
  - calc\_vol, 202
  - sgtl5000\_adc\_high\_pass\_filter\_disable, 202
  - sgtl5000\_adc\_high\_pass\_filter\_enable, 202
  - sgtl5000\_adc\_high\_pass\_filter\_freeze, 202
  - sgtl5000\_automate, 203
  - sgtl5000\_dap\_audio\_eq\_band, 203
  - sgtl5000\_enable, 203
  - sgtl5000\_eq\_select, 203
  - sgtl5000\_kill\_automation, 203
  - sgtl5000\_line\_in\_level, 203
  - sgtl5000\_line\_out\_level, 203
  - sgtl5000\_modify\_reg, 203
  - sgtl5000\_mute\_headphone, 203
  - sgtl5000\_mute\_line\_out, 204
  - sgtl5000\_read\_reg, 204
  - sgtl5000\_set\_address, 204
  - sgtl5000\_start, 204
  - sgtl5000\_unmute\_headphone, 204
  - sgtl5000\_unmute\_line\_out, 204
  - sgtl5000\_volum\_eng\_integer, 204
  - sgtl5000\_volume, 204
  - sgtl5000\_write\_reg, 204
- m\_eng\_sgtl5000.h, 124, 128
  - calc\_vol, 125
  - sgtl5000\_adc\_high\_pass\_filter\_disable, 125
  - sgtl5000\_adc\_high\_pass\_filter\_enable, 125
  - sgtl5000\_adc\_high\_pass\_filter\_freeze, 125
  - sgtl5000\_automate, 125
  - sgtl5000\_dap\_audio\_eq\_band, 125
  - sgtl5000\_enable, 126
  - sgtl5000\_healthy, 126
  - sgtl5000\_kill\_automation, 126
  - sgtl5000\_line\_in\_level, 126
  - sgtl5000\_line\_out\_level, 126
  - sgtl5000\_mic\_gain, 126
  - sgtl5000\_modify\_reg, 126
  - sgtl5000\_mute\_headphone, 126
  - sgtl5000\_mute\_line\_out, 126
  - sgtl5000\_read\_reg, 127
  - sgtl5000\_set\_address, 127
  - sgtl5000\_set\_master\_mode, 127
  - sgtl5000\_soft\_reboot, 127
  - sgtl5000\_start, 127
  - sgtl5000\_unmute\_headphone, 127
  - sgtl5000\_unmute\_line\_out, 127
  - sgtl5000\_volum\_eng\_integer, 127
  - sgtl5000\_volume, 127
  - sgtl5000\_write\_reg, 128
- m\_eng\_sgtl5000\_defs.h, 128, 137
  - AUDIO\_HEADPHONE\_DAC, 130
  - AUDIO\_HEADPHONE\_LINEIN, 130
  - CHIP\_ADCDAC\_CTRL, 130
  - CHIP\_ANA\_ADC\_CTRL, 130
  - CHIP\_ANA\_CTRL, 130
  - CHIP\_ANA\_HP\_CTRL, 130
  - CHIP\_ANA\_POWER, 130
  - CHIP\_ANA\_STATUS, 130
  - CHIP\_ANA\_TEST1, 130
  - CHIP\_ANA\_TEST2, 131
  - CHIP\_CLK\_CTRL, 131
  - CHIP\_CLK\_TOP\_CTRL, 131
  - CHIP\_DAC\_VOL, 131
  - CHIP\_DIG\_POWER, 131
  - CHIP\_I2S\_CTRL, 131
  - CHIP\_ID, 131
  - CHIP\_LINE\_OUT\_CTRL, 131
  - CHIP\_LINE\_OUT\_VOL, 131
  - CHIP\_LINREG\_CTRL, 131
  - CHIP\_MIC\_CTRL, 132
  - CHIP\_PAD\_STRENGTH, 132
  - CHIP\_PLL\_CTRL, 132
  - CHIP\_REF\_CTRL, 132
  - CHIP\_SHORT\_CTRL, 132
  - CHIP\_SSS\_CTRL, 132
  - DAP\_AUDIO\_EQ, 132
  - DAP\_AUDIO\_EQ\_BAND1, 132
  - DAP\_AUDIO\_EQ\_BAND2, 132
  - DAP\_AUDIO\_EQ\_BAND3, 132
  - DAP\_AUDIO\_EQ\_BASS\_BAND0, 133
  - DAP\_AUDIO\_EQ\_TREBLE\_BAND4, 133
  - DAP\_AVC\_ATTACK, 133
  - DAP\_AVC\_CTRL, 133
  - DAP\_AVC\_DECAY, 133
  - DAP\_AVC\_THRESHOLD, 133
  - DAP\_BASS\_ENHANCE, 133
  - DAP\_BASS\_ENHANCE\_CTRL, 133
  - DAP\_COEF\_WR\_A1\_LSB, 133
  - DAP\_COEF\_WR\_A1\_MSB, 133
  - DAP\_COEF\_WR\_A2\_LSB, 134
  - DAP\_COEF\_WR\_A2\_MSB, 134
  - DAP\_COEF\_WR\_B0\_LSB, 134
  - DAP\_COEF\_WR\_B0\_MSB, 134
  - DAP\_COEF\_WR\_B1\_LSB, 134
  - DAP\_COEF\_WR\_B1\_MSB, 134
  - DAP\_COEF\_WR\_B2\_LSB, 134
  - DAP\_COEF\_WR\_B2\_MSB, 134
  - DAP\_CONTROL, 134
  - DAP\_FILTER\_COEF\_ACCESS, 134
  - DAP\_MAIN\_CHAN, 135
  - DAP\_MIX\_CHAN, 135
  - DAP\_PEQ, 135
  - DAP\_SGTL\_SURROUND, 135
  - FILTER\_BANDPASS, 135
  - FILTER\_HIPASS, 135
  - FILTER\_HISHELF, 135
  - FILTER\_LOPASS, 135
  - FILTER\_LOSHELF, 135
  - FILTER\_NOTCH, 135
  - FILTER\_PARAEQ, 136
  - FLAT\_FREQUENCY, 136
  - GRAPHIC\_EQUALIZER, 136

- PARAMETRIC\_EQUALIZER, 136
- SGTL5000\_I2C\_ADDR\_CS\_HIGH, 136
- SGTL5000\_I2C\_ADDR\_CS\_LOW, 136
- TONE\_CONTROLS, 136
- m\_eng\_simple\_distortion.c, 205
  - calc\_simple\_distortion, 205
  - init\_simple\_distortion\_str, 205
  - reconfigure\_simple\_distortion, 205
- m\_eng\_simple\_distortion.h, 142, 143
  - calc\_simple\_distortion, 143
  - init\_simple\_distortion\_str, 143
  - reconfigure\_simple\_distortion, 143
- m\_eng\_simple\_distortion\_str, 36
  - postgain, 37
  - pregain, 37
- m\_eng\_software\_isr
  - m\_eng\_update.c, 210
  - m\_eng\_update.h, 151
- M\_ENG\_TRACE\_FUNCTION\_ENTER
  - m\_eng\_logging.h, 96
- M\_ENG\_TRACE\_FUNCTION\_RETURN
  - m\_eng\_logging.h, 96
- m\_eng\_trace\_log\_begin
  - m\_eng\_logging.cpp, 185
  - m\_eng\_logging.h, 100
- M\_ENG\_TRACE\_LOG\_ENTRY
  - m\_eng\_logging.h, 96
- M\_ENG\_TRACE\_LOG\_RETURN
  - m\_eng\_logging.h, 97
- m\_eng\_trace\_log\_return
  - m\_eng\_logging.cpp, 185
  - m\_eng\_logging.h, 100
- m\_eng\_transformer.c, 205
  - clone\_transformer, 206
  - free\_transformer, 206
  - MAX\_BLOCK\_DIVIDER, 206
  - run\_bypass, 206
  - run\_transformer, 206
  - transformer\_add\_parameter, 206
  - transformer\_add\_setting, 207
  - transformer\_get\_parameter, 207
  - transformer\_get\_setting, 207
  - transformer\_init\_controls, 207
  - transformer\_init\_parameter\_array, 207
  - transformer\_init\_setting\_array, 207
- m\_eng\_transformer.h, 143, 147
  - clone\_transformer, 145
  - FADER\_FADE\_IN, 144
  - FADER\_FADE\_OUT, 144
  - free\_transformer, 145
  - N\_NATIVE\_PARAMETERS, 144
  - N\_NATIVE\_SETTINGS, 144
  - PRINT\_TRANSFORMER\_INFO, 144
  - run\_transformer, 146
  - transformer\_add\_parameter, 146
  - transformer\_add\_setting, 146
  - transformer\_get\_parameter, 146
  - transformer\_get\_setting, 146
- transformer\_init\_controls, 146
- transformer\_init\_parameter\_array, 146
- transformer\_init\_setting\_array, 146
- TRANSFORMER\_MAX\_INPUTS, 145
- TRANSFORMER\_MAX\_OUTPUTS, 145
- TRANSFORMER\_SWITCH\_ACTION\_BYPASS, 145
- TRANSFORMER\_TRANSITION\_BIBLOCK\_LINEAR, 145
- TRANSFORMER\_TRANSITION\_INSTANT, 145
- TRANSFORMER\_TRANSITION\_MONOBLOCK\_LINEAR, 145
- TRANSFORMER\_TRANSITION\_QUADBLOCK\_LINEAR, 145
- TRANSFORMER\_TRANSITION\_TAIL, 145
- transformer\_type\_to\_string, 147
- m\_eng\_transformer\_init.c, 207
  - init\_3\_band\_eq, 208
  - init\_amplifier, 208
  - init\_band\_pass\_filter, 208
  - init\_compressor, 208
  - init\_delay, 208
  - init\_dirty\_octave, 208
  - init\_distortion, 209
  - init\_envelope, 209
  - init\_flanger, 209
  - init\_high\_pass\_filter, 209
  - init\_low\_end\_compressor, 209
  - init\_low\_pass\_filter, 209
  - init\_noise\_suppressor, 209
  - init\_percussifier, 209
  - init\_transformer, 209
  - init\_warbler, 210
- m\_eng\_transformer\_init.h, 147, 148
  - init\_transformer, 148
- m\_eng\_transformer\_template.c, 210
- m\_eng\_transformer\_template.h, 148, 149
  - calc\_transformer, 148
  - init\_transformer\_str, 148
  - reconfigure\_transformer, 148
- m\_eng\_transition.h, 149, 150
  - TAIL\_INPUT\_FADE\_SAMPLES, 149
  - TAIL\_NEW\_FADE\_IN\_SAMPLES, 149
  - TRANSITION\_MONOBLOCK\_COS2, 149
  - TRANSITION\_OCTOBLOCK\_COS2, 149
  - TRANSITION\_QUADBLOCK\_COS2, 150
  - TRANSITION\_TAIL, 150
- m\_eng\_update.c, 210
  - m\_eng\_software\_isr, 210
  - update\_all, 210
  - update\_setup, 210
  - update\_stop, 210
- m\_eng\_update.h, 150, 151
  - m\_eng\_disable\_software\_interrupts, 150
  - m\_eng\_enable\_software\_interrupts, 150
  - m\_eng\_software\_isr, 151
  - update\_all, 151
  - update\_setup, 151

- update\_stop, 151
- m\_eng\_useful\_functions.c, 211
  - convert\_block\_float\_to\_int, 211
  - convert\_block\_int\_to\_float, 211
  - DENORMAL\_THRESHOLD, 211
  - FLOAT\_TO\_INT16\_MAX, 211
  - hard\_clip, 212
  - identity\_function, 212
  - MAX\_INT, 211
  - normalised\_arctan, 212
  - SCALE\_FACTOR, 211
  - soft\_fold, 212
  - trig\_transition\_function, 212
- m\_eng\_useful\_functions.h, 152, 153
  - convert\_block\_float\_to\_int, 152
  - convert\_block\_int\_to\_float, 152
  - hard\_clip, 152
  - identity\_function, 152
  - normalised\_arctan, 152
  - soft\_fold, 152
  - trig\_transition\_function, 152
- m\_eng\_warbler.c, 212
  - calc\_warbler, 213
  - init\_warbler\_str, 213
  - reconfigure\_warbler, 213
- m\_eng\_warbler.h, 153, 154
  - calc\_warbler, 153
  - init\_warbler\_str, 153
  - reconfigure\_warbler, 154
- m\_eng\_warbler\_str, 37
  - alpha, 37
  - center, 37
  - e, 37
  - filter, 38
  - max\_rate, 38
  - min\_rate, 38
  - rate, 38
  - reactivity, 38
  - sensitivity, 38
  - t, 38
  - width, 38
- m\_eng\_waveshaper.c, 213
  - calc\_waveshaper, 213
  - init\_waveshaper\_str, 213
- m\_eng\_waveshaper.h, 154, 155
  - calc\_waveshaper, 155
  - init\_waveshaper\_str, 155
  - WAVESHAPER\_ENVELOPE\_ATTACK, 155
  - WAVESHAPER\_ENVELOPE\_RELEASE, 155
- m\_eng\_waveshaper\_str, 39
  - coefficient, 39
  - shape, 39
- M\_ENGINE
  - m\_eng.h, 54
- m\_error\_code\_to\_string
  - m\_error\_codes.c, 231
  - m\_error\_codes.h, 236
- m\_error\_codes.c, 231
  - m\_error\_code\_to\_string, 231
- m\_error\_codes.h, 231, 236
  - ERR\_ALLOC\_FAIL, 232
  - ERR\_ARRAY\_MALFORMED, 232
  - ERR\_BAD\_ARGS, 232
  - ERR\_BAD\_REQUEST, 232
  - ERR\_BUSTED\_MSG, 232
  - ERR\_COMMS\_FAIL, 232
  - ERR\_FIXED\_ARRAY\_FULL, 232
  - ERR\_FOPEN\_FAIL, 233
  - ERR\_I2C\_FAIL, 233
  - ERR\_INCONSISTENT\_BACK\_PIPELINE, 233
  - ERR\_INVALID\_MESSAGE, 233
  - ERR\_INVALID\_PARAMETER\_ID, 233
  - ERR\_INVALID\_PROFILE\_ID, 233
  - ERR\_INVALID\_SETTING\_ID, 233
  - ERR\_INVALID\_TRANSFORMER\_ID, 233
  - ERR\_LOOP\_DETECTED, 233
  - ERR\_MANGLED\_FILE, 233
  - ERR\_MUTEX\_UNAVAILABLE, 234
  - ERR\_NO\_RESPONSE, 234
  - ERR\_NODE\_PRIVATE, 234
  - ERR\_NULL\_PTR, 234
  - ERR\_PIPELINE\_BUSTED, 234
  - ERR\_PIPELINE\_FULL, 234
  - ERR\_PIPELINE\_NULL, 234
  - ERR\_POSITION\_ILLEGAL, 234
  - ERR\_POSITION\_OCCUPIED, 234
  - ERR\_POT\_LINK\_MALFORMED, 234
  - ERR\_QUEUE\_FULL, 235
  - ERR\_QUEUE\_SEND\_FAILED, 235
  - ERR\_SD\_INIT\_FAIL, 235
  - ERR\_SD\_MOUNT\_FAIL, 235
  - ERR\_SGTL5000\_WRITE\_FAIL, 235
  - ERR\_SPI\_INIT\_FAIL, 235
  - ERR\_SWITCH\_LINK\_MALFORMED, 235
  - ERR\_TRANSFORMER\_MALFORMED, 235
  - ERR\_UNFINISHED\_WRITE, 235
  - ERR\_UNIMPLEMENTED, 235
  - ERR\_UNKNOWN\_ERR, 236
  - ERR\_VALUE\_OUT\_OF\_BOUNDS, 236
  - m\_error\_code\_to\_string, 236
  - NO\_ERROR, 236
- m\_free
  - m\_alloc.c, 214
  - m\_alloc.h, 215
- m\_int\_lv\_free
  - m\_alloc.h, 215
- m\_int\_lv\_malloc
  - m\_alloc.h, 215
- m\_linked\_list.h, 237, 239
  - DECLARE\_LINKED\_LIST, 237
  - DECLARE\_LINKED\_PTR\_LIST, 238
  - IMPLEMENT\_LINKED\_LIST, 238
  - IMPLEMENT\_LINKED\_PTR\_LIST, 238
  - LL\_FREE, 238
  - LL\_MALLOC, 238
- M\_LOG

- m\_eng\_logging.h, [97](#)
- M\_LOG\_ERROR
  - m\_eng\_logging.h, [97](#)
- m\_lr\_high\_pass\_filter\_str, [39](#)
  - a0, [40](#)
  - a1, [40](#)
  - a2, [40](#)
  - a3, [40](#)
  - a4, [40](#)
  - cutoff\_frequency, [40](#)
  - x\_11, [40](#)
  - x\_12, [40](#)
  - x\_21, [40](#)
  - x\_22, [40](#)
  - y\_11, [40](#)
  - y\_12, [41](#)
  - y\_21, [41](#)
  - y\_22, [41](#)
- m\_lr\_low\_pass\_filter\_str, [41](#)
  - a0, [41](#)
  - a1, [41](#)
  - a2, [42](#)
  - a3, [42](#)
  - a4, [42](#)
  - cutoff\_frequency, [42](#)
  - x\_11, [42](#)
  - x\_12, [42](#)
  - x\_21, [42](#)
  - x\_22, [42](#)
  - y\_11, [42](#)
  - y\_12, [42](#)
  - y\_21, [43](#)
  - y\_22, [43](#)
- m\_message, [43](#)
  - callback, [43](#)
  - cb\_arg, [43](#)
  - data, [43](#)
  - retries, [43](#)
  - type, [44](#)
- M\_MESSAGE\_APPEND\_TRANSFORMER
  - m\_comms.h, [221](#)
- m\_message\_code\_to\_string
  - m\_comms.c, [218](#)
  - m\_comms.h, [228](#)
- M\_MESSAGE\_CRC\_FAIL
  - m\_comms.h, [221](#)
- M\_MESSAGE\_CREATE\_PROFILE
  - m\_comms.h, [221](#)
- M\_MESSAGE\_DELETE\_PROFILE
  - m\_comms.h, [221](#)
- M\_MESSAGE\_ENTER\_TUNER\_MODE
  - m\_comms.h, [221](#)
- M\_MESSAGE\_EXIT\_TUNER\_MODE
  - m\_comms.h, [221](#)
- M\_MESSAGE\_GET\_N\_PARAMETERS
  - m\_comms.h, [221](#)
- M\_MESSAGE\_GET\_N\_PROFILES
  - m\_comms.h, [221](#)
- M\_MESSAGE\_GET\_N\_SETTINGS
  - m\_comms.h, [221](#)
- M\_MESSAGE\_GET\_N\_TRANSFORMERS
  - m\_comms.h, [222](#)
- M\_MESSAGE\_GET\_PARAM\_VALUE
  - m\_comms.h, [222](#)
- M\_MESSAGE\_GET\_SETTING\_VALUE
  - m\_comms.h, [222](#)
- M\_MESSAGE\_GET\_TRANSFORMER\_ID
  - m\_comms.h, [222](#)
- M\_MESSAGE\_GET\_TRANSFORMER\_TYPE
  - m\_comms.h, [222](#)
- M\_MESSAGE\_HI
  - m\_comms.h, [222](#)
- M\_MESSAGE\_INVALID
  - m\_comms.h, [222](#)
- M\_MESSAGE\_MAX\_DATA\_LEN
  - m\_comms.h, [222](#)
- M\_MESSAGE\_MAX\_TRANSFER\_LEN
  - m\_comms.h, [222](#)
- M\_MESSAGE\_MOVE\_TRANSFORMER
  - m\_comms.h, [222](#)
- M\_MESSAGE\_NO\_MESSAGE
  - m\_comms.h, [223](#)
- M\_MESSAGE\_REBOOT
  - m\_comms.h, [223](#)
- M\_MESSAGE\_REMOVE\_TRANSFORMER
  - m\_comms.h, [223](#)
- M\_MESSAGE\_REPEAT\_MESSAGE
  - m\_comms.h, [223](#)
- M\_MESSAGE\_RESET
  - m\_comms.h, [223](#)
- m\_message\_sanity\_check
  - m\_eng\_comms.cpp, [161](#)
- M\_MESSAGE\_SET\_PARAM\_VALUE
  - m\_comms.h, [223](#)
- M\_MESSAGE\_SET\_SETTING\_VALUE
  - m\_comms.h, [223](#)
- M\_MESSAGE\_STRING\_CONTINUE
  - m\_comms.h, [223](#)
- M\_MESSAGE\_STRING\_CONTINUING
  - m\_comms.h, [223](#)
- M\_MESSAGE\_SWITCH\_PROFILE
  - m\_comms.h, [223](#)
- M\_MESSAGE\_TYPE\_MAX
  - m\_comms.h, [224](#)
- m\_mute\_voice
  - m\_eng\_printf.cpp, [198](#)
  - m\_eng\_printf.h, [121](#)
- m\_parameter, [44](#)
  - max, [44](#)
  - min, [44](#)
  - new\_value, [44](#)
  - old\_value, [44](#)
  - scale, [44](#)
  - updated, [45](#)
  - value, [45](#)
- m\_parameter.h, [247](#), [249](#)

- DECLARE\_LINKED\_PTR\_LIST, 248
- PARAMETER\_SCALE\_LINEAR, 248
- PARAMETER\_SCALE\_LOGARITHMIC, 248
- TRANSFORMER\_SETTING\_BOOL, 248
- TRANSFORMER\_SETTING\_ENUM, 248
- TRANSFORMER\_SETTING\_INT, 248
- TRANSFORMER\_SETTING\_PAGE\_MAIN, 248
- TRANSFORMER\_SETTING\_PAGE\_SETTINGS, 248
- m\_parameter\_id, 45
  - parameter\_id, 45
  - profile\_id, 45
  - transformer\_id, 45
- m\_pipeline, 46
- m\_pipeline.h, 250
- m\_pipeline\_mod, 46
  - data, 46
  - sdata, 46
  - tid, 46
  - type, 46
- m\_printf
  - m\_eng\_printf.cpp, 198
  - m\_eng\_printf.h, 121
- m\_profile, 47
  - active, 47
- m\_profile.h, 250, 251
- M\_PROFILE\_SWITCH\_SAMPLES
  - m\_eng\_context.h, 68
- m\_realloc
  - m\_alloc.c, 214
  - m\_alloc.h, 215
- m\_response, 47
  - data, 47
  - extra, 47
  - type, 47
- M\_RESPONSE\_BAD\_MESSAGE
  - m\_comms.h, 224
- M\_RESPONSE\_BAD\_REQUEST
  - m\_comms.h, 224
- m\_response\_code\_to\_string
  - m\_comms.c, 218
  - m\_comms.h, 228
- M\_RESPONSE\_CRC\_FAIL
  - m\_comms.h, 224
- M\_RESPONSE\_DELETED\_PROFILE
  - m\_comms.h, 224
- M\_RESPONSE\_ERROR
  - m\_comms.h, 224
- M\_RESPONSE\_HI
  - m\_comms.h, 224
- M\_RESPONSE\_INVALID
  - m\_comms.h, 224
- M\_RESPONSE\_MAX\_DATA\_LEN
  - m\_comms.h, 224
- M\_RESPONSE\_MAX\_TRANSFER\_LEN
  - m\_comms.h, 224
- M\_RESPONSE\_N\_PARAMETERS
  - m\_comms.h, 225
- M\_RESPONSE\_N\_PROFILES
  - m\_comms.h, 225
- M\_RESPONSE\_N\_SETTINGS
  - m\_comms.h, 225
- M\_RESPONSE\_N\_TRANSFORMERS
  - m\_comms.h, 225
- M\_RESPONSE\_NO\_MESSAGE
  - m\_comms.h, 225
- M\_RESPONSE\_OK
  - m\_comms.h, 225
- M\_RESPONSE\_PARAM\_VALUE
  - m\_comms.h, 225
- M\_RESPONSE\_PROFILE\_ID
  - m\_comms.h, 225
- M\_RESPONSE\_REPEAT\_MESSAGE
  - m\_comms.h, 225
- M\_RESPONSE\_SETTING\_VALUE
  - m\_comms.h, 225
- M\_RESPONSE\_START\_OVER
  - m\_comms.h, 226
- M\_RESPONSE\_STRING\_CONTINUING
  - m\_comms.h, 226
- M\_RESPONSE\_SWITCHING\_PROFILE
  - m\_comms.h, 226
- M\_RESPONSE\_TRANSFORMER\_ID
  - m\_comms.h, 226
- M\_RESPONSE\_TRANSFORMER\_TYPE
  - m\_comms.h, 226
- M\_RESPONSE\_TRY\_AGAIN
  - m\_comms.h, 226
- M\_RESPONSE\_TYPE\_MAX
  - m\_comms.h, 226
- M\_RESPONSE\_WAIT
  - m\_comms.h, 226
- m\_setting, 48
  - new\_value, 48
  - old\_value, 48
  - updated, 48
  - value, 48
- m\_setting\_id, 48
  - profile\_id, 49
  - setting\_id, 49
  - transformer\_id, 49
- m\_status.h, 251, 252
  - M\_STATUS\_BOOTING, 251
  - M\_STATUS\_FRESH\_BOOT, 251
  - M\_STATUS\_OK, 252
- M\_STATUS\_BOOTING
  - m\_status.h, 251
- M\_STATUS\_FRESH\_BOOT
  - m\_status.h, 251
- M\_STATUS\_OK
  - m\_status.h, 252
- m\_strndup
  - m\_alloc.c, 214
  - m\_alloc.h, 215
- m\_transformer, 49
  - band\_center, 49

- band\_hp\_cutoff, [49](#)
- band\_lp\_cutoff, [49](#)
- band\_mode, [50](#)
- band\_width, [50](#)
- id, [50](#)
- type, [50](#)
- wet\_mix, [50](#)
- m\_transformer.h, [252](#), [253](#)
  - DECLARE\_LINKED\_PTR\_LIST, [252](#)
- m\_transformer\_enum.c, [253](#)
  - transformer\_type\_to\_string, [254](#)
  - transformer\_type\_valid, [254](#)
- m\_transformer\_enum.h, [254](#), [258](#)
  - band\_pass, [258](#)
  - biquad\_type, [257](#)
  - DISTORTION\_ARCTAN, [255](#)
  - DISTORTION\_CLIP, [255](#)
  - DISTORTION\_SOFT\_FOLD, [255](#)
  - DISTORTION\_TANH, [255](#)
  - high\_pass, [258](#)
  - high\_shelf, [258](#)
  - low\_pass, [258](#)
  - low\_shelf, [258](#)
  - notch, [258](#)
  - peaking\_band\_eq, [258](#)
  - TRANSFORMER\_3\_BAND\_EQ, [255](#)
  - TRANSFORMER\_AMPLIFIER, [255](#)
  - TRANSFORMER\_BAND\_HP\_CUTOFF\_PID, [255](#)
  - TRANSFORMER\_BAND\_LP\_CUTOFF\_PID, [255](#)
  - TRANSFORMER\_BAND\_MODE\_SID, [256](#)
  - TRANSFORMER\_BAND\_PASS\_FILTER, [256](#)
  - TRANSFORMER\_COMPRESSOR, [256](#)
  - TRANSFORMER\_DELAY, [256](#)
  - TRANSFORMER\_DIRTY\_OCTAVE, [256](#)
  - TRANSFORMER\_DISTORTION, [256](#)
  - TRANSFORMER\_ENVELOPE, [256](#)
  - TRANSFORMER\_FLANGER, [256](#)
  - TRANSFORMER\_HIGH\_PASS\_FILTER, [256](#)
  - TRANSFORMER\_LOW\_END\_COMPRESSOR, [256](#)
  - TRANSFORMER\_LOW\_PASS\_FILTER, [257](#)
  - TRANSFORMER\_MODE\_BAND, [257](#)
  - TRANSFORMER\_MODE\_FULL\_SPECTRUM, [257](#)
  - TRANSFORMER\_MODE\_LOWER\_SPECTRUM, [257](#)
  - TRANSFORMER\_MODE\_UPPER\_SPECTRUM, [257](#)
  - TRANSFORMER\_NOISE\_SUPPRESSOR, [257](#)
  - TRANSFORMER\_PERCUSSIFIER, [257](#)
  - transformer\_type\_to\_string, [258](#)
  - transformer\_type\_valid, [258](#)
  - TRANSFORMER\_WARBLER, [257](#)
  - TRANSFORMER\_WET\_MIX\_PID, [257](#)
- m\_transformer\_str, [50](#)
  - param, [50](#)
- m\_unmute\_voice
  - m\_eng\_printf.cpp, [198](#)
  - m\_eng\_printf.h, [121](#)
- M\_VOICE\_COMMS
  - m\_eng.h, [54](#)
- M\_VOICE\_CXT
  - m\_eng.h, [54](#)
- M\_VOICE\_ERR
  - m\_eng.h, [54](#)
- M\_VOICE\_LOG
  - m\_eng.h, [54](#)
- M\_VOICE\_PL
  - m\_eng.h, [54](#)
- M\_VOICE\_PR
  - m\_eng.h, [54](#)
- M\_VOICE\_PRF
  - m\_eng.h, [54](#)
- m\_voice\_printf
  - m\_eng\_printf.cpp, [198](#)
  - m\_eng\_printf.h, [121](#)
- M\_VOICE\_TR
  - m\_eng.h, [54](#)
- main
  - m\_eng.cpp, [157](#)
- max
  - m\_parameter, [44](#)
- MAX\_AUDIO\_MEMORY
  - m\_eng\_mempool.h, [105](#)
- MAX\_BLOCK\_DIVIDER
  - m\_eng\_transformer.c, [206](#)
- max\_center
  - m\_eng\_envelope\_str, [21](#)
- MAX\_INT
  - m\_eng\_useful\_functions.c, [211](#)
- max\_rate
  - m\_eng\_warbler\_str, [38](#)
- max\_reduction
  - m\_eng\_noise\_suppressor\_str, [30](#)
- mem\_pools\_initialised
  - m\_eng\_mempool.c, [188](#)
- MEM\_REPORT\_MILLIS
  - m\_eng.cpp, [156](#)
- MEM\_SIZE
  - m\_eng\_mempool.h, [105](#)
- memcpy\_tointerleaveL
  - m\_eng\_memcpy\_audio.h, [104](#)
- memcpy\_tointerleaveLR
  - m\_eng\_memcpy\_audio.h, [104](#)
- memcpy\_tointerleaveQuad
  - m\_eng\_memcpy\_audio.h, [104](#)
- memcpy\_tointerleaveR
  - m\_eng\_memcpy\_audio.h, [104](#)
- memory\_used
  - m\_eng\_globals.c, [177](#)
  - m\_eng\_globals.h, [88](#)
- memory\_used\_max
  - m\_eng\_globals.c, [177](#)
  - m\_eng\_globals.h, [88](#)
- MEMPOOL\_MALLOC\_TRIES
  - m\_eng\_mempool.c, [187](#)

- message
  - m\_eng\_log\_entry, 26
- MESSAGE\_BEGIN\_COL
  - m\_eng\_logging.cpp, 183
- MESSAGE\_LEN\_VARIABLE
  - m\_comms.h, 226
- message\_pending
  - m\_eng\_comms.cpp, 161
- mid
  - m\_eng\_3\_band\_eq\_str, 9
- mid\_pass
  - m\_eng\_low\_end\_compressor\_str, 27
- mids\_comp
  - m\_eng\_low\_end\_compressor\_str, 27
- min
  - m\_parameter, 44
- min\_center
  - m\_eng\_envelope\_str, 22
- min\_rate
  - m\_eng\_warbler\_str, 38
- mix
  - m\_eng\_flanger\_str, 23
- mode
  - m\_eng\_amplifier\_str, 10
- MS\_TO\_SAMPLES
  - m\_eng.h, 55
- n\_buffers
  - m\_delay\_buffer, 7
- N\_NATIVE\_PARAMETERS
  - m\_eng\_transformer.h, 144
- N\_NATIVE\_SETTINGS
  - m\_eng\_transformer.h, 144
- n\_profiles
  - m\_eng\_context, 16
- new\_profile
  - m\_eng\_context, 16
- new\_value
  - m\_parameter, 44
  - m\_setting, 48
- NO\_ERROR
  - m\_error\_codes.h, 236
- normalised\_arctan
  - m\_eng\_useful\_functions.c, 212
  - m\_eng\_useful\_functions.h, 152
- notch
  - m\_transformer\_enum.h, 258
- note
  - m\_eng\_delay\_str, 19
  - m\_eng\_flanger\_str, 23
  - m\_eng\_percussifier\_str, 32
- nullify\_profile
  - m\_eng\_profile.c, 200
  - m\_eng\_profile.h, 123
- NUM\_MASKS
  - m\_eng.h, 55
- old\_value
  - m\_parameter, 44
  - m\_setting, 48
- open\_cycle
  - m\_eng\_profiler\_entry, 36
- output\_amp
  - m\_eng\_context, 16
  - m\_eng\_profile, 34
- output\_hpf
  - m\_eng\_context, 17
- param
  - m\_transformer\_str, 50
- PARAM\_NAM\_ENG\_MAX\_LEN
  - m\_eng\_parameter.h, 108
- parameter\_id
  - m\_parameter\_id, 45
- PARAMETER\_SCALE\_LINEAR
  - m\_parameter.h, 248
- PARAMETER\_SCALE\_LOGARITHMIC
  - m\_parameter.h, 248
- PARAMETER\_UPDATE\_BIBLOCK\_LINEAR
  - m\_eng\_parameter.h, 108
- PARAMETER\_UPDATE\_INSTANT
  - m\_eng\_parameter.h, 108
- PARAMETER\_UPDATE\_MONOBLOCK\_LINEAR
  - m\_eng\_parameter.h, 108
- PARAMETER\_UPDATE\_QUADBLOCK\_LINEAR
  - m\_eng\_parameter.h, 108
- PARAMETRIC\_EQUALIZER
  - m\_eng\_sgtl5000\_defs.h, 136
- pcxt\_profile\_id\_valid
  - m\_eng\_context.c, 169
- peak\_cycles
  - m\_eng\_profiler\_entry, 36
- peaking\_band\_eq
  - m\_transformer\_enum.h, 258
- PERCUSSIFIER\_FADE\_IN
  - m\_eng\_percussifier.h, 112
- PERCUSSIFIER\_FADE\_OUT
  - m\_eng\_percussifier.h, 112
- PERCUSSIFIER\_HOLD
  - m\_eng\_percussifier.h, 112
- PERCUSSIFIER\_MUTE
  - m\_eng\_percussifier.h, 112
- PERCUSSIFIER\_REFRACTORY
  - m\_eng\_percussifier.h, 112
- period
  - m\_eng\_flanger\_str, 23
- pipeline\_append\_transformer
  - m\_eng\_pipeline.c, 194
  - m\_eng\_pipeline.h, 115
- pipeline\_append\_transformer\_type
  - m\_eng\_pipeline.c, 194
  - m\_eng\_pipeline.h, 115
- pipeline\_change\_transformer\_setting
  - m\_eng\_pipeline.c, 194
  - m\_eng\_pipeline.h, 115
- pipeline\_clone\_transformer\_into\_position
  - m\_eng\_pipeline.c, 194
  - m\_eng\_pipeline.h, 115



pipeline\_compare  
     m\_eng\_pipeline.c, 194  
     m\_eng\_pipeline.h, 115  
 pipeline\_expand\_transformer\_array  
     m\_eng\_pipeline.c, 194  
     m\_eng\_pipeline.h, 116  
 pipeline\_expand\_transformer\_array\_to  
     m\_eng\_pipeline.c, 194  
     m\_eng\_pipeline.h, 116  
 pipeline\_get\_transformer\_by\_id  
     m\_eng\_pipeline.c, 195  
     m\_eng\_pipeline.h, 116  
 pipeline\_get\_transformer\_position  
     m\_eng\_pipeline.c, 195  
     m\_eng\_pipeline.h, 116  
 pipeline\_insert\_transformer  
     m\_eng\_pipeline.c, 195  
     m\_eng\_pipeline.h, 116  
 pipeline\_insert\_transformer\_type  
     m\_eng\_pipeline.c, 195  
     m\_eng\_pipeline.h, 116  
 PIPELINE\_MOD\_APPEND\_TRANSFORMER  
     m\_eng\_pipeline\_mod.h, 119  
 PIPELINE\_MOD\_CHANGE\_TRANSFORMER\_SETTING  
     m\_eng\_pipeline\_mod.h, 119  
 PIPELINE\_MOD\_MOVE\_TRANSFORMER  
     m\_eng\_pipeline\_mod.h, 119  
 PIPELINE\_MOD\_REMOVE\_TRANSFORMER  
     m\_eng\_pipeline\_mod.h, 119  
 pipeline\_mod\_type\_string  
     m\_eng\_pipeline\_mod.c, 197  
     m\_eng\_pipeline\_mod.h, 120  
 pipeline\_move\_transformer  
     m\_eng\_pipeline.c, 195  
     m\_eng\_pipeline.h, 116  
 pipeline\_prepend\_transformer  
     m\_eng\_pipeline.c, 195  
     m\_eng\_pipeline.h, 117  
 pipeline\_prepend\_transformer\_type  
     m\_eng\_pipeline.c, 195  
     m\_eng\_pipeline.h, 117  
 pipeline\_print\_transformer\_array  
     m\_eng\_pipeline.c, 196  
 pipeline\_remove\_transformer  
     m\_eng\_pipeline.c, 196  
     m\_eng\_pipeline.h, 117  
 pipeline\_swap\_progress  
     m\_eng\_profile, 35  
 pipeline\_swap\_samples  
     m\_eng\_profile, 35  
 pipeline\_swap\_transformers  
     m\_eng\_pipeline.c, 196  
     m\_eng\_pipeline.h, 117  
 pipeline\_swap\_type  
     m\_eng\_profile, 35  
 pipeline\_update\_transition\_policy  
     m\_eng\_pipeline.c, 196  
     m\_eng\_pipeline.h, 117  
 pipeline\_valid  
     m\_eng\_pipeline.c, 196  
     m\_eng\_pipeline.h, 117  
 pipelines\_swapping  
     m\_eng\_profile, 35  
 pos  
     m\_delay\_buffer, 7  
 postgain  
     m\_eng\_simple\_distortion\_str, 37  
 pregain  
     m\_eng\_simple\_distortion\_str, 37  
 pretty\_print\_block  
     m\_eng\_printf.cpp, 199  
     m\_eng\_printf.h, 121  
 pretty\_print\_block\_float  
     m\_eng\_printf.cpp, 199  
     m\_eng\_printf.h, 121  
 prev\_block  
     m\_eng\_context, 17  
     m\_eng\_profile, 35  
 prev\_response  
     m\_eng\_comms.cpp, 161  
 print\_binary  
     m\_eng\_debugging.c, 169  
 print\_context\_info  
     m\_eng\_debugging.c, 169  
     m\_eng\_debugging.h, 74  
 PRINT\_LOG  
     m\_eng.cpp, 156  
 print\_memory\_report  
     m\_alloc.c, 214  
     m\_alloc.h, 215  
 print\_mempool\_info  
     m\_eng\_mempool.c, 187  
     m\_eng\_mempool.h, 106  
 print\_pipeline\_info  
     m\_eng\_debugging.c, 170  
     m\_eng\_debugging.h, 74  
 print\_profile\_info  
     m\_eng\_debugging.c, 170  
     m\_eng\_debugging.h, 74  
 PRINT\_RESPONSE\_BYTES  
     m\_eng\_comms.cpp, 160  
 PRINT\_TRANSFORMER\_INFO  
     m\_eng\_transformer.h, 144  
 print\_transformer\_info  
     m\_eng\_debugging.c, 170  
     m\_eng\_debugging.h, 74  
 profile\_apply\_pipeline\_mod  
     m\_eng\_profile.c, 200  
     m\_eng\_profile.h, 123  
 PROFILE\_ARRAY\_INITIAL\_SIZE  
     m\_eng\_context.h, 68  
 profile\_array\_size  
     m\_eng\_context, 17  
 profile\_id  
     m\_parameter\_id, 45  
     m\_setting\_id, 49



- profile\_maintenance\_index
  - m\_eng\_context, [17](#)
- profile\_print\_job\_list
  - m\_eng\_profile.c, [201](#)
- profile\_print\_ujob\_list
  - m\_eng\_profile.c, [201](#)
- profile\_process
  - m\_eng\_profile.c, [201](#)
  - m\_eng\_profile.h, [123](#)
- profile\_regenerate\_back\_pipeline
  - m\_eng\_profile.c, [201](#)
- profile\_scheduled\_maintenance
  - m\_eng\_profile.c, [201](#)
  - m\_eng\_profile.h, [123](#)
- profile\_switch\_progress
  - m\_eng\_context, [17](#)
- profile\_switch\_samples
  - m\_eng\_context, [17](#)
- profile\_switch\_triggered
  - m\_eng\_context, [17](#)
- profile\_switch\_type
  - m\_eng\_context, [17](#)
- profile\_trigger\_pipeline\_swap
  - m\_eng\_profile.c, [201](#)
  - m\_eng\_profile.h, [123](#)
- profile\_update
  - m\_eng\_profile.c, [201](#)
  - m\_eng\_profile.h, [124](#)
- PROFILER\_PRINT\_MILLIS
  - m\_eng.cpp, [156](#)
- profiles
  - m\_eng\_context, [17](#)
- PROFILES\_MALLOC\_CHUNK\_SIZE
  - m\_eng\_context.h, [68](#)
- profiles\_switching
  - m\_eng\_context, [17](#)
- r
  - m\_eng\_flanger\_str, [23](#)
  - m\_eng\_noise\_suppressor\_str, [30](#)
  - m\_eng\_percussifier\_str, [32](#)
- ra\_cycles
  - m\_eng\_profiler\_entry, [36](#)
- range
  - m\_eng\_flanger\_str, [23](#)
- rate
  - m\_eng\_warbler\_str, [38](#)
- ratio
  - m\_eng\_compressor\_str, [15](#)
  - m\_eng\_mixer\_str, [29](#)
  - m\_eng\_noise\_suppressor\_str, [30](#)
- raw\_sample\_t
  - m\_eng\_i2s\_dma.h, [90](#)
- reactivity
  - m\_eng\_warbler\_str, [38](#)
- receive\_buffer
  - m\_eng\_comms.cpp, [161](#)
- received
  - m\_eng\_comms.cpp, [162](#)
- received\_length
  - m\_eng\_comms.cpp, [162](#)
- RECIEVING\_NEW\_PARAM\_NAM\_ENG\_LONG
  - m\_eng\_comms.cpp, [160](#)
- reconfigure\_3\_band\_eq
  - m\_eng\_equaliser.c, [175](#)
  - m\_eng\_equaliser.h, [82](#)
- reconfigure\_3\_band\_splitter
  - m\_eng\_band\_splitter.h, [61](#)
- reconfigure\_amplifier
  - m\_eng\_buffer\_mixer\_amp.c, [159](#)
  - m\_eng\_buffer\_mixer\_amp.h, [63](#)
- reconfigure\_band\_pass\_filter
  - m\_eng\_pass\_filter.c, [191](#)
  - m\_eng\_pass\_filter.h, [111](#)
- reconfigure\_biquad
  - m\_eng\_biquad.c, [158](#)
  - m\_eng\_biquad.h, [62](#)
- reconfigure\_compressor
  - m\_eng\_compressor.c, [163](#)
  - m\_eng\_compressor.h, [66](#)
- reconfigure\_delay
  - m\_eng\_delay.c, [171](#)
  - m\_eng\_delay.h, [75](#)
- reconfigure\_dirty\_octave
  - m\_eng\_dirty\_octave.c, [173](#)
  - m\_eng\_dirty\_octave.h, [78](#)
- reconfigure\_distortion
  - m\_eng\_distortion.c, [173](#)
  - m\_eng\_distortion.h, [80](#)
- reconfigure\_envelope
  - m\_eng\_envelope.c, [174](#)
  - m\_eng\_envelope.h, [81](#)
- reconfigure\_flanger
  - m\_eng\_flanger.c, [176](#)
  - m\_eng\_flanger.h, [83](#)
- reconfigure\_high\_pass\_filter
  - m\_eng\_pass\_filter.c, [191](#)
  - m\_eng\_pass\_filter.h, [111](#)
- reconfigure\_low\_end\_compressor
  - m\_eng\_low\_end\_compressor.c, [186](#)
  - m\_eng\_low\_end\_compressor.h, [103](#)
- reconfigure\_low\_pass\_filter
  - m\_eng\_pass\_filter.c, [191](#)
  - m\_eng\_pass\_filter.h, [111](#)
- reconfigure\_lr\_high\_pass\_filter
  - m\_eng\_linkowitz\_riley.c, [182](#)
  - m\_eng\_linkowitz\_riley.h, [92](#)
- reconfigure\_lr\_low\_pass\_filter
  - m\_eng\_linkowitz\_riley.c, [182](#)
  - m\_eng\_linkowitz\_riley.h, [92](#)
- reconfigure\_n\_band\_splitter
  - m\_eng\_band\_splitter.h, [61](#)
- reconfigure\_noise\_suppressor
  - m\_eng\_noise\_suppressor.c, [189](#)
  - m\_eng\_noise\_suppressor.h, [107](#)
- reconfigure\_percussifier
  - m\_eng\_percussifier.c, [192](#)

- m\_eng\_percussifier.h, 113
- reconfigure\_simple\_distortion
  - m\_eng\_simple\_distortion.c, 205
  - m\_eng\_simple\_distortion.h, 143
- reconfigure\_transformer
  - m\_eng\_transformer\_template.h, 148
- reconfigure\_warbler
  - m\_eng\_warbler.c, 213
  - m\_eng\_warbler.h, 154
- refractory\_period
  - m\_eng\_percussifier\_str, 32
- refractory\_samples
  - m\_eng\_percussifier\_str, 33
- release
  - m\_eng\_compressor\_str, 15
- release\_buffer
  - m\_eng\_mempool.c, 188
  - m\_eng\_mempool.h, 106
- reset\_context
  - m\_eng\_context.c, 169
  - m\_eng\_context.h, 72
- response
  - m\_eng\_comms.cpp, 162
- response\_buffer
  - m\_eng\_comms.cpp, 162
- response\_length
  - m\_eng\_comms.cpp, 162
- response\_ready
  - m\_eng\_comms.cpp, 162
- RESTRICT
  - m\_eng\_flops.h, 84
- retries
  - m\_message, 43
- RETURN
  - m\_eng\_logging.h, 97
- RETURN\_ERR\_CODE
  - m\_eng\_logging.h, 97
- RETURN\_INT
  - m\_eng\_logging.h, 97
- RETURN\_NEG\_ERR\_CODE
  - m\_eng\_logging.h, 97
- RETURN\_PTR
  - m\_eng\_logging.h, 98
- RETURN\_VOID
  - m\_eng\_logging.h, 98
- rho
  - m\_eng\_compressor\_str, 15
- rms\_long
  - m\_eng\_percussifier\_str, 33
- rms\_short
  - m\_eng\_percussifier\_str, 33
- run\_bypass
  - m\_eng\_transformer.c, 206
- run\_transformer
  - m\_eng\_transformer.c, 206
  - m\_eng\_transformer.h, 146
- runs
  - m\_eng\_context, 18
- m\_eng\_profile, 35
- S
  - m\_eng\_flanger\_str, 23
- S\_TO\_SAMPLES
  - m\_eng.h, 55
- SAMPLE\_FREQUENCY
  - m\_eng.h, 55
- SAMPLES\_TO\_MS
  - m\_eng.h, 55
- SAMPLES\_TO\_S
  - m\_eng.h, 55
- scale
  - m\_parameter, 44
- SCALE\_FACTOR
  - m\_eng\_useful\_functions.c, 211
- SCHEDULED\_MAINTAINANCE
  - m\_eng.cpp, 156
- SCHEDULED\_MAINTAINANCE\_MILLIS
  - m\_eng.cpp, 157
- sdata
  - m\_pipeline\_mod, 46
- SECONDS\_TO\_CYCLES
  - m\_eng\_logging.h, 98
- SENDING\_STRING
  - m\_eng\_comms.cpp, 160
- sensitivity
  - m\_eng\_envelope\_str, 22
  - m\_eng\_warbler\_str, 38
- serial\_print\_blocks
  - m\_eng\_printf.cpp, 199
  - m\_eng\_printf.h, 121
- setting\_id
  - m\_setting\_id, 49
- sgtl5000\_adc\_high\_pass\_filter\_disable
  - m\_eng\_sgtl5000.cpp, 202
  - m\_eng\_sgtl5000.h, 125
- sgtl5000\_adc\_high\_pass\_filter\_enable
  - m\_eng\_sgtl5000.cpp, 202
  - m\_eng\_sgtl5000.h, 125
- sgtl5000\_adc\_high\_pass\_filter\_freeze
  - m\_eng\_sgtl5000.cpp, 202
  - m\_eng\_sgtl5000.h, 125
- sgtl5000\_automate
  - m\_eng\_sgtl5000.cpp, 203
  - m\_eng\_sgtl5000.h, 125
- SGTL5000\_CHECK\_PERIOD
  - m\_eng.cpp, 157
- sgtl5000\_dap\_audio\_eq\_band
  - m\_eng\_sgtl5000.cpp, 203
  - m\_eng\_sgtl5000.h, 125
- sgtl5000\_enable
  - m\_eng\_sgtl5000.cpp, 203
  - m\_eng\_sgtl5000.h, 126
- sgtl5000\_eq\_select
  - m\_eng\_sgtl5000.cpp, 203
- sgtl5000\_healthy
  - m\_eng\_sgtl5000.h, 126
- SGTL5000\_I2C\_ADDR\_CS\_HIGH

- m\_eng\_sgtl5000\_defs.h, 136
- SGTL5000\_I2C\_ADDR\_CS\_LOW
  - m\_eng\_sgtl5000\_defs.h, 136
- sgtl5000\_kill\_automation
  - m\_eng\_sgtl5000.cpp, 203
  - m\_eng\_sgtl5000.h, 126
- sgtl5000\_line\_in\_level
  - m\_eng\_sgtl5000.cpp, 203
  - m\_eng\_sgtl5000.h, 126
- sgtl5000\_line\_out\_level
  - m\_eng\_sgtl5000.cpp, 203
  - m\_eng\_sgtl5000.h, 126
- sgtl5000\_mic\_gain
  - m\_eng\_sgtl5000.h, 126
- sgtl5000\_modify\_reg
  - m\_eng\_sgtl5000.cpp, 203
  - m\_eng\_sgtl5000.h, 126
- sgtl5000\_mute\_headphone
  - m\_eng\_sgtl5000.cpp, 203
  - m\_eng\_sgtl5000.h, 126
- sgtl5000\_mute\_line\_out
  - m\_eng\_sgtl5000.cpp, 204
  - m\_eng\_sgtl5000.h, 126
- sgtl5000\_read\_reg
  - m\_eng\_sgtl5000.cpp, 204
  - m\_eng\_sgtl5000.h, 127
- sgtl5000\_set\_address
  - m\_eng\_sgtl5000.cpp, 204
  - m\_eng\_sgtl5000.h, 127
- sgtl5000\_set\_master\_mode
  - m\_eng\_sgtl5000.h, 127
- sgtl5000\_soft\_reboot
  - m\_eng\_sgtl5000.h, 127
- sgtl5000\_start
  - m\_eng\_sgtl5000.cpp, 204
  - m\_eng\_sgtl5000.h, 127
- sgtl5000\_unmute\_headphone
  - m\_eng\_sgtl5000.cpp, 204
  - m\_eng\_sgtl5000.h, 127
- sgtl5000\_unmute\_line\_out
  - m\_eng\_sgtl5000.cpp, 204
  - m\_eng\_sgtl5000.h, 127
- sgtl5000\_volum\_eng\_integer
  - m\_eng\_sgtl5000.cpp, 204
  - m\_eng\_sgtl5000.h, 127
- sgtl5000\_volume
  - m\_eng\_sgtl5000.cpp, 204
  - m\_eng\_sgtl5000.h, 127
- sgtl5000\_write\_reg
  - m\_eng\_sgtl5000.cpp, 204
  - m\_eng\_sgtl5000.h, 128
- shape
  - m\_eng\_adaptive\_waveshaper\_str, 10
  - m\_eng\_waveshaper\_str, 39
- SILENCE\_BLOCKS\_THRESHOLD
  - m\_eng\_context.h, 68
- SILENCE\_ENERGY\_THRESHOLD
  - m\_eng\_context.h, 68
- sink\_buffer
  - m\_eng\_mempool.c, 188
  - m\_eng\_mempool.h, 106
- smoothness
  - m\_eng\_envelope\_str, 22
- soft\_fold
  - m\_eng\_useful\_functions.c, 212
  - m\_eng\_useful\_functions.h, 152
- SPACING
  - m\_eng\_printf.cpp, 198
- speed
  - m\_eng\_envelope\_str, 22
- sqr
  - m\_eng.h, 55
- state
  - m\_eng\_percussifier\_str, 33
- status\_flags
  - m\_eng\_context, 18
- STR
  - m\_eng\_logging.h, 98
- string\_in
  - m\_eng\_comms.cpp, 162
- string\_in\_pos
  - m\_eng\_comms.cpp, 162
- string\_out
  - m\_eng\_comms.cpp, 162
- string\_out\_pos
  - m\_eng\_comms.cpp, 162
- t
  - m\_eng\_flanger\_str, 23
  - m\_eng\_warbler\_str, 38
- tail
  - m\_eng\_mempool.c, 188
- TAIL\_INPUT\_FADE\_SAMPLES
  - m\_eng\_transition.h, 149
- TAIL\_NEW\_FADE\_IN\_SAMPLES
  - m\_eng\_transition.h, 149
- te\_message\_data\_length
  - m\_comms.c, 219
  - m\_comms.h, 229
- TEENSY\_ADDR
  - m\_comms.h, 226
- TEENSY\_I2C\_SLAVE\_ADDR
  - m\_eng\_comms.h, 64
- tempo
  - m\_eng\_delay\_str, 19
  - m\_eng\_flanger\_str, 24
  - m\_eng\_percussifier\_str, 33
- threshold
  - m\_eng\_compressor\_str, 15
  - m\_eng\_noise\_suppressor\_str, 30
- tid
  - m\_pipeline\_mod, 46
- timer
  - m\_eng\_percussifier\_str, 33
- TONE\_CONTROLS
  - m\_eng\_sgtl5000\_defs.h, 136
- total\_cycles

- m\_eng\_profiler\_entry, [36](#)
- trace\_depth
  - m\_eng\_globals.c, [177](#)
  - m\_eng\_globals.h, [88](#)
  - m\_eng\_log\_entry, [26](#)
- TRANSFORMER\_3\_BAND\_EQ
  - m\_transformer\_enum.h, [255](#)
- transformer\_add\_parameter
  - m\_eng\_transformer.c, [206](#)
  - m\_eng\_transformer.h, [146](#)
- transformer\_add\_setting
  - m\_eng\_transformer.c, [207](#)
  - m\_eng\_transformer.h, [146](#)
- TRANSFORMER\_AMPLIFIER
  - m\_transformer\_enum.h, [255](#)
- TRANSFORMER\_BAND\_HP\_CUTOFF\_PID
  - m\_transformer\_enum.h, [255](#)
- TRANSFORMER\_BAND\_LP\_CUTOFF\_PID
  - m\_transformer\_enum.h, [255](#)
- TRANSFORMER\_BAND\_MODE\_SID
  - m\_transformer\_enum.h, [256](#)
- TRANSFORMER\_BAND\_PASS\_FILTER
  - m\_transformer\_enum.h, [256](#)
- TRANSFORMER\_COMPRESSOR
  - m\_transformer\_enum.h, [256](#)
- TRANSFORMER\_DELAY
  - m\_transformer\_enum.h, [256](#)
- TRANSFORMER\_DIRTY\_OCTAVE
  - m\_transformer\_enum.h, [256](#)
- TRANSFORMER\_DISTORTION
  - m\_transformer\_enum.h, [256](#)
- TRANSFORMER\_ENVELOPE
  - m\_transformer\_enum.h, [256](#)
- TRANSFORMER\_FLANGER
  - m\_transformer\_enum.h, [256](#)
- transformer\_get\_parameter
  - m\_eng\_transformer.c, [207](#)
  - m\_eng\_transformer.h, [146](#)
- transformer\_get\_setting
  - m\_eng\_transformer.c, [207](#)
  - m\_eng\_transformer.h, [146](#)
- TRANSFORMER\_HIGH\_PASS\_FILTER
  - m\_transformer\_enum.h, [256](#)
- transformer\_id
  - m\_parameter\_id, [45](#)
  - m\_setting\_id, [49](#)
- transformer\_init\_controls
  - m\_eng\_transformer.c, [207](#)
  - m\_eng\_transformer.h, [146](#)
- transformer\_init\_parameter\_array
  - m\_eng\_transformer.c, [207](#)
  - m\_eng\_transformer.h, [146](#)
- transformer\_init\_setting\_array
  - m\_eng\_transformer.c, [207](#)
  - m\_eng\_transformer.h, [146](#)
- TRANSFORMER\_LOW\_END\_COMPRESSOR
  - m\_transformer\_enum.h, [256](#)
- TRANSFORMER\_LOW\_PASS\_FILTER
  - m\_transformer\_enum.h, [257](#)
- TRANSFORMER\_MAX\_INPUTS
  - m\_eng\_transformer.h, [145](#)
- TRANSFORMER\_MAX\_OUTPUTS
  - m\_eng\_transformer.h, [145](#)
- TRANSFORMER\_MODE\_BAND
  - m\_transformer\_enum.h, [257](#)
- TRANSFORMER\_MODE\_FULL\_SPECTRUM
  - m\_transformer\_enum.h, [257](#)
- TRANSFORMER\_MODE\_LOWER\_SPECTRUM
  - m\_transformer\_enum.h, [257](#)
- TRANSFORMER\_MODE\_UPPER\_SPECTRUM
  - m\_transformer\_enum.h, [257](#)
- TRANSFORMER\_NOISE\_SUPPRESSOR
  - m\_transformer\_enum.h, [257](#)
- TRANSFORMER\_PERCUSSIFIER
  - m\_transformer\_enum.h, [257](#)
- TRANSFORMER\_SETTING\_BOOL
  - m\_parameter.h, [248](#)
- TRANSFORMER\_SETTING\_ENUM
  - m\_parameter.h, [248](#)
- TRANSFORMER\_SETTING\_INT
  - m\_parameter.h, [248](#)
- TRANSFORMER\_SETTING\_PAGE\_MAIN
  - m\_parameter.h, [248](#)
- TRANSFORMER\_SETTING\_PAGE\_SETTINGS
  - m\_parameter.h, [248](#)
- TRANSFORMER\_SWITCH\_ACTION\_BYPASS
  - m\_eng\_transformer.h, [145](#)
- TRANSFORMER\_TRANSITION\_BIBLOCK\_LINEAR
  - m\_eng\_transformer.h, [145](#)
- TRANSFORMER\_TRANSITION\_INSTANT
  - m\_eng\_transformer.h, [145](#)
- TRANSFORMER\_TRANSITION\_MONOBLOCK\_LINEAR
  - m\_eng\_transformer.h, [145](#)
- TRANSFORMER\_TRANSITION\_QUADBLOCK\_LINEAR
  - m\_eng\_transformer.h, [145](#)
- TRANSFORMER\_TRANSITION\_TAIL
  - m\_eng\_transformer.h, [145](#)
- transformer\_type\_to\_string
  - m\_eng\_transformer.h, [147](#)
  - m\_transformer\_enum.c, [254](#)
  - m\_transformer\_enum.h, [258](#)
- transformer\_type\_valid
  - m\_transformer\_enum.c, [254](#)
  - m\_transformer\_enum.h, [258](#)
- TRANSFORMER\_WARBLER
  - m\_transformer\_enum.h, [257](#)
- TRANSFORMER\_WET\_MIX\_PID
  - m\_transformer\_enum.h, [257](#)
- TRANSFORMERS\_MALLOC\_CHUNK\_SIZE
  - m\_eng\_pipeline.c, [193](#)
- TRANSITION\_MONOBLOCK\_COS2
  - m\_eng\_transition.h, [149](#)
- TRANSITION\_OCTOBLOCK\_COS2
  - m\_eng\_transition.h, [149](#)
- transition\_policy
  - m\_eng\_profile, [35](#)

- TRANSITION\_QUADBLOCK\_COS2
  - m\_eng\_transition.h, 150
- TRANSITION\_TAIL
  - m\_eng\_transition.h, 150
- trig\_transition\_function
  - m\_eng.h, 56
  - m\_eng\_useful\_functions.c, 212
  - m\_eng\_useful\_functions.h, 152
- trigger\_threshold
  - m\_eng\_percussifier\_str, 33
- type
  - m\_eng\_biquad\_str, 14
  - m\_eng\_log\_entry, 27
  - m\_message, 44
  - m\_pipeline\_mod, 46
  - m\_response, 47
  - m\_transformer, 50
- update\_all
  - m\_eng\_update.c, 210
  - m\_eng\_update.h, 151
- update\_scheduled
  - m\_eng\_globals.c, 177
  - m\_eng\_globals.h, 89
- update\_setting
  - m\_eng\_parameter.c, 190
- update\_setup
  - m\_eng\_update.c, 210
  - m\_eng\_update.h, 151
- update\_stop
  - m\_eng\_update.c, 210
  - m\_eng\_update.h, 151
- update\_upper\_cycles
  - m\_eng\_globals.h, 88
- updated
  - m\_parameter, 45
  - m\_setting, 48
- USE\_GLOBAL\_TEMP\_BUFFERS
  - m\_eng\_distortion.h, 79
- valid
  - m\_delay\_buffer, 8
- valid\_m\_message\_type
  - m\_comms.c, 219
  - m\_comms.h, 229
- valid\_m\_response\_type
  - m\_comms.c, 219
  - m\_comms.h, 229
- value
  - m\_parameter, 45
  - m\_setting, 48
- voice\_colour\_table
  - m\_eng\_printf.cpp, 199
- wait\_message
  - m\_eng\_comms.cpp, 163
- WAVESHAPER\_ENVELOPE\_ATTACK
  - m\_eng\_waveshaper.h, 155
- WAVESHAPER\_ENVELOPE\_RELEASE
  - m\_eng\_waveshaper.h, 155
- wet\_mix
  - m\_eng\_distortion\_str, 20
  - m\_eng\_flanger\_str, 24
  - m\_transformer, 50
- width
  - m\_eng\_envelope\_str, 22
  - m\_eng\_warbler\_str, 38
- x1
  - m\_eng\_band\_pass\_filter\_str, 12
  - m\_eng\_biquad\_str, 14
  - m\_eng\_high\_pass\_filter\_str, 25
  - m\_eng\_low\_pass\_filter\_str, 28
- x2
  - m\_eng\_band\_pass\_filter\_str, 12
  - m\_eng\_biquad\_str, 14
  - m\_eng\_high\_pass\_filter\_str, 25
  - m\_eng\_low\_pass\_filter\_str, 29
- x\_11
  - m\_lr\_high\_pass\_filter\_str, 40
  - m\_lr\_low\_pass\_filter\_str, 42
- x\_12
  - m\_lr\_high\_pass\_filter\_str, 40
  - m\_lr\_low\_pass\_filter\_str, 42
- x\_21
  - m\_lr\_high\_pass\_filter\_str, 40
  - m\_lr\_low\_pass\_filter\_str, 42
- x\_22
  - m\_lr\_high\_pass\_filter\_str, 40
  - m\_lr\_low\_pass\_filter\_str, 42
- XSTR
  - m\_eng\_logging.h, 98
- y1
  - m\_eng\_band\_pass\_filter\_str, 12
  - m\_eng\_biquad\_str, 14
  - m\_eng\_high\_pass\_filter\_str, 25
  - m\_eng\_low\_pass\_filter\_str, 29
- y2
  - m\_eng\_band\_pass\_filter\_str, 12
  - m\_eng\_biquad\_str, 14
  - m\_eng\_high\_pass\_filter\_str, 25
  - m\_eng\_low\_pass\_filter\_str, 29
- y\_11
  - m\_lr\_high\_pass\_filter\_str, 40
  - m\_lr\_low\_pass\_filter\_str, 42
- y\_12
  - m\_lr\_high\_pass\_filter\_str, 41
  - m\_lr\_low\_pass\_filter\_str, 42
- y\_21
  - m\_lr\_high\_pass\_filter\_str, 41
  - m\_lr\_low\_pass\_filter\_str, 43
- y\_22
  - m\_lr\_high\_pass\_filter\_str, 41
  - m\_lr\_low\_pass\_filter\_str, 43
- zero\_buffer
  - m\_eng\_mempool.c, 188

m\_eng\_mempool.h, [106](#)