

CELT
0.7.0

Generated by Doxygen 1.6.1

Mon Oct 26 10:06:35 2009

Contents

1	Module Index	1
1.1	Modules	1
2	Data Structure Index	3
2.1	Data Structures	3
3	File Index	5
3.1	File List	5
4	Module Documentation	7
4.1	Encoding and decoding	7
4.1.1	Detailed Description	7
4.1.2	Function Documentation	7
4.1.2.1	celt_decode	7
4.1.2.2	celt_decode_float	8
4.1.2.3	celt_decoder_create	8
4.1.2.4	celt_decoder_ctl	9
4.1.2.5	celt_decoder_destroy	9
4.1.2.6	celt_encode	9
4.1.2.7	celt_encode_float	10
4.1.2.8	celt_encoder_create	10
4.1.2.9	celt_encoder_ctl	11
4.1.2.10	celt_encoder_destroy	11
4.1.2.11	celt_mode_create	11
4.1.2.12	celt_mode_destroy	12
4.1.2.13	celt_mode_info	12
4.1.2.14	celt_strerror	12
5	Data Structure Documentation	13
5.1	c64_fft_t Struct Reference	13

5.1.1	Detailed Description	13
5.1.2	Field Documentation	13
5.1.2.1	itwiddle	13
5.1.2.2	nfft	13
5.1.2.3	shift	13
5.1.2.4	twiddle	14
5.2	CELTDecoder Struct Reference	15
5.2.1	Detailed Description	15
5.2.2	Field Documentation	15
5.2.2.1	block_size	15
5.2.2.2	buf	15
5.2.2.3	channels	15
5.2.2.4	decode_mem	15
5.2.2.5	enc	16
5.2.2.6	frame_size	16
5.2.2.7	last_pitch_index	16
5.2.2.8	loss_count	16
5.2.2.9	marker	16
5.2.2.10	mode	16
5.2.2.11	oldBandE	16
5.2.2.12	out_mem	16
5.2.2.13	overlap	16
5.2.2.14	preemph_memD	17
5.3	CELTEncoder Struct Reference	18
5.3.1	Detailed Description	18
5.3.2	Field Documentation	18
5.3.2.1	block_size	18
5.3.2.2	channels	18
5.3.2.3	delayedIntra	19
5.3.2.4	fold_decision	19
5.3.2.5	force_intra	19
5.3.2.6	frame_size	19
5.3.2.7	gain_prod	19
5.3.2.8	in_mem	19
5.3.2.9	marker	19
5.3.2.10	mode	19

5.3.2.11	oldBandE	19
5.3.2.12	out_mem	20
5.3.2.13	overlap	20
5.3.2.14	pitch_available	20
5.3.2.15	pitch_enabled	20
5.3.2.16	pitch_permitted	20
5.3.2.17	preemph_memD	20
5.3.2.18	preemph_memE	20
5.3.2.19	tonal_average	20
5.3.2.20	vbr_count	20
5.3.2.21	vbr_drift	21
5.3.2.22	vbr_offset	21
5.3.2.23	vbr_rate	21
5.3.2.24	vbr_reservoir	21
5.4	CELTHHeader Struct Reference	22
5.4.1	Detailed Description	22
5.4.2	Field Documentation	22
5.4.2.1	bytes_per_packet	22
5.4.2.2	codec_id	22
5.4.2.3	codec_version	22
5.4.2.4	extra_headers	23
5.4.2.5	frame_size	23
5.4.2.6	header_size	23
5.4.2.7	nb_channels	23
5.4.2.8	overlap	23
5.4.2.9	sample_rate	23
5.4.2.10	version_id	23
5.5	CELTMode Struct Reference	25
5.5.1	Detailed Description	25
5.5.2	Field Documentation	25
5.5.2.1	allocVectors	25
5.5.2.2	bits	26
5.5.2.3	eBands	26
5.5.2.4	ePredCoef	26
5.5.2.5	fft	26
5.5.2.6	Fs	26

5.5.2.7	marker_end	26
5.5.2.8	marker_start	26
5.5.2.9	mdct	27
5.5.2.10	mdctSize	27
5.5.2.11	nbAllocVectors	27
5.5.2.12	nbEBands	27
5.5.2.13	nbShortMdcts	27
5.5.2.14	overlap	27
5.5.2.15	pitchEnd	27
5.5.2.16	prob	28
5.5.2.17	psy	28
5.5.2.18	shortMdct	28
5.5.2.19	shortMdctSize	28
5.5.2.20	shortWindow	28
5.5.2.21	window	28
5.6	ec_byte_buffer Struct Reference	29
5.6.1	Detailed Description	29
5.6.2	Field Documentation	29
5.6.2.1	buf	29
5.6.2.2	end_ptr	29
5.6.2.3	ptr	29
5.6.2.4	storage	29
5.7	ec_dec Struct Reference	30
5.7.1	Detailed Description	30
5.7.2	Field Documentation	30
5.7.2.1	buf	30
5.7.2.2	dif	30
5.7.2.3	end_bits_left	30
5.7.2.4	end_byte	30
5.7.2.5	nb_end_bits	30
5.7.2.6	nrm	31
5.7.2.7	rem	31
5.7.2.8	rng	31
5.8	ec_enc Struct Reference	32
5.8.1	Detailed Description	32
5.8.2	Field Documentation	32

5.8.2.1	buf	32
5.8.2.2	end_bits_left	32
5.8.2.3	end_byte	32
5.8.2.4	ext	32
5.8.2.5	low	32
5.8.2.6	nb_end_bits	33
5.8.2.7	rem	33
5.8.2.8	rng	33
5.9	kiss_fft_cpx Struct Reference	34
5.9.1	Detailed Description	34
5.9.2	Field Documentation	34
5.9.2.1	i	34
5.9.2.2	r	34
5.10	kiss_fft_state Struct Reference	35
5.10.1	Detailed Description	35
5.10.2	Field Documentation	35
5.10.2.1	bitrev	35
5.10.2.2	factors	35
5.10.2.3	nfft	35
5.10.2.4	scale	35
5.10.2.5	twiddles	35
5.11	kiss_fftr_state Struct Reference	36
5.11.1	Detailed Description	36
5.11.2	Field Documentation	36
5.11.2.1	substate	36
5.11.2.2	super_twiddles	36
5.12	kiss_twiddle_cpx Struct Reference	37
5.12.1	Detailed Description	37
5.12.2	Field Documentation	37
5.12.2.1	i	37
5.12.2.2	r	37
5.13	mdct_lookup Struct Reference	38
5.13.1	Detailed Description	38
5.13.2	Field Documentation	38
5.13.2.1	kfft	38
5.13.2.2	n	38

5.13.2.3	trig	38
5.14	PsyDecay Struct Reference	39
5.14.1	Detailed Description	39
5.14.2	Field Documentation	39
5.14.2.1	decayR	39
6	File Documentation	41
6.1	libcelt/_kiss_fft_guts.h File Reference	41
6.1.1	Define Documentation	42
6.1.1.1	C_ADD	42
6.1.1.2	C_ADDTO	42
6.1.1.3	C_FIXDIV	42
6.1.1.4	C_MUL	42
6.1.1.5	C_MUL4	42
6.1.1.6	C_MULBYSCALAR	42
6.1.1.7	C_MULC	43
6.1.1.8	C_SUB	43
6.1.1.9	C_SUBFROM	43
6.1.1.10	CHECK_OVERFLOW_OP	43
6.1.1.11	EXT32	43
6.1.1.12	HALF_OF	43
6.1.1.13	kf_cexp	44
6.1.1.14	kf_cexp2	44
6.1.1.15	KISS_FFT_COS	44
6.1.1.16	KISS_FFT_SIN	44
6.1.1.17	MAX	44
6.1.1.18	MAXFACTORS	44
6.1.1.19	MIN	44
6.1.1.20	S_MUL	44
6.2	libcelt/arch.h File Reference	45
6.2.1	Detailed Description	47
6.2.2	Define Documentation	47
6.2.2.1	ABS	47
6.2.2.2	ABS16	47
6.2.2.3	ABS32	47
6.2.2.4	ADD16	47
6.2.2.5	ADD32	47

6.2.2.6	BITS_PER_CHAR	48
6.2.2.7	BYTES_PER_CHAR	48
6.2.2.8	celt_assert	48
6.2.2.9	celt_assert2	48
6.2.2.10	celt_fatal	48
6.2.2.11	CELT_SIG_SCALE	48
6.2.2.12	DIV32	48
6.2.2.13	DIV32_16	48
6.2.2.14	ENER_SCALING	48
6.2.2.15	ENER_SCALING_1	48
6.2.2.16	EPSILON	49
6.2.2.17	EXTEND32	49
6.2.2.18	EXTRACT16	49
6.2.2.19	GLOBAL_STACK_SIZE	49
6.2.2.20	HALF32	49
6.2.2.21	IMAX	49
6.2.2.22	IMIN	49
6.2.2.23	IMUL32	49
6.2.2.24	LOG2_BITS_PER_CHAR	50
6.2.2.25	MAC16_16	50
6.2.2.26	MAC16_16_P13	50
6.2.2.27	MAC16_16_Q11	50
6.2.2.28	MAC16_16_Q13	50
6.2.2.29	MAC16_32_Q11	50
6.2.2.30	MAC16_32_Q15	50
6.2.2.31	MAX16	50
6.2.2.32	MAX32	50
6.2.2.33	MIN16	51
6.2.2.34	MIN32	51
6.2.2.35	MULT16_16	51
6.2.2.36	MULT16_16_16	51
6.2.2.37	MULT16_16_P13	51
6.2.2.38	MULT16_16_P14	51
6.2.2.39	MULT16_16_P15	51
6.2.2.40	MULT16_16_Q11_32	51
6.2.2.41	MULT16_16_Q13	51

6.2.2.42	MULT16_16_Q14	52
6.2.2.43	MULT16_16_Q15	52
6.2.2.44	MULT16_32_P15	52
6.2.2.45	MULT16_32_Q11	52
6.2.2.46	MULT16_32_Q13	52
6.2.2.47	MULT16_32_Q14	52
6.2.2.48	MULT16_32_Q15	52
6.2.2.49	MULT16_32_Q16	52
6.2.2.50	MULT32_32_Q31	52
6.2.2.51	NEG16	52
6.2.2.52	NEG32	53
6.2.2.53	NORM_SCALING	53
6.2.2.54	NORM_SCALING_1	53
6.2.2.55	PDIV32	53
6.2.2.56	PDIV32_16	53
6.2.2.57	PGAIN_SCALING	53
6.2.2.58	PGAIN_SCALING_1	53
6.2.2.59	PRINT_MIPS	53
6.2.2.60	PSHR	53
6.2.2.61	PSHR16	53
6.2.2.62	PSHR32	54
6.2.2.63	Q15_ONE	54
6.2.2.64	Q15_ONE_1	54
6.2.2.65	Q15ONE	54
6.2.2.66	Q30ONE	54
6.2.2.67	QCONST16	54
6.2.2.68	QCONST32	54
6.2.2.69	ROUND16	54
6.2.2.70	SATURATE	54
6.2.2.71	SATURATE16	55
6.2.2.72	SATURATE32	55
6.2.2.73	SCALEIN	55
6.2.2.74	SCALEOUT	55
6.2.2.75	SHL	55
6.2.2.76	SHL16	55
6.2.2.77	SHL32	55

6.2.2.78	SHR	55
6.2.2.79	SHR16	55
6.2.2.80	SHR32	56
6.2.2.81	SUB16	56
6.2.2.82	SUB32	56
6.2.2.83	UADD32	56
6.2.2.84	UMUL16_16	56
6.2.2.85	UMUL32	56
6.2.2.86	USUB32	56
6.2.2.87	VERY_LARGE16	56
6.2.2.88	VERY_LARGE32	56
6.2.2.89	VERY_SMALL	56
6.2.2.90	VSHR32	57
6.2.3	Typedef Documentation	57
6.2.3.1	celt_ener	57
6.2.3.2	celt_mask	57
6.2.3.3	celt_norm	57
6.2.3.4	celt_pgain	57
6.2.3.5	celt_sig	57
6.2.3.6	celt_word16	57
6.2.3.7	celt_word32	57
6.3	libcelt/bands.c File Reference	58
6.3.1	Function Documentation	58
6.3.1.1	apply_pitch	58
6.3.1.2	compute_band_energies	58
6.3.1.3	compute_pitch_gain	59
6.3.1.4	denormalise_bands	59
6.3.1.5	folding_decision	60
6.3.1.6	normalise_bands	60
6.3.1.7	quant_bands	60
6.3.1.8	quant_bands_stereo	60
6.3.1.9	renormalise_bands	61
6.3.1.10	unquant_bands_stereo	61
6.4	libcelt/bands.h File Reference	62
6.4.1	Function Documentation	62
6.4.1.1	apply_pitch	62

6.4.1.2	compute_band_energies	62
6.4.1.3	compute_pitch_gain	63
6.4.1.4	denormalise_bands	63
6.4.1.5	folding_decision	63
6.4.1.6	normalise_bands	64
6.4.1.7	quant_bands	64
6.4.1.8	quant_bands_stereo	64
6.4.1.9	renormalise_bands	64
6.4.1.10	stereo_decision	65
6.4.1.11	unquant_bands	65
6.4.1.12	unquant_bands_stereo	65
6.5	libcelt/c64_fft.c File Reference	66
6.5.1	Define Documentation	66
6.5.1.1	NBCACHE	66
6.5.1.2	PI	66
6.5.2	Function Documentation	66
6.5.2.1	c64_fft16_alloc	66
6.5.2.2	c64_fft16_free	67
6.5.2.3	c64_fft16_inplace	67
6.5.2.4	c64_fft32	67
6.5.2.5	c64_fft32_alloc	67
6.5.2.6	c64_fft32_free	67
6.5.2.7	c64_ifft16	67
6.5.2.8	c64_ifft32	67
6.5.2.9	gen_twiddle16	67
6.5.2.10	gen_twiddle32	68
6.6	libcelt/c64_fft.h File Reference	69
6.6.1	Function Documentation	69
6.6.1.1	c64_fft16_alloc	69
6.6.1.2	c64_fft16_free	69
6.6.1.3	c64_fft16_inplace	69
6.6.1.4	c64_fft32	69
6.6.1.5	c64_fft32_alloc	69
6.6.1.6	c64_fft32_free	70
6.6.1.7	c64_ifft16	70
6.6.1.8	c64_ifft32	70

6.7	libcelt/celt.c File Reference	71
6.7.1	Define Documentation	72
6.7.1.1	CELT_C	72
6.7.1.2	DECODE_BUFFER_SIZE	72
6.7.1.3	DECODERFREED	72
6.7.1.4	DECODERPARTIAL	72
6.7.1.5	DECODERVALID	72
6.7.1.6	ENCODERFREED	73
6.7.1.7	ENCODERPARTIAL	73
6.7.1.8	ENCODERVALID	73
6.7.1.9	FLAG_FOLD	73
6.7.1.10	FLAG_INTRA	73
6.7.1.11	FLAG_MASK	73
6.7.1.12	FLAG_NONE	73
6.7.1.13	FLAG_PITCH	73
6.7.1.14	FLAG_SHORT	73
6.7.2	Function Documentation	73
6.7.2.1	celt_decode	73
6.7.2.2	celt_decode_float	74
6.7.2.3	celt_decoder_ctl	74
6.7.2.4	celt_encode	74
6.7.2.5	celt_encode_float	74
6.7.2.6	celt_encoder_ctl	75
6.7.2.7	check_decoder	75
6.8	libcelt/celt.h File Reference	76
6.8.1	Detailed Description	77
6.8.2	Define Documentation	77
6.8.2.1	_celt_check_int	77
6.8.2.2	_celt_check_mode_ptr_ptr	77
6.8.2.3	CELT_ALLOC_FAIL	77
6.8.2.4	CELT_BAD_ARG	77
6.8.2.5	CELT_CORRUPTED_DATA	77
6.8.2.6	CELT_GET_BITSTREAM_VERSION	78
6.8.2.7	CELT_GET_FRAME_SIZE	78
6.8.2.8	CELT_GET_LOOKAHEAD	78
6.8.2.9	CELT_GET_MODE	78

6.8.2.10	CELT_GET_MODE_REQUEST	78
6.8.2.11	CELT_GET_SAMPLE_RATE	78
6.8.2.12	CELT_INTERNAL_ERROR	78
6.8.2.13	CELT_INVALID_MODE	78
6.8.2.14	CELT_INVALID_STATE	79
6.8.2.15	CELT_OK	79
6.8.2.16	CELT_RESET_STATE	79
6.8.2.17	CELT_RESET_STATE_REQUEST	79
6.8.2.18	CELT_SET_COMPLEXITY	79
6.8.2.19	CELT_SET_COMPLEXITY_REQUEST	79
6.8.2.20	CELT_SET_PREDICTION	80
6.8.2.21	CELT_SET_PREDICTION_REQUEST	80
6.8.2.22	CELT_SET_VBR_RATE	80
6.8.2.23	CELT_SET_VBR_RATE_REQUEST	80
6.8.2.24	CELT_UNIMPLEMENTED	80
6.8.2.25	EXPORT	80
6.8.3	Typedef Documentation	80
6.8.3.1	CELTDecoder	80
6.8.3.2	CELTEncoder	80
6.8.3.3	CELTMode	81
6.9	libcelt/celt_header.h File Reference	82
6.9.1	Function Documentation	82
6.9.1.1	celt_header_from_packet	82
6.9.1.2	celt_header_init	82
6.9.1.3	celt_header_to_packet	82
6.10	libcelt/celt_types.h File Reference	83
6.10.1	Detailed Description	83
6.10.2	Typedef Documentation	83
6.10.2.1	celt_int16	83
6.10.2.2	celt_int32	83
6.10.2.3	celt_uint16	83
6.10.2.4	celt_uint32	83
6.11	libcelt/cwrs.c File Reference	84
6.11.1	Define Documentation	84
6.11.1.1	MASK32	84
6.11.2	Function Documentation	84

6.11.2.1	decode_pulses	84
6.11.2.2	encode_pulses	84
6.11.2.3	fits_in32	85
6.11.2.4	get_required_bits	85
6.11.2.5	icwrs	85
6.11.2.6	log2_frac	85
6.12	libcelt/cwrs.h File Reference	86
6.12.1	Function Documentation	86
6.12.1.1	decode_pulses	86
6.12.1.2	encode_pulses	86
6.12.1.3	fits_in32	86
6.12.1.4	get_required_bits	86
6.12.1.5	log2_frac	86
6.13	libcelt/dump_modes.c File Reference	88
6.13.1	Define Documentation	88
6.13.1.1	FLOAT	88
6.13.1.2	INT16	88
6.13.1.3	INT32	88
6.13.1.4	WORD16	88
6.13.1.5	WORD32	88
6.13.2	Function Documentation	89
6.13.2.1	dump_header	89
6.13.2.2	dump_modes	89
6.13.2.3	main	89
6.14	libcelt/ecintrin.h File Reference	90
6.14.1	Define Documentation	90
6.14.1.1	_ecintrin_H	90
6.14.1.2	EC_CLAMPI	90
6.14.1.3	EC_ILOG	90
6.14.1.4	EC_ILOG64	90
6.14.1.5	EC_MAXI	90
6.14.1.6	EC_MINI	90
6.14.1.7	EC_SIGNI	91
6.14.1.8	EC_SIGNMASK	91
6.15	libcelt/entcode.c File Reference	92
6.15.1	Function Documentation	92

6.15.1.1	ec_ilog	92
6.16	libcelt/entcode.h File Reference	93
6.16.1	Define Documentation	93
6.16.1.1	_entcode_H	93
6.16.1.2	EC_UNIT_BITS	94
6.16.1.3	EC_UNIT_MASK	94
6.16.2	Typedef Documentation	94
6.16.2.1	ec_byte_buffer	94
6.16.2.2	ec_int32	94
6.16.2.3	ec_uint32	94
6.16.3	Function Documentation	94
6.16.3.1	ec_byte_adv1	94
6.16.3.2	ec_byte_adv4	94
6.16.3.3	ec_byte_look1	94
6.16.3.4	ec_byte_look4	94
6.16.3.5	ec_byte_look_at_end	94
6.16.3.6	ec_byte_read1	94
6.16.3.7	ec_byte_read4	95
6.16.3.8	ec_byte_readinit	95
6.16.3.9	ec_byte_shrink	95
6.16.3.10	ec_byte_writel	95
6.16.3.11	ec_byte_write4	95
6.16.3.12	ec_byte_write_at_end	95
6.16.3.13	ec_byte_writeclear	95
6.16.3.14	ec_byte_writecopy	95
6.16.3.15	ec_byte_writeinit	95
6.16.3.16	ec_byte_writeinit_buffer	95
6.16.3.17	ec_byte_writetrunc	96
6.16.3.18	ec_ilog	96
6.17	libcelt/entdec.c File Reference	97
6.17.1	Function Documentation	97
6.17.1.1	ec_byte_adv1	97
6.17.1.2	ec_byte_look_at_end	97
6.17.1.3	ec_byte_read1	97
6.17.1.4	ec_byte_readinit	97
6.17.1.5	ec_dec_bits	97

6.17.1.6	<code>ec_dec_uint</code>	98
6.18	<code>libcelt/entdec.h</code> File Reference	99
6.18.1	Define Documentation	99
6.18.1.1	<code>_entdec_H</code>	99
6.18.2	Typedef Documentation	99
6.18.2.1	<code>ec_dec</code>	99
6.18.3	Function Documentation	99
6.18.3.1	<code>ec_dec_bits</code>	99
6.18.3.2	<code>ec_dec_init</code>	100
6.18.3.3	<code>ec_dec_tell</code>	100
6.18.3.4	<code>ec_dec_uint</code>	100
6.18.3.5	<code>ec_dec_update</code>	100
6.18.3.6	<code>ec_decode</code>	100
6.18.3.7	<code>ec_decode_bin</code>	100
6.18.3.8	<code>ec_decode_raw</code>	100
6.19	<code>libcelt/entenc.c</code> File Reference	102
6.19.1	Define Documentation	102
6.19.1.1	<code>EC_BUFFER_INCREMENT</code>	102
6.19.2	Function Documentation	102
6.19.2.1	<code>ec_byte_shrink</code>	102
6.19.2.2	<code>ec_byte_writel</code>	102
6.19.2.3	<code>ec_byte_write_at_end</code>	102
6.19.2.4	<code>ec_byte_writeinit_buffer</code>	103
6.19.2.5	<code>ec_enc_bits</code>	103
6.19.2.6	<code>ec_enc_uint</code>	103
6.20	<code>libcelt/entenc.h</code> File Reference	104
6.20.1	Define Documentation	104
6.20.1.1	<code>_entenc_H</code>	104
6.20.2	Typedef Documentation	104
6.20.2.1	<code>ec_enc</code>	104
6.20.3	Function Documentation	104
6.20.3.1	<code>ec_enc_bits</code>	104
6.20.3.2	<code>ec_enc_done</code>	105
6.20.3.3	<code>ec_enc_init</code>	105
6.20.3.4	<code>ec_enc_tell</code>	105
6.20.3.5	<code>ec_enc_uint</code>	105

6.20.3.6	ec_encode	105
6.20.3.7	ec_encode_bin	105
6.20.3.8	ec_encode_raw	105
6.21	libcelt/fixed_c5x.h File Reference	107
6.21.1	Detailed Description	107
6.21.2	Define Documentation	107
6.21.2.1	celt_ilog2	107
6.21.2.2	celt_maxabs16	107
6.21.2.3	MAX16	107
6.21.2.4	MAX32	107
6.21.2.5	MIN16	108
6.21.2.6	MIN32	108
6.21.2.7	MULT16_16_Q15	108
6.21.2.8	MULT16_16SU	108
6.21.2.9	MULT16_32_Q15	108
6.21.2.10	MULT_16_16	108
6.21.2.11	OVERRIDE_CELT_ILOG2	108
6.21.2.12	OVERRIDE_CELT_MAXABS16	108
6.21.2.13	OVERRIDE_FIND_MAX16	108
6.21.2.14	VSHR32	108
6.22	libcelt/fixed_c6x.h File Reference	109
6.22.1	Detailed Description	109
6.22.2	Define Documentation	109
6.22.2.1	celt_ilog2	109
6.22.2.2	MULT16_16SU	109
6.22.2.3	MULT16_32_Q15	109
6.22.2.4	MULT_16_16	109
6.22.2.5	OVERRIDE_CELT_ILOG2	109
6.23	libcelt/fixed_debug.h File Reference	110
6.23.1	Detailed Description	111
6.23.2	Define Documentation	111
6.23.2.1	ADD16	111
6.23.2.2	ADD32	111
6.23.2.3	DIV32	111
6.23.2.4	DIV32_16	111
6.23.2.5	EXTEND32	111

6.23.2.6	EXTRACT16	111
6.23.2.7	HALF32	112
6.23.2.8	MAC16_16	112
6.23.2.9	MAC16_16_P13	112
6.23.2.10	MAC16_16_Q11	112
6.23.2.11	MAC16_16_Q13	112
6.23.2.12	MAC16_32_Q11	112
6.23.2.13	MAC16_32_Q15	112
6.23.2.14	MIPS_INC	112
6.23.2.15	MULT16_16	112
6.23.2.16	MULT16_16_Q15	112
6.23.2.17	MULT16_16SU	112
6.23.2.18	MULT16_32_P15	113
6.23.2.19	MULT16_32_Q11	113
6.23.2.20	MULT16_32_Q12	113
6.23.2.21	MULT16_32_Q13	113
6.23.2.22	MULT16_32_Q14	113
6.23.2.23	MULT16_32_Q15	113
6.23.2.24	MULT16_32_Q16	113
6.23.2.25	MULT16_32_QX	113
6.23.2.26	MULT32_32_Q31	113
6.23.2.27	PDIV32	113
6.23.2.28	PDIV32_16	114
6.23.2.29	PRINT_MIPS	114
6.23.2.30	PSHR	114
6.23.2.31	PSHR16	114
6.23.2.32	PSHR32	114
6.23.2.33	QCONST16	114
6.23.2.34	QCONST32	114
6.23.2.35	ROUND16	114
6.23.2.36	SATURATE16	114
6.23.2.37	SATURATE32	114
6.23.2.38	SHL16	114
6.23.2.39	SHR	115
6.23.2.40	SHR16	115
6.23.2.41	SUB16	115

6.23.2.42	SUB32	115
6.23.2.43	UADD32	115
6.23.2.44	USUB32	115
6.23.2.45	VERIFY_INT	115
6.23.2.46	VERIFY_SHORT	115
6.23.2.47	VERIFY_UINT	115
6.23.2.48	VSHR32	115
6.23.3	Variable Documentation	115
6.23.3.1	celt_mips	115
6.24	libcelt/fixed_generic.h File Reference	116
6.24.1	Detailed Description	117
6.24.2	Define Documentation	117
6.24.2.1	ADD16	117
6.24.2.2	ADD32	117
6.24.2.3	DIV32	117
6.24.2.4	DIV32_16	118
6.24.2.5	EXTEND32	118
6.24.2.6	EXTRACT16	118
6.24.2.7	HALF32	118
6.24.2.8	MAC16_16	118
6.24.2.9	MAC16_16_P13	118
6.24.2.10	MAC16_16_Q11	118
6.24.2.11	MAC16_16_Q13	118
6.24.2.12	MAC16_32_Q11	118
6.24.2.13	MAC16_32_Q15	119
6.24.2.14	MULT16_16	119
6.24.2.15	MULT16_16_16	119
6.24.2.16	MULT16_16_P13	119
6.24.2.17	MULT16_16_P14	119
6.24.2.18	MULT16_16_P15	119
6.24.2.19	MULT16_16_Q11_32	119
6.24.2.20	MULT16_16_Q13	119
6.24.2.21	MULT16_16_Q14	119
6.24.2.22	MULT16_16_Q15	119
6.24.2.23	MULT16_16SU	120
6.24.2.24	MULT16_32_P15	120

6.24.2.25 MULT16_32_Q11	120
6.24.2.26 MULT16_32_Q12	120
6.24.2.27 MULT16_32_Q13	120
6.24.2.28 MULT16_32_Q14	120
6.24.2.29 MULT16_32_Q15	120
6.24.2.30 MULT16_32_Q16	120
6.24.2.31 MULT32_32_Q31	121
6.24.2.32 MULT32_32_Q32	121
6.24.2.33 NEG16	121
6.24.2.34 NEG32	121
6.24.2.35 PDIV32	121
6.24.2.36 PDIV32_16	121
6.24.2.37 PSHR	121
6.24.2.38 PSHR16	121
6.24.2.39 PSHR32	122
6.24.2.40 QCONST16	122
6.24.2.41 QCONST32	122
6.24.2.42 ROUND16	122
6.24.2.43 SATURATE	122
6.24.2.44 SATURATE16	122
6.24.2.45 SATURATE32	122
6.24.2.46 SHL	122
6.24.2.47 SHL16	122
6.24.2.48 SHL32	123
6.24.2.49 SHR	123
6.24.2.50 SHR16	123
6.24.2.51 SHR32	123
6.24.2.52 SUB16	123
6.24.2.53 SUB32	123
6.24.2.54 VSHR32	123
6.25 libcelt/float_cast.h File Reference	124
6.25.1 Define Documentation	124
6.25.1.1 float2int	124
6.26 libcelt/header.c File Reference	125
6.26.1 Function Documentation	125
6.26.1.1 celt_header_from_packet	125

6.26.1.2	celt_header_init	125
6.26.1.3	celt_header_to_packet	125
6.27	libcelt/kfft_double.h File Reference	126
6.27.1	Define Documentation	126
6.27.1.1	cpx32_fft	126
6.27.1.2	cpx32_fft_alloc	126
6.27.1.3	cpx32_fft_free	126
6.27.1.4	cpx32_ifft	126
6.28	libcelt/kfft_single.c File Reference	127
6.29	libcelt/kfft_single.h File Reference	128
6.29.1	Define Documentation	128
6.29.1.1	BITREV	128
6.29.1.2	real16_fft_alloc	128
6.29.1.3	real16_fft_free	128
6.29.1.4	real16_fft_inplace	128
6.29.1.5	real16_ifft	128
6.30	libcelt/kiss_fft.c File Reference	129
6.30.1	Function Documentation	129
6.30.1.1	kf_work	129
6.30.1.2	ki_work	129
6.30.1.3	kiss_fft	129
6.30.1.4	kiss_fft_alloc	130
6.30.1.5	kiss_fft_stride	130
6.30.1.6	kiss_ifft	130
6.30.1.7	kiss_ifft_stride	130
6.31	libcelt/kiss_fft.h File Reference	131
6.31.1	Define Documentation	131
6.31.1.1	CAT_SUFFIX	131
6.31.1.2	KF_SUFFIX	132
6.31.1.3	kf_work	132
6.31.1.4	ki_work	132
6.31.1.5	kiss_fft	132
6.31.1.6	kiss_fft_alloc	132
6.31.1.7	kiss_fft_free	132
6.31.1.8	KISS_FFT_MALLOC	132
6.31.1.9	kiss_fft_scalar	132

6.31.1.10	<code>kiss_fft_stride</code>	132
6.31.1.11	<code>kiss_ifft</code>	133
6.31.1.12	<code>kiss_ifft_stride</code>	133
6.31.1.13	<code>kiss_twiddle_scalar</code>	133
6.31.1.14	<code>SUF</code>	133
6.31.2	Typedef Documentation	133
6.31.2.1	<code>kiss_fft_cfg</code>	133
6.31.3	Function Documentation	133
6.31.3.1	<code>kf_work</code>	133
6.31.3.2	<code>ki_work</code>	133
6.31.3.3	<code>kiss_fft</code>	133
6.31.3.4	<code>kiss_fft_alloc</code>	134
6.31.3.5	<code>kiss_fft_stride</code>	134
6.31.3.6	<code>kiss_ifft</code>	134
6.31.3.7	<code>kiss_ifft_stride</code>	134
6.32	<code>libcelt/kiss_fftr.c</code> File Reference	135
6.32.1	Function Documentation	135
6.32.1.1	<code>kiss_fftr</code>	135
6.32.1.2	<code>kiss_fftr_alloc</code>	135
6.32.1.3	<code>kiss_fftr_inplace</code>	135
6.32.1.4	<code>kiss_fftr_twiddles</code>	135
6.32.1.5	<code>kiss_fftri</code>	135
6.33	<code>libcelt/kiss_fftr.h</code> File Reference	136
6.33.1	Define Documentation	136
6.33.1.1	<code>kiss_fftr</code>	136
6.33.1.2	<code>kiss_fftr_alloc</code>	136
6.33.1.3	<code>kiss_fftr_alloc</code>	136
6.33.1.4	<code>kiss_fftr_free</code>	136
6.33.1.5	<code>kiss_fftr_inplace</code>	137
6.33.1.6	<code>kiss_fftr_twiddles</code>	137
6.33.1.7	<code>kiss_fftri</code>	137
6.33.2	Typedef Documentation	137
6.33.2.1	<code>kiss_fftr_cfg</code>	137
6.33.3	Function Documentation	137
6.33.3.1	<code>kiss_fftr</code>	137
6.33.3.2	<code>kiss_fftr_alloc</code>	137

6.33.3.3	<code>kiss_fftr_inplace</code>	137
6.33.3.4	<code>kiss_fftr_twiddles</code>	137
6.33.3.5	<code>kiss_fftri</code>	138
6.34	<code>libcelt/laplace.c</code> File Reference	139
6.34.1	Function Documentation	139
6.34.1.1	<code>ec_laplace_decode</code>	139
6.34.1.2	<code>ec_laplace_decode_start</code>	139
6.34.1.3	<code>ec_laplace_encode</code>	139
6.34.1.4	<code>ec_laplace_encode_start</code>	140
6.34.1.5	<code>ec_laplace_get_start_freq</code>	140
6.35	<code>libcelt/laplace.h</code> File Reference	141
6.35.1	Function Documentation	141
6.35.1.1	<code>ec_laplace_decode</code>	141
6.35.1.2	<code>ec_laplace_decode_start</code>	141
6.35.1.3	<code>ec_laplace_encode</code>	141
6.35.1.4	<code>ec_laplace_encode_start</code>	142
6.35.1.5	<code>ec_laplace_get_start_freq</code>	142
6.36	<code>libcelt/mathops.h</code> File Reference	143
6.36.1	Detailed Description	143
6.36.2	Define Documentation	143
6.36.2.1	<code>celt_acos</code>	143
6.36.2.2	<code>celt_atan</code>	143
6.36.2.3	<code>celt_cos_norm</code>	143
6.36.2.4	<code>celt_div</code>	143
6.36.2.5	<code>celt_exp</code>	144
6.36.2.6	<code>celt_exp2</code>	144
6.36.2.7	<code>celt_log2</code>	144
6.36.2.8	<code>celt_psqrt</code>	144
6.36.2.9	<code>celt_rcp</code>	144
6.36.2.10	<code>celt_rsqr</code>	144
6.36.2.11	<code>celt_sqrt</code>	144
6.36.2.12	<code>FRAC_MUL16</code>	144
6.37	<code>libcelt/mdct.c</code> File Reference	145
6.37.1	Define Documentation	145
6.37.1.1	<code>M_PI</code>	145
6.37.2	Function Documentation	145

6.37.2.1	mdct_backward	145
6.37.2.2	mdct_clear	145
6.37.2.3	mdct_forward	146
6.37.2.4	mdct_init	146
6.38	libcelt/mdct.h File Reference	147
6.38.1	Function Documentation	147
6.38.1.1	mdct_backward	147
6.38.1.2	mdct_clear	147
6.38.1.3	mdct_forward	147
6.38.1.4	mdct_init	147
6.39	libcelt/mfrngcod.h File Reference	148
6.39.1	Define Documentation	148
6.39.1.1	_mfrngcode_H	148
6.39.1.2	EC_CODE_BITS	148
6.39.1.3	EC_CODE_BOT	148
6.39.1.4	EC_CODE_CARRY	148
6.39.1.5	EC_CODE_EXTRA	148
6.39.1.6	EC_CODE_MASK	148
6.39.1.7	EC_CODE_SHIFT	149
6.39.1.8	EC_CODE_TOP	149
6.39.1.9	EC_SYM_BITS	149
6.39.1.10	EC_SYM_MAX	149
6.40	libcelt/mfrngdec.c File Reference	150
6.40.1	Function Documentation	150
6.40.1.1	ec_dec_init	150
6.40.1.2	ec_dec_tell	150
6.40.1.3	ec_dec_update	150
6.40.1.4	ec_decode	150
6.40.1.5	ec_decode_bin	150
6.41	libcelt/mfrngenc.c File Reference	151
6.41.1	Function Documentation	151
6.41.1.1	ec_enc_done	151
6.41.1.2	ec_enc_init	151
6.41.1.3	ec_enc_tell	151
6.41.1.4	ec_encode	151
6.41.1.5	ec_encode_bin	151

6.42	libcelt/modes.c File Reference	152
6.42.1	Define Documentation	152
6.42.1.1	BARK_BANDS	152
6.42.1.2	BITALLOC_SIZE	152
6.42.1.3	MODEFREED	152
6.42.1.4	MODEPARTIAL	152
6.42.1.5	MODEVALID	153
6.42.2	Function Documentation	153
6.42.2.1	check_mode	153
6.43	libcelt/modes.h File Reference	154
6.43.1	Define Documentation	154
6.43.1.1	CELT_BITSTREAM_VERSION	154
6.43.1.2	CHANNELS	154
6.43.1.3	FRAMESIZE	155
6.43.1.4	MAX_PERIOD	155
6.43.1.5	MCHANNELS	155
6.43.1.6	MDCT	155
6.43.1.7	OVERLAP	155
6.43.2	Function Documentation	155
6.43.2.1	check_mode	155
6.44	libcelt/os_support.h File Reference	156
6.44.1	Define Documentation	156
6.44.1.1	CELT_COPY	156
6.44.1.2	CELT_MEMSET	156
6.44.1.3	CELT_MOVE	156
6.45	libcelt/pitch.c File Reference	157
6.45.1	Detailed Description	157
6.45.2	Define Documentation	157
6.45.2.1	INPUT_SHIFT	157
6.45.2.2	normalise16	157
6.45.3	Function Documentation	158
6.45.3.1	find_spectral_pitch	158
6.45.3.2	pitch_state_alloc	158
6.45.3.3	pitch_state_free	158
6.46	libcelt/pitch.h File Reference	159
6.46.1	Detailed Description	159

6.46.2	Function Documentation	159
6.46.2.1	find_spectral_pitch	159
6.46.2.2	pitch_state_alloc	159
6.46.2.3	pitch_state_free	159
6.47	libcelt/psy.c File Reference	160
6.47.1	Define Documentation	160
6.47.1.1	fromBARK	160
6.47.1.2	toBARK	160
6.47.2	Function Documentation	160
6.47.2.1	compute_masking	160
6.47.2.2	psydecay_clear	160
6.47.2.3	psydecay_init	161
6.48	libcelt/psy.h File Reference	162
6.48.1	Function Documentation	162
6.48.1.1	compute_masking	162
6.48.1.2	compute_mdct_masking	162
6.48.1.3	compute_tonality	162
6.48.1.4	psydecay_clear	162
6.48.1.5	psydecay_init	162
6.49	libcelt/quant_bands.c File Reference	164
6.49.1	Function Documentation	164
6.49.1.1	intra_decision	164
6.49.1.2	quant_coarse_energy	164
6.49.1.3	quant_energy_finalise	165
6.49.1.4	quant_fine_energy	165
6.49.1.5	quant_prob_alloc	165
6.49.1.6	quant_prob_free	165
6.49.1.7	unquant_coarse_energy	165
6.49.1.8	unquant_energy_finalise	165
6.49.1.9	unquant_fine_energy	166
6.49.2	Variable Documentation	166
6.49.2.1	eMeans	166
6.50	libcelt/quant_bands.h File Reference	167
6.50.1	Function Documentation	167
6.50.1.1	compute_fine_allocation	167
6.50.1.2	intra_decision	167

6.50.1.3	quant_coarse_energy	167
6.50.1.4	quant_energy_finalise	168
6.50.1.5	quant_fine_energy	168
6.50.1.6	quant_prob_alloc	168
6.50.1.7	quant_prob_free	168
6.50.1.8	unquant_coarse_energy	168
6.50.1.9	unquant_energy_finalise	168
6.50.1.10	unquant_fine_energy	169
6.51	libcelt/rangedec.c File Reference	170
6.51.1	Function Documentation	170
6.51.1.1	ec_dec_init	170
6.51.1.2	ec_dec_tell	170
6.51.1.3	ec_dec_update	170
6.51.1.4	ec_decode	170
6.51.1.5	ec_decode_bin	171
6.51.1.6	ec_decode_raw	171
6.52	libcelt/rangeenc.c File Reference	172
6.52.1	Function Documentation	172
6.52.1.1	ec_enc_done	172
6.52.1.2	ec_enc_init	172
6.52.1.3	ec_enc_tell	172
6.52.1.4	ec_encode	172
6.52.1.5	ec_encode_bin	173
6.52.1.6	ec_encode_raw	173
6.53	libcelt/rate.c File Reference	174
6.53.1	Function Documentation	174
6.53.1.1	compute_alloc_cache	174
6.53.1.2	compute_allocation	174
6.54	libcelt/rate.h File Reference	175
6.54.1	Define Documentation	175
6.54.1.1	BITOVERFLOW	175
6.54.1.2	BITRES	175
6.54.1.3	FINE_OFFSET	175
6.54.1.4	LOG_MAX_PSEUDO	175
6.54.1.5	LOG_MAX_PULSES	175
6.54.1.6	MAX_PSEUDO	176

6.54.1.7	MAX_PULSES	176
6.54.1.8	QTHETA_OFFSET	176
6.54.2	Function Documentation	176
6.54.2.1	compute_alloc_cache	176
6.54.2.2	compute_allocation	176
6.55	libcelt/stack_alloc.h File Reference	177
6.55.1	Detailed Description	177
6.55.2	Define Documentation	177
6.55.2.1	ALIGN	177
6.55.2.2	ALLOC	177
6.55.2.3	ALLOC_STACK	178
6.55.2.4	PUSH	178
6.55.2.5	RESTORE_STACK	178
6.55.2.6	SAVE_STACK	178
6.55.2.7	VARDECL	178
6.55.3	Variable Documentation	179
6.55.3.1	global_stack	179
6.56	libcelt/testcelt.c File Reference	180
6.56.1	Define Documentation	180
6.56.1.1	MAX_PACKET	180
6.56.2	Function Documentation	180
6.56.2.1	main	180
6.57	libcelt/vq.c File Reference	181
6.57.1	Function Documentation	181
6.57.1.1	alg_quant	181
6.57.1.2	alg_unquant	181
6.57.1.3	intra_fold	182
6.57.1.4	renormalise_vector	182
6.58	libcelt/vq.h File Reference	183
6.58.1	Detailed Description	183
6.58.2	Function Documentation	183
6.58.2.1	alg_quant	183
6.58.2.2	alg_unquant	183
6.58.2.3	intra_fold	184
6.58.2.4	renormalise_vector	184

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Encoding and decoding	7
---------------------------------	---

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

c64_fft_t	13
CELTD decoder (Decoder state)	15
CELTEncoder (Encoder state)	18
CELTH header (Header data)	22
CELTMode (Mode definition)	25
ec_byte_buffer	29
ec_dec	30
ec_enc	32
kiss_fft_cpx	34
kiss_fft_state	35
kiss_fftr_state	36
kiss_twiddle_cpx	37
mdct_lookup	38
PsyDecay	39

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

libcelt/_kiss_fft_guts.h	41
libcelt/arch.h (Various architecture definitions for CELT)	45
libcelt/bands.c	58
libcelt/bands.h	62
libcelt/c64_fft.c	66
libcelt/c64_fft.h	69
libcelt/celt.c	71
libcelt/celt.h (Contains all the functions for encoding and decoding audio)	76
libcelt/celt_header.h	82
libcelt/celt_types.h (CELT types)	83
libcelt/cwrs.c	84
libcelt/cwrs.h	86
libcelt/dump_modes.c	88
libcelt/ecintrin.h	90
libcelt/entcode.c	92
libcelt/entcode.h	93
libcelt/entdec.c	97
libcelt/entdec.h	99
libcelt/entenc.c	102
libcelt/entenc.h	104
libcelt/fixed_c5x.h (Fixed-point operations for the TI C5x DSP family)	107
libcelt/fixed_c6x.h (Fixed-point operations for the TI C6x DSP family)	109
libcelt/fixed_debug.h (Fixed-point operations with debugging)	110
libcelt/fixed_generic.h (Generic fixed-point operations)	116
libcelt/float_cast.h	124
libcelt/header.c	125
libcelt/kfft_double.h	126
libcelt/kfft_single.c	127
libcelt/kfft_single.h	128
libcelt/kiss_fft.c	129
libcelt/kiss_fft.h	131
libcelt/kiss_fftr.c	135
libcelt/kiss_fftr.h	136

libcelt/laplace.c	139
libcelt/laplace.h	141
libcelt/mathops.h (Various math functions)	143
libcelt/mdct.c	145
libcelt/mdct.h	147
libcelt/mfrngcod.h	148
libcelt/mfrngdec.c	150
libcelt/mfrngenc.c	151
libcelt/modes.c	152
libcelt/modes.h	154
libcelt/os_support.h	156
libcelt/pitch.c (Pitch analysis)	157
libcelt/pitch.h (Pitch analysis)	159
libcelt/psy.c	160
libcelt/psy.h	162
libcelt/quant_bands.c	164
libcelt/quant_bands.h	167
libcelt/rangedec.c	170
libcelt/rangeenc.c	172
libcelt/rate.c	174
libcelt/rate.h	175
libcelt/stack_alloc.h (Temporary memory allocation on stack)	177
libcelt/testcelt.c	180
libcelt/vq.c	181
libcelt/vq.h (Vector quantisation of the residual)	183

Chapter 4

Module Documentation

4.1 Encoding and decoding

Functions

- EXPORT `CELTMode` * `celt_mode_create` (`celt_int32` Fs, int frame_size, int *error)
- EXPORT void `celt_mode_destroy` (`CELTMode` *mode)
- EXPORT int `celt_mode_info` (const `CELTMode` *mode, int request, `celt_int32` *value)
- EXPORT `CELTEncoder` * `celt_encoder_create` (const `CELTMode` *mode, int channels, int *error)
- EXPORT void `celt_encoder_destroy` (`CELTEncoder` *st)
- EXPORT int `celt_encode_float` (`CELTEncoder` *st, const float *pcm, float *optional_synthesis, unsigned char *compressed, int nbCompressedBytes)
- EXPORT int `celt_encode` (`CELTEncoder` *st, const `celt_int16` *pcm, `celt_int16` *optional_synthesis, unsigned char *compressed, int nbCompressedBytes)
- EXPORT int `celt_encoder_ctl` (`CELTEncoder` *st, int request,...)
- EXPORT `CELTDDecoder` * `celt_decoder_create` (const `CELTMode` *mode, int channels, int *error)
- EXPORT void `celt_decoder_destroy` (`CELTDDecoder` *st)
- EXPORT int `celt_decode_float` (`CELTDDecoder` *st, const unsigned char *data, int len, float *pcm)
- EXPORT int `celt_decode` (`CELTDDecoder` *st, const unsigned char *data, int len, `celt_int16` *pcm)
- EXPORT int `celt_decoder_ctl` (`CELTDDecoder` *st, int request,...)
- EXPORT const char * `celt_strerror` (int error)

4.1.1 Detailed Description

4.1.2 Function Documentation

4.1.2.1 EXPORT int `celt_decode` (`CELTDDecoder` * *st*, const unsigned char * *data*, int *len*, `celt_int16` * *pcm*)

Decodes a frame of audio.

Parameters:

st Decoder state

data Compressed data produced by an encoder

len Number of bytes to read from "data". This MUST be exactly the number of bytes returned by the encoder. Using a larger value WILL NOT WORK.

pcm One frame (frame_size samples per channel) of decoded PCM will be returned here in 16-bit PCM format (native endian).

Returns:

Error code.

4.1.2.2 EXPORT int celt_decode_float (CELTDecoder * st, const unsigned char * data, int len, float * pcm)

Decodes a frame of audio.

Parameters:

st Decoder state

data Compressed data produced by an encoder

len Number of bytes to read from "data". This MUST be exactly the number of bytes returned by the encoder. Using a larger value WILL NOT WORK.

pcm One frame (frame_size samples per channel) of decoded PCM will be returned here in float format.

Returns:

Error code.

4.1.2.3 EXPORT CELTDecoder* celt_decoder_create (const CELTMode * mode, int channels, int * error)

Creates a new decoder state. Each stream needs its own decoder state (can't be shared across simultaneous streams).

Parameters:

mode Contains all the information about the characteristics of the stream (must be the same characteristics as used for the encoder)

channels Number of channels

error Returns an error code

Returns:

Newly created decoder state.

Definition at line 1122 of file celt.c.

References CELTDecoder::block_size, CELT_ALLOC_FAIL, CELT_BAD_ARG, celt_decoder_destroy(), CELT_INVALID_MODE, CELT_OK, CELTDecoder::channels, CHANNELS, check_mode(), DECODE_BUFFER_SIZE, CELTDecoder::decode_mem, DECODERPARTIAL, DECODERVALID, CELTDecoder::frame_size, CELTDecoder::loss_count, CELTDecoder::marker, MAX_PERIOD, CELTMode::mdctSize, CELTDecoder::mode, CELTMode::nbEBands, CELTDecoder::oldBandE, CELTDecoder::out_mem, CELTMode::overlap, CELTDecoder::overlap, and CELTDecoder::preemph_memD.

Referenced by main().

4.1.2.4 EXPORT int celt_decoder_ctl (CELTDecoder * *st*, int *request*, ...)

Query and set decoder parameters

Parameters:

- st* Decoder state
- request* Parameter to change or query
- value* Pointer to a 32-bit int value

Returns:

Error code

4.1.2.5 EXPORT void celt_decoder_destroy (CELTDecoder * *st*)

Destroys a a decoder state.

Parameters:

- st* Decoder state to be destroyed

Definition at line 1193 of file celt.c.

References check_mode(), CELTDecoder::decode_mem, DECODERFREED, DECODERPARTIAL, DECODERVALID, CELTDecoder::marker, CELTDecoder::mode, CELTDecoder::oldBandE, and CELTDecoder::preemph_memD.

Referenced by celt_decoder_create(), and main().

4.1.2.6 EXPORT int celt_encode (CELTDecoder * *st*, const celt_int16 * *pcm*, celt_int16 * *optional_synthesis*, unsigned char * *compressed*, int *nbCompressedBytes*)

Encodes a frame of audio.

Parameters:

- st* Encoder state
- pcm* PCM audio in signed 16-bit format (native endian). There must be exactly frame_size samples per channel.
- optional_synthesis* If not NULL, the encoder copies the audio signal that the decoder would decode. It is the same as calling the decoder on the compressed data, just faster. This may alias pcm.
- compressed* The compressed data is written here. This may not alias pcm or optional_synthesis.
- nbCompressedBytes* Maximum number of bytes to use for compressing the frame (can change from one frame to another)

Returns:

Number of bytes written to "compressed". Will be the same as "nbCompressedBytes" unless the stream is VBR and will never be larger. If negative, an error has occurred (see error codes). It is IMPORTANT that the length returned be somehow transmitted to the decoder. Otherwise, no decoding is possible.

4.1.2.7 EXPORT int celt_encode_float (CELTEncoder * *st*, const float * *pcm*, float * *optional_synthesis*, unsigned char * *compressed*, int *nbCompressedBytes*)

Encodes a frame of audio.

Parameters:

st Encoder state

pcm PCM audio in float format, with a normal range of ± 1.0 . Samples with a range beyond ± 1.0 are supported but will be clipped by decoders using the integer API and should only be used if it is known that the far end supports extended dynamic range. There must be exactly *frame_size* samples per channel.

optional_synthesis If not NULL, the encoder copies the audio signal that the decoder would decode. It is the same as calling the decoder on the compressed data, just faster. This may alias *pcm*.

compressed The compressed data is written here. This may not alias *pcm* or *optional_synthesis*.

nbCompressedBytes Maximum number of bytes to use for compressing the frame (can change from one frame to another)

Returns:

Number of bytes written to "compressed". Will be the same as "nbCompressedBytes" unless the stream is VBR and will never be larger. If negative, an error has occurred (see error codes). It is IMPORTANT that the length returned be somehow transmitted to the decoder. Otherwise, no decoding is possible.

4.1.2.8 EXPORT CELTEncoder* celt_encoder_create (const CELTMode * *mode*, int *channels*, int * *error*)

Creates a new encoder state. Each stream needs its own encoder state (can't be shared across simultaneous streams).

Parameters:

mode Contains all the information about the characteristics of the stream (must be the same characteristics as used for the decoder)

channels Number of channels

error Returns an error code

Returns:

Newly created encoder state.

Definition at line 131 of file celt.c.

References CELTEncoder::block_size, CELT_ALLOC_FAIL, CELT_BAD_ARG, celt_encoder_destroy(), CELT_INVALID_MODE, CELT_OK, CELTEncoder::channels, check_mode(), CELTEncoder::delayedIntra, ENCODERPARTIAL, ENCODERINVALID, CELTEncoder::fold_decision, CELTEncoder::force_intra, CELTEncoder::frame_size, CELTMode::Fs, CELTEncoder::in_mem, CELTEncoder::marker, MAX_PERIOD, CELTMode::mdctSize, CELTEncoder::mode, CELTMode::nbEBands, CELTEncoder::oldBandE, CELTEncoder::out_mem, CELTMode::overlap, CELTEncoder::overlap, CELTEncoder::pitch_available, CELTEncoder::pitch_enabled, CELTEncoder::pitch_permitted, CELTEncoder::preemph_memD, CELTEncoder::preemph_memE, psydecay_init(), QCONST16, CELTEncoder::tonal_average, and CELTEncoder::vbr_rate.

Referenced by main().

4.1.2.9 EXPORT int celt_encoder_ctl (CELTEncoder * st, int request, ...)

Query and set encoder parameters

Parameters:

st Encoder state
request Parameter to change or query
value Pointer to a 32-bit int value

Returns:

Error code

4.1.2.10 EXPORT void celt_encoder_destroy (CELTEncoder * st)

Destroys a an encoder state.

Parameters:

st Encoder state to be destroyed

Definition at line 217 of file celt.c.

References check_mode(), ENCODERFREED, ENCODERPARTIAL, ENCODERVALID, CELTEncoder::in_mem, CELTEncoder::marker, CELTEncoder::mode, CELTEncoder::oldBandE, CELTEncoder::out_mem, CELTEncoder::preemph_memD, CELTEncoder::preemph_memE, and psydecay_clear().

Referenced by celt_encoder_create(), and main().

4.1.2.11 EXPORT CELTMode* celt_mode_create (celt_int32 Fs, int frame_size, int * error)

Creates a new mode struct. This will be passed to an encoder or decoder. The mode **MUST NOT BE DESTROYED** until the encoders and decoders that use it are destroyed as well.

Parameters:

Fs Sampling rate (32000 to 96000 Hz)
frame_size Number of samples (per channel) to encode in each packet (even values; 64 - 512)
error Returned error code (if NULL, no error will be returned)

Returns:

A newly created mode

Definition at line 228 of file modes.c.

References ALLOC_STACK, CELTMode::allocVectors, CELTMode::bits, CELT_BAD_ARG, CELT_COPY, CELT_INVALID_MODE, celt_mode_destroy(), CELT_OK, compute_alloc_cache(), PsyDecay::decayR, CELTMode::eBands, CELTMode::ePredCoef, CELTMode::fft, CELTMode::Fs, global_stack, mdct_lookup::kfft, M_PI, CELTMode::marker_end, CELTMode::marker_start, MAX_PERIOD, CELTMode::mdct, mdct_init(), CELTMode::mdctSize, MIN32, MODEPARTIAL, MODEVALID,

CELTMode::nbEBands, CELTMode::nbShortMdcts, CELTMode::overlap, pitch_state_alloc(), CELTMode::pitchEnd, CELTMode::prob, CELTMode::psy, psydecay_init(), Q15ONE, QCONST16, quant_prob_alloc(), CELTMode::shortMdct, CELTMode::shortMdctSize, CELTMode::shortWindow, mdct_lookup::trig, and CELTMode::window.

Referenced by main().

4.1.2.12 EXPORT void celt_mode_destroy (CELTMode * mode)

Destroys a mode struct. Only call this after all encoders and decoders using this mode are destroyed as well.

Parameters:

mode Mode to be destroyed

Definition at line 405 of file modes.c.

References CELTMode::allocVectors, CELTMode::bits, CELTMode::eBands, CELTMode::fft, CELTMode::marker_end, CELTMode::marker_start, CELTMode::mdct, mdct_clear(), MODEFREED, MODEPARTIAL, MODEVALID, CELTMode::nbEBands, pitch_state_free(), CELTMode::prob, CELTMode::psy, psydecay_clear(), quant_prob_free(), CELTMode::shortMdct, and CELTMode::window.

Referenced by celt_mode_create(), and main().

4.1.2.13 EXPORT int celt_mode_info (const CELTMode * mode, int request, celt_int32 * value)

Query information from a mode

Definition at line 58 of file modes.c.

References CELT_BITSTREAM_VERSION, CELT_GET_BITSTREAM_VERSION, CELT_GET_FRAME_SIZE, CELT_GET_LOOKAHEAD, CELT_GET_SAMPLE_RATE, CELT_INVALID_MODE, CELT_OK, CELT_UNIMPLEMENTED, check_mode(), CELTMode::Fs, CELTMode::mdctSize, and CELTMode::overlap.

Referenced by celt_header_init(), and main().

4.1.2.14 EXPORT const char* celt_strerror (int error)

Returns the English string that corresponds to an error code

Parameters:

error Error code (negative for an error, 0 for success)

Returns:

Constant string (must NOT be freed)

Definition at line 1546 of file celt.c.

Referenced by main().

Chapter 5

Data Structure Documentation

5.1 `c64_fft_t` Struct Reference

```
#include <c64_fft.h>
```

Data Fields

- `int nfft`
- `int shift`
- `celt_int32 * twiddle`
- `celt_int32 * itwiddle`

5.1.1 Detailed Description

Definition at line 41 of file `c64_fft.h`.

5.1.2 Field Documentation

5.1.2.1 `celt_int32* c64_fft_t::itwiddle`

Definition at line 45 of file `c64_fft.h`.

Referenced by `c64_fft16_alloc()`, and `c64_fft32_alloc()`.

5.1.2.2 `int c64_fft_t::nfft`

Definition at line 42 of file `c64_fft.h`.

Referenced by `c64_fft16_alloc()`, and `c64_fft32_alloc()`.

5.1.2.3 `int c64_fft_t::shift`

Definition at line 43 of file `c64_fft.h`.

Referenced by `c64_fft16_alloc()`, and `c64_fft32_alloc()`.

5.1.2.4 `celt_int32* c64_fft_t::twiddle`

Definition at line 44 of file `c64_fft.h`.

Referenced by `c64_fft16_alloc()`, and `c64_fft32_alloc()`.

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

- [libcelt/c64_fft.h](#)

5.2 CELTDecoder Struct Reference

Decoder state. Collaboration diagram for CELTDecoder:

Data Fields

- [celt_uint32](#) marker
- const [CELTMode](#) * mode
- int frame_size
- int block_size
- int overlap
- int channels
- [ec_byte_buffer](#) buf
- [ec_enc](#) enc
- [celt_sig](#) *restrict preemph_memD
- [celt_sig](#) * out_mem
- [celt_sig](#) * decode_mem
- [celt_word16](#) * oldBandE
- int last_pitch_index
- int loss_count

5.2.1 Detailed Description

Decoder state. Decoder state

Definition at line 1084 of file celt.c.

5.2.2 Field Documentation

5.2.2.1 int CELTDecoder::block_size

Definition at line 1088 of file celt.c.

Referenced by [celt_decoder_create\(\)](#).

5.2.2.2 ec_byte_buffer CELTDecoder::buf

Definition at line 1092 of file celt.c.

5.2.2.3 int CELTDecoder::channels

Definition at line 1090 of file celt.c.

Referenced by [celt_decoder_create\(\)](#).

5.2.2.4 celt_sig* CELTDecoder::decode_mem

Definition at line 1098 of file celt.c.

Referenced by [celt_decoder_create\(\)](#), and [celt_decoder_destroy\(\)](#).

5.2.2.5 ec_enc CELTDecoder::enc

Definition at line 1093 of file celt.c.

5.2.2.6 int CELTDecoder::frame_size

Definition at line 1087 of file celt.c.

Referenced by celt_decoder_create().

5.2.2.7 int CELTDecoder::last_pitch_index

Definition at line 1102 of file celt.c.

5.2.2.8 int CELTDecoder::loss_count

Definition at line 1103 of file celt.c.

Referenced by celt_decoder_create().

5.2.2.9 celt_uint32 CELTDecoder::marker

Definition at line 1085 of file celt.c.

Referenced by celt_decoder_create(), celt_decoder_destroy(), and check_decoder().

5.2.2.10 const CELTMode* CELTDecoder::mode

Definition at line 1086 of file celt.c.

Referenced by celt_decoder_create(), and celt_decoder_destroy().

5.2.2.11 celt_word16* CELTDecoder::oldBandE

Definition at line 1100 of file celt.c.

Referenced by celt_decoder_create(), and celt_decoder_destroy().

5.2.2.12 celt_sig* CELTDecoder::out_mem

Definition at line 1097 of file celt.c.

Referenced by celt_decoder_create().

5.2.2.13 int CELTDecoder::overlap

Definition at line 1089 of file celt.c.

Referenced by celt_decoder_create().

5.2.2.14 celt_sig* restrict CELTDecoder::preemph_memD

Definition at line 1095 of file celt.c.

Referenced by `celt_decoder_create()`, and `celt_decoder_destroy()`.

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

- [libcelt/celt.c](#)

5.3 CELTEncoder Struct Reference

Encoder state. Collaboration diagram for CELTEncoder:

Data Fields

- [celt_uint32](#) marker
- const [CELTMode](#) * mode
- int [frame_size](#)
- int [block_size](#)
- int [overlap](#)
- int [channels](#)
- int [pitch_enabled](#)
- int [pitch_permitted](#)
- int [pitch_available](#)
- int [force_intra](#)
- int [delayedIntra](#)
- [celt_word16](#) tonal_average
- int [fold_decision](#)
- [celt_word16](#) gain_prod
- [celt_int32](#) vbr_reservoir
- [celt_int32](#) vbr_drift
- [celt_int32](#) vbr_offset
- [celt_int32](#) vbr_count
- [celt_int32](#) vbr_rate
- [celt_word16](#) *restrict preemph_memE
- [celt_sig](#) *restrict preemph_memD
- [celt_sig](#) * in_mem
- [celt_sig](#) * out_mem
- [celt_word16](#) * oldBandE

5.3.1 Detailed Description

Encoder state. Encoder state

Definition at line 78 of file celt.c.

5.3.2 Field Documentation

5.3.2.1 int CELTEncoder::block_size

Definition at line 82 of file celt.c.

Referenced by [celt_encoder_create\(\)](#).

5.3.2.2 int CELTEncoder::channels

Definition at line 84 of file celt.c.

Referenced by [celt_encoder_create\(\)](#).

5.3.2.3 int CELTEncoder::delayedIntra

Definition at line 90 of file celt.c.

Referenced by celt_encoder_create().

5.3.2.4 int CELTEncoder::fold_decision

Definition at line 92 of file celt.c.

Referenced by celt_encoder_create().

5.3.2.5 int CELTEncoder::force_intra

Definition at line 89 of file celt.c.

Referenced by celt_encoder_create().

5.3.2.6 int CELTEncoder::frame_size

Definition at line 81 of file celt.c.

Referenced by celt_encoder_create().

5.3.2.7 celt_word16 CELTEncoder::gain_prod

Definition at line 93 of file celt.c.

5.3.2.8 celt_sig* CELTEncoder::in_mem

Definition at line 105 of file celt.c.

Referenced by celt_encoder_create(), and celt_encoder_destroy().

5.3.2.9 celt_uint32 CELTEncoder::marker

Definition at line 79 of file celt.c.

Referenced by celt_encoder_create(), and celt_encoder_destroy().

5.3.2.10 const CELTMode* CELTEncoder::mode

Mode used by the encoder

Definition at line 80 of file celt.c.

Referenced by celt_encoder_create(), and celt_encoder_destroy().

5.3.2.11 celt_word16* CELTEncoder::oldBandE

Definition at line 108 of file celt.c.

Referenced by celt_encoder_create(), and celt_encoder_destroy().

5.3.2.12 celt_sig* CELTEncoder::out_mem

Definition at line 106 of file celt.c.

Referenced by celt_encoder_create(), and celt_encoder_destroy().

5.3.2.13 int CELTEncoder::overlap

Definition at line 83 of file celt.c.

Referenced by celt_encoder_create().

5.3.2.14 int CELTEncoder::pitch_available

Definition at line 88 of file celt.c.

Referenced by celt_encoder_create().

5.3.2.15 int CELTEncoder::pitch_enabled

Definition at line 86 of file celt.c.

Referenced by celt_encoder_create().

5.3.2.16 int CELTEncoder::pitch_permitted

Definition at line 87 of file celt.c.

Referenced by celt_encoder_create().

5.3.2.17 celt_sig* restrict CELTEncoder::preemph_memD

Definition at line 103 of file celt.c.

Referenced by celt_encoder_create(), and celt_encoder_destroy().

5.3.2.18 celt_word16* restrict CELTEncoder::preemph_memE

Definition at line 102 of file celt.c.

Referenced by celt_encoder_create(), and celt_encoder_destroy().

5.3.2.19 celt_word16 CELTEncoder::tonal_average

Definition at line 91 of file celt.c.

Referenced by celt_encoder_create().

5.3.2.20 celt_int32 CELTEncoder::vbr_count

Definition at line 99 of file celt.c.

5.3.2.21 celt_int32 CELTEncoder::vbr_drift

Definition at line 97 of file celt.c.

5.3.2.22 celt_int32 CELTEncoder::vbr_offset

Definition at line 98 of file celt.c.

5.3.2.23 celt_int32 CELTEncoder::vbr_rate

Definition at line 101 of file celt.c.

Referenced by `celt_encoder_create()`.

5.3.2.24 celt_int32 CELTEncoder::vbr_reservoir

Definition at line 96 of file celt.c.

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

- [libcelt/celt.c](#)

5.4 CELTHeader Struct Reference

Header data.

```
#include <celt_header.h>
```

Data Fields

- char `codec_id` [8]
- char `codec_version` [20]
- `celt_int32` `version_id`
- `celt_int32` `header_size`
- `celt_int32` `sample_rate`
- `celt_int32` `nb_channels`
- `celt_int32` `frame_size`
- `celt_int32` `overlap`
- `celt_int32` `bytes_per_packet`
- `celt_int32` `extra_headers`

5.4.1 Detailed Description

Header data. Header data to be used for Ogg files (or possibly other encapsulation)

Definition at line 46 of file `celt_header.h`.

5.4.2 Field Documentation

5.4.2.1 `celt_int32 CELTHeader::bytes_per_packet`

Number of bytes per compressed packet (0 if unknown)

Definition at line 55 of file `celt_header.h`.

Referenced by `celt_header_init()`.

5.4.2.2 `char CELTHeader::codec_id[8]`

MUST be "CELT " (four spaces)

Definition at line 47 of file `celt_header.h`.

Referenced by `celt_header_init()`.

5.4.2.3 `char CELTHeader::codec_version[20]`

Version used (as string)

Definition at line 48 of file `celt_header.h`.

Referenced by `celt_header_init()`.

5.4.2.4 celt_int32 CELTHeader::extra_headers

Number of additional headers that follow this header

Definition at line 56 of file celt_header.h.

Referenced by celt_header_init().

5.4.2.5 celt_int32 CELTHeader::frame_size

Samples per frame (per channel)

Definition at line 53 of file celt_header.h.

Referenced by celt_header_init().

5.4.2.6 celt_int32 CELTHeader::header_size

Size of this header

Definition at line 50 of file celt_header.h.

Referenced by celt_header_init().

5.4.2.7 celt_int32 CELTHeader::nb_channels

Number of channels

Definition at line 52 of file celt_header.h.

Referenced by celt_header_init().

5.4.2.8 celt_int32 CELTHeader::overlap

Overlapping samples (per channel)

Definition at line 54 of file celt_header.h.

Referenced by celt_header_init().

5.4.2.9 celt_int32 CELTHeader::sample_rate

Sampling rate of the original audio

Definition at line 51 of file celt_header.h.

Referenced by celt_header_init().

5.4.2.10 celt_int32 CELTHeader::version_id

Version id (negative for until stream is frozen)

Definition at line 49 of file celt_header.h.

Referenced by celt_header_init().

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

- [libcelt/celt_header.h](#)

5.5 CELTMode Struct Reference

Mode definition.

`#include <modes.h>` Collaboration diagram for CELTMode:

Data Fields

- [celt_uint32 marker_start](#)
- [celt_int32 Fs](#)
- [int overlap](#)
- [int mdctSize](#)
- [int nbEBands](#)
- [int pitchEnd](#)
- [const celt_int16 * eBands](#)
- [celt_word16 ePredCoef](#)
- [int nbAllocVectors](#)
- [const celt_int16 * allocVectors](#)
- [const celt_int16 *const * bits](#)
- [mdct_lookup mdct](#)
- [kiss_fftr_cfg fft](#)
- [const celt_word16 * window](#)
- [int nbShortMdcts](#)
- [int shortMdctSize](#)
- [mdct_lookup shortMdct](#)
- [const celt_word16 * shortWindow](#)
- [struct PsyDecay psy](#)
- [int * prob](#)
- [celt_uint32 marker_end](#)

5.5.1 Detailed Description

Mode definition. Mode definition (opaque)

Definition at line 81 of file modes.h.

5.5.2 Field Documentation

5.5.2.1 `const celt_int16* CELTMode::allocVectors`

Number of bits in each band for several rates

Definition at line 95 of file modes.h.

Referenced by `celt_mode_create()`, `celt_mode_destroy()`, `compute_allocation()`, and `dump_modes()`.

5.5.2.2 `const celt_int16* const* CELTMode::bits`

Cache for pulses->bits mapping in each band

Definition at line 97 of file modes.h.

Referenced by `celt_mode_create()`, `celt_mode_destroy()`, `dump_modes()`, `quant_bands()`, `quant_bands_stereo()`, and `unquant_bands_stereo()`.

5.5.2.3 `const celt_int16* CELTMode::eBands`

Definition for each "pseudo-critical band"

Definition at line 90 of file modes.h.

Referenced by `celt_mode_create()`, `celt_mode_destroy()`, `compute_alloc_cache()`, `compute_band_energies()`, `denormalise_bands()`, `dump_modes()`, `folding_decision()`, `normalise_bands()`, `quant_bands()`, `quant_bands_stereo()`, `renormalise_bands()`, and `unquant_bands_stereo()`.

5.5.2.4 `celt_word16 CELTMode::ePredCoef`

Prediction coefficient for the energy encoding

Definition at line 92 of file modes.h.

Referenced by `celt_mode_create()`, `dump_modes()`, `quant_coarse_energy()`, and `unquant_coarse_energy()`.

5.5.2.5 `kiss_fftr_cfg CELTMode::fft`

Definition at line 101 of file modes.h.

Referenced by `celt_mode_create()`, and `celt_mode_destroy()`.

5.5.2.6 `celt_int32 CELTMode::Fs`

Definition at line 83 of file modes.h.

Referenced by `celt_encoder_create()`, `celt_header_init()`, `celt_mode_create()`, `celt_mode_info()`, and `dump_modes()`.

5.5.2.7 `celt_uint32 CELTMode::marker_end`

Definition at line 113 of file modes.h.

Referenced by `celt_mode_create()`, `celt_mode_destroy()`, and `check_mode()`.

5.5.2.8 `celt_uint32 CELTMode::marker_start`

Definition at line 82 of file modes.h.

Referenced by `celt_mode_create()`, `celt_mode_destroy()`, and `check_mode()`.

5.5.2.9 mdct_lookup CELTMode::mdct

Definition at line 100 of file modes.h.

Referenced by celt_mode_create(), and celt_mode_destroy().

5.5.2.10 int CELTMode::mdctSize

Definition at line 85 of file modes.h.

Referenced by celt_decoder_create(), celt_encoder_create(), celt_header_init(), celt_mode_create(), celt_mode_info(), dump_header(), and dump_modes().

5.5.2.11 int CELTMode::nbAllocVectors

Number of lines in the matrix below

Definition at line 94 of file modes.h.

Referenced by compute_allocation(), and dump_modes().

5.5.2.12 int CELTMode::nbEBands

Definition at line 87 of file modes.h.

Referenced by celt_decoder_create(), celt_decoder_ctl(), celt_encoder_create(), celt_encoder_ctl(), celt_mode_create(), celt_mode_destroy(), compute_alloc_cache(), compute_allocation(), compute_band_energies(), denormalise_bands(), dump_modes(), folding_decision(), normalise_bands(), quant_bands(), quant_bands_stereo(), quant_coarse_energy(), quant_energy_finalise(), quant_fine_energy(), quant_prob_alloc(), renormalise_bands(), unquant_bands_stereo(), unquant_coarse_energy(), unquant_energy_finalise(), and unquant_fine_energy().

5.5.2.13 int CELTMode::nbShortMdcts

Definition at line 105 of file modes.h.

Referenced by celt_encode_float(), celt_mode_create(), dump_modes(), quant_bands(), quant_bands_stereo(), and unquant_bands_stereo().

5.5.2.14 int CELTMode::overlap

Definition at line 84 of file modes.h.

Referenced by celt_decoder_create(), celt_encoder_create(), celt_header_init(), celt_mode_create(), celt_mode_info(), dump_header(), and dump_modes().

5.5.2.15 int CELTMode::pitchEnd

Definition at line 88 of file modes.h.

Referenced by apply_pitch(), celt_mode_create(), compute_alloc_cache(), compute_pitch_gain(), and dump_modes().

5.5.2.16 int* CELTMode::prob

Definition at line 112 of file modes.h.

Referenced by `celt_mode_create()`, and `celt_mode_destroy()`.

5.5.2.17 struct PsyDecay CELTMode::psy [read]

Definition at line 110 of file modes.h.

Referenced by `celt_mode_create()`, `celt_mode_destroy()`, and `dump_modes()`.

5.5.2.18 mdct_lookup CELTMode::shortMdct

Definition at line 107 of file modes.h.

Referenced by `celt_mode_create()`, and `celt_mode_destroy()`.

5.5.2.19 int CELTMode::shortMdctSize

Definition at line 106 of file modes.h.

Referenced by `celt_mode_create()`, and `dump_modes()`.

5.5.2.20 const celt_word16* CELTMode::shortWindow

Definition at line 108 of file modes.h.

Referenced by `celt_mode_create()`.

5.5.2.21 const celt_word16* CELTMode::window

Definition at line 103 of file modes.h.

Referenced by `celt_mode_create()`, `celt_mode_destroy()`, and `dump_modes()`.

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

- [libcelt/modes.h](#)

5.6 `ec_byte_buffer` Struct Reference

```
#include <entcode.h>
```

Data Fields

- unsigned char * [buf](#)
- unsigned char * [ptr](#)
- unsigned char * [end_ptr](#)
- long [storage](#)

5.6.1 Detailed Description

Definition at line 55 of file `entcode.h`.

5.6.2 Field Documentation

5.6.2.1 unsigned char* `ec_byte_buffer::buf`

Definition at line 56 of file `entcode.h`.

Referenced by `ec_byte_look_at_end()`, `ec_byte_read1()`, `ec_byte_readinit()`, `ec_byte_shrink()`, `ec_byte_write1()`, and `ec_byte_writeinit_buffer()`.

5.6.2.2 unsigned char* `ec_byte_buffer::end_ptr`

Definition at line 58 of file `entcode.h`.

Referenced by `ec_byte_look_at_end()`, `ec_byte_readinit()`, `ec_byte_shrink()`, `ec_byte_write_at_end()`, `ec_byte_writeinit_buffer()`, and `ec_enc_done()`.

5.6.2.3 unsigned char* `ec_byte_buffer::ptr`

Definition at line 57 of file `entcode.h`.

Referenced by `ec_byte_adv1()`, `ec_byte_read1()`, `ec_byte_readinit()`, `ec_byte_write1()`, `ec_byte_write_at_end()`, `ec_byte_writeinit_buffer()`, and `ec_enc_done()`.

5.6.2.4 long `ec_byte_buffer::storage`

Definition at line 59 of file `entcode.h`.

Referenced by `ec_byte_read1()`, `ec_byte_readinit()`, `ec_byte_shrink()`, `ec_byte_write1()`, and `ec_byte_writeinit_buffer()`.

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

- [libcelt/entcode.h](#)

5.7 ec_dec Struct Reference

#include <entdec.h> Collaboration diagram for ec_dec:

Data Fields

- [ec_byte_buffer](#) * buf
- int rem
- [ec_uint32](#) rng
- [ec_uint32](#) dif
- [ec_uint32](#) nrm
- unsigned char [end_byte](#)
- int [end_bits_left](#)
- int [nb_end_bits](#)

5.7.1 Detailed Description

Definition at line 43 of file entdec.h.

5.7.2 Field Documentation

5.7.2.1 [ec_byte_buffer](#)* [ec_dec::buf](#)

Definition at line 45 of file entdec.h.

Referenced by [ec_dec_init\(\)](#), [ec_dec_tell\(\)](#), [ec_decode_bin\(\)](#), and [ec_decode_raw\(\)](#).

5.7.2.2 [ec_uint32](#) [ec_dec::dif](#)

Definition at line 52 of file entdec.h.

Referenced by [ec_dec_init\(\)](#), [ec_dec_update\(\)](#), [ec_decode\(\)](#), and [ec_decode_bin\(\)](#).

5.7.2.3 int [ec_dec::end_bits_left](#)

Definition at line 58 of file entdec.h.

Referenced by [ec_dec_init\(\)](#), [ec_decode_bin\(\)](#), and [ec_decode_raw\(\)](#).

5.7.2.4 unsigned char [ec_dec::end_byte](#)

Definition at line 56 of file entdec.h.

Referenced by [ec_decode_bin\(\)](#), and [ec_decode_raw\(\)](#).

5.7.2.5 int [ec_dec::nb_end_bits](#)

Definition at line 59 of file entdec.h.

Referenced by [ec_dec_init\(\)](#), [ec_dec_tell\(\)](#), [ec_decode_bin\(\)](#), and [ec_decode_raw\(\)](#).

5.7.2.6 ec_uint32 ec_dec::nrm

Definition at line 54 of file entdec.h.

Referenced by ec_dec_update(), ec_decode(), and ec_decode_bin().

5.7.2.7 int ec_dec::rem

Definition at line 47 of file entdec.h.

Referenced by ec_dec_init().

5.7.2.8 ec_uint32 ec_dec::rng

Definition at line 49 of file entdec.h.

Referenced by ec_dec_init(), ec_dec_tell(), ec_dec_update(), ec_decode(), and ec_decode_bin().

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

- [libcelt/entdec.h](#)

5.8 ec_enc Struct Reference

#include <entenc.h> Collaboration diagram for ec_enc:

Data Fields

- [ec_byte_buffer * buf](#)
- [int rem](#)
- [size_t ext](#)
- [ec_uint32 rng](#)
- [ec_uint32 low](#)
- [unsigned char end_byte](#)
- [int end_bits_left](#)
- [int nb_end_bits](#)

5.8.1 Detailed Description

Definition at line 44 of file entenc.h.

5.8.2 Field Documentation

5.8.2.1 ec_byte_buffer* ec_enc::buf

Definition at line 46 of file entenc.h.

Referenced by [ec_enc_done\(\)](#), [ec_enc_init\(\)](#), [ec_enc_tell\(\)](#), [ec_encode_bin\(\)](#), and [ec_encode_raw\(\)](#).

5.8.2.2 int ec_enc::end_bits_left

Definition at line 58 of file entenc.h.

Referenced by [ec_enc_done\(\)](#), [ec_enc_init\(\)](#), [ec_encode_bin\(\)](#), and [ec_encode_raw\(\)](#).

5.8.2.3 unsigned char ec_enc::end_byte

Definition at line 56 of file entenc.h.

Referenced by [ec_enc_done\(\)](#), [ec_enc_init\(\)](#), [ec_encode_bin\(\)](#), and [ec_encode_raw\(\)](#).

5.8.2.4 size_t ec_enc::ext

Definition at line 50 of file entenc.h.

Referenced by [ec_enc_done\(\)](#), [ec_enc_init\(\)](#), and [ec_enc_tell\(\)](#).

5.8.2.5 ec_uint32 ec_enc::low

Definition at line 54 of file entenc.h.

Referenced by [ec_enc_done\(\)](#), [ec_enc_init\(\)](#), [ec_encode\(\)](#), and [ec_encode_bin\(\)](#).

5.8.2.6 int ec_enc::nb_end_bits

Definition at line 59 of file entenc.h.

Referenced by `ec_enc_init()`, `ec_enc_tell()`, `ec_encode_bin()`, and `ec_encode_raw()`.

5.8.2.7 int ec_enc::rem

Definition at line 48 of file entenc.h.

Referenced by `ec_enc_done()`, `ec_enc_init()`, and `ec_enc_tell()`.

5.8.2.8 ec_uint32 ec_enc::rng

Definition at line 52 of file entenc.h.

Referenced by `ec_enc_done()`, `ec_enc_init()`, `ec_enc_tell()`, `ec_encode()`, and `ec_encode_bin()`.

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

- [libcelt/entenc.h](#)

5.9 kiss_fft_cpx Struct Reference

```
#include <kiss_fft.h>
```

Data Fields

- [kiss_fft_scalar r](#)
- [kiss_fft_scalar i](#)

5.9.1 Detailed Description

Definition at line 87 of file `kiss_fft.h`.

5.9.2 Field Documentation

5.9.2.1 `kiss_fft_scalar kiss_fft_cpx::i`

Definition at line 89 of file `kiss_fft.h`.

Referenced by `kiss_fft_stride()`, `kiss_fftr_twiddles()`, and `kiss_fftri()`.

5.9.2.2 `kiss_fft_scalar kiss_fft_cpx::r`

Definition at line 88 of file `kiss_fft.h`.

Referenced by `kiss_fft_stride()`, `kiss_fftr_twiddles()`, and `kiss_fftri()`.

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

- [libcelt/kiss_fft.h](#)

5.10 kiss_fft_state Struct Reference

`#include <_kiss_fft_guts.h>` Collaboration diagram for `kiss_fft_state`:

Data Fields

- `int` `nfft`
- `kiss_fft_scalar` `scale`
- `int` `factors` [2 *MAXFACTORS]
- `int *` `bitrev`
- `kiss_twiddle_cpx` `twiddles` [1]

5.10.1 Detailed Description

Definition at line 33 of file `_kiss_fft_guts.h`.

5.10.2 Field Documentation

5.10.2.1 `int* kiss_fft_state::bitrev`

Definition at line 39 of file `_kiss_fft_guts.h`.

Referenced by `kiss_fft_alloc()`, `kiss_fft_stride()`, `kiss_fftri()`, and `kiss_ifft_stride()`.

5.10.2.2 `int kiss_fft_state::factors[2 *MAXFACTORS]`

Definition at line 38 of file `_kiss_fft_guts.h`.

Referenced by `kiss_fft_alloc()`, `kiss_fft_stride()`, `kiss_fftr_inplace()`, `kiss_fftri()`, and `kiss_ifft_stride()`.

5.10.2.3 `int kiss_fft_state::nfft`

Definition at line 34 of file `_kiss_fft_guts.h`.

Referenced by `kiss_fft_alloc()`, `kiss_fft_stride()`, `kiss_fftr_twiddles()`, `kiss_fftri()`, and `kiss_ifft_stride()`.

5.10.2.4 `kiss_fft_scalar kiss_fft_state::scale`

Definition at line 36 of file `_kiss_fft_guts.h`.

Referenced by `kiss_fft_alloc()`, `kiss_fft_stride()`, and `kiss_fftr_alloc()`.

5.10.2.5 `kiss_twiddle_cpx kiss_fft_state::twiddles[1]`

Definition at line 40 of file `_kiss_fft_guts.h`.

Referenced by `kiss_fft_alloc()`.

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

- `libcelt/_kiss_fft_guts.h`

5.11 kiss_fftr_state Struct Reference

`#include <kiss_fftr.h>` Collaboration diagram for `kiss_fftr_state`:

Data Fields

- [kiss_fft_cfg](#) `substate`
- [kiss_twiddle_cpx](#) * `super_twiddles`

5.11.1 Detailed Description

Definition at line 42 of file `kiss_fftr.h`.

5.11.2 Field Documentation

5.11.2.1 `kiss_fft_cfg` `kiss_fftr_state::substate`

Definition at line 43 of file `kiss_fftr.h`.

Referenced by `kiss_fftr()`, `kiss_fftr_alloc()`, `kiss_fftr_inplace()`, `kiss_fftr_twiddles()`, and `kiss_fftri()`.

5.11.2.2 `kiss_twiddle_cpx`* `kiss_fftr_state::super_twiddles`

Definition at line 44 of file `kiss_fftr.h`.

Referenced by `kiss_fftr_alloc()`, `kiss_fftr_twiddles()`, and `kiss_fftri()`.

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

- [libcelt/kiss_fftr.h](#)

5.12 kiss_twiddle_cpx Struct Reference

```
#include <kiss_fft.h>
```

Data Fields

- [kiss_twiddle_scalar r](#)
- [kiss_twiddle_scalar i](#)

5.12.1 Detailed Description

Definition at line 92 of file [kiss_fft.h](#).

5.12.2 Field Documentation

5.12.2.1 [kiss_twiddle_scalar](#) [kiss_twiddle_cpx::i](#)

Definition at line 94 of file [kiss_fft.h](#).

5.12.2.2 [kiss_twiddle_scalar](#) [kiss_twiddle_cpx::r](#)

Definition at line 93 of file [kiss_fft.h](#).

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

- [libcelt/kiss_fft.h](#)

5.13 mdct_lookup Struct Reference

`#include <mdct.h>` Collaboration diagram for mdct_lookup:

Data Fields

- `int n`
- `kiss_fft_cfg kfft`
- `kiss_twiddle_scalar *restrict trig`

5.13.1 Detailed Description

Definition at line 52 of file mdct.h.

5.13.2 Field Documentation

5.13.2.1 `kiss_fft_cfg mdct_lookup::kfft`

Definition at line 54 of file mdct.h.

Referenced by `celt_mode_create()`, `mdct_backward()`, `mdct_clear()`, `mdct_forward()`, and `mdct_init()`.

5.13.2.2 `int mdct_lookup::n`

Definition at line 53 of file mdct.h.

Referenced by `mdct_backward()`, `mdct_forward()`, and `mdct_init()`.

5.13.2.3 `kiss_twiddle_scalar* restrict mdct_lookup::trig`

Definition at line 55 of file mdct.h.

Referenced by `celt_mode_create()`, `mdct_backward()`, `mdct_clear()`, `mdct_forward()`, and `mdct_init()`.

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

- `libcelt/mdct.h`

5.14 PsyDecay Struct Reference

```
#include <psy.h>
```

Data Fields

- const [celt_word16](#) *restrict [decayR](#)

5.14.1 Detailed Description

Definition at line 38 of file psy.h.

5.14.2 Field Documentation

5.14.2.1 const celt_word16* restrict PsyDecay::decayR

Definition at line 40 of file psy.h.

Referenced by [celt_mode_create\(\)](#), [dump_modes\(\)](#), [psydecay_clear\(\)](#), and [psydecay_init\(\)](#).

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

- [libcelt/psy.h](#)

Chapter 6

File Documentation

6.1 libcelt/_kiss_fft_guts.h File Reference

```
#include "kiss_fft.h"
```

Data Structures

- struct [kiss_fft_state](#)

Defines

- #define [MIN](#)(a, b) ((a)<(b) ? (a):(b))
- #define [MAX](#)(a, b) ((a)>(b) ? (a):(b))
- #define [MAXFACTORS](#) 32
- #define [EXT32](#)(a) (a)
- #define [S_MUL](#)(a, b) ((a)*(b))
- #define [C_MUL](#)(m, a, b)
- #define [C_MULC](#)(m, a, b)
- #define [C_MUL4](#)(m, a, b) C_MUL(m,a,b)
- #define [C_FIXDIV](#)(c, div)
- #define [C_MULBYSCALAR](#)(c, s)
- #define [CHECK_OVERFLOW_OP](#)(a, op, b)
- #define [C_ADD](#)(res, a, b)
- #define [C_SUB](#)(res, a, b)
- #define [C_ADDTO](#)(res, a)
- #define [C_SUBFROM](#)(res, a)
- #define [KISS_FFT_COS](#)(phase) (kiss_fft_scalar) cos(phase)
- #define [KISS_FFT_SIN](#)(phase) (kiss_fft_scalar) sin(phase)
- #define [HALF_OF](#)(x) ((x)*.5)
- #define [kf_cexp](#)(x, phase)
- #define [kf_cexp2](#)(x, phase)

6.1.1 Define Documentation

6.1.1.1 #define C_ADD(res, a, b)

Value:

```
do { \
    CHECK_OVERFLOW_OP((a).r,+,(b).r)\
    CHECK_OVERFLOW_OP((a).i,+,(b).i)\
    (res).r=(a).r+(b).r; (res).i=(a).i+(b).i; \
}while(0)
```

Definition at line 189 of file `_kiss_fft_guts.h`.

Referenced by `kiss_fftri()`.

6.1.1.2 #define C_ADDTO(res, a)

Value:

```
do { \
    CHECK_OVERFLOW_OP((res).r,+,(a).r)\
    CHECK_OVERFLOW_OP((res).i,+,(a).i)\
    (res).r += (a).r; (res).i += (a).i;\
}while(0)
```

Definition at line 201 of file `_kiss_fft_guts.h`.

6.1.1.3 #define C_FIXDIV(c, div)

Definition at line 176 of file `_kiss_fft_guts.h`.

Referenced by `kiss_fftr_twiddles()`.

6.1.1.4 #define C_MUL(m, a, b)

Value:

```
do{ (m).r = (a).r*(b).r - (a).i*(b).i;\
    (m).i = (a).r*(b).i + (a).i*(b).r; }while(0)
```

Definition at line 167 of file `_kiss_fft_guts.h`.

Referenced by `kiss_fftri()`.

6.1.1.5 #define C_MUL4(m, a, b) C_MUL(m,a,b)

Definition at line 174 of file `_kiss_fft_guts.h`.

6.1.1.6 #define C_MULBYSCALAR(c, s)

Value:

```
do{ (c).r *= (s);\
    (c).i *= (s); }while(0)
```

Definition at line 177 of file `_kiss_fft_guts.h`.

6.1.1.7 #define C_MULC(m, a, b)**Value:**

```
do{ (m).r = (a).r*(b).r + (a).i*(b).i;\
    (m).i = (a).i*(b).r - (a).r*(b).i; }while(0)
```

Definition at line 170 of file _kiss_fft_guts.h.

Referenced by kiss_fftr_twiddles().

6.1.1.8 #define C_SUB(res, a, b)**Value:**

```
do { \
    CHECK_OVERFLOW_OP((a).r,-,(b).r)\
    CHECK_OVERFLOW_OP((a).i,-,(b).i)\
    (res).r=(a).r-(b).r; (res).i=(a).i-(b).i; \
}while(0)
```

Definition at line 195 of file _kiss_fft_guts.h.

Referenced by kiss_fftri().

6.1.1.9 #define C_SUBFROM(res, a)**Value:**

```
do {\
    CHECK_OVERFLOW_OP((res).r,-,(a).r)\
    CHECK_OVERFLOW_OP((res).i,-,(a).i)\
    (res).r -= (a).r; (res).i -= (a).i; \
}while(0)
```

Definition at line 208 of file _kiss_fft_guts.h.

6.1.1.10 #define CHECK_OVERFLOW_OP(a, op, b)

Definition at line 185 of file _kiss_fft_guts.h.

Referenced by kiss_fftr_twiddles().

6.1.1.11 #define EXT32(a) (a)

Definition at line 164 of file _kiss_fft_guts.h.

Referenced by kiss_fftr_twiddles().

6.1.1.12 #define HALF_OF(x) ((x)*.5)

Definition at line 229 of file _kiss_fft_guts.h.

Referenced by kiss_fftr_twiddles().

6.1.1.13 #define kf_cexp(x, phase)**Value:**

```
do{ \
    (x)->r = KISS_FFT_COS(phase);\
    (x)->i = KISS_FFT_SIN(phase);\
}while(0)
```

Definition at line 232 of file `_kiss_fft_guts.h`.

Referenced by `kiss_fft_alloc()`, and `kiss_fftr_alloc()`.

6.1.1.14 #define kf_cexp2(x, phase)**Value:**

```
do{ \
    (x)->r = TRIG_UPSCALE*celt_cos_norm((phase));\
    (x)->i = TRIG_UPSCALE*celt_cos_norm((phase)-32768);\
}while(0)
```

Definition at line 238 of file `_kiss_fft_guts.h`.

Referenced by `kiss_fft_alloc()`, and `kiss_fftr_alloc()`.

6.1.1.15 #define KISS_FFT_COS(phase) (kiss_fft_scalar) cos(phase)

Definition at line 227 of file `_kiss_fft_guts.h`.

6.1.1.16 #define KISS_FFT_SIN(phase) (kiss_fft_scalar) sin(phase)

Definition at line 228 of file `_kiss_fft_guts.h`.

6.1.1.17 #define MAX(a, b) ((a)>(b) ? (a):(b))

Definition at line 19 of file `_kiss_fft_guts.h`.

6.1.1.18 #define MAXFACTORS 32

Definition at line 27 of file `_kiss_fft_guts.h`.

6.1.1.19 #define MIN(a, b) ((a)<(b) ? (a):(b))

Definition at line 18 of file `_kiss_fft_guts.h`.

6.1.1.20 #define S_MUL(a, b) ((a)*(b))

Definition at line 166 of file `_kiss_fft_guts.h`.

Referenced by `mdct_backward()`, and `mdct_forward()`.

6.2 libcelt/arch.h File Reference

Various architecture definitions for CELT. `#include "celt_types.h"`

Defines

- `#define CELT_SIG_SCALE` 32768.
- `#define celt_fatal`(str) `_celt_fatal`(str, `__FILE__`, `__LINE__`);
- `#define celt_assert`(cond)
- `#define celt_assert2`(cond, message)
- `#define IMUL32`(a, b) ((a)*(b))
- `#define UMUL32`(a, b) ((`celt_int32`)(a)*(`celt_int32`)(b))
- `#define UMUL16_16`(a, b) ((`celt_int32`)(a)*(`celt_int32`)(b))
- `#define ABS`(x) ((x) < 0 ? (-x) : (x))
- `#define ABS16`(x) ((x) < 0 ? (-x) : (x))
- `#define MIN16`(a, b) ((a) < (b) ? (a) : (b))
- `#define MAX16`(a, b) ((a) > (b) ? (a) : (b))
- `#define ABS32`(x) ((x) < 0 ? (-x) : (x))
- `#define MIN32`(a, b) ((a) < (b) ? (a) : (b))
- `#define MAX32`(a, b) ((a) > (b) ? (a) : (b))
- `#define IMIN`(a, b) ((a) < (b) ? (a) : (b))
- `#define IMAX`(a, b) ((a) > (b) ? (a) : (b))
- `#define UADD32`(a, b) ((a)+(b))
- `#define USUB32`(a, b) ((a)-(b))
- `#define PRINT_MIPS`(file)
- `#define Q15ONE` 1.0f
- `#define Q30ONE` 1.0f
- `#define NORM_SCALING` 1.f
- `#define NORM_SCALING_1` 1.f
- `#define ENER_SCALING` 1.f
- `#define ENER_SCALING_1` 1.f
- `#define PGAIN_SCALING` 1.f
- `#define PGAIN_SCALING_1` 1.f
- `#define EPSILON` 1e-15f
- `#define VERY_SMALL` 1e-15f
- `#define VERY_LARGE32` 1e15f
- `#define VERY_LARGE16` 1e15f
- `#define Q15_ONE` ((`celt_word16`)1.f)
- `#define Q15_ONE_1` ((`celt_word16`)1.f)
- `#define QCONST16`(x, bits) (x)
- `#define QCONST32`(x, bits) (x)
- `#define NEG16`(x) (-x)
- `#define NEG32`(x) (-x)
- `#define EXTRACT16`(x) (x)
- `#define EXTEND32`(x) (x)
- `#define SHR16`(a, shift) (a)
- `#define SHL16`(a, shift) (a)
- `#define SHR32`(a, shift) (a)
- `#define SHL32`(a, shift) (a)

- #define `PSHR16(a, shift)` (a)
- #define `PSHR32(a, shift)` (a)
- #define `VSHR32(a, shift)` (a)
- #define `SATURATE16(x, a)` (x)
- #define `SATURATE32(x, a)` (x)
- #define `PSHR(a, shift)` (a)
- #define `SHR(a, shift)` (a)
- #define `SHL(a, shift)` (a)
- #define `SATURATE(x, a)` (x)
- #define `ROUND16(a, shift)` (a)
- #define `HALF32(x)` (.5f*(x))
- #define `ADD16(a, b)` ((a)+(b))
- #define `SUB16(a, b)` ((a)-(b))
- #define `ADD32(a, b)` ((a)+(b))
- #define `SUB32(a, b)` ((a)-(b))
- #define `MULT16_16_16(a, b)` ((a)*(b))
- #define `MULT16_16(a, b)` ((`celt_word32`)(a)*(`celt_word32`)(b))
- #define `MAC16_16(c, a, b)` ((c)+(`celt_word32`)(a)*(`celt_word32`)(b))
- #define `MULT16_32_Q11(a, b)` ((a)*(b))
- #define `MULT16_32_Q13(a, b)` ((a)*(b))
- #define `MULT16_32_Q14(a, b)` ((a)*(b))
- #define `MULT16_32_Q15(a, b)` ((a)*(b))
- #define `MULT16_32_Q16(a, b)` ((a)*(b))
- #define `MULT16_32_P15(a, b)` ((a)*(b))
- #define `MULT32_32_Q31(a, b)` ((a)*(b))
- #define `MAC16_32_Q11(c, a, b)` ((c)+(a)*(b))
- #define `MAC16_32_Q15(c, a, b)` ((c)+(a)*(b))
- #define `MAC16_16_Q11(c, a, b)` ((c)+(a)*(b))
- #define `MAC16_16_Q13(c, a, b)` ((c)+(a)*(b))
- #define `MAC16_16_P13(c, a, b)` ((c)+(a)*(b))
- #define `MULT16_16_Q11_32(a, b)` ((a)*(b))
- #define `MULT16_16_Q13(a, b)` ((a)*(b))
- #define `MULT16_16_Q14(a, b)` ((a)*(b))
- #define `MULT16_16_Q15(a, b)` ((a)*(b))
- #define `MULT16_16_P15(a, b)` ((a)*(b))
- #define `MULT16_16_P13(a, b)` ((a)*(b))
- #define `MULT16_16_P14(a, b)` ((a)*(b))
- #define `DIV32_16(a, b)` (((`celt_word32`)(a))/(`celt_word16`)(b))
- #define `PDIV32_16(a, b)` (((`celt_word32`)(a))/(`celt_word16`)(b))
- #define `DIV32(a, b)` (((`celt_word32`)(a))/(`celt_word32`)(b))
- #define `PDIV32(a, b)` (((`celt_word32`)(a))/(`celt_word32`)(b))
- #define `SCALEIN(a)` ((a)*`CELT_SIG_SCALE`)
- #define `SCALEOUT(a)` ((a)*(1/`CELT_SIG_SCALE`))
- #define `BYTES_PER_CHAR` 1
- #define `BITS_PER_CHAR` 8
- #define `LOG2_BITS_PER_CHAR` 3
- #define `GLOBAL_STACK_SIZE` 100000

Typedefs

- typedef float [celt_word16](#)
- typedef float [celt_word32](#)
- typedef float [celt_sig](#)
- typedef float [celt_norm](#)
- typedef float [celt_ener](#)
- typedef float [celt_pgain](#)
- typedef float [celt_mask](#)

6.2.1 Detailed Description

Various architecture definitions for CELT.

Definition in file [arch.h](#).

6.2.2 Define Documentation

6.2.2.1 #define ABS(x) ((x) < 0 ? (-x) : (x))

Absolute integer value.

Definition at line 58 of file arch.h.

6.2.2.2 #define ABS16(x) ((x) < 0 ? (-x) : (x))

Absolute 16-bit value.

Definition at line 59 of file arch.h.

Referenced by `folding_decision()`.

6.2.2.3 #define ABS32(x) ((x) < 0 ? (-x) : (x))

Absolute 32-bit value.

Definition at line 62 of file arch.h.

Referenced by `celt_encode_float()`, and `compute_pitch_gain()`.

6.2.2.4 #define ADD16(a, b) ((a)+(b))

Definition at line 185 of file arch.h.

Referenced by `apply_pitch()`, `compute_pitch_gain()`, and `find_spectral_pitch()`.

6.2.2.5 #define ADD32(a, b) ((a)+(b))

Definition at line 187 of file arch.h.

Referenced by `compute_masking()`, `compute_pitch_gain()`, `folding_decision()`, `kiss_fftr_twiddles()`, and `mdct_init()`.

6.2.2.6 #define BITS_PER_CHAR 8

Definition at line 237 of file arch.h.

6.2.2.7 #define BYTES_PER_CHAR 1

Definition at line 236 of file arch.h.

6.2.2.8 #define celt_assert(cond)

Definition at line 50 of file arch.h.

Referenced by c64_iff32(), ec_dec_uint(), ec_enc_uint(), get_required_bits(), and icwrs().

6.2.2.9 #define celt_assert2(cond, message)

Definition at line 51 of file arch.h.

Referenced by alg_quant().

6.2.2.10 #define celt_fatal(str) _celt_fatal(str, __FILE__, __LINE__);

Definition at line 45 of file arch.h.

Referenced by denormalise_bands(), ec_byte_look_at_end(), ec_byte_write1(), ec_byte_write_at_end(), kf_work(), ki_work(), kiss_fft_stride(), and kiss_iff_stride().

6.2.2.11 #define CELT_SIG_SCALE 32768.

Definition at line 43 of file arch.h.

6.2.2.12 #define DIV32(a, b) (((celt_word32)(a))/(celt_word32)(b))

Definition at line 218 of file arch.h.

Referenced by compute_pitch_gain(), kiss_fft_alloc(), kiss_fftr_alloc(), and mdct_init().

6.2.2.13 #define DIV32_16(a, b) (((celt_word32)(a))/(celt_word16)(b))

Definition at line 216 of file arch.h.

Referenced by folding_decision().

6.2.2.14 #define ENER_SCALING 1.f

Definition at line 148 of file arch.h.

6.2.2.15 #define ENER_SCALING_1 1.f

Definition at line 149 of file arch.h.

6.2.2.16 #define EPSILON 1e-15f

Definition at line 153 of file arch.h.

Referenced by `alg_quant()`, `compute_pitch_gain()`, `find_spectral_pitch()`, `folding_decision()`, and `renormalise_vector()`.

6.2.2.17 #define EXTEND32(x) (x)

Definition at line 166 of file arch.h.

Referenced by `celt_encode_float()`, `folding_decision()`, and `mdct_init()`.

6.2.2.18 #define EXTRACT16(x) (x)

Definition at line 165 of file arch.h.

Referenced by `alg_quant()`, `celt_encode_float()`, `compute_pitch_gain()`, `find_spectral_pitch()`, and `folding_decision()`.

6.2.2.19 #define GLOBAL_STACK_SIZE 100000

Definition at line 246 of file arch.h.

6.2.2.20 #define HALF32(x) (.5f*(x))

Definition at line 183 of file arch.h.

Referenced by `folding_decision()`.

6.2.2.21 #define IMAX(a, b) ((a) > (b) ? (a) : (b))

Maximum int value.

Definition at line 66 of file arch.h.

Referenced by `celt_encode_float()`.

6.2.2.22 #define IMIN(a, b) ((a) < (b) ? (a) : (b))

Minimum int value.

Definition at line 65 of file arch.h.

Referenced by `celt_encode_float()`, `quant_bands_stereo()`, and `unquant_bands_stereo()`.

6.2.2.23 #define IMUL32(a, b) ((a)*(b))

Definition at line 54 of file arch.h.

Referenced by `ec_dec_update()`, `ec_encode()`, and `ec_encode_bin()`.

6.2.2.24 #define LOG2_BITS_PER_CHAR 3

Definition at line 238 of file arch.h.

6.2.2.25 #define MAC16_16(c, a, b) ((c)+(celt_word32)(a)*(celt_word32)(b))

Definition at line 191 of file arch.h.

Referenced by alg_quant(), alg_unquant(), compute_pitch_gain(), intra_decision(), and renormalise_vector().

6.2.2.26 #define MAC16_16_P13(c, a, b) ((c)+(a)*(b))

Definition at line 207 of file arch.h.

6.2.2.27 #define MAC16_16_Q11(c, a, b) ((c)+(a)*(b))

Definition at line 205 of file arch.h.

6.2.2.28 #define MAC16_16_Q13(c, a, b) ((c)+(a)*(b))

Definition at line 206 of file arch.h.

6.2.2.29 #define MAC16_32_Q11(c, a, b) ((c)+(a)*(b))

Definition at line 202 of file arch.h.

6.2.2.30 #define MAC16_32_Q15(c, a, b) ((c)+(a)*(b))

Definition at line 203 of file arch.h.

6.2.2.31 #define MAX16(a, b) ((a) > (b) ? (a) : (b))

Maximum 16-bit value.

Definition at line 61 of file arch.h.

Referenced by compute_pitch_gain().

6.2.2.32 #define MAX32(a, b) ((a) > (b) ? (a) : (b))

Maximum 32-bit value.

Definition at line 64 of file arch.h.

Referenced by compute_pitch_gain(), and folding_decision().

6.2.2.33 #define MIN16(a, b) ((a) < (b) ? (a) : (b))

Minimum 16-bit value.

Definition at line 60 of file arch.h.

6.2.2.34 #define MIN32(a, b) ((a) < (b) ? (a) : (b))

Minimum 32-bit value.

Definition at line 63 of file arch.h.

Referenced by celt_mode_create().

6.2.2.35 #define MULT16_16(a, b) ((celt_word32)(a)*(celt_word32)(b))

Definition at line 190 of file arch.h.

Referenced by alg_quant(), celt_encode_float(), compute_masking(), compute_pitch_gain(), find_spectral_pitch(), folding_decision(), and renormalise_vector().

6.2.2.36 #define MULT16_16_16(a, b) ((a)*(b))

Definition at line 189 of file arch.h.

Referenced by apply_pitch(), compute_pitch_gain(), and find_spectral_pitch().

6.2.2.37 #define MULT16_16_P13(a, b) ((a)*(b))

Definition at line 213 of file arch.h.

6.2.2.38 #define MULT16_16_P14(a, b) ((a)*(b))

Definition at line 214 of file arch.h.

6.2.2.39 #define MULT16_16_P15(a, b) ((a)*(b))

Definition at line 212 of file arch.h.

Referenced by compute_pitch_gain().

6.2.2.40 #define MULT16_16_Q11_32(a, b) ((a)*(b))

Definition at line 208 of file arch.h.

6.2.2.41 #define MULT16_16_Q13(a, b) ((a)*(b))

Definition at line 209 of file arch.h.

6.2.2.42 #define MULT16_16_Q14(a, b) ((a)*(b))

Definition at line 210 of file arch.h.

Referenced by compute_pitch_gain().

6.2.2.43 #define MULT16_16_Q15(a, b) ((a)*(b))

Definition at line 211 of file arch.h.

Referenced by alg_quant(), find_spectral_pitch(), quant_bands(), quant_bands_stereo(), quant_coarse_energy(), renormalise_vector(), unquant_bands_stereo(), and unquant_coarse_energy().

6.2.2.44 #define MULT16_32_P15(a, b) ((a)*(b))

Definition at line 198 of file arch.h.

6.2.2.45 #define MULT16_32_Q11(a, b) ((a)*(b))

Definition at line 193 of file arch.h.

6.2.2.46 #define MULT16_32_Q13(a, b) ((a)*(b))

Definition at line 194 of file arch.h.

6.2.2.47 #define MULT16_32_Q14(a, b) ((a)*(b))

Definition at line 195 of file arch.h.

6.2.2.48 #define MULT16_32_Q15(a, b) ((a)*(b))

Definition at line 196 of file arch.h.

Referenced by apply_pitch(), celt_encode_float(), compute_pitch_gain(), denormalise_bands(), mdct_backward(), and mdct_forward().

6.2.2.49 #define MULT16_32_Q16(a, b) ((a)*(b))

Definition at line 197 of file arch.h.

Referenced by alg_quant().

6.2.2.50 #define MULT32_32_Q31(a, b) ((a)*(b))

Definition at line 200 of file arch.h.

6.2.2.51 #define NEG16(x) (-x)

Definition at line 163 of file arch.h.

6.2.2.52 #define NEG32(x) (-x)

Definition at line 164 of file arch.h.

6.2.2.53 #define NORM_SCALING 1.f

Definition at line 146 of file arch.h.

6.2.2.54 #define NORM_SCALING_1 1.f

Definition at line 147 of file arch.h.

6.2.2.55 #define PDIV32(a, b) (((celt_word32)(a))/(celt_word32)(b))

Definition at line 219 of file arch.h.

6.2.2.56 #define PDIV32_16(a, b) (((celt_word32)(a))/(celt_word16)(b))

Definition at line 217 of file arch.h.

Referenced by apply_pitch(), and compute_pitch_gain().

6.2.2.57 #define PGAIN_SCALING 1.f

Definition at line 150 of file arch.h.

6.2.2.58 #define PGAIN_SCALING_1 1.f

Definition at line 151 of file arch.h.

6.2.2.59 #define PRINT_MIPS(file)

Definition at line 70 of file arch.h.

Referenced by main().

6.2.2.60 #define PSHR(a, shift) (a)

Definition at line 177 of file arch.h.

6.2.2.61 #define PSHR16(a, shift) (a)

Definition at line 171 of file arch.h.

6.2.2.62 #define PSHR32(a, shift) (a)

Definition at line 172 of file arch.h.

Referenced by kiss_fftr_twiddles(), and renormalise_vector().

6.2.2.63 #define Q15_ONE ((celt_word16)1.f)

Definition at line 157 of file arch.h.

6.2.2.64 #define Q15_ONE_1 ((celt_word16)1.f)

Definition at line 158 of file arch.h.

6.2.2.65 #define Q15ONE 1.0f

Definition at line 143 of file arch.h.

Referenced by celt_encode_float(), celt_mode_create(), compute_pitch_gain(), intra_fold(), psydecay_init(), quant_bands_stereo(), quant_coarse_energy(), renormalise_bands(), renormalise_vector(), unquant_bands_stereo(), and unquant_coarse_energy().

6.2.2.66 #define Q30ONE 1.0f

Definition at line 144 of file arch.h.

6.2.2.67 #define QCONST16(x, bits) (x)

Definition at line 160 of file arch.h.

Referenced by alg_quant(), apply_pitch(), celt_encode_float(), celt_encoder_create(), celt_mode_create(), compute_pitch_gain(), folding_decision(), quant_bands_stereo(), quant_coarse_energy(), quant_energy_finalise(), quant_fine_energy(), unquant_bands_stereo(), unquant_coarse_energy(), unquant_energy_finalise(), and unquant_fine_energy().

6.2.2.68 #define QCONST32(x, bits) (x)

Definition at line 161 of file arch.h.

Referenced by compute_pitch_gain(), and folding_decision().

6.2.2.69 #define ROUND16(a, shift) (a)

Definition at line 182 of file arch.h.

6.2.2.70 #define SATURATE(x, a) (x)

Definition at line 180 of file arch.h.

6.2.2.71 #define SATURATE16(x, a) (x)

Definition at line 174 of file arch.h.

6.2.2.72 #define SATURATE32(x, a) (x)

Definition at line 175 of file arch.h.

6.2.2.73 #define SCALEIN(a) ((a)*CELT_SIG_SCALE)

Definition at line 221 of file arch.h.

Referenced by celt_encode_float().

6.2.2.74 #define SCALEOUT(a) ((a)*(1/CELT_SIG_SCALE))

Definition at line 222 of file arch.h.

Referenced by celt_encode().

6.2.2.75 #define SHL(a, shift) (a)

Definition at line 179 of file arch.h.

6.2.2.76 #define SHL16(a, shift) (a)

Definition at line 168 of file arch.h.

Referenced by alg_quant(), quant_bands(), quant_bands_stereo(), quant_coarse_energy(), quant_energy_finalise(), quant_fine_energy(), unquant_bands_stereo(), unquant_coarse_energy(), unquant_energy_finalise(), and unquant_fine_energy().

6.2.2.77 #define SHL32(a, shift) (a)

Definition at line 170 of file arch.h.

Referenced by apply_pitch(), celt_encode_float(), compute_pitch_gain(), denormalise_bands(), folding_decision(), kiss_fft_alloc(), kiss_fftr_alloc(), mdct_init(), and renormalise_vector().

6.2.2.78 #define SHR(a, shift) (a)

Definition at line 178 of file arch.h.

6.2.2.79 #define SHR16(a, shift) (a)

Definition at line 167 of file arch.h.

Referenced by quant_energy_finalise(), quant_fine_energy(), unquant_energy_finalise(), and unquant_fine_energy().

6.2.2.80 #define SHR32(a, shift) (a)

Definition at line 169 of file arch.h.

Referenced by `alg_quant()`, `celt_encode_float()`, `compute_pitch_gain()`, `denormalise_bands()`, `find_spectral_pitch()`, `intra_decision()`, and `kiss_fftr_twiddles()`.

6.2.2.81 #define SUB16(a, b) ((a)-(b))

Definition at line 186 of file arch.h.

Referenced by `compute_pitch_gain()`, `find_spectral_pitch()`, `intra_decision()`, `quant_fine_energy()`, and `unquant_fine_energy()`.

6.2.2.82 #define SUB32(a, b) ((a)-(b))

Definition at line 188 of file arch.h.

Referenced by `celt_encode_float()`, and `kiss_fftr_twiddles()`.

6.2.2.83 #define UADD32(a, b) ((a)+(b))

Definition at line 67 of file arch.h.

6.2.2.84 #define UMUL16_16(a, b) ((celt_int32)(a)*(celt_int32)(b))

Definition at line 56 of file arch.h.

6.2.2.85 #define UMUL32(a, b) ((celt_int32)(a)*(celt_int32)(b))

Definition at line 55 of file arch.h.

6.2.2.86 #define USUB32(a, b) ((a)-(b))

Definition at line 68 of file arch.h.

6.2.2.87 #define VERY_LARGE16 1e15f

Definition at line 156 of file arch.h.

Referenced by `alg_quant()`.

6.2.2.88 #define VERY_LARGE32 1e15f

Definition at line 155 of file arch.h.

6.2.2.89 #define VERY_SMALL 1e-15f

Definition at line 154 of file arch.h.

6.2.2.90 #define VSHR32(a, shift) (a)

Definition at line 173 of file arch.h.

6.2.3 Typedef Documentation**6.2.3.1 typedef float celt_ener**

Definition at line 139 of file arch.h.

6.2.3.2 typedef float celt_mask

Definition at line 141 of file arch.h.

6.2.3.3 typedef float celt_norm

Definition at line 138 of file arch.h.

6.2.3.4 typedef float celt_pgain

Definition at line 140 of file arch.h.

6.2.3.5 typedef float celt_sig

Definition at line 137 of file arch.h.

6.2.3.6 typedef float celt_word16

Definition at line 134 of file arch.h.

6.2.3.7 typedef float celt_word32

Definition at line 135 of file arch.h.

6.3 libcelt/bands.c File Reference

```
#include <math.h>
#include "bands.h"
#include "modes.h"
#include "vq.h"
#include "cwrs.h"
#include "stack_alloc.h"
#include "os_support.h"
#include "mathops.h"
#include "rate.h"
```

Functions

- void `compute_band_energies` (const `CELTMode` **m*, const `celt_sig` **X*, `celt_ener` **bank*, int *_C*)
- void `normalise_bands` (const `CELTMode` **m*, const `celt_sig` *restrict *freq*, `celt_norm` *restrict *X*, const `celt_ener` **bank*, int *_C*)
- void `renormalise_bands` (const `CELTMode` **m*, `celt_norm` *restrict *X*, int *_C*)
- void `denormalise_bands` (const `CELTMode` **m*, const `celt_norm` *restrict *X*, `celt_sig` *restrict *freq*, const `celt_ener` **bank*, int *_C*)
- int `compute_pitch_gain` (const `CELTMode` **m*, const `celt_sig` **X*, const `celt_sig` **P*, int *norm_rate*, int **gain_id*, int *_C*, `celt_word16` **gain_prod*)
- void `apply_pitch` (const `CELTMode` **m*, `celt_sig` **X*, const `celt_sig` **P*, int *gain_id*, int *pred*, int *_C*)
- int `folding_decision` (const `CELTMode` **m*, `celt_norm` **X*, `celt_word16` **average*, int **last_decision*, int *_C*)
- void `quant_bands` (const `CELTMode` **m*, `celt_norm` *restrict *X*, const `celt_ener` **bandE*, int **pulses*, int *shortBlocks*, int *fold*, int *total_bits*, int *encode*, void **enc_dec*)
- void `quant_bands_stereo` (const `CELTMode` **m*, `celt_norm` **_X*, const `celt_ener` **bandE*, int **pulses*, int *shortBlocks*, int *fold*, int *total_bits*, `ec_enc` **enc*)
- void `unquant_bands_stereo` (const `CELTMode` **m*, `celt_norm` **_X*, const `celt_ener` **bandE*, int **pulses*, int *shortBlocks*, int *fold*, int *total_bits*, `ec_dec` **dec*)

6.3.1 Function Documentation

6.3.1.1 void apply_pitch (const `CELTMode` * *m*, `celt_sig` * *X*, const `celt_sig` * *P*, int *gain_id*, int *pred*, int *_C*)

Definition at line 309 of file `bands.c`.

References `ADD16`, `CHANNELS`, `FRAMESIZE`, `MULT16_16_16`, `MULT16_32_Q15`, `PDIV32_16`, `CELTMode::pitchEnd`, `QCONST16`, and `SHL32`.

Referenced by `celt_decode_float()`, and `celt_encode_float()`.

6.3.1.2 void compute_band_energies (const `CELTMode` * *m*, const `celt_sig` * *X*, `celt_ener` * *bands*, int *_C*)

Compute the amplitude (sqrt energy) in each of the bands

Parameters:

- m* Mode data
- X* Spectrum
- bands* Square root of the energy for each band (returned)

Definition at line 114 of file bands.c.

References CHANNELS, CELTMode::eBands, FRAMESIZE, and CELTMode::nbEBands.

Referenced by celt_encode_float().

6.3.1.3 int compute_pitch_gain (const CELTMode * *m*, const celt_sig * *X*, const celt_sig * *P*, int *norm_rate*, int * *gain_id*, int *_C*, celt_word16 * *gain_prod*)

Compute the pitch predictor gain for each pitch band

Parameters:

- m* Mode data
- X* Spectrum to predict
- P* Pitch vector (normalised)
- gains* Gain computed for each pitch band (returned)
- bank* Square root of the energy for each band

Definition at line 216 of file bands.c.

References ABS32, ADD16, ADD32, celt_ilog2, celt_sqrt, CHANNELS, DIV32, EPSILON, EXTRACT16, FRAMESIZE, MAC16_16, MAX16, MAX32, MULT16_16, MULT16_16_16, MULT16_16_P15, MULT16_16_Q14, MULT16_32_Q15, PDIV32_16, CELTMode::pitchEnd, Q15ONE, QCONST16, QCONST32, SHL32, SHR32, and SUB16.

Referenced by celt_encode_float().

6.3.1.4 void denormalise_bands (const CELTMode * *m*, const celt_norm *restrict *X*, celt_sig *restrict *freq*, const celt_ener * *bands*, int *_C*)

Denormalise each band of *X* to restore full amplitude

Parameters:

- m* Mode data
- X* Spectrum (returned de-normalised)
- bands* Square root of the energy for each band

Definition at line 193 of file bands.c.

References celt_fatal, CHANNELS, CELTMode::eBands, FRAMESIZE, MULT16_32_Q15, CELTMode::nbEBands, SHL32, and SHR32.

Referenced by celt_decode_float(), and celt_encode_float().

6.3.1.5 `int folding_decision (const CELTMode * m, celt_norm * X, celt_word16 * average, int * last_decision, int _C)`

Definition at line 373 of file bands.c.

References ABS16, ADD32, celt_sqrt, CHANNELS, DIV32_16, CELTMode::eBands, EPSILON, EXTEND32, EXTRACT16, FRAMESIZE, HALF32, MAX32, MULT16_16, CELTMode::nbEBands, QCONST16, QCONST32, and SHL32.

Referenced by celt_encode_float().

6.3.1.6 `void normalise_bands (const CELTMode * m, const celt_sig * restrict freq, celt_norm * restrict X, const celt_ener * bands, int _C)`

Normalise each band of X such that the energy in each band is equal to 1

Parameters:

- m* Mode data
- X* Spectrum (returned normalised)
- bands* Square root of the energy for each band

Definition at line 159 of file bands.c.

References CHANNELS, CELTMode::eBands, FRAMESIZE, and CELTMode::nbEBands.

Referenced by celt_encode_float().

6.3.1.7 `void quant_bands (const CELTMode * m, celt_norm * restrict X, const celt_ener * bandE, int * pulses, int time_domain, int fold, int total_bits, int encode, void * enc_dec)`

Quantisation/encoding of the residual spectrum

Parameters:

- m* Mode data
- X* Residual (normalised)
- total_bits* Total number of bits that can be used for the frame (including the ones already spent)
- enc* Entropy encoder

Definition at line 444 of file bands.c.

References alg_quant(), alg_unquant(), ALLOC, BITRES, CELTMode::bits, celt_sqrt, CELTMode::eBands, ec_dec_tell(), ec_enc_tell(), intra_fold(), MULT16_16_Q15, CELTMode::nbEBands, CELTMode::nbShortMdcts, RESTORE_STACK, SAVE_STACK, SHL16, and VARDECL.

Referenced by celt_decode_float(), and celt_encode_float().

6.3.1.8 `void quant_bands_stereo (const CELTMode * m, celt_norm * _X, const celt_ener * bandE, int * pulses, int shortBlocks, int fold, int total_bits, ec_enc * enc)`

Definition at line 514 of file bands.c.

References alg_quant(), ALLOC, BITRES, CELTMode::bits, celt_sqrt, CELTMode::eBands, ec_enc_bits(), ec_enc_tell(), ec_enc_uint(), IMIN, intra_fold(), log2_frac(), MULT16_16_Q15,

CELTMode::nbEBands, CELTMode::nbShortMdcts, Q15ONE, QCONST16, QTHETA_OFFSET, renormalise_vector(), RESTORE_STACK, SAVE_STACK, SHL16, and VARDECL.

Referenced by celt_encode_float().

6.3.1.9 void renormalise_bands (const CELTMode * m, celt_norm *restrict X, int _C)

Definition at line 179 of file bands.c.

References CHANNELS, CELTMode::eBands, CELTMode::nbEBands, Q15ONE, and renormalise_vector().

Referenced by celt_encode_float().

6.3.1.10 void unquant_bands_stereo (const CELTMode * m, celt_norm * _X, const celt_ener * bandE, int * pulses, int shortBlocks, int fold, int total_bits, ec_dec * dec)

Definition at line 746 of file bands.c.

References alg_unquant(), ALLOC, BITRES, CELTMode::bits, celt_sqrt, CELTMode::eBands, ec_dec_bits(), ec_dec_tell(), ec_dec_uint(), IMIN, intra_fold(), log2_frac(), MULT16_16_Q15, CELTMode::nbEBands, CELTMode::nbShortMdcts, Q15ONE, QCONST16, QTHETA_OFFSET, renormalise_vector(), RESTORE_STACK, SAVE_STACK, SHL16, and VARDECL.

Referenced by celt_decode_float().

6.4 libcelt/bands.h File Reference

```
#include "arch.h"
#include "modes.h"
#include "entenc.h"
#include "entdec.h"
#include "rate.h"
```

Functions

- void `compute_band_energies` (const `CELTMode` *m, const `celt_sig` *X, `celt_ener` *bands, int _C)
- void `normalise_bands` (const `CELTMode` *m, const `celt_sig` *restrict freq, `celt_norm` *restrict X, const `celt_ener` *bands, int _C)
- void `renormalise_bands` (const `CELTMode` *m, `celt_norm` *restrict X, int _C)
- void `denormalise_bands` (const `CELTMode` *m, const `celt_norm` *restrict X, `celt_sig` *restrict freq, const `celt_ener` *bands, int _C)
- int `compute_pitch_gain` (const `CELTMode` *m, const `celt_sig` *X, const `celt_sig` *P, int norm_rate, int *gain_id, int _C, `celt_word16` *gain_prod)
- void `apply_pitch` (const `CELTMode` *m, `celt_sig` *X, const `celt_sig` *P, int gain_id, int pred, int _C)
- int `folding_decision` (const `CELTMode` *m, `celt_norm` *X, `celt_word16` *average, int *last_decision, int _C)
- void `quant_bands` (const `CELTMode` *m, `celt_norm` *restrict X, const `celt_ener` *bandE, int *pulses, int time_domain, int fold, int total_bits, int encode, void *enc_dec)
- void `quant_bands_stereo` (const `CELTMode` *m, `celt_norm` *restrict X, const `celt_ener` *bandE, int *pulses, int time_domain, int fold, int total_bits, `ec_enc` *enc)
- void `unquant_bands` (const `CELTMode` *m, `celt_norm` *restrict X, const `celt_ener` *bandE, int *pulses, int time_domain, int fold, int total_bits, `ec_dec` *dec)
- void `unquant_bands_stereo` (const `CELTMode` *m, `celt_norm` *restrict X, const `celt_ener` *bandE, int *pulses, int time_domain, int fold, int total_bits, `ec_dec` *dec)
- void `stereo_decision` (const `CELTMode` *m, `celt_norm` *restrict X, int *stereo_mode, int len)

6.4.1 Function Documentation

6.4.1.1 void `apply_pitch` (const `CELTMode` * m, `celt_sig` * X, const `celt_sig` * P, int *gain_id*, int *pred*, int _C)

Definition at line 309 of file bands.c.

References `ADD16`, `CHANNELS`, `FRAMESIZE`, `MULT16_16_16`, `MULT16_32_Q15`, `PDIV32_16`, `CELTMode::pitchEnd`, `QCONST16`, and `SHL32`.

Referenced by `celt_decode_float()`, and `celt_encode_float()`.

6.4.1.2 void `compute_band_energies` (const `CELTMode` * m, const `celt_sig` * X, `celt_ener` * bands, int _C)

Compute the amplitude (sqrt energy) in each of the bands

Parameters:

m Mode data

X Spectrum

bands Square root of the energy for each band (returned)

Definition at line 114 of file bands.c.

References CHANNELS, CELTMode::eBands, FRAMESIZE, and CELTMode::nbEBands.

Referenced by celt_encode_float().

6.4.1.3 int compute_pitch_gain (const CELTMode * *m*, const celt_sig * *X*, const celt_sig * *P*, int *norm_rate*, int * *gain_id*, int *_C*, celt_word16 * *gain_prod*)

Compute the pitch predictor gain for each pitch band

Parameters:

m Mode data

X Spectrum to predict

P Pitch vector (normalised)

gains Gain computed for each pitch band (returned)

bank Square root of the energy for each band

Definition at line 216 of file bands.c.

References ABS32, ADD16, ADD32, celt_ilog2, celt_sqrt, CHANNELS, DIV32, EPSILON, EXTRACT16, FRAMESIZE, MAC16_16, MAX16, MAX32, MULT16_16, MULT16_16_16, MULT16_16_P15, MULT16_16_Q14, MULT16_32_Q15, PDIV32_16, CELTMode::pitchEnd, Q15ONE, QCONST16, QCONST32, SHL32, SHR32, and SUB16.

Referenced by celt_encode_float().

6.4.1.4 void denormalise_bands (const CELTMode * *m*, const celt_norm * *restrict X*, celt_sig * *restrict freq*, const celt_ener * *bands*, int *_C*)

Denormalise each band of *X* to restore full amplitude

Parameters:

m Mode data

X Spectrum (returned de-normalised)

bands Square root of the energy for each band

Definition at line 193 of file bands.c.

References celt_fatal, CHANNELS, CELTMode::eBands, FRAMESIZE, MULT16_32_Q15, CELTMode::nbEBands, SHL32, and SHR32.

Referenced by celt_decode_float(), and celt_encode_float().

6.4.1.5 int folding_decision (const CELTMode * *m*, celt_norm * *X*, celt_word16 * *average*, int * *last_decision*, int *_C*)

Definition at line 373 of file bands.c.

References ABS16, ADD32, celt_sqrt, CHANNELS, DIV32_16, CELTMode::eBands, EPSILON, EXTEND32, EXTRACT16, FRAMESIZE, HALF32, MAX32, MULT16_16, CELTMode::nbEBands, QCONST16, QCONST32, and SHL32.

Referenced by celt_encode_float().

6.4.1.6 void normalise_bands (const CELTMode * *m*, const celt_sig *restrict *freq*, celt_norm *restrict *X*, const celt_ener * *bands*, int *_C*)

Normalise each band of *X* such that the energy in each band is equal to 1

Parameters:

m Mode data

X Spectrum (returned normalised)

bands Square root of the energy for each band

Definition at line 159 of file bands.c.

References CHANNELS, CELTMode::eBands, FRAMESIZE, and CELTMode::nbEBands.

Referenced by celt_encode_float().

6.4.1.7 void quant_bands (const CELTMode * *m*, celt_norm *restrict *X*, const celt_ener * *bandE*, int * *pulses*, int *time_domain*, int *fold*, int *total_bits*, int *encode*, void * *enc_dec*)

Quantisation/encoding of the residual spectrum

Parameters:

m Mode data

X Residual (normalised)

total_bits Total number of bits that can be used for the frame (including the ones already spent)

enc Entropy encoder

Definition at line 444 of file bands.c.

References alg_quant(), alg_unquant(), ALLOC, BITRES, CELTMode::bits, celt_sqrt, CELTMode::eBands, ec_dec_tell(), ec_enc_tell(), intra_fold(), MULT16_16_Q15, CELTMode::nbEBands, CELTMode::nbShortMdcts, RESTORE_STACK, SAVE_STACK, SHL16, and VARDECL.

Referenced by celt_decode_float(), and celt_encode_float().

6.4.1.8 void quant_bands_stereo (const CELTMode * *m*, celt_norm *restrict *X*, const celt_ener * *bandE*, int * *pulses*, int *time_domain*, int *fold*, int *total_bits*, ec_enc * *enc*)

6.4.1.9 void renormalise_bands (const CELTMode * *m*, celt_norm *restrict *X*, int *_C*)

Definition at line 179 of file bands.c.

References CHANNELS, CELTMode::eBands, CELTMode::nbEBands, Q15ONE, and renormalise_vector().

Referenced by celt_encode_float().

6.4.1.10 void `stereo_decision` (const CELTMode * *m*, celt_norm *restrict *X*, int * *stereo_mode*, int *len*)

6.4.1.11 void `unquant_bands` (const CELTMode * *m*, celt_norm *restrict *X*, const celt_ener * *bandE*, int * *pulses*, int *time_domain*, int *fold*, int *total_bits*, ec_dec * *dec*)

Decoding of the residual spectrum

Parameters:

m Mode data

X Residual (normalised)

total_bits Total number of bits that can be used for the frame (including the ones already spent)

dec Entropy decoder

6.4.1.12 void `unquant_bands_stereo` (const CELTMode * *m*, celt_norm *restrict *X*, const celt_ener * *bandE*, int * *pulses*, int *time_domain*, int *fold*, int *total_bits*, ec_dec * *dec*)

6.5 libcelt/c64_fft.c File Reference

```
#include "c64_fft.h"
#include "dsp_fft16x16t.h"
#include "dsp_fft32x32s.h"
#include "dsp_ifft32x32.h"
```

Defines

- #define [PI](#) 3.14159265358979323846
- #define [NBCACHE](#) 3

Functions

- int [gen_twiddle16](#) (short *w, int n, double scale)
- int [gen_twiddle32](#) (int *w, int n, double scale)
- [c64_fft_t](#) * [c64_fft16_alloc](#) (int length, int x, int y)
- [c64_fft_t](#) * [c64_fft32_alloc](#) (int length, int x, int y)
- void [c64_fft16_free](#) ([c64_fft_t](#) *state)
- void [c64_fft32_free](#) ([c64_fft_t](#) *state)
- void [c64_fft16_inplace](#) ([c64_fft_t](#) *restrict state, [celt_int16](#) *X)
- void [c64_fft32](#) ([c64_fft_t](#) *restrict state, const [celt_int32](#) *X, [celt_int32](#) *Y)
- void [c64_ifft16](#) ([c64_fft_t](#) *restrict state, const [celt_int16](#) *X, [celt_int16](#) *Y)
- void [c64_ifft32](#) ([c64_fft_t](#) *restrict state, const [celt_int32](#) *X, [celt_int32](#) *Y)

6.5.1 Define Documentation

6.5.1.1 #define NBCACHE 3

Definition at line 157 of file `c64_fft.c`.

Referenced by `c64_fft16_alloc()`, and `c64_fft32_alloc()`.

6.5.1.2 #define PI 3.14159265358979323846

Definition at line 41 of file `c64_fft.c`.

Referenced by `gen_twiddle16()`, and `gen_twiddle32()`.

6.5.2 Function Documentation

6.5.2.1 `c64_fft_t*` `c64_fft16_alloc` (int *length*, int *x*, int *y*)

Definition at line 161 of file `c64_fft.c`.

References `gen_twiddle16()`, `c64_fft_t::itwiddle`, `NBCACHE`, `c64_fft_t::nfft`, `c64_fft_t::shift`, and `c64_fft_t::twiddle`.

6.5.2.2 void c64_fft16_free (c64_fft_t * state)

Definition at line 235 of file c64_fft.c.

References c64_fft32_free().

6.5.2.3 void c64_fft16_inplace (c64_fft_t *restrict state, celt_int16 * X)

Definition at line 246 of file c64_fft.c.

References ALLOC, RESTORE_STACK, SAVE_STACK, and VARDECL.

6.5.2.4 void c64_fft32 (c64_fft_t *restrict state, const celt_int32 * X, celt_int32 * Y)

Definition at line 273 of file c64_fft.c.

References ALLOC, RESTORE_STACK, SAVE_STACK, and VARDECL.

6.5.2.5 c64_fft_t* c64_fft32_alloc (int length, int x, int y)

Definition at line 200 of file c64_fft.c.

References gen_twiddle32(), c64_fft_t::itwiddle, NBCACHE, c64_fft_t::nfft, c64_fft_t::shift, and c64_fft_t::twiddle.

6.5.2.6 void c64_fft32_free (c64_fft_t * state)

Definition at line 241 of file c64_fft.c.

Referenced by c64_fft16_free().

6.5.2.7 void c64_iff16 (c64_fft_t *restrict state, const celt_int16 * X, celt_int16 * Y)

Definition at line 291 of file c64_fft.c.

References ALLOC, RESTORE_STACK, SAVE_STACK, and VARDECL.

6.5.2.8 void c64_iff32 (c64_fft_t *restrict state, const celt_int32 * X, celt_int32 * Y)

Definition at line 323 of file c64_fft.c.

References ALLOC, celt_assert, RESTORE_STACK, SAVE_STACK, and VARDECL.

6.5.2.9 int gen_twiddle16 (short * w, int n, double scale)

Definition at line 85 of file c64_fft.c.

References PI.

Referenced by c64_fft16_alloc().

6.5.2.10 int gen_twiddle32 (int * *w*, int *n*, double *scale*)

Definition at line 133 of file c64_fft.c.

References PI.

Referenced by c64_fft32_alloc().

6.6 libcelt/c64_fft.h File Reference

```
#include "config.h"
#include "arch.h"
#include "os_support.h"
#include "mathops.h"
#include "stack_alloc.h"
```

Data Structures

- struct [c64_fft_t](#)

Functions

- [c64_fft_t * c64_fft16_alloc](#) (int length, int x, int y)
- void [c64_fft16_free](#) (c64_fft_t *state)
- void [c64_fft16_inplace](#) (c64_fft_t *state, celt_int16 *X)
- void [c64_ifft16](#) (c64_fft_t *state, const celt_int16 *X, celt_int16 *Y)
- [c64_fft_t * c64_fft32_alloc](#) (int length, int x, int y)
- void [c64_fft32_free](#) (c64_fft_t *state)
- void [c64_fft32](#) (c64_fft_t *state, const celt_int32 *X, celt_int32 *Y)
- void [c64_ifft32](#) (c64_fft_t *state, const celt_int32 *X, celt_int32 *Y)

6.6.1 Function Documentation

6.6.1.1 [c64_fft_t* c64_fft16_alloc](#) (int length, int x, int y)

Definition at line 161 of file [c64_fft.c](#).

References [gen_twiddle16\(\)](#), [c64_fft_t::itwiddle](#), [NBCACHE](#), [c64_fft_t::nfft](#), [c64_fft_t::shift](#), and [c64_fft_t::twiddle](#).

6.6.1.2 void [c64_fft16_free](#) (c64_fft_t * state)

Definition at line 235 of file [c64_fft.c](#).

References [c64_fft32_free\(\)](#).

6.6.1.3 void [c64_fft16_inplace](#) (c64_fft_t * state, celt_int16 * X)

6.6.1.4 void [c64_fft32](#) (c64_fft_t * state, const celt_int32 * X, celt_int32 * Y)

6.6.1.5 [c64_fft_t* c64_fft32_alloc](#) (int length, int x, int y)

Definition at line 200 of file [c64_fft.c](#).

References [gen_twiddle32\(\)](#), [c64_fft_t::itwiddle](#), [NBCACHE](#), [c64_fft_t::nfft](#), [c64_fft_t::shift](#), and [c64_fft_t::twiddle](#).

6.6.1.6 void c64_fft32_free (c64_fft_t * state)

Definition at line 241 of file c64_fft.c.

Referenced by c64_fft16_free().

6.6.1.7 void c64_iff16 (c64_fft_t * state, const celt_int16 * X, celt_int16 * Y)**6.6.1.8 void c64_iff32 (c64_fft_t * state, const celt_int32 * X, celt_int32 * Y)**

6.7 libcelt/celt.c File Reference

```
#include "os_support.h"
#include "mdct.h"
#include <math.h>
#include "celt.h"
#include "pitch.h"
#include "kiss_fftr.h"
#include "bands.h"
#include "modes.h"
#include "entcode.h"
#include "quant_bands.h"
#include "psy.h"
#include "rate.h"
#include "stack_alloc.h"
#include "mathops.h"
#include "float_cast.h"
#include <stdarg.h>
```

Data Structures

- struct [CELTEncoder](#)
Encoder state.
- struct [CELTDecoder](#)
Decoder state.

Defines

- #define [CELT_C](#)
- #define [ENCODERINVALID](#) 0x4c434554
- #define [ENCODERPARTIAL](#) 0x54445434c
- #define [ENCODERFREED](#) 0x4c004500
- #define [FLAG_NONE](#) 0
- #define [FLAG_INTRA](#) (1U<<13)
- #define [FLAG_PITCH](#) (1U<<12)
- #define [FLAG_SHORT](#) (1U<<11)
- #define [FLAG_FOLD](#) (1U<<10)
- #define [FLAG_MASK](#) (FLAG_INTRA|FLAG_PITCH|FLAG_SHORT|FLAG_FOLD)
- #define [DECODE_BUFFER_SIZE](#) MAX_PERIOD
- #define [DECODERINVALID](#) 0x4c434454
- #define [DECODERPARTIAL](#) 0x54444434c
- #define [DECODERFREED](#) 0x4c004400

Functions

- `CELTEncoder * celt_encoder_create` (const `CELTMode *mode`, int channels, int *error)
- void `celt_encoder_destroy` (`CELTEncoder *st`)
- int `celt_encode_float` (`CELTEncoder *restrict st`, const `celt_sig *pcm`, `celt_sig *optional_synthesis`, unsigned char *compressed, int nbCompressedBytes)
- int `celt_encode` (`CELTEncoder *restrict st`, const `celt_int16 *pcm`, `celt_int16 *optional_synthesis`, unsigned char *compressed, int nbCompressedBytes)
- int `celt_encoder_ctl` (`CELTEncoder *restrict st`, int request,...)
- int `check_decoder` (const `CELTDDecoder *st`)
- `CELTDDecoder * celt_decoder_create` (const `CELTMode *mode`, int channels, int *error)
- void `celt_decoder_destroy` (`CELTDDecoder *st`)
- int `celt_decode_float` (`CELTDDecoder *restrict st`, const unsigned char *data, int len, `celt_sig *restrict pcm`)
- int `celt_decode` (`CELTDDecoder *restrict st`, const unsigned char *data, int len, `celt_int16 *restrict pcm`)
- int `celt_decoder_ctl` (`CELTDDecoder *restrict st`, int request,...)
- const char * `celt_strerror` (int error)

6.7.1 Define Documentation

6.7.1.1 #define CELT_C

Definition at line 38 of file celt.c.

6.7.1.2 #define DECODE_BUFFER_SIZE MAX_PERIOD

Definition at line 1074 of file celt.c.

Referenced by `celt_decode_float()`, `celt_decoder_create()`, and `celt_decoder_ctl()`.

6.7.1.3 #define DECODERFREED 0x4c004400

Definition at line 1079 of file celt.c.

Referenced by `celt_decoder_destroy()`, and `check_decoder()`.

6.7.1.4 #define DECODERPARTIAL 0x5444434c

Definition at line 1078 of file celt.c.

Referenced by `celt_decoder_create()`, and `celt_decoder_destroy()`.

6.7.1.5 #define DECODERINVALID 0x4c434454

Definition at line 1077 of file celt.c.

Referenced by `celt_decoder_create()`, `celt_decoder_destroy()`, and `check_decoder()`.

6.7.1.6 #define ENCODERFREED 0x4c004500

Definition at line 73 of file celt.c.

Referenced by celt_encoder_destroy().

6.7.1.7 #define ENCODERPARTIAL 0x5445434c

Definition at line 72 of file celt.c.

Referenced by celt_encoder_create(), and celt_encoder_destroy().

6.7.1.8 #define ENCODERVALID 0x4c434554

Definition at line 71 of file celt.c.

Referenced by celt_encoder_create(), and celt_encoder_destroy().

6.7.1.9 #define FLAG_FOLD (1U<<10)

Definition at line 445 of file celt.c.

6.7.1.10 #define FLAG_INTRA (1U<<13)

Definition at line 442 of file celt.c.

6.7.1.11 #define FLAG_MASK (FLAG_INTRA|FLAG_PITCH|FLAG_SHORT|FLAG_FOLD)

Definition at line 446 of file celt.c.

6.7.1.12 #define FLAG_NONE 0

Definition at line 441 of file celt.c.

6.7.1.13 #define FLAG_PITCH (1U<<12)

Definition at line 443 of file celt.c.

6.7.1.14 #define FLAG_SHORT (1U<<11)

Definition at line 444 of file celt.c.

6.7.2 Function Documentation**6.7.2.1 int celt_decode (CELTDecoder *restrict st, const unsigned char * data, int len, celt_int16 *restrict pcm)**

Definition at line 1468 of file celt.c.

References ALLOC, CELT_BAD_ARG, celt_decode_float(), CELT_INVALID_MODE, CELT_INVALID_STATE, CELT_OK, CHANNELS, check_decoder(), check_mode(), RESTORE_STACK, SAVE_STACK, and VARDECL.

Referenced by main().

6.7.2.2 int celt_decode_float (CELTDecoder *restrict st, const unsigned char * data, int len, celt_sig *restrict pcm)

< Interleaved signal MDCTs

< Interleaved normalised MDCTs

< Interleaved signal MDCTs

Definition at line 1294 of file celt.c.

References ALLOC, apply_pitch(), CELT_BAD_ARG, CELT_INVALID_MODE, CELT_INVALID_STATE, CELT_MOVE, CELT_OK, CHANNELS, check_decoder(), check_mode(), compute_allocation(), DECODE_BUFFER_SIZE, denormalise_bands(), ec_byte_readinit(), ec_dec_init(), ec_dec_tell(), ec_dec_uint(), MAX_PERIOD, quant_bands(), RESTORE_STACK, SAVE_STACK, unquant_bands_stereo(), unquant_coarse_energy(), unquant_energy_finalise(), unquant_fine_energy(), and VARDECL.

Referenced by celt_decode().

6.7.2.3 int celt_decoder_ctl (CELTDecoder *restrict st, int request, ...)

Definition at line 1497 of file celt.c.

References CELT_BAD_ARG, CELT_GET_MODE_REQUEST, CELT_INVALID_MODE, CELT_INVALID_STATE, CELT_MEMSET, CELT_OK, CELT_RESET_STATE, CELT_UNIMPLEMENTED, check_decoder(), check_mode(), DECODE_BUFFER_SIZE, and CELTMode::nbEBands.

6.7.2.4 int celt_encode (CELTEncoder *restrict st, const celt_int16 * pcm, celt_int16 * optional_synthesis, unsigned char * compressed, int nbCompressedBytes)

Definition at line 935 of file celt.c.

References ALLOC, CELT_BAD_ARG, celt_encode_float(), CELT_INVALID_MODE, CELT_INVALID_STATE, CELT_OK, CHANNELS, check_mode(), RESTORE_STACK, SAVE_STACK, SCALEOUT, and VARDECL.

Referenced by main().

6.7.2.5 int celt_encode_float (CELTEncoder *restrict st, const celt_sig * pcm, celt_sig * optional_synthesis, unsigned char * compressed, int nbCompressedBytes)

< Interleaved signal MDCTs

< Interleaved normalised MDCTs

< Interleaved signal MDCTs

Definition at line 541 of file celt.c.

References ABS32, ALLOC, apply_pitch(), BITRES, CELT_BAD_ARG, CELT_COPY, CELT_INVALID_MODE, CELT_INVALID_STATE, CELT_MEMSET, CELT_MOVE, CELT_OK, celt_rcp, CHANNELS, check_mode(), compute_allocation(), compute_band_energies(), compute_pitch_gain(),

denormalise_bands(), ec_byte_shrink(), ec_byte_writeinit_buffer(), ec_enc_done(), ec_enc_init(), ec_enc_tell(), ec_enc_uint(), EXTEND32, EXTRACT16, find_spectral_pitch(), folding_decision(), IMAX, IMIN, intra_decision(), MAX_PERIOD, MULT16_16, MULT16_32_Q15, CELTMode::nbShortMdcts, normalise_bands(), Q15ONE, QCONST16, quant_bands(), quant_bands_stereo(), quant_coarse_energy(), quant_energy_finalise(), quant_fine_energy(), renormalise_bands(), RESTORE_STACK, SAVE_STACK, SCALEIN, SHL32, SHR32, SUB32, and VARDECL.

Referenced by celt_encode().

6.7.2.6 int celt_encoder_ctl (CELTEncoder *restrict st, int request, ...)

Definition at line 969 of file celt.c.

References CELT_BAD_ARG, CELT_GET_MODE_REQUEST, CELT_INVALID_MODE, CELT_INVALID_STATE, CELT_MEMSET, CELT_OK, CELT_RESET_STATE, CELT_SET_COMPLEXITY_REQUEST, CELT_SET_PREDICTION_REQUEST, CELT_SET_VBR_RATE_REQUEST, CELT_UNIMPLEMENTED, check_mode(), MAX_PERIOD, and CELTMode::nbEBands.

Referenced by main().

6.7.2.7 int check_decoder (const CELTDecoder * st)

Definition at line 1106 of file celt.c.

References CELT_INVALID_STATE, CELT_OK, DECODERFREED, DECODERVALID, and CELTDecoder::marker.

Referenced by celt_decode(), celt_decode_float(), and celt_decoder_ctl().

6.8 libcelt/celt.h File Reference

Contains all the functions for encoding and decoding audio. `#include "celt_types.h"`

Defines

- `#define EXPORT`
- `#define _celt_check_int(x) (((void)((x) == (celt_int32)0)), (celt_int32)(x))`
- `#define _celt_check_mode_ptr_ptr(ptr) ((ptr) + ((ptr) - (CELTMode**)(ptr)))`
- `#define CELT_OK 0`
- `#define CELT_BAD_ARG -1`
- `#define CELT_INVALID_MODE -2`
- `#define CELT_INTERNAL_ERROR -3`
- `#define CELT_CORRUPTED_DATA -4`
- `#define CELT_UNIMPLEMENTED -5`
- `#define CELT_INVALID_STATE -6`
- `#define CELT_ALLOC_FAIL -7`
- `#define CELT_GET_MODE_REQUEST 1`
- `#define CELT_GET_MODE(x) CELT_GET_MODE_REQUEST, _celt_check_mode_ptr_ptr(x)`
- `#define CELT_SET_COMPLEXITY_REQUEST 2`
- `#define CELT_SET_COMPLEXITY(x) CELT_SET_COMPLEXITY_REQUEST, _celt_check_int(x)`
- `#define CELT_SET_PREDICTION_REQUEST 4`
- `#define CELT_SET_PREDICTION(x) CELT_SET_PREDICTION_REQUEST, _celt_check_int(x)`
- `#define CELT_SET_VBR_RATE_REQUEST 6`
- `#define CELT_SET_VBR_RATE(x) CELT_SET_VBR_RATE_REQUEST, _celt_check_int(x)`
- `#define CELT_RESET_STATE_REQUEST 8`
- `#define CELT_RESET_STATE CELT_RESET_STATE_REQUEST`
- `#define CELT_GET_FRAME_SIZE 1000`
- `#define CELT_GET_LOOKAHEAD 1001`
- `#define CELT_GET_SAMPLE_RATE 1003`
- `#define CELT_GET_BITSTREAM_VERSION 2000`

Typedefs

- typedef struct CELTEncoder CELTEncoder
Encoder state.
- typedef struct CELTDecoder CELTDecoder
- typedef struct CELTMode CELTMode

Functions

- EXPORT CELTMode * `celt_mode_create` (celt_int32 Fs, int frame_size, int *error)
- EXPORT void `celt_mode_destroy` (CELTMode *mode)
- EXPORT int `celt_mode_info` (const CELTMode *mode, int request, celt_int32 *value)
- EXPORT CELTEncoder * `celt_encoder_create` (const CELTMode *mode, int channels, int *error)
- EXPORT void `celt_encoder_destroy` (CELTEncoder *st)

- EXPORT int `celt_encode_float` (`CELTEncoder` *st, const float *pcm, float *optional_synthesis, unsigned char *compressed, int nbCompressedBytes)
- EXPORT int `celt_encode` (`CELTEncoder` *st, const `celt_int16` *pcm, `celt_int16` *optional_synthesis, unsigned char *compressed, int nbCompressedBytes)
- EXPORT int `celt_encoder_ctl` (`CELTEncoder` *st, int request,...)
- EXPORT `CELTDDecoder` * `celt_decoder_create` (const `CELTMode` *mode, int channels, int *error)
- EXPORT void `celt_decoder_destroy` (`CELTDDecoder` *st)
- EXPORT int `celt_decode_float` (`CELTDDecoder` *st, const unsigned char *data, int len, float *pcm)
- EXPORT int `celt_decode` (`CELTDDecoder` *st, const unsigned char *data, int len, `celt_int16` *pcm)
- EXPORT int `celt_decoder_ctl` (`CELTDDecoder` *st, int request,...)
- EXPORT const char * `celt_strerror` (int error)

6.8.1 Detailed Description

Contains all the functions for encoding and decoding audio.

Definition in file [celt.h](#).

6.8.2 Define Documentation

6.8.2.1 #define `_celt_check_int(x)` (((void)((x) == (`celt_int32`)0)), (`celt_int32`)(x))

Definition at line 56 of file [celt.h](#).

6.8.2.2 #define `_celt_check_mode_ptr_ptr(ptr)` ((ptr) + ((ptr) - (`CELTMode**`)(ptr)))

Definition at line 57 of file [celt.h](#).

6.8.2.3 #define `CELT_ALLOC_FAIL` -7

Memory allocation has failed

Definition at line 75 of file [celt.h](#).

Referenced by `celt_decoder_create()`, and `celt_encoder_create()`.

6.8.2.4 #define `CELT_BAD_ARG` -1

An (or more) invalid argument (e.g. out of range)

Definition at line 63 of file [celt.h](#).

Referenced by `celt_decode()`, `celt_decode_float()`, `celt_decoder_create()`, `celt_decoder_ctl()`, `celt_encode()`, `celt_encode_float()`, `celt_encoder_create()`, `celt_encoder_ctl()`, `celt_header_from_packet()`, `celt_header_init()`, `celt_header_to_packet()`, and `celt_mode_create()`.

6.8.2.5 #define `CELT_CORRUPTED_DATA` -4

The data passed (e.g. compressed data to decoder) is corrupted

Definition at line 69 of file [celt.h](#).

6.8.2.6 #define CELT_GET_BITSTREAM_VERSION 2000

GET the bit-stream version for compatibility check

Definition at line 106 of file celt.h.

Referenced by celt_header_init(), and celt_mode_info().

6.8.2.7 #define CELT_GET_FRAME_SIZE 1000

GET the frame size used in the current mode

Definition at line 99 of file celt.h.

Referenced by celt_mode_info(), and main().

6.8.2.8 #define CELT_GET_LOOKAHEAD 1001

GET the lookahead used in the current mode

Definition at line 101 of file celt.h.

Referenced by celt_mode_info(), and main().

**6.8.2.9 #define CELT_GET_MODE(x) CELT_GET_MODE_REQUEST,
_celt_check_mode_ptr(x)**

Get the [CELTMode](#) used by an encoder or decoder

Definition at line 80 of file celt.h.

6.8.2.10 #define CELT_GET_MODE_REQUEST 1

Definition at line 78 of file celt.h.

Referenced by celt_decoder_ctl(), and celt_encoder_ctl().

6.8.2.11 #define CELT_GET_SAMPLE_RATE 1003

GET the sample rate used in the current mode

Definition at line 103 of file celt.h.

Referenced by celt_mode_info().

6.8.2.12 #define CELT_INTERNAL_ERROR -3

An internal error was detected

Definition at line 67 of file celt.h.

6.8.2.13 #define CELT_INVALID_MODE -2

The mode struct passed is invalid

Definition at line 65 of file celt.h.

Referenced by `celt_decode()`, `celt_decode_float()`, `celt_decoder_create()`, `celt_decoder_ctl()`, `celt_encode()`, `celt_encode_float()`, `celt_encoder_create()`, `celt_encoder_ctl()`, `celt_header_init()`, `celt_mode_create()`, `celt_mode_info()`, and `check_mode()`.

6.8.2.14 #define CELT_INVALID_STATE -6

An encoder or decoder structure is invalid or already freed

Definition at line 73 of file celt.h.

Referenced by `celt_decode()`, `celt_decode_float()`, `celt_decoder_ctl()`, `celt_encode()`, `celt_encode_float()`, `celt_encoder_ctl()`, and `check_decoder()`.

6.8.2.15 #define CELT_OK 0

No error

Definition at line 61 of file celt.h.

Referenced by `celt_decode()`, `celt_decode_float()`, `celt_decoder_create()`, `celt_decoder_ctl()`, `celt_encode()`, `celt_encode_float()`, `celt_encoder_create()`, `celt_encoder_ctl()`, `celt_header_init()`, `celt_mode_create()`, `celt_mode_info()`, `check_decoder()`, and `check_mode()`.

6.8.2.16 #define CELT_RESET_STATE CELT_RESET_STATE_REQUEST

Definition at line 96 of file celt.h.

Referenced by `celt_decoder_ctl()`, and `celt_encoder_ctl()`.

6.8.2.17 #define CELT_RESET_STATE_REQUEST 8

Reset the encoder/decoder memories to zero

Definition at line 95 of file celt.h.

6.8.2.18 #define CELT_SET_COMPLEXITY(x) CELT_SET_COMPLEXITY_REQUEST, _celt_check_int(x)

Controls the complexity from 0-10 (int)

Definition at line 83 of file celt.h.

Referenced by `main()`.

6.8.2.19 #define CELT_SET_COMPLEXITY_REQUEST 2

Definition at line 81 of file celt.h.

Referenced by `celt_encoder_ctl()`.

6.8.2.20 #define CELT_SET_PREDICTION(x) CELT_SET_PREDICTION_REQUEST, _celt_check_int(x)

Controls the use of interframe prediction. 0=Independent frames 1=Short term interframe prediction allowed 2=Long term prediction allowed

Definition at line 90 of file celt.h.

6.8.2.21 #define CELT_SET_PREDICTION_REQUEST 4

Definition at line 84 of file celt.h.

Referenced by celt_encoder_ctl().

6.8.2.22 #define CELT_SET_VBR_RATE(x) CELT_SET_VBR_RATE_REQUEST, _celt_check_int(x)

Set the target VBR rate in bits per second(int); 0=CBR (default)

Definition at line 93 of file celt.h.

6.8.2.23 #define CELT_SET_VBR_RATE_REQUEST 6

Definition at line 91 of file celt.h.

Referenced by celt_encoder_ctl().

6.8.2.24 #define CELT_UNIMPLEMENTED -5

Invalid/unsupported request number

Definition at line 71 of file celt.h.

Referenced by celt_decoder_ctl(), celt_encoder_ctl(), and celt_mode_info().

6.8.2.25 #define EXPORT

Definition at line 53 of file celt.h.

6.8.3 Typedef Documentation**6.8.3.1 typedef struct CELTDecoder CELTDecoder**

State of the decoder. One decoder state is needed for each stream. It is initialised once at the beginning of the stream. Do *not* re-initialise the state for every frame

Definition at line 119 of file celt.h.

6.8.3.2 typedef struct CELTEncoder CELTEncoder

Encoder state. Contains the state of an encoder. One encoder state is needed for each stream. It is initialised once at the beginning of the stream. Do *not* re-initialise the state for every frame.

Definition at line 114 of file celt.h.

6.8.3.3 typedef struct CELTMode CELTMode

The mode contains all the information necessary to create an encoder. Both the encoder and decoder need to be initialised with exactly the same mode, otherwise the quality will be very bad

Definition at line 125 of file celt.h.

6.9 libcelt/celt_header.h File Reference

```
#include "celt.h"
#include "celt_types.h"
```

Data Structures

- struct [CELTHdr](#)
Header data.

Functions

- EXPORT int [celt_header_init](#) ([CELTHdr](#) *header, const [CELTMode](#) *m, int channels)
- EXPORT int [celt_header_to_packet](#) (const [CELTHdr](#) *header, unsigned char *packet, [celt_uint32](#) size)
- EXPORT int [celt_header_from_packet](#) (const unsigned char *packet, [celt_uint32](#) size, [CELTHdr](#) *header)

6.9.1 Function Documentation

6.9.1.1 EXPORT int [celt_header_from_packet](#) (const unsigned char * *packet*, [celt_uint32](#) *size*, [CELTHdr](#) * *header*)

Definition at line 103 of file header.c.

References [CELT_BAD_ARG](#), [CELT_COPY](#), and [CELT_MEMSET](#).

6.9.1.2 EXPORT int [celt_header_init](#) ([CELTHdr](#) * *header*, const [CELTMode](#) * *m*, int *channels*)

Creates a basic header struct

Definition at line 54 of file header.c.

References [CELTHdr::bytes_per_packet](#), [CELT_BAD_ARG](#), [CELT_COPY](#), [CELT_GET_BITSTREAM_VERSION](#), [CELT_INVALID_MODE](#), [celt_mode_info\(\)](#), [CELT_OK](#), [check_mode\(\)](#), [CELTHdr::codec_id](#), [CELTHdr::codec_version](#), [CELTHdr::extra_headers](#), [CELTHdr::frame_size](#), [CELTMode::Fs](#), [CELTHdr::header_size](#), [CELTMode::mdctSize](#), [CELTHdr::nb_channels](#), [CELTMode::overlap](#), [CELTHdr::overlap](#), [CELTHdr::sample_rate](#), and [CELTHdr::version_id](#).

6.9.1.3 EXPORT int [celt_header_to_packet](#) (const [CELTHdr](#) * *header*, unsigned char * *packet*, [celt_uint32](#) *size*)

Definition at line 76 of file header.c.

References [CELT_BAD_ARG](#), [CELT_COPY](#), and [CELT_MEMSET](#).

6.10 libcelt/celt_types.h File Reference

CELT types.

Typedefs

- typedef short [celt_int16](#)
- typedef unsigned short [celt_uint16](#)
- typedef int [celt_int32](#)
- typedef unsigned int [celt_uint32](#)

6.10.1 Detailed Description

CELT types.

Definition in file [celt_types.h](#).

6.10.2 Typedef Documentation

6.10.2.1 typedef short celt_int16

Definition at line 133 of file [celt_types.h](#).

6.10.2.2 typedef int celt_int32

Definition at line 135 of file [celt_types.h](#).

6.10.2.3 typedef unsigned short celt_uint16

Definition at line 134 of file [celt_types.h](#).

6.10.2.4 typedef unsigned int celt_uint32

Definition at line 136 of file [celt_types.h](#).

6.11 libcelt/cwrs.c File Reference

```
#include "os_support.h"
#include <stdlib.h>
#include <string.h>
#include "cwrs.h"
#include "mathops.h"
#include "arch.h"
```

Defines

- #define [MASK32](#) (0xFFFFFFFF)

Functions

- int [log2_frac](#) ([ec_uint32](#) val, int frac)
- int [fits_in32](#) (int _n, int _k)
- [celt_uint32](#) [icwrs](#) (int _n, int _k, [celt_uint32](#) *_nc, const int *_y, [celt_uint32](#) *_u)
- void [get_required_bits](#) ([celt_int16](#) *_bits, int _n, int _maxk, int _frac)
- void [encode_pulses](#) (int *_y, int N, int K, [ec_enc](#) *enc)
- void [decode_pulses](#) (int *_y, int N, int K, [ec_dec](#) *dec)

6.11.1 Define Documentation

6.11.1.1 #define MASK32 (0xFFFFFFFF)

Definition at line 79 of file cwrs.c.

6.11.2 Function Documentation

6.11.2.1 void [decode_pulses](#) (int *_y, int N, int K, [ec_dec](#) *dec)

Definition at line 891 of file cwrs.c.

References [decode_pulses\(\)](#), [ec_dec_uint\(\)](#), and [fits_in32\(\)](#).

Referenced by [alg_unquant\(\)](#), and [decode_pulses\(\)](#).

6.11.2.2 void [encode_pulses](#) (int *_y, int N, int K, [ec_enc](#) *enc)

Definition at line 849 of file cwrs.c.

References [ec_enc_uint\(\)](#), [encode_pulses\(\)](#), and [fits_in32\(\)](#).

Referenced by [alg_quant\(\)](#), and [encode_pulses\(\)](#).

6.11.2.3 int fits_in32 (int *_n*, int *_k*)

Definition at line 301 of file cwrs.c.

Referenced by decode_pulses(), encode_pulses(), and get_required_bits().

6.11.2.4 void get_required_bits (celt_int16 * *_bits*, int *_n*, int *_maxk*, int *_frac*)

Definition at line 780 of file cwrs.c.

References ALLOC, celt_assert, fits_in32(), log2_frac(), RESTORE_STACK, SAVE_STACK, and VARDECL.

Referenced by compute_alloc_cache().

6.11.2.5 celt_uint32 icwrs (int *_n*, int *_k*, celt_uint32 * *_nc*, const int * *_y*, celt_uint32 * *_u*)

Definition at line 671 of file cwrs.c.

References celt_assert.

6.11.2.6 int log2_frac (ec_uint32 *val*, int *frac*)

Definition at line 49 of file cwrs.c.

References EC_ILOG.

Referenced by get_required_bits(), quant_bands_stereo(), and unquant_bands_stereo().

6.12 libcelt/cwrs.h File Reference

```
#include "arch.h"
#include "stack_alloc.h"
#include "entenc.h"
#include "entdec.h"
```

Functions

- int [log2_frac](#) ([ec_uint32](#) val, int frac)
- int [fits_in32](#) (int _n, int _m)
- void [get_required_bits](#) ([celt_int16](#) *bits, int N, int K, int frac)
- void [encode_pulses](#) (int *_y, int N, int K, [ec_enc](#) *enc)
- void [decode_pulses](#) (int *_y, int N, int K, [ec_dec](#) *dec)

6.12.1 Function Documentation

6.12.1.1 void [decode_pulses](#) (int *_y, int N, int K, [ec_dec](#) *dec)

Definition at line 891 of file cwrs.c.

References [decode_pulses\(\)](#), [ec_dec_uint\(\)](#), and [fits_in32\(\)](#).

Referenced by [alg_unquant\(\)](#), and [decode_pulses\(\)](#).

6.12.1.2 void [encode_pulses](#) (int *_y, int N, int K, [ec_enc](#) *enc)

Definition at line 849 of file cwrs.c.

References [ec_enc_uint\(\)](#), [encode_pulses\(\)](#), and [fits_in32\(\)](#).

Referenced by [alg_quant\(\)](#), and [encode_pulses\(\)](#).

6.12.1.3 int [fits_in32](#) (int _n, int _m)

Definition at line 301 of file cwrs.c.

Referenced by [decode_pulses\(\)](#), [encode_pulses\(\)](#), and [get_required_bits\(\)](#).

6.12.1.4 void [get_required_bits](#) ([celt_int16](#) *bits, int N, int K, int frac)

Definition at line 780 of file cwrs.c.

References [ALLOC](#), [celt_assert](#), [fits_in32\(\)](#), [log2_frac\(\)](#), [RESTORE_STACK](#), [SAVE_STACK](#), and [VARDECL](#).

Referenced by [compute_alloc_cache\(\)](#).

6.12.1.5 int [log2_frac](#) ([ec_uint32](#) val, int frac)

Definition at line 49 of file cwrs.c.

References EC_ILOG.

Referenced by `get_required_bits()`, `quant_bands_stereo()`, and `unquant_bands_stereo()`.

6.13 libcelt/dump_modes.c File Reference

```
#include <stdio.h>
#include "modes.h"
#include "celt.h"
#include "rate.h"
```

Defines

- #define [INT16](#) "%d"
- #define [INT32](#) "%d"
- #define [FLOAT](#) "%f"
- #define [WORD16](#) FLOAT
- #define [WORD32](#) FLOAT

Functions

- void [dump_modes](#) (FILE *file, [CELTMode](#) **modes, int nb_modes)
- void [dump_header](#) (FILE *file, [CELTMode](#) **modes, int nb_modes)
- int [main](#) (int argc, char **argv)

6.13.1 Define Documentation

6.13.1.1 #define [FLOAT](#) "%f"

Definition at line 44 of file dump_modes.c.

6.13.1.2 #define [INT16](#) "%d"

Definition at line 42 of file dump_modes.c.

6.13.1.3 #define [INT32](#) "%d"

Definition at line 43 of file dump_modes.c.

Referenced by [dump_modes\(\)](#).

6.13.1.4 #define [WORD16](#) FLOAT

Definition at line 50 of file dump_modes.c.

Referenced by [dump_modes\(\)](#).

6.13.1.5 #define [WORD32](#) FLOAT

Definition at line 51 of file dump_modes.c.

6.13.2 Function Documentation

6.13.2.1 void dump_header (FILE *file, CELTMode **modes, int nb_modes)

Definition at line 171 of file dump_modes.c.

References CELTMode::mdctSize, and CELTMode::overlap.

Referenced by main().

6.13.2.2 void dump_modes (FILE *file, CELTMode **modes, int nb_modes)

Definition at line 55 of file dump_modes.c.

References CELTMode::allocVectors, CELTMode::bits, PsyDecay::decayR, CELTMode::eBands, CELTMode::ePredCoef, CELTMode::Fs, INT32, MAX_PERIOD, MAX_PULSES, CELTMode::mdctSize, CELTMode::nbAllocVectors, CELTMode::nbEBands, CELTMode::nbShortMdcts, CELTMode::overlap, CELTMode::pitchEnd, CELTMode::psy, CELTMode::shortMdctSize, CELTMode::window, and WORD16.

Referenced by main().

6.13.2.3 int main (int argc, char **argv)

Definition at line 206 of file dump_modes.c.

References celt_mode_create(), celt_mode_destroy(), dump_header(), and dump_modes().

6.14 libcelt/ecintrin.h File Reference

```
#include <math.h>
#include <limits.h>
```

Defines

- `#define _ecintrin_H (1)`
- `#define EC_MAXI(_a, _b) ((_a)-((_a)-(_b))&-((_b)>(_a)))`
- `#define EC_MINI(_a, _b) ((_a)+((_b)-(_a))&-((_b)<(_a)))`
- `#define EC_SIGNI(_a) (((_a)>0)-((_a)<0))`
- `#define EC_SIGNMASK(_a) (-((_a)<0))`
- `#define EC_CLAMPI(_a, _b, _c) (EC_MAXI(_a, EC_MINI(_b, _c)))`
- `#define EC_ILOG(_x) (ec_ilog(_x))`
- `#define EC_ILOG64(_x) (ec_ilog64(_x))`

6.14.1 Define Documentation

6.14.1.1 `#define _ecintrin_H (1)`

Definition at line 36 of file `ecintrin.h`.

6.14.1.2 `#define EC_CLAMPI(_a, _b, _c) (EC_MAXI(_a, EC_MINI(_b, _c)))`

Definition at line 80 of file `ecintrin.h`.

6.14.1.3 `#define EC_ILOG(_x) (ec_ilog(_x))`

Definition at line 108 of file `ecintrin.h`.

Referenced by `ec_dec_tell()`, `ec_dec_uint()`, `ec_decode()`, `ec_enc_done()`, `ec_enc_tell()`, `ec_enc_uint()`, `ec_encode()`, and `log2_frac()`.

6.14.1.4 `#define EC_ILOG64(_x) (ec_ilog64(_x))`

Definition at line 133 of file `ecintrin.h`.

6.14.1.5 `#define EC_MAXI(_a, _b) ((_a)-((_a)-(_b))&-((_b)>(_a)))`

Definition at line 63 of file `ecintrin.h`.

Referenced by `ec_decode()`.

6.14.1.6 `#define EC_MINI(_a, _b) ((_a)+((_b)-(_a))&-((_b)<(_a)))`

Definition at line 64 of file `ecintrin.h`.

Referenced by `ec_dec_update()`, `ec_decode()`, `ec_decode_bin()`, and `ec_encode()`.

6.14.1.7 #define EC_SIGNI(_a) (((_a)>0)-((_a)<0))

Definition at line 69 of file ecintrin.h.

6.14.1.8 #define EC_SIGNMASK(_a) (-((_a)<0))

Definition at line 73 of file ecintrin.h.

6.15 libcelt/entcode.c File Reference

```
#include "entcode.h"
```

Functions

- [int ec_ilog \(ec_uint32 _v\)](#)

6.15.1 Function Documentation

6.15.1.1 [int ec_ilog \(ec_uint32 _v\)](#)

Definition at line 44 of file entcode.c.

6.16 libcelt/entcode.h File Reference

```
#include "celt_types.h"
#include <limits.h>
#include "ecintrin.h"
```

Data Structures

- struct [ec_byte_buffer](#)

Defines

- #define [_entcode_H](#) (1)
- #define [EC_UNIT_BITS](#) (8)
- #define [EC_UNIT_MASK](#) ((1U<<EC_UNIT_BITS)-1)

Typedefs

- typedef [celt_int32](#) [ec_int32](#)
- typedef [celt_uint32](#) [ec_uint32](#)
- typedef struct [ec_byte_buffer](#) [ec_byte_buffer](#)

Functions

- void [ec_byte_writeinit_buffer](#) ([ec_byte_buffer](#) *_b, unsigned char *_buf, long _size)
- void [ec_byte_shrink](#) ([ec_byte_buffer](#) *_b, long _size)
- void [ec_byte_writeinit](#) ([ec_byte_buffer](#) *_b)
- void [ec_byte_writetrunc](#) ([ec_byte_buffer](#) *_b, long _bytes)
- void [ec_byte_write1](#) ([ec_byte_buffer](#) *_b, unsigned _value)
- void [ec_byte_write_at_end](#) ([ec_byte_buffer](#) *_b, unsigned _value)
- void [ec_byte_write4](#) ([ec_byte_buffer](#) *_b, [ec_uint32](#) _value)
- void [ec_byte_writecopy](#) ([ec_byte_buffer](#) *_b, void *_source, long _bytes)
- void [ec_byte_writeclear](#) ([ec_byte_buffer](#) *_b)
- void [ec_byte_readinit](#) ([ec_byte_buffer](#) *_b, unsigned char *_buf, long _bytes)
- int [ec_byte_look1](#) ([ec_byte_buffer](#) *_b)
- unsigned char [ec_byte_look_at_end](#) ([ec_byte_buffer](#) *_b)
- int [ec_byte_look4](#) ([ec_byte_buffer](#) *_b, [ec_uint32](#) *_val)
- void [ec_byte_adv1](#) ([ec_byte_buffer](#) *_b)
- void [ec_byte_adv4](#) ([ec_byte_buffer](#) *_b)
- int [ec_byte_read1](#) ([ec_byte_buffer](#) *_b)
- int [ec_byte_read4](#) ([ec_byte_buffer](#) *_b, [ec_uint32](#) *_val)
- int [ec_ilog](#) ([ec_uint32](#) _v)

6.16.1 Define Documentation

6.16.1.1 #define [_entcode_H](#) (1)

Definition at line 35 of file entcode.h.

6.16.1.2 `#define EC_UNIT_BITS (8)`

Definition at line 48 of file `encode.h`.

Referenced by `ec_dec_bits()`, `ec_dec_uint()`, `ec_enc_bits()`, and `ec_enc_uint()`.

6.16.1.3 `#define EC_UNIT_MASK ((1U<<EC_UNIT_BITS)-1)`

Definition at line 50 of file `encode.h`.

Referenced by `ec_enc_bits()`.

6.16.2 Typedef Documentation

6.16.2.1 `typedef struct ec_byte_buffer ec_byte_buffer`

Definition at line 43 of file `encode.h`.

6.16.2.2 `typedef celt_int32 ec_int32`

Definition at line 41 of file `encode.h`.

6.16.2.3 `typedef celt_uint32 ec_uint32`

Definition at line 42 of file `encode.h`.

6.16.3 Function Documentation

6.16.3.1 `void ec_byte_adv1 (ec_byte_buffer * b)`

Definition at line 55 of file `entdec.c`.

References `ec_byte_buffer::ptr`.

6.16.3.2 `void ec_byte_adv4 (ec_byte_buffer * b)`

6.16.3.3 `int ec_byte_look1 (ec_byte_buffer * b)`

6.16.3.4 `int ec_byte_look4 (ec_byte_buffer * b, ec_uint32 * val)`

6.16.3.5 `unsigned char ec_byte_look_at_end (ec_byte_buffer * b)`

Definition at line 47 of file `entdec.c`.

References `ec_byte_buffer::buf`, `celt_fatal`, and `ec_byte_buffer::end_ptr`.

Referenced by `ec_decode_bin()`, and `ec_decode_raw()`.

6.16.3.6 `int ec_byte_read1 (ec_byte_buffer * b)`

Definition at line 59 of file `entdec.c`.

References `ec_byte_buffer::buf`, `ec_byte_buffer::ptr`, and `ec_byte_buffer::storage`.

6.16.3.7 `int ec_byte_read4 (ec_byte_buffer * b, ec_uint32 * val)`

6.16.3.8 `void ec_byte_readinit (ec_byte_buffer * b, unsigned char * buf, long bytes)`

Definition at line 41 of file `entdec.c`.

References `ec_byte_buffer::buf`, `ec_byte_buffer::end_ptr`, `ec_byte_buffer::ptr`, and `ec_byte_buffer::storage`.

Referenced by `celt_decode_float()`.

6.16.3.9 `void ec_byte_shrink (ec_byte_buffer * b, long size)`

Definition at line 49 of file `entenc.c`.

References `ec_byte_buffer::buf`, `ec_byte_buffer::end_ptr`, and `ec_byte_buffer::storage`.

Referenced by `celt_encode_float()`.

6.16.3.10 `void ec_byte_write1 (ec_byte_buffer * b, unsigned value)`

Definition at line 54 of file `entenc.c`.

References `ec_byte_buffer::buf`, `celt_fatal`, `ec_byte_buffer::ptr`, and `ec_byte_buffer::storage`.

6.16.3.11 `void ec_byte_write4 (ec_byte_buffer * b, ec_uint32 value)`

6.16.3.12 `void ec_byte_write_at_end (ec_byte_buffer * b, unsigned value)`

Definition at line 63 of file `entenc.c`.

References `celt_fatal`, `ec_byte_buffer::end_ptr`, and `ec_byte_buffer::ptr`.

Referenced by `ec_encode_bin()`, and `ec_encode_raw()`.

6.16.3.13 `void ec_byte_writeclear (ec_byte_buffer * b)`

6.16.3.14 `void ec_byte_writecopy (ec_byte_buffer * b, void * source, long bytes)`

6.16.3.15 `void ec_byte_writeinit (ec_byte_buffer * b)`

6.16.3.16 `void ec_byte_writeinit_buffer (ec_byte_buffer * b, unsigned char * buf, long size)`

Definition at line 43 of file `entenc.c`.

References `ec_byte_buffer::buf`, `ec_byte_buffer::end_ptr`, `ec_byte_buffer::ptr`, and `ec_byte_buffer::storage`.

Referenced by `celt_encode_float()`.

6.16.3.17 void `ec_byte_writetrunc` (`ec_byte_buffer` * *b*, long *bytes*)

6.16.3.18 int `ec_ilog` (`ec_uint32` *v*)

Definition at line 44 of file `entcode.c`.

6.17 libcelt/entdec.c File Reference

```
#include <stddef.h>
#include "entdec.h"
#include "os_support.h"
#include "arch.h"
```

Functions

- void `ec_byte_readinit` (`ec_byte_buffer * _b`, unsigned char * `_buf`, long `_bytes`)
- unsigned char `ec_byte_look_at_end` (`ec_byte_buffer * _b`)
- void `ec_byte_adv1` (`ec_byte_buffer * _b`)
- int `ec_byte_read1` (`ec_byte_buffer * _b`)
- `ec_uint32 ec_dec_bits` (`ec_dec * _this`, int `_ftb`)
- `ec_uint32 ec_dec_uint` (`ec_dec * _this`, `ec_uint32 _ft`)

6.17.1 Function Documentation

6.17.1.1 void `ec_byte_adv1` (`ec_byte_buffer * _b`)

Definition at line 55 of file `entdec.c`.

References `ec_byte_buffer::ptr`.

6.17.1.2 unsigned char `ec_byte_look_at_end` (`ec_byte_buffer * _b`)

Definition at line 47 of file `entdec.c`.

References `ec_byte_buffer::buf`, `celt_fatal`, and `ec_byte_buffer::end_ptr`.

Referenced by `ec_decode_bin()`, and `ec_decode_raw()`.

6.17.1.3 int `ec_byte_read1` (`ec_byte_buffer * _b`)

Definition at line 59 of file `entdec.c`.

References `ec_byte_buffer::buf`, `ec_byte_buffer::ptr`, and `ec_byte_buffer::storage`.

6.17.1.4 void `ec_byte_readinit` (`ec_byte_buffer * _b`, unsigned char * `_buf`, long `_bytes`)

Definition at line 41 of file `entdec.c`.

References `ec_byte_buffer::buf`, `ec_byte_buffer::end_ptr`, `ec_byte_buffer::ptr`, and `ec_byte_buffer::storage`.

Referenced by `celt_decode_float()`.

6.17.1.5 `ec_uint32 ec_dec_bits` (`ec_dec * _this`, int `_ftb`)

Definition at line 67 of file `entdec.c`.

References `ec_decode_raw()`, and `EC_UNIT_BITS`.

Referenced by `ec_dec_uint()`, `unquant_bands_stereo()`, `unquant_energy_finalise()`, and `unquant_fine_energy()`.

6.17.1.6 `ec_uint32 ec_dec_uint (ec_dec * this, ec_uint32 ft)`

Definition at line 83 of file `entdec.c`.

References `celt_assert`, `ec_dec_bits()`, `ec_dec_update()`, `ec_decode()`, `EC_ILOG`, and `EC_UNIT_BITS`.

Referenced by `celt_decode_float()`, `decode_pulses()`, and `unquant_bands_stereo()`.

6.18 libcelt/entdec.h File Reference

```
#include "entcode.h"
```

Data Structures

- struct [ec_dec](#)

Defines

- #define [_entdec_H](#) (1)

Typedefs

- typedef struct [ec_dec](#) [ec_dec](#)

Functions

- void [ec_dec_init](#) ([ec_dec](#) *_this, [ec_byte_buffer](#) *_buf)
- unsigned [ec_decode](#) ([ec_dec](#) *_this, unsigned _ft)
- unsigned [ec_decode_bin](#) ([ec_dec](#) *_this, unsigned _bits)
- unsigned [ec_decode_raw](#) ([ec_dec](#) *_this, unsigned bits)
- void [ec_dec_update](#) ([ec_dec](#) *_this, unsigned _fl, unsigned _fh, unsigned _ft)
- [ec_uint32](#) [ec_dec_bits](#) ([ec_dec](#) *_this, int _ftb)
- [ec_uint32](#) [ec_dec_uint](#) ([ec_dec](#) *_this, [ec_uint32](#) _ft)
- long [ec_dec_tell](#) ([ec_dec](#) *_this, int _b)

6.18.1 Define Documentation

6.18.1.1 #define [_entdec_H](#) (1)

Definition at line 33 of file entdec.h.

6.18.2 Typedef Documentation

6.18.2.1 typedef struct [ec_dec](#) [ec_dec](#)

Definition at line 38 of file entdec.h.

6.18.3 Function Documentation

6.18.3.1 [ec_uint32](#) [ec_dec_bits](#) ([ec_dec](#) * *_this*, int *_ftb*)

Definition at line 67 of file entdec.c.

References [ec_decode_raw\(\)](#), and [EC_UNIT_BITS](#).

Referenced by [ec_dec_uint\(\)](#), [unquant_bands_stereo\(\)](#), [unquant_energy_finalise\(\)](#), and [unquant_fine_energy\(\)](#).

6.18.3.2 void ec_dec_init (ec_dec * *this*, ec_byte_buffer * *buf*)

Definition at line 155 of file mfrngdec.c.

References ec_dec::buf, ec_dec::dif, EC_CODE_EXTRA, EC_SYM_BITS, ec_dec::end_bits_left, ec_dec::nb_end_bits, ec_dec::rem, and ec_dec::rng.

Referenced by celt_decode_float().

6.18.3.3 long ec_dec_tell (ec_dec * *this*, int *b*)

Definition at line 227 of file mfrngdec.c.

References ec_dec::buf, EC_CODE_BITS, EC_ILOG, EC_SYM_BITS, ec_dec::nb_end_bits, and ec_dec::rng.

Referenced by celt_decode_float(), quant_bands(), unquant_bands_stereo(), and unquant_coarse_energy().

6.18.3.4 ec_uint32 ec_dec_uint (ec_dec * *this*, ec_uint32 *ft*)

Definition at line 83 of file entdec.c.

References celt_assert, ec_dec_bits(), ec_dec_update(), ec_decode(), EC_ILOG, and EC_UNIT_BITS.

Referenced by celt_decode_float(), decode_pulses(), and unquant_bands_stereo().

6.18.3.5 void ec_dec_update (ec_dec * *this*, unsigned *fl*, unsigned *fh*, unsigned *ft*)

Definition at line 206 of file mfrngdec.c.

References ec_dec::dif, EC_MINI, IMUL32, ec_dec::nrm, and ec_dec::rng.

Referenced by ec_dec_uint(), and ec_laplace_decode_start().

6.18.3.6 unsigned ec_decode (ec_dec * *this*, unsigned *ft*)

Definition at line 167 of file mfrngdec.c.

References ec_dec::dif, EC_ILOG, EC_MAXI, EC_MINI, ec_dec::nrm, and ec_dec::rng.

Referenced by ec_dec_uint(), and ec_decode_bin().

6.18.3.7 unsigned ec_decode_bin (ec_dec * *this*, unsigned *bits*)

Definition at line 185 of file mfrngdec.c.

References ec_dec::buf, ec_dec::dif, ec_byte_look_at_end(), ec_decode(), EC_MINI, ec_dec::end_bits_left, ec_dec::end_byte, ec_dec::nb_end_bits, ec_dec::nrm, and ec_dec::rng.

Referenced by ec_laplace_decode_start().

6.18.3.8 unsigned ec_decode_raw (ec_dec * *this*, unsigned *bits*)

Definition at line 164 of file rangedec.c.

References `ec_dec::buf`, `ec_byte_look_at_end()`, `ec_dec::end_bits_left`, `ec_dec::end_byte`, and `ec_dec::nb_end_bits`.

Referenced by `ec_dec_bits()`.

6.19 libcelt/entenc.c File Reference

```
#include "os_support.h"
#include "entenc.h"
#include "arch.h"
```

Defines

- #define [EC_BUFFER_INCREMENT](#) (256)

Functions

- void [ec_byte_writeinit_buffer](#) ([ec_byte_buffer](#) *_b, unsigned char *_buf, long _size)
- void [ec_byte_shrink](#) ([ec_byte_buffer](#) *_b, long _size)
- void [ec_byte_write1](#) ([ec_byte_buffer](#) *_b, unsigned _value)
- void [ec_byte_write_at_end](#) ([ec_byte_buffer](#) *_b, unsigned _value)
- void [ec_enc_bits](#) ([ec_enc](#) *_this, [ec_uint32](#) _fl, int _ftb)
- void [ec_enc_uint](#) ([ec_enc](#) *_this, [ec_uint32](#) _fl, [ec_uint32](#) _ft)

6.19.1 Define Documentation

6.19.1.1 #define [EC_BUFFER_INCREMENT](#) (256)

Definition at line 41 of file entenc.c.

6.19.2 Function Documentation

6.19.2.1 void [ec_byte_shrink](#) ([ec_byte_buffer](#) *_b, long _size)

Definition at line 49 of file entenc.c.

References [ec_byte_buffer::buf](#), [ec_byte_buffer::end_ptr](#), and [ec_byte_buffer::storage](#).

Referenced by [celt_encode_float\(\)](#).

6.19.2.2 void [ec_byte_write1](#) ([ec_byte_buffer](#) *_b, unsigned _value)

Definition at line 54 of file entenc.c.

References [ec_byte_buffer::buf](#), [celt_fatal](#), [ec_byte_buffer::ptr](#), and [ec_byte_buffer::storage](#).

6.19.2.3 void [ec_byte_write_at_end](#) ([ec_byte_buffer](#) *_b, unsigned _value)

Definition at line 63 of file entenc.c.

References [celt_fatal](#), [ec_byte_buffer::end_ptr](#), and [ec_byte_buffer::ptr](#).

Referenced by [ec_encode_bin\(\)](#), and [ec_encode_raw\(\)](#).

6.19.2.4 void ec_byte_writeinit_buffer (ec_byte_buffer * *_b*, unsigned char * *_buf*, long *_size*)

Definition at line 43 of file entenc.c.

References ec_byte_buffer::buf, ec_byte_buffer::end_ptr, ec_byte_buffer::ptr, and ec_byte_buffer::storage.

Referenced by celt_encode_float().

6.19.2.5 void ec_enc_bits (ec_enc * *_this*, ec_uint32 *_fl*, int *_fib*)

Definition at line 72 of file entenc.c.

References ec_encode_raw(), EC_UNIT_BITS, and EC_UNIT_MASK.

Referenced by ec_enc_uint(), quant_bands_stereo(), quant_energy_finalise(), and quant_fine_energy().

6.19.2.6 void ec_enc_uint (ec_enc * *_this*, ec_uint32 *_fl*, ec_uint32 *_ft*)

Definition at line 85 of file entenc.c.

References celt_assert, ec_enc_bits(), ec_encode(), EC_ILOG, and EC_UNIT_BITS.

Referenced by celt_encode_float(), encode_pulses(), and quant_bands_stereo().

6.20 libcelt/entenc.h File Reference

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

Data Structures

- struct [ec_enc](#)

Defines

- #define [_entenc_H](#) (1)

Typedefs

- typedef struct [ec_enc](#) [ec_enc](#)

Functions

- void [ec_enc_init](#) ([ec_enc](#) *_this, [ec_byte_buffer](#) *_buf)
- void [ec_encode](#) ([ec_enc](#) *_this, unsigned _fl, unsigned _fh, unsigned _ft)
- void [ec_encode_bin](#) ([ec_enc](#) *_this, unsigned _fl, unsigned _fh, unsigned _bits)
- void [ec_encode_raw](#) ([ec_enc](#) *_this, unsigned _fl, unsigned _fh, unsigned bits)
- void [ec_enc_bits](#) ([ec_enc](#) *_this, [ec_uint32](#) _fl, int _ftb)
- void [ec_enc_uint](#) ([ec_enc](#) *_this, [ec_uint32](#) _fl, [ec_uint32](#) _ft)
- long [ec_enc_tell](#) ([ec_enc](#) *_this, int _b)
- void [ec_enc_done](#) ([ec_enc](#) *_this)

6.20.1 Define Documentation

6.20.1.1 #define [_entenc_H](#) (1)

Definition at line 33 of file entenc.h.

6.20.2 Typedef Documentation

6.20.2.1 typedef struct [ec_enc](#) [ec_enc](#)

Definition at line 39 of file entenc.h.

6.20.3 Function Documentation

6.20.3.1 void [ec_enc_bits](#) ([ec_enc](#) *_this, [ec_uint32](#) _fl, int _ftb)

Definition at line 72 of file entenc.c.

References [ec_encode_raw\(\)](#), [EC_UNIT_BITS](#), and [EC_UNIT_MASK](#).

Referenced by [ec_enc_uint\(\)](#), [quant_bands_stereo\(\)](#), [quant_energy_finalise\(\)](#), and [quant_fine_energy\(\)](#).

6.20.3.2 void ec_enc_done (ec_enc * *this*)

Definition at line 205 of file mfrngenc.c.

References ec_enc::buf, EC_CODE_BITS, EC_CODE_SHIFT, EC_CODE_TOP, EC_ILOG, ec_enc::end_bits_left, ec_enc::end_byte, ec_byte_buffer::end_ptr, ec_enc::ext, ec_enc::low, ec_byte_buffer::ptr, ec_enc::rem, and ec_enc::rng.

Referenced by celt_encode_float().

6.20.3.3 void ec_enc_init (ec_enc * *this*, ec_byte_buffer * *buf*)

Definition at line 117 of file mfrngenc.c.

References ec_enc::buf, EC_CODE_TOP, ec_enc::end_bits_left, ec_enc::end_byte, ec_enc::ext, ec_enc::low, ec_enc::nb_end_bits, ec_enc::rem, and ec_enc::rng.

Referenced by celt_encode_float().

6.20.3.4 long ec_enc_tell (ec_enc * *this*, int *b*)

Definition at line 183 of file mfrngenc.c.

References ec_enc::buf, EC_CODE_BITS, EC_ILOG, ec_enc::ext, ec_enc::nb_end_bits, ec_enc::rem, and ec_enc::rng.

Referenced by celt_encode_float(), quant_bands(), quant_bands_stereo(), and quant_coarse_energy().

6.20.3.5 void ec_enc_uint (ec_enc * *this*, ec_uint32 *fl*, ec_uint32 *ft*)

Definition at line 85 of file entenc.c.

References celt_assert, ec_enc_bits(), ec_encode(), EC_ILOG, and EC_UNIT_BITS.

Referenced by celt_encode_float(), encode_pulses(), and quant_bands_stereo().

6.20.3.6 void ec_encode (ec_enc * *this*, unsigned *fl*, unsigned *fh*, unsigned *ft*)

Definition at line 128 of file mfrngenc.c.

References EC_ILOG, EC_MINI, IMUL32, ec_enc::low, and ec_enc::rng.

Referenced by ec_enc_uint(), and ec_encode_bin().

6.20.3.7 void ec_encode_bin (ec_enc * *this*, unsigned *fl*, unsigned *fh*, unsigned *bits*)

Definition at line 164 of file mfrngenc.c.

References ec_enc::buf, ec_byte_write_at_end(), ec_encode(), ec_enc::end_bits_left, ec_enc::end_byte, IMUL32, ec_enc::low, ec_enc::nb_end_bits, and ec_enc::rng.

Referenced by ec_laplace_encode_start().

6.20.3.8 void ec_encode_raw (ec_enc * *this*, unsigned *fl*, unsigned *fh*, unsigned *bits*)

Definition at line 141 of file rangeenc.c.

References `ec_enc::buf`, `ec_byte_write_at_end()`, `ec_enc::end_bits_left`, `ec_enc::end_byte`, and `ec_enc::nb_end_bits`.

Referenced by `ec_enc_bits()`.

6.21 libcelt/fixed_c5x.h File Reference

Fixed-point operations for the TI C5x DSP family. `#include "dsplib.h"`

Defines

- `#define MAX16(a, b) _max(a,b)`
- `#define MIN16(a, b) _min(a,b)`
- `#define MAX32(a, b) _lmax(a,b)`
- `#define MIN32(a, b) _lmin(a,b)`
- `#define VSHR32(a, shift) _lshl(a,-(shift))`
- `#define MULT16_16_Q15(a, b) (_smpy(a,b))`
- `#define MULT16_16SU(a, b) _lmpysu(a,b)`
- `#define MULT_16_16(a, b) _lmpy(a,b)`
- `#define MULT16_32_Q15(a, b) ADD32(SHL(MULT16_16((a),SHR((b),16)),1), SHR(MULT16_16SU((a),(b)),15))`
- `#define celt_ilog2(x) (30 - _lnorm(x))`
- `#define OVERRIDE_CELT_ILOG2`
- `#define celt_maxabs16(x, len) MAX16(maxval((DATA *)x, len),-minval((DATA *)x, len))`
- `#define OVERRIDE_CELT_MAXABS16`
- `#define OVERRIDE_FIND_MAX16`

6.21.1 Detailed Description

Fixed-point operations for the TI C5x DSP family.

Definition in file [fixed_c5x.h](#).

6.21.2 Define Documentation

6.21.2.1 `#define celt_ilog2(x) (30 - _lnorm(x))`

Definition at line 78 of file [fixed_c5x.h](#).

Referenced by [alg_quant\(\)](#), [compute_pitch_gain\(\)](#), and [find_spectral_pitch\(\)](#).

6.21.2.2 `#define celt_maxabs16(x, len) MAX16(maxval((DATA *)x, len),-minval((DATA *)x, len))`

Definition at line 81 of file [fixed_c5x.h](#).

6.21.2.3 `#define MAX16(a, b) _max(a,b)`

Definition at line 50 of file [fixed_c5x.h](#).

6.21.2.4 `#define MAX32(a, b) _lmax(a,b)`

Definition at line 56 of file [fixed_c5x.h](#).

6.21.2.5 #define MIN16(a, b) _min(a,b)

Definition at line 53 of file fixed_c5x.h.

6.21.2.6 #define MIN32(a, b) _lmin(a,b)

Definition at line 59 of file fixed_c5x.h.

6.21.2.7 #define MULT16_16_Q15(a, b) (_smpy(a,b))

Definition at line 65 of file fixed_c5x.h.

6.21.2.8 #define MULT16_16SU(a, b) _lmpysu(a,b)

Definition at line 68 of file fixed_c5x.h.

**6.21.2.9 #define MULT16_32_Q15(a, b) ADD32(SHL(MULT16_16((a),SHR((b),16)),1),
SHR(MULT16_16SU((a),(b)),15))**

Definition at line 75 of file fixed_c5x.h.

6.21.2.10 #define MULT_16_16(a, b) _lmpy(a,b)

Definition at line 71 of file fixed_c5x.h.

6.21.2.11 #define OVERRIDE_CELT_ILOG2

Definition at line 79 of file fixed_c5x.h.

6.21.2.12 #define OVERRIDE_CELT_MAXABS16

Definition at line 82 of file fixed_c5x.h.

6.21.2.13 #define OVERRIDE_FIND_MAX16

Definition at line 84 of file fixed_c5x.h.

6.21.2.14 #define VSHR32(a, shift) _lshl(a,-(shift))

Definition at line 62 of file fixed_c5x.h.

6.22 libcelt/fixed_c6x.h File Reference

Fixed-point operations for the TI C6x DSP family.

Defines

- #define [MULT16_16SU](#)(a, b) _mpysu(a,b)
- #define [MULT_16_16](#)(a, b) _mpy(a,b)
- #define [celt_ilog2](#)(x) (30 - _norm(x))
- #define [OVERRIDE_CELT_ILOG2](#)
- #define [MULT16_32_Q15](#)(a, b) (_mpylill(a, b) >> 15)

6.22.1 Detailed Description

Fixed-point operations for the TI C6x DSP family.

Definition in file [fixed_c6x.h](#).

6.22.2 Define Documentation

6.22.2.1 #define [celt_ilog2](#)(x) (30 - _norm(x))

Definition at line 44 of file [fixed_c6x.h](#).

6.22.2.2 #define [MULT16_16SU](#)(a, b) _mpysu(a,b)

Definition at line 39 of file [fixed_c6x.h](#).

6.22.2.3 #define [MULT16_32_Q15](#)(a, b) (_mpylill(a, b) >> 15)

Definition at line 48 of file [fixed_c6x.h](#).

6.22.2.4 #define [MULT_16_16](#)(a, b) _mpy(a,b)

Definition at line 42 of file [fixed_c6x.h](#).

6.22.2.5 #define [OVERRIDE_CELT_ILOG2](#)

Definition at line 45 of file [fixed_c6x.h](#).

6.23 libcelt/fixe_debug.h File Reference

Fixed-point operations with debugging. #include <stdio.h>

Defines

- #define MIPS_INC celt_mips++,
- #define MULT16_16SU(a, b) ((celt_word32)(celt_word16)(a)*(celt_word32)(celt_uint16)(b))
- #define MULT32_32_Q31(a, b) ADD32(ADD32(SHL32(MULT16_16(SHR32((a),16),SHR((b),16)),1), SHR32(MULT16_16SU(SHR32((a),16),((b)&0x0000ffff)),15)), SHR32(MULT16_16SU(SHR32((b),16),((a)&0x0000ffff)),15))
- #define MULT16_32_Q16(a, b) ADD32(MULT16_16((a),SHR32((b),16)), SHR32(MULT16_16SU((a),((b)&0x0000ffff)),16))
- #define QCONST16(x, bits) ((celt_word16)(.5+(x)*(((celt_word32)1)<<(bits))))
- #define QCONST32(x, bits) ((celt_word32)(.5+(x)*(((celt_word32)1)<<(bits))))
- #define VERIFY_SHORT(x) ((x)<=-32767&&(x)>=-32768)
- #define VERIFY_INT(x) ((x)<=-2147483647LL&&(x)>=-2147483648LL)
- #define VERIFY_UINT(x) ((x)<=(2147483647LLU<<1))
- #define SHR(a, b) SHR32(a,b)
- #define PSHR(a, b) PSHR32(a,b)
- #define EXTRACT16(x) _EXTRACT16(x, __FILE__, __LINE__)
- #define EXTEND32(x) _EXTEND32(x, __FILE__, __LINE__)
- #define SHR16(a, shift) _SHR16(a, shift, __FILE__, __LINE__)
- #define SHL16(a, shift) _SHL16(a, shift, __FILE__, __LINE__)
- #define PSHR16(a, shift) (celt_mips--,SHR16(ADD16((a),((1<<((shift))>>1))),shift))
- #define PSHR32(a, shift) (celt_mips--,SHR32(ADD32((a),(((celt_word32)1)<<((shift))>>1))),shift))
- #define VSHR32(a, shift) (((shift)>0) ? SHR32(a, shift) : SHL32(a, -(shift)))
- #define SATURATE16(x, a) (((x)>(a) ? (a) : (x)<-(a) ? -(a) : (x)))
- #define SATURATE32(x, a) (((x)>(a) ? (a) : (x)<-(a) ? -(a) : (x)))
- #define ROUND16(x, a) (celt_mips--,EXTRACT16(PSHR32((x),(a))))
- #define HALF32(x) (SHR32(x,1))
- #define ADD16(a, b) _ADD16(a, b, __FILE__, __LINE__)
- #define SUB16(a, b) _SUB16(a, b, __FILE__, __LINE__)
- #define ADD32(a, b) _ADD32(a, b, __FILE__, __LINE__)
- #define SUB32(a, b) _SUB32(a, b, __FILE__, __LINE__)
- #define UADD32(a, b) _UADD32(a, b, __FILE__, __LINE__)
- #define USUB32(a, b) _USUB32(a, b, __FILE__, __LINE__)
- #define MULT16_16(a, b) _MULT16_16(a, b, __FILE__, __LINE__)
- #define MAC16_16(c, a, b) (celt_mips-=2,ADD32((c),MULT16_16((a),(b))))
- #define MAC16_16_Q11(c, a, b) (ADD16((c),EXTRACT16(SHR32(MULT16_16((a),(b)),11))))
- #define MAC16_16_Q13(c, a, b) (ADD16((c),EXTRACT16(SHR32(MULT16_16((a),(b)),13))))
- #define MAC16_16_P13(c, a, b) (ADD16((c),SHR32(ADD32(4096,MULT16_16((a),(b))),13)))
- #define MULT16_32_QX(a, b, Q) _MULT16_32_QX(a, b, Q, __FILE__, __LINE__)
- #define MULT16_32_Q11(a, b) MULT16_32_QX(a,b,11)
- #define MAC16_32_Q11(c, a, b) ADD32((c),MULT16_32_Q11((a),(b)))
- #define MULT16_32_Q12(a, b) MULT16_32_QX(a,b,12)
- #define MULT16_32_Q13(a, b) MULT16_32_QX(a,b,13)
- #define MULT16_32_Q14(a, b) MULT16_32_QX(a,b,14)
- #define MULT16_32_Q15(a, b) MULT16_32_QX(a,b,15)

- #define `MULT16_32_P15(a, b) MULT16_32_PX(a,b,15)`
- #define `MAC16_32_Q15(c, a, b) (celt_mips==2,ADD32((c),MULT16_32_Q15((a),(b))))`
- #define `MULT16_16_Q15(a, b) _MULT16_16_Q15(a, b, __FILE__, __LINE__)`
- #define `DIV32_16(a, b) _DIV32_16(a, b, __FILE__, __LINE__)`
- #define `DIV32(a, b) _DIV32(a, b, __FILE__, __LINE__)`
- #define `PDIV32(a, b) DIV32(ADD32((a),(b)>>1),b)`
- #define `PDIV32_16(a, b) DIV32_16(ADD32((a),(b)>>1),b)`
- #define `PRINT_MIPS(file) do {fprintf (file, "total complexity = %llu MIPS\n", celt_mips);} while (0);`

Variables

- long long `celt_mips = 0`

6.23.1 Detailed Description

Fixed-point operations with debugging.

Definition in file [fixed_debug.h](#).

6.23.2 Define Documentation

6.23.2.1 #define `ADD16(a, b) _ADD16(a, b, __FILE__, __LINE__)`

Definition at line 192 of file [fixed_debug.h](#).

6.23.2.2 #define `ADD32(a, b) _ADD32(a, b, __FILE__, __LINE__)`

Definition at line 224 of file [fixed_debug.h](#).

6.23.2.3 #define `DIV32(a, b) _DIV32(a, b, __FILE__, __LINE__)`

Definition at line 529 of file [fixed_debug.h](#).

6.23.2.4 #define `DIV32_16(a, b) _DIV32_16(a, b, __FILE__, __LINE__)`

Definition at line 502 of file [fixed_debug.h](#).

6.23.2.5 #define `EXTEND32(x) _EXTEND32(x, __FILE__, __LINE__)`

Definition at line 106 of file [fixed_debug.h](#).

6.23.2.6 #define `EXTRACT16(x) _EXTRACT16(x, __FILE__, __LINE__)`

Definition at line 93 of file [fixed_debug.h](#).

6.23.2.7 #define HALF32(x) (SHR32(x,1))

Definition at line 187 of file fixed_debug.h.

6.23.2.8 #define MAC16_16(c, a, b) (celt_mips-=2,ADD32((c),MULT16_16((a),(b))))

Definition at line 324 of file fixed_debug.h.

6.23.2.9 #define MAC16_16_P13(c, a, b) (ADD16((c),SHR32(ADD32(4096,MULT16_16((a),(b))),13)))

Definition at line 327 of file fixed_debug.h.

6.23.2.10 #define MAC16_16_Q11(c, a, b) (ADD16((c),EXTRACT16(SHR32(MULT16_16((a),(b)),11))))

Definition at line 325 of file fixed_debug.h.

6.23.2.11 #define MAC16_16_Q13(c, a, b) (ADD16((c),EXTRACT16(SHR32(MULT16_16((a),(b)),13))))

Definition at line 326 of file fixed_debug.h.

6.23.2.12 #define MAC16_32_Q11(c, a, b) ADD32((c),MULT16_32_Q11((a),(b)))

Definition at line 371 of file fixed_debug.h.

6.23.2.13 #define MAC16_32_Q15(c, a, b) (celt_mips-=2,ADD32((c),MULT16_32_Q15((a),(b))))

Definition at line 377 of file fixed_debug.h.

6.23.2.14 #define MIPS_INC celt_mips++,

Definition at line 47 of file fixed_debug.h.

6.23.2.15 #define MULT16_16(a, b) _MULT16_16(a, b, __FILE__, __LINE__)

Definition at line 309 of file fixed_debug.h.

6.23.2.16 #define MULT16_16_Q15(a, b) _MULT16_16_Q15(a, b, __FILE__, __LINE__)

Definition at line 432 of file fixed_debug.h.

6.23.2.17 #define MULT16_16SU(a, b) ((celt_word32)(celt_word16)(a)*(celt_word32)(celt_uint16)(b))

Definition at line 49 of file fixed_debug.h.

6.23.2.18 #define MULT16_32_P15(a, b) MULT16_32_PX(a,b,15)

Definition at line 376 of file fixed_debug.h.

6.23.2.19 #define MULT16_32_Q11(a, b) MULT16_32_QX(a,b,11)

Definition at line 370 of file fixed_debug.h.

6.23.2.20 #define MULT16_32_Q12(a, b) MULT16_32_QX(a,b,12)

Definition at line 372 of file fixed_debug.h.

6.23.2.21 #define MULT16_32_Q13(a, b) MULT16_32_QX(a,b,13)

Definition at line 373 of file fixed_debug.h.

6.23.2.22 #define MULT16_32_Q14(a, b) MULT16_32_QX(a,b,14)

Definition at line 374 of file fixed_debug.h.

6.23.2.23 #define MULT16_32_Q15(a, b) MULT16_32_QX(a,b,15)

Definition at line 375 of file fixed_debug.h.

**6.23.2.24 #define MULT16_32_Q16(a, b) ADD32(MULT16_16((a),SHR32((b),16)),
SHR32(MULT16_16SU((a),((b)&0x0000ffff)),16))**

16x32 multiplication, followed by a 16-bit shift right. Results fits in 32 bits

Definition at line 53 of file fixed_debug.h.

6.23.2.25 #define MULT16_32_QX(a, b, Q) _MULT16_32_QX(a, b, Q, __FILE__, __LINE__)

Definition at line 330 of file fixed_debug.h.

**6.23.2.26 #define MULT32_32_Q31(a, b) ADD32(ADD32(SHL32(MULT16_-
16(SHR32((a),16),SHR((b),16)),1), SHR32(MULT16_-
16SU(SHR32((a),16),((b)&0x0000ffff)),15)), SHR32(MULT16_-
16SU(SHR32((b),16),((a)&0x0000ffff)),15))**

Definition at line 50 of file fixed_debug.h.

6.23.2.27 #define PDIV32(a, b) DIV32(ADD32((a),(b)>>1),b)

Definition at line 549 of file fixed_debug.h.

6.23.2.28 `#define PDIV32_16(a, b) DIV32_16(ADD32((a),(b)>>1),b)`

Definition at line 550 of file fixed_debug.h.

6.23.2.29 `#define PRINT_MIPS(file) do {fprintf (file, "total complexity = %llu MIPS\n", celt_mips);} while (0);`

Definition at line 553 of file fixed_debug.h.

6.23.2.30 `#define PSHR(a, b) PSHR32(a,b)`

Definition at line 64 of file fixed_debug.h.

6.23.2.31 `#define PSHR16(a, shift) (celt_mips--,SHR16(ADD16((a),(1<<((shift))>>1))),shift))`

Definition at line 179 of file fixed_debug.h.

6.23.2.32 `#define PSHR32(a, shift) (celt_mips--,SHR32(ADD32((a),(((celt_word32)1)<<((shift))>>1))),shift))`

Definition at line 180 of file fixed_debug.h.

6.23.2.33 `#define QCONST16(x, bits) ((celt_word16)(.5+(x)*(((celt_word32)1)<<(bits))))`

Definition at line 55 of file fixed_debug.h.

6.23.2.34 `#define QCONST32(x, bits) ((celt_word32)(.5+(x)*(((celt_word32)1)<<(bits))))`

Definition at line 56 of file fixed_debug.h.

6.23.2.35 `#define ROUND16(x, a) (celt_mips--,EXTRACT16(PSHR32((x),(a))))`

Definition at line 186 of file fixed_debug.h.

6.23.2.36 `#define SATURATE16(x, a) (((x)>(a) ? (a) : (x)<-(a) ? -(a) : (x)))`

Definition at line 183 of file fixed_debug.h.

6.23.2.37 `#define SATURATE32(x, a) (((x)>(a) ? (a) : (x)<-(a) ? -(a) : (x)))`

Definition at line 184 of file fixed_debug.h.

6.23.2.38 `#define SHL16(a, shift) _SHL16(a, shift, __FILE__, __LINE__)`

Definition at line 133 of file fixed_debug.h.

6.23.2.39 #define SHR(a, b) SHR32(a,b)

Definition at line 63 of file fixed_debug.h.

6.23.2.40 #define SHR16(a, shift) _SHR16(a, shift, __FILE__, __LINE__)

Definition at line 119 of file fixed_debug.h.

6.23.2.41 #define SUB16(a, b) _SUB16(a, b, __FILE__, __LINE__)

Definition at line 209 of file fixed_debug.h.

6.23.2.42 #define SUB32(a, b) _SUB32(a, b, __FILE__, __LINE__)

Definition at line 241 of file fixed_debug.h.

6.23.2.43 #define UADD32(a, b) _UADD32(a, b, __FILE__, __LINE__)

Definition at line 257 of file fixed_debug.h.

6.23.2.44 #define USUB32(a, b) _USUB32(a, b, __FILE__, __LINE__)

Definition at line 275 of file fixed_debug.h.

6.23.2.45 #define VERIFY_INT(x) ((x)<=2147483647LL&&(x)>=-2147483648LL)

Definition at line 60 of file fixed_debug.h.

6.23.2.46 #define VERIFY_SHORT(x) ((x)<=32767&&(x)>=-32768)

Definition at line 59 of file fixed_debug.h.

6.23.2.47 #define VERIFY_UINT(x) ((x)<=(2147483647LLU<<1))

Definition at line 61 of file fixed_debug.h.

6.23.2.48 #define VSHR32(a, shift) (((shift)>0) ? SHR32(a, shift) : SHL32(a, -(shift)))

Definition at line 181 of file fixed_debug.h.

6.23.3 Variable Documentation**6.23.3.1 long long celt_mips = 0**

Definition at line 42 of file fixed_debug.h.

6.24 libcelt/fixed_generic.h File Reference

Generic fixed-point operations.

Defines

- #define **MULT16_16SU**(a, b) ((**celt_word32**)(**celt_word16**)(a)*(**celt_word32**)(**celt_uint16**)(b))
- #define **MULT16_32_Q16**(a, b) ADD32(MULT16_16((a),SHR((b),16)), SHR(MULT16_16SU((a),((b)&0x0000ffff)),16))
- #define **MULT16_32_Q15**(a, b) ADD32(SHL(MULT16_16((a),SHR((b),16)),1), SHR(MULT16_16SU((a),((b)&0x0000ffff)),15))
- #define **MULT32_32_Q31**(a, b) ADD32(ADD32(SHL(MULT16_16(SHR((a),16),SHR((b),16)),1), SHR(MULT16_16SU(SHR((a),16),((b)&0x0000ffff)),15)), SHR(MULT16_16SU(SHR((b),16),((a)&0x0000ffff)),15))
- #define **MULT32_32_Q32**(a, b) ADD32(ADD32(MULT16_16(SHR((a),16),SHR((b),16)), SHR(MULT16_16SU(SHR((a),16),((b)&0x0000ffff)),16)), SHR(MULT16_16SU(SHR((b),16),((a)&0x0000ffff)),16))
- #define **QCONST16**(x, bits) ((**celt_word16**)(.5+(x)*(((**celt_word32**)1)<<(bits))))
- #define **QCONST32**(x, bits) ((**celt_word32**)(.5+(x)*(((**celt_word32**)1)<<(bits))))
- #define **NEG16**(x) -(x)
- #define **NEG32**(x) -(x)
- #define **EXTRACT16**(x) ((**celt_word16**)(x))
- #define **EXTEND32**(x) ((**celt_word32**)(x))
- #define **SHR16**(a, shift) ((a) >> (shift))
- #define **SHL16**(a, shift) ((a) << (shift))
- #define **SHR32**(a, shift) ((a) >> (shift))
- #define **SHL32**(a, shift) ((**celt_word32**)(a) << (shift))
- #define **PSHR16**(a, shift) (SHR16((a)+((1<<((shift))>>1)),shift))
- #define **PSHR32**(a, shift) (SHR32((a)+((EXTEND32(1)<<((shift))>>1)),shift))
- #define **VSHR32**(a, shift) (((shift)>0) ? SHR32(a, shift) : SHL32(a, -(shift)))
- #define **SATURATE16**(x, a) (((x)>a) ? (a) : (x)<-(a) ? -(a) : (x))
- #define **SATURATE32**(x, a) (((x)>a) ? (a) : (x)<-(a) ? -(a) : (x))
- #define **SHR**(a, shift) ((a) >> (shift))
- #define **SHL**(a, shift) ((**celt_word32**)(a) << (shift))
- #define **PSHR**(a, shift) (SHR((a)+((EXTEND32(1)<<((shift))>>1)),shift))
- #define **SATURATE**(x, a) (((x)>a) ? (a) : (x)<-(a) ? -(a) : (x))
- #define **ROUND16**(x, a) (EXTRACT16(PSHR32((x),(a))))
- #define **HALF32**(x) (SHR32(x,1))
- #define **ADD16**(a, b) ((**celt_word16**)((**celt_word16**)(a)+(**celt_word16**)(b)))
- #define **SUB16**(a, b) ((**celt_word16**)(a)-(**celt_word16**)(b))
- #define **ADD32**(a, b) ((**celt_word32**)(a)+(**celt_word32**)(b))
- #define **SUB32**(a, b) ((**celt_word32**)(a)-(**celt_word32**)(b))
- #define **MULT16_16_16**(a, b) (((**celt_word16**)(a))*((**celt_word16**)(b)))
- #define **MULT16_16**(a, b) (((**celt_word32**)(**celt_word16**)(a))*((**celt_word32**)(**celt_word16**)(b)))
- #define **MAC16_16**(c, a, b) (ADD32((c),MULT16_16((a),(b))))
- #define **MULT16_32_Q12**(a, b) ADD32(MULT16_16((a),SHR((b),12)), SHR(MULT16_16((a),((b)&0x0000fff)),12))
- #define **MULT16_32_Q13**(a, b) ADD32(MULT16_16((a),SHR((b),13)), SHR(MULT16_16((a),((b)&0x00001fff)),13))

- #define `MULT16_32_Q14`(a, b) `ADD32(MULT16_16((a),SHR((b),14)), SHR(MULT16_16((a),((b)&0x00003fff)),14))`
- #define `MULT16_32_Q11`(a, b) `ADD32(MULT16_16((a),SHR((b),11)), SHR(MULT16_16((a),((b)&0x000007ff)),11))`
- #define `MAC16_32_Q11`(c, a, b) `ADD32(c,ADD32(MULT16_16((a),SHR((b),11)), SHR(MULT16_16((a),((b)&0x000007ff)),11)))`
- #define `MULT16_32_P15`(a, b) `ADD32(MULT16_16((a),SHR((b),15)), PSHR(MULT16_16((a),((b)&0x00007fff)),15))`
- #define `MAC16_32_Q15`(c, a, b) `ADD32(c,ADD32(MULT16_16((a),SHR((b),15)), SHR(MULT16_16((a),((b)&0x00007fff)),15)))`
- #define `MAC16_16_Q11`(c, a, b) `(ADD32((c),SHR(MULT16_16((a),(b)),11)))`
- #define `MAC16_16_Q13`(c, a, b) `(ADD32((c),SHR(MULT16_16((a),(b)),13)))`
- #define `MAC16_16_P13`(c, a, b) `(ADD32((c),SHR(ADD32(4096,MULT16_16((a),(b))),13)))`
- #define `MULT16_16_Q11_32`(a, b) `(SHR(MULT16_16((a),(b)),11))`
- #define `MULT16_16_Q13`(a, b) `(SHR(MULT16_16((a),(b)),13))`
- #define `MULT16_16_Q14`(a, b) `(SHR(MULT16_16((a),(b)),14))`
- #define `MULT16_16_Q15`(a, b) `(SHR(MULT16_16((a),(b)),15))`
- #define `MULT16_16_P13`(a, b) `(SHR(ADD32(4096,MULT16_16((a),(b))),13))`
- #define `MULT16_16_P14`(a, b) `(SHR(ADD32(8192,MULT16_16((a),(b))),14))`
- #define `MULT16_16_P15`(a, b) `(SHR(ADD32(16384,MULT16_16((a),(b))),15))`
- #define `DIV32_16`(a, b) `((celt_word16)(((celt_word32)(a))/((celt_word16)(b))))`
- #define `PDIV32_16`(a, b) `((celt_word16)(((celt_word32)(a)+((celt_word16)(b)>>1))/((celt_word16)(b))))`
- #define `DIV32`(a, b) `((((celt_word32)(a))/((celt_word32)(b))))`
- #define `PDIV32`(a, b) `((((celt_word32)(a)+((celt_word16)(b)>>1))/((celt_word32)(b))))`

6.24.1 Detailed Description

Generic fixed-point operations.

Definition in file [fixed_generic.h](#).

6.24.2 Define Documentation

6.24.2.1 #define `ADD16`(a, b) `((celt_word16)((celt_word16)(a)+(celt_word16)(b)))`

Add two 16-bit values

Definition at line 103 of file [fixed_generic.h](#).

6.24.2.2 #define `ADD32`(a, b) `((celt_word32)(a)+(celt_word32)(b))`

Add two 32-bit values

Definition at line 107 of file [fixed_generic.h](#).

6.24.2.3 #define `DIV32`(a, b) `((((celt_word32)(a))/((celt_word32)(b))))`

Divide a 32-bit value by a 32-bit value. Result fits in 32 bits

Definition at line 157 of file [fixed_generic.h](#).

6.24.2.4 #define DIV32_16(a, b) ((celt_word16)(((celt_word32)(a))/((celt_word16)(b))))

Divide a 32-bit value by a 16-bit value. Result fits in 16 bits

Definition at line 153 of file fixed_generic.h.

6.24.2.5 #define EXTEND32(x) ((celt_word32)(x))

Change a 16-bit value into a 32-bit value

Definition at line 68 of file fixed_generic.h.

6.24.2.6 #define EXTRACT16(x) ((celt_word16)(x))

Change a 32-bit value into a 16-bit value. The value is assumed to fit in 16-bit, otherwise the result is undefined

Definition at line 66 of file fixed_generic.h.

6.24.2.7 #define HALF32(x) (SHR32(x,1))

Divide by two

Definition at line 100 of file fixed_generic.h.

6.24.2.8 #define MAC16_16(c, a, b) (ADD32((c),MULT16_16((a),(b))))

16x16 multiply-add where the result fits in 32 bits

Definition at line 120 of file fixed_generic.h.

6.24.2.9 #define MAC16_16_P13(c, a, b) (ADD32((c),SHR(ADD32(4096,MULT16_16((a),(b))),13)))

Definition at line 141 of file fixed_generic.h.

6.24.2.10 #define MAC16_16_Q11(c, a, b) (ADD32((c),SHR(MULT16_16((a),(b)),11)))

Definition at line 139 of file fixed_generic.h.

6.24.2.11 #define MAC16_16_Q13(c, a, b) (ADD32((c),SHR(MULT16_16((a),(b)),13)))

Definition at line 140 of file fixed_generic.h.

6.24.2.12 #define MAC16_32_Q11(c, a, b) ADD32(c,ADD32(MULT16_16((a),SHR((b),11)),SHR(MULT16_16((a),(b)&0x000007ff),11)))

16x32 multiply-add, followed by an 11-bit shift right. Results fits in 32 bits

Definition at line 131 of file fixed_generic.h.

6.24.2.13 `#define MAC16_32_Q15(c, a, b) ADD32(c,ADD32(MULT16_16((a),SHR((b),15)),
SHR(MULT16_16((a),(b)&0x00007fff),15)))`

16x32 multiply-add, followed by a 15-bit shift right. Results fits in 32 bits

Definition at line 136 of file fixed_generic.h.

6.24.2.14 `#define MULT16_16(a, b) (((celt_word32)(celt_word16)(a))*((celt_word32)(celt_ -
word16)(b)))`

16x16 multiplication where the result fits in 32 bits

Definition at line 117 of file fixed_generic.h.

6.24.2.15 `#define MULT16_16_16(a, b) (((celt_word16)(a))*((celt_word16)(b)))`

16x16 multiplication where the result fits in 16 bits

Definition at line 113 of file fixed_generic.h.

6.24.2.16 `#define MULT16_16_P13(a, b) (SHR(ADD32(4096,MULT16_16((a),(b))),13))`

Definition at line 148 of file fixed_generic.h.

6.24.2.17 `#define MULT16_16_P14(a, b) (SHR(ADD32(8192,MULT16_16((a),(b))),14))`

Definition at line 149 of file fixed_generic.h.

6.24.2.18 `#define MULT16_16_P15(a, b) (SHR(ADD32(16384,MULT16_16((a),(b))),15))`

Definition at line 150 of file fixed_generic.h.

6.24.2.19 `#define MULT16_16_Q11_32(a, b) (SHR(MULT16_16((a),(b)),11))`

Definition at line 143 of file fixed_generic.h.

6.24.2.20 `#define MULT16_16_Q13(a, b) (SHR(MULT16_16((a),(b)),13))`

Definition at line 144 of file fixed_generic.h.

6.24.2.21 `#define MULT16_16_Q14(a, b) (SHR(MULT16_16((a),(b)),14))`

Definition at line 145 of file fixed_generic.h.

6.24.2.22 `#define MULT16_16_Q15(a, b) (SHR(MULT16_16((a),(b)),15))`

Definition at line 146 of file fixed_generic.h.

6.24.2.23 `#define MULT16_16SU(a, b) ((celt_word32)(celt_word16)(a)*(celt_word32)(celt_uint16)(b))`

Multiply a 16-bit signed value by a 16-bit unsigned value. The result is a 32-bit signed value

Definition at line 41 of file fixed_generic.h.

6.24.2.24 `#define MULT16_32_P15(a, b) ADD32(MULT16_16((a),SHR((b),15)), PSHR(MULT16_16((a),(b)&0x00007fff),15))`

16x32 multiplication, followed by a 15-bit shift right (round-to-nearest). Results fits in 32 bits

Definition at line 134 of file fixed_generic.h.

6.24.2.25 `#define MULT16_32_Q11(a, b) ADD32(MULT16_16((a),SHR((b),11)), SHR(MULT16_16((a),(b)&0x000007ff),11))`

16x32 multiplication, followed by an 11-bit shift right. Results fits in 32 bits

Definition at line 129 of file fixed_generic.h.

6.24.2.26 `#define MULT16_32_Q12(a, b) ADD32(MULT16_16((a),SHR((b),12)), SHR(MULT16_16((a),(b)&0x00000fff),12))`

16x32 multiplication, followed by a 12-bit shift right. Results fits in 32 bits

Definition at line 122 of file fixed_generic.h.

6.24.2.27 `#define MULT16_32_Q13(a, b) ADD32(MULT16_16((a),SHR((b),13)), SHR(MULT16_16((a),(b)&0x00001fff),13))`

16x32 multiplication, followed by a 13-bit shift right. Results fits in 32 bits

Definition at line 124 of file fixed_generic.h.

6.24.2.28 `#define MULT16_32_Q14(a, b) ADD32(MULT16_16((a),SHR((b),14)), SHR(MULT16_16((a),(b)&0x00003fff),14))`

16x32 multiplication, followed by a 14-bit shift right. Results fits in 32 bits

Definition at line 126 of file fixed_generic.h.

6.24.2.29 `#define MULT16_32_Q15(a, b) ADD32(SHL(MULT16_16((a),SHR((b),16)),1), SHR(MULT16_16SU((a),(b)&0x0000ffff),15))`

16x32 multiplication, followed by a 15-bit shift right. Results fits in 32 bits

Definition at line 47 of file fixed_generic.h.

6.24.2.30 `#define MULT16_32_Q16(a, b) ADD32(MULT16_16((a),SHR((b),16)), SHR(MULT16_16SU((a),(b)&0x0000ffff),16))`

16x32 multiplication, followed by a 16-bit shift right. Results fits in 32 bits

Definition at line 44 of file fixed_generic.h.

6.24.2.31 `#define MULT32_32_Q31(a, b) ADD32(ADD32(SHL(MULT16_16(SHR((a),16),SHR((b),16)),1), SHR(MULT16_16SU(SHR((a),16),((b)&0x0000ffff),15)), SHR(MULT16_16SU(SHR((b),16),((a)&0x0000ffff),15))`

32x32 multiplication, followed by a 31-bit shift right. Results fits in 32 bits

Definition at line 50 of file fixed_generic.h.

6.24.2.32 `#define MULT32_32_Q32(a, b) ADD32(ADD32(MULT16_16(SHR((a),16),SHR((b),16)), SHR(MULT16_16SU(SHR((a),16),((b)&0x0000ffff),16)), SHR(MULT16_16SU(SHR((b),16),((a)&0x0000ffff),16))`

32x32 multiplication, followed by a 32-bit shift right. Results fits in 32 bits

Definition at line 53 of file fixed_generic.h.

6.24.2.33 `#define NEG16(x) (-x)`

Negate a 16-bit value

Definition at line 61 of file fixed_generic.h.

6.24.2.34 `#define NEG32(x) (-x)`

Negate a 32-bit value

Definition at line 63 of file fixed_generic.h.

6.24.2.35 `#define PDIV32(a, b) (((celt_word32)(a)+((celt_word16)(b)>>1))/((celt_word32)(b)))`

Divide a 32-bit value by a 32-bit value and round to nearest. Result fits in 32 bits

Definition at line 159 of file fixed_generic.h.

6.24.2.36 `#define PDIV32_16(a, b) ((celt_word16)(((celt_word32)(a)+((celt_word16)(b)>>1))/((celt_word16)(b))))`

Divide a 32-bit value by a 16-bit value and round to nearest. Result fits in 16 bits

Definition at line 155 of file fixed_generic.h.

6.24.2.37 `#define PSHR(a, shift) (SHR((a)+((EXTEND32(1)<<((shift))>>1)),shift))`

Definition at line 94 of file fixed_generic.h.

6.24.2.38 `#define PSHR16(a, shift) (SHR16((a)+((1<<((shift))>>1)),shift))`

16-bit arithmetic shift right with rounding-to-nearest instead of rounding down

Definition at line 80 of file fixed_generic.h.

6.24.2.39 #define PSHR32(a, shift) (SHR32((a)+((EXTEND32(1)<<((shift))>>1)),shift))

32-bit arithmetic shift right with rounding-to-nearest instead of rounding down

Definition at line 82 of file fixed_generic.h.

6.24.2.40 #define QCONST16(x, bits) ((celt_word16)(.5+(x)*(((celt_word32)1)<<(bits))))

Compile-time conversion of float constant to 16-bit value

Definition at line 56 of file fixed_generic.h.

6.24.2.41 #define QCONST32(x, bits) ((celt_word32)(.5+(x)*(((celt_word32)1)<<(bits))))

Compile-time conversion of float constant to 32-bit value

Definition at line 58 of file fixed_generic.h.

6.24.2.42 #define ROUND16(x, a) (EXTRACT16(PSHR32((x),(a))))

Shift by a and round-to-nearest 32-bit value. Result is a 16-bit value

Definition at line 98 of file fixed_generic.h.

6.24.2.43 #define SATURATE(x, a) (((x)>(a) ? (a) : (x)<-(a) ? -(a) : (x)))

Definition at line 95 of file fixed_generic.h.

6.24.2.44 #define SATURATE16(x, a) (((x)>(a) ? (a) : (x)<-(a) ? -(a) : (x)))

Saturates 16-bit value to +/- a

Definition at line 87 of file fixed_generic.h.

6.24.2.45 #define SATURATE32(x, a) (((x)>(a) ? (a) : (x)<-(a) ? -(a) : (x)))

Saturates 32-bit value to +/- a

Definition at line 89 of file fixed_generic.h.

6.24.2.46 #define SHL(a, shift) ((celt_word32)(a) << (shift))

Definition at line 93 of file fixed_generic.h.

6.24.2.47 #define SHL16(a, shift) ((a) << (shift))

Arithmetic shift-left of a 16-bit value

Definition at line 73 of file fixed_generic.h.

6.24.2.48 #define SHL32(a, shift) ((celt_word32)(a) << (shift))

Arithmetic shift-left of a 32-bit value

Definition at line 77 of file fixed_generic.h.

6.24.2.49 #define SHR(a, shift) ((a) >> (shift))

"RAW" macros, should not be used outside of this header file

Definition at line 92 of file fixed_generic.h.

6.24.2.50 #define SHR16(a, shift) ((a) >> (shift))

Arithmetic shift-right of a 16-bit value

Definition at line 71 of file fixed_generic.h.

6.24.2.51 #define SHR32(a, shift) ((a) >> (shift))

Arithmetic shift-right of a 32-bit value

Definition at line 75 of file fixed_generic.h.

6.24.2.52 #define SUB16(a, b) ((celt_word16)(a)-(celt_word16)(b))

Subtract two 16-bit values

Definition at line 105 of file fixed_generic.h.

6.24.2.53 #define SUB32(a, b) ((celt_word32)(a)-(celt_word32)(b))

Subtract two 32-bit values

Definition at line 109 of file fixed_generic.h.

6.24.2.54 #define VSHR32(a, shift) (((shift)>0) ? SHR32(a, shift) : SHL32(a, -(shift)))

32-bit arithmetic shift right where the argument can be negative

Definition at line 84 of file fixed_generic.h.

6.25 libcelt/float_cast.h File Reference

```
#include <math.h>
```

Defines

- #define `float2int(float)` `((int)(floor(.5+float))`

6.25.1 Define Documentation

6.25.1.1 #define `float2int(float)` `((int)(floor(.5+float))`

Definition at line 104 of file float_cast.h.

6.26 libcelt/header.c File Reference

```
#include "celt_header.h"
#include "os_support.h"
#include "modes.h"
```

Functions

- int `celt_header_init` (`CELTHHeader *header`, const `CELTMode *m`, int channels)
- int `celt_header_to_packet` (const `CELTHHeader *header`, unsigned char *`packet`, `celt_uint32` size)
- int `celt_header_from_packet` (const unsigned char *`packet`, `celt_uint32` size, `CELTHHeader *header`)

6.26.1 Function Documentation

6.26.1.1 int `celt_header_from_packet` (const unsigned char * *packet*, `celt_uint32` size, `CELTHHeader * header`)

Definition at line 103 of file header.c.

References `CELT_BAD_ARG`, `CELT_COPY`, and `CELT_MEMSET`.

6.26.1.2 int `celt_header_init` (`CELTHHeader * header`, const `CELTMode * m`, int *channels*)

Creates a basic header struct

Definition at line 54 of file header.c.

References `CELTHHeader::bytes_per_packet`, `CELT_BAD_ARG`, `CELT_COPY`, `CELT_GET_BITSTREAM_VERSION`, `CELT_INVALID_MODE`, `celt_mode_info()`, `CELT_OK`, `check_mode()`, `CELTHHeader::codec_id`, `CELTHHeader::codec_version`, `CELTHHeader::extra_headers`, `CELTHHeader::frame_size`, `CELTMode::Fs`, `CELTHHeader::header_size`, `CELTMode::mdctSize`, `CELTHHeader::nb_channels`, `CELTMode::overlap`, `CELTHHeader::overlap`, `CELTHHeader::sample_rate`, and `CELTHHeader::version_id`.

6.26.1.3 int `celt_header_to_packet` (const `CELTHHeader * header`, unsigned char * *packet*, `celt_uint32` size)

Definition at line 76 of file header.c.

References `CELT_BAD_ARG`, `CELT_COPY`, and `CELT_MEMSET`.

6.27 libcelt/kfft_double.h File Reference

```
#include "kiss_fft.h"  
#include "_kiss_fft_guts.h"
```

Defines

- #define `cpx32_fft_alloc`(length) `kiss_fft_alloc`(length, 0, 0);
- #define `cpx32_fft_free`(state) `kiss_fft_free`(state)
- #define `cpx32_fft`(state, X, Y, nx) `kiss_fft`(state,(const `kiss_fft_cpx` *)`(X)`, (`kiss_fft_cpx` *)`(Y)`)
- #define `cpx32_ifft`(state, X, Y, nx) `kiss_ifft`(state,(const `kiss_fft_cpx` *)`(X)`, (`kiss_fft_cpx` *)`(Y)`)

6.27.1 Define Documentation

6.27.1.1 #define `cpx32_fft`(state, X, Y, nx) `kiss_fft`(state,(const `kiss_fft_cpx` *)`(X)`, (`kiss_fft_cpx` *)`(Y)`)

Definition at line 74 of file `kfft_double.h`.

Referenced by `mdct_forward()`.

6.27.1.2 #define `cpx32_fft_alloc`(length) `kiss_fft_alloc`(length, 0, 0);

Definition at line 72 of file `kfft_double.h`.

Referenced by `mdct_init()`.

6.27.1.3 #define `cpx32_fft_free`(state) `kiss_fft_free`(state)

Definition at line 73 of file `kfft_double.h`.

Referenced by `mdct_clear()`.

6.27.1.4 #define `cpx32_ifft`(state, X, Y, nx) `kiss_ifft`(state,(const `kiss_fft_cpx` *)`(X)`, (`kiss_fft_cpx` *)`(Y)`)

Definition at line 75 of file `kfft_double.h`.

Referenced by `mdct_backward()`.

6.28 libcelt/kfft_single.c File Reference

6.29 libcelt/kfft_single.h File Reference

```
#include "kiss_fft.h"  
#include "kiss_fftr.h"  
#include "_kiss_fft_guts.h"
```

Defines

- #define `real16_fft_alloc`(length) `kiss_fftr_alloc_celt_single`(length, 0, 0);
- #define `real16_fft_free`(state) `kiss_fft_free`(state)
- #define `real16_fft_inplace`(state, X, nx) `kiss_fftr_inplace`(state,X)
- #define `BITREV`(state, i) ((state)->substate->bitrev[i])
- #define `real16_ifft`(state, X, Y, nx) `kiss_fftri`(state,X, Y)

6.29.1 Define Documentation

6.29.1.1 #define `BITREV`(state, i) ((state)->substate->bitrev[i])

Definition at line 79 of file `kfft_single.h`.

Referenced by `find_spectral_pitch()`.

6.29.1.2 #define `real16_fft_alloc`(length) `kiss_fftr_alloc_celt_single`(length, 0, 0);

Definition at line 76 of file `kfft_single.h`.

Referenced by `pitch_state_alloc()`.

6.29.1.3 #define `real16_fft_free`(state) `kiss_fft_free`(state)

Definition at line 77 of file `kfft_single.h`.

Referenced by `pitch_state_free()`.

6.29.1.4 #define `real16_fft_inplace`(state, X, nx) `kiss_fftr_inplace`(state,X)

Definition at line 78 of file `kfft_single.h`.

Referenced by `find_spectral_pitch()`.

6.29.1.5 #define `real16_ifft`(state, X, Y, nx) `kiss_fftri`(state,X, Y)

Definition at line 80 of file `kfft_single.h`.

Referenced by `find_spectral_pitch()`.

6.30 libcelt/kiss_fft.c File Reference

```
#include "_kiss_fft_guts.h"
#include "arch.h"
#include "os_support.h"
#include "mathops.h"
#include "stack_alloc.h"
```

Functions

- void [kf_work](#) ([kiss_fft_cpx](#) *Fout, const [kiss_fft_cpx](#) *f, const size_t fstride, int in_stride, int *factors, const [kiss_fft_cfg](#) st, int N, int s2, int m2)
- void [ki_work](#) ([kiss_fft_cpx](#) *Fout, const [kiss_fft_cpx](#) *f, const size_t fstride, int in_stride, int *factors, const [kiss_fft_cfg](#) st, int N, int s2, int m2)
- [kiss_fft_cfg](#) [kiss_fft_alloc](#) (int nfft, void *mem, size_t *lenmem)
- void [kiss_fft_stride](#) ([kiss_fft_cfg](#) st, const [kiss_fft_cpx](#) *fin, [kiss_fft_cpx](#) *fout, int in_stride)
- void [kiss_fft](#) ([kiss_fft_cfg](#) cfg, const [kiss_fft_cpx](#) *fin, [kiss_fft_cpx](#) *fout)
- void [kiss_ifft_stride](#) ([kiss_fft_cfg](#) st, const [kiss_fft_cpx](#) *fin, [kiss_fft_cpx](#) *fout, int in_stride)
- void [kiss_ifft](#) ([kiss_fft_cfg](#) cfg, const [kiss_fft_cpx](#) *fin, [kiss_fft_cpx](#) *fout)

6.30.1 Function Documentation

6.30.1.1 void [kf_work](#) ([kiss_fft_cpx](#) * *Fout*, const [kiss_fft_cpx](#) * *f*, const size_t *fstride*, int *in_stride*, int * *factors*, const [kiss_fft_cfg](#) *st*, int *N*, int *s2*, int *m2*)

Definition at line 512 of file [kiss_fft.c](#).

References [celt_fatal](#), and [kf_work](#).

6.30.1.2 void [ki_work](#) ([kiss_fft_cpx](#) * *Fout*, const [kiss_fft_cpx](#) * *f*, const size_t *fstride*, int *in_stride*, int * *factors*, const [kiss_fft_cfg](#) *st*, int *N*, int *s2*, int *m2*)

Internal function. Can be useful when you want to do the bit-reversing yourself

Definition at line 548 of file [kiss_fft.c](#).

References [celt_fatal](#), and [ki_work](#).

6.30.1.3 void [kiss_fft](#) ([kiss_fft_cfg](#) *cfg*, const [kiss_fft_cpx](#) * *fin*, [kiss_fft_cpx](#) * *fout*)

[kiss_fft\(cfg,in_out_buf\)](#)

Perform an FFT on a complex input buffer. for a forward FFT, fin should be f[0] , f[1] , ... ,f[nfft-1] fout will be F[0] , F[1] , ... ,F[nfft-1] Note that each element is complex and can be accessed like f[k].r and f[k].i

Definition at line 678 of file [kiss_fft.c](#).

References [kiss_fft_stride](#).

6.30.1.4 kiss_fft_cfg kiss_fft_alloc (int *nfft*, void * *mem*, size_t * *lenmem*)

kiss_fft_alloc

Initialize a FFT (or IFFT) algorithm's cfg/state buffer.

typical usage: kiss_fft_cfg mycfg=kiss_fft_alloc(1024,0,NULL,NULL);

The return value from fft_alloc is a cfg buffer used internally by the fft routine or NULL.

If lenmem is NULL, then kiss_fft_alloc will allocate a cfg buffer using malloc. The returned value should be free()d when done to avoid memory leaks.

The state can be placed in a user supplied buffer 'mem': If lenmem is not NULL and mem is not NULL and *lenmem is large enough, then the function places the cfg in mem and the size used in *lenmem and returns mem.

If lenmem is not NULL and (mem is NULL or *lenmem is not large enough), then the function returns NULL and places the minimum cfg buffer size in *lenmem.

Definition at line 615 of file kiss_fft.c.

References kiss_fft_state::bitrev, DIV32, kiss_fft_state::factors, kf_cexp, kf_cexp2, KISS_FFT_MALLOC, kiss_fft_state::nfft, kiss_fft_state::scale, SHL32, and kiss_fft_state::twiddles.

6.30.1.5 void kiss_fft_stride (kiss_fft_cfg *cfg*, const kiss_fft_cpx * *fin*, kiss_fft_cpx * *fout*, int *fin_stride*)

A more generic version of the above function. It reads its input from every Nth sample.

Definition at line 658 of file kiss_fft.c.

References kiss_fft_state::bitrev, celt_fatal, kiss_fft_state::factors, kiss_fft_cpx::i, kf_work, kiss_fft_state::nfft, kiss_fft_cpx::r, and kiss_fft_state::scale.

6.30.1.6 void kiss_ifft (kiss_fft_cfg *cfg*, const kiss_fft_cpx * *fin*, kiss_fft_cpx * *fout*)

Definition at line 697 of file kiss_fft.c.

References kiss_ifft_stride.

6.30.1.7 void kiss_ifft_stride (kiss_fft_cfg *st*, const kiss_fft_cpx * *fin*, kiss_fft_cpx * *fout*, int *in_stride*)

Definition at line 683 of file kiss_fft.c.

References kiss_fft_state::bitrev, celt_fatal, kiss_fft_state::factors, ki_work, and kiss_fft_state::nfft.

6.31 libcelt/kiss_fft.h File Reference

```
#include <stdlib.h>
#include <math.h>
#include "arch.h"
```

Data Structures

- struct [kiss_fft_cpx](#)
- struct [kiss_twiddle_cpx](#)

Defines

- #define [KISS_FFT_MALLOC](#) celt_alloc
- #define [kiss_fft_scalar](#) float
- #define [kiss_twiddle_scalar](#) float
- #define [KF_SUFFIX](#) _celt_single
- #define [CAT_SUFFIX](#)(a, b) a ## b
- #define [SUF](#)(a, b) [CAT_SUFFIX](#)(a, b)
- #define [kiss_fft_alloc](#) [SUF](#)(kiss_fft_alloc, [KF_SUFFIX](#))
- #define [kf_work](#) [SUF](#)(kf_work, [KF_SUFFIX](#))
- #define [ki_work](#) [SUF](#)(ki_work, [KF_SUFFIX](#))
- #define [kiss_fft](#) [SUF](#)(kiss_fft, [KF_SUFFIX](#))
- #define [kiss_ifft](#) [SUF](#)(kiss_ifft, [KF_SUFFIX](#))
- #define [kiss_fft_stride](#) [SUF](#)(kiss_fft_stride, [KF_SUFFIX](#))
- #define [kiss_ifft_stride](#) [SUF](#)(kiss_ifft_stride, [KF_SUFFIX](#))
- #define [kiss_fft_free](#) celt_free

Typedefs

- typedef struct [kiss_fft_state](#) * [kiss_fft_cfg](#)

Functions

- [kiss_fft_cfg](#) [kiss_fft_alloc](#) (int nfft, void *mem, size_t *lenmem)
- void [kf_work](#) ([kiss_fft_cpx](#) *Fout, const [kiss_fft_cpx](#) *f, const size_t fstride, int in_stride, int *factors, const [kiss_fft_cfg](#) st, int N, int s2, int m2)
- void [ki_work](#) ([kiss_fft_cpx](#) *Fout, const [kiss_fft_cpx](#) *f, const size_t fstride, int in_stride, int *factors, const [kiss_fft_cfg](#) st, int N, int s2, int m2)
- void [kiss_fft](#) ([kiss_fft_cfg](#) cfg, const [kiss_fft_cpx](#) *fin, [kiss_fft_cpx](#) *fout)
- void [kiss_ifft](#) ([kiss_fft_cfg](#) cfg, const [kiss_fft_cpx](#) *fin, [kiss_fft_cpx](#) *fout)
- void [kiss_fft_stride](#) ([kiss_fft_cfg](#) cfg, const [kiss_fft_cpx](#) *fin, [kiss_fft_cpx](#) *fout, int fin_stride)
- void [kiss_ifft_stride](#) ([kiss_fft_cfg](#) cfg, const [kiss_fft_cpx](#) *fin, [kiss_fft_cpx](#) *fout, int fin_stride)

6.31.1 Define Documentation

6.31.1.1 #define [CAT_SUFFIX](#)(a, b) a ## b

Definition at line 75 of file kiss_fft.h.

6.31.1.2 #define KF_SUFFIX_celt_single

Definition at line 68 of file kiss_fft.h.

6.31.1.3 #define kf_work SUF(kf_work,KF_SUFFIX)

Definition at line 79 of file kiss_fft.h.

Referenced by kf_work(), kiss_fft_stride(), and kiss_fftr_inplace().

6.31.1.4 #define ki_work SUF(ki_work,KF_SUFFIX)

Definition at line 80 of file kiss_fft.h.

Referenced by ki_work(), kiss_fftri(), and kiss_ifft_stride().

6.31.1.5 #define kiss_fft SUF(kiss_fft,KF_SUFFIX)

Definition at line 81 of file kiss_fft.h.

Referenced by kiss_fftr().

6.31.1.6 #define kiss_fft_alloc SUF(kiss_fft_alloc,KF_SUFFIX)

Definition at line 78 of file kiss_fft.h.

Referenced by kiss_fftr_alloc().

6.31.1.7 #define kiss_fft_free celt_free

If kiss_fft_alloc allocated a buffer, it is one contiguous buffer and can be simply free()d when no longer needed

Definition at line 152 of file kiss_fft.h.

6.31.1.8 #define KISS_FFT_MALLOC celt_alloc

Definition at line 48 of file kiss_fft.h.

Referenced by kiss_fft_alloc(), and kiss_fftr_alloc().

6.31.1.9 #define kiss_fft_scalar float

Definition at line 66 of file kiss_fft.h.

Referenced by mdct_backward(), and mdct_forward().

6.31.1.10 #define kiss_fft_stride SUF(kiss_fft_stride,KF_SUFFIX)

Definition at line 83 of file kiss_fft.h.

Referenced by kiss_fft().

6.31.1.11 #define kiss_iftt SUF(kiss_iftt,KF_SUFFIX)

Definition at line 82 of file kiss_fft.h.

6.31.1.12 #define kiss_iftt_stride SUF(kiss_iftt_stride,KF_SUFFIX)

Definition at line 84 of file kiss_fft.h.

Referenced by kiss_iftt().

6.31.1.13 #define kiss_twiddle_scalar float

Definition at line 67 of file kiss_fft.h.

Referenced by mdct_init().

6.31.1.14 #define SUF(a, b) CAT_SUFFIX(a, b)

Definition at line 76 of file kiss_fft.h.

6.31.2 Typedef Documentation**6.31.2.1 typedef struct kiss_fft_state* kiss_fft_cfg**

Definition at line 97 of file kiss_fft.h.

6.31.3 Function Documentation**6.31.3.1 void kf_work (kiss_fft_cpx * *Fout*, const kiss_fft_cpx * *f*, const size_t *fstride*, int *in_stride*, int * *factors*, const kiss_fft_cfg *st*, int *N*, int *s2*, int *m2*)**

Definition at line 512 of file kiss_fft.c.

References celt_fatal, and kf_work.

6.31.3.2 void ki_work (kiss_fft_cpx * *Fout*, const kiss_fft_cpx * *f*, const size_t *fstride*, int *in_stride*, int * *factors*, const kiss_fft_cfg *st*, int *N*, int *s2*, int *m2*)

Internal function. Can be useful when you want to do the bit-reversing yourself

Definition at line 548 of file kiss_fft.c.

References celt_fatal, and ki_work.

6.31.3.3 void kiss_fft (kiss_fft_cfg *cfg*, const kiss_fft_cpx * *fin*, kiss_fft_cpx * *fout*)

[kiss_fft\(cfg,in_out_buf\)](#)

Perform an FFT on a complex input buffer. for a forward FFT, fin should be f[0] , f[1] , ... ,f[nfft-1] fout will be F[0] , F[1] , ... ,F[nfft-1] Note that each element is complex and can be accessed like f[k].r and f[k].i

Definition at line 678 of file kiss_fft.c.

References `kiss_fft_stride`.

6.31.3.4 `kiss_fft_cfg kiss_fft_alloc (int nfft, void * mem, size_t * lenmem)`

`kiss_fft_alloc`

Initialize a FFT (or IFFT) algorithm's cfg/state buffer.

typical usage: `kiss_fft_cfg mycfg=kiss_fft_alloc(1024,0,NULL,NULL);`

The return value from `fft_alloc` is a cfg buffer used internally by the `fft` routine or `NULL`.

If `lenmem` is `NULL`, then `kiss_fft_alloc` will allocate a cfg buffer using `malloc`. The returned value should be `free()`d when done to avoid memory leaks.

The state can be placed in a user supplied buffer '`mem`': If `lenmem` is not `NULL` and `mem` is not `NULL` and `*lenmem` is large enough, then the function places the cfg in `mem` and the size used in `*lenmem` and returns `mem`.

If `lenmem` is not `NULL` and (`mem` is `NULL` or `*lenmem` is not large enough), then the function returns `NULL` and places the minimum cfg buffer size in `*lenmem`.

Definition at line 615 of file `kiss_fft.c`.

References `kiss_fft_state::bitrev`, `DIV32`, `kiss_fft_state::factors`, `kf_cexp`, `kf_cexp2`, `KISS_FFT_MALLOC`, `kiss_fft_state::nfft`, `kiss_fft_state::scale`, `SHL32`, and `kiss_fft_state::twiddles`.

6.31.3.5 `void kiss_fft_stride (kiss_fft_cfg cfg, const kiss_fft_cpx * fin, kiss_fft_cpx * fout, int fn_stride)`

A more generic version of the above function. It reads its input from every Nth sample.

Definition at line 658 of file `kiss_fft.c`.

References `kiss_fft_state::bitrev`, `celt_fatal`, `kiss_fft_state::factors`, `kiss_fft_cpx::i`, `kf_work`, `kiss_fft_state::nfft`, `kiss_fft_cpx::r`, and `kiss_fft_state::scale`.

6.31.3.6 `void kiss_ifft (kiss_fft_cfg cfg, const kiss_fft_cpx * fin, kiss_fft_cpx * fout)`

Definition at line 697 of file `kiss_fft.c`.

References `kiss_ifft_stride`.

6.31.3.7 `void kiss_ifft_stride (kiss_fft_cfg cfg, const kiss_fft_cpx * fin, kiss_fft_cpx * fout, int fn_stride)`

Definition at line 683 of file `kiss_fft.c`.

References `kiss_fft_state::bitrev`, `celt_fatal`, `kiss_fft_state::factors`, `ki_work`, and `kiss_fft_state::nfft`.

6.32 libcelt/kiss_fftr.c File Reference

```
#include "os_support.h"
#include "mathops.h"
#include "kiss_fftr.h"
#include "_kiss_fft_guts.h"
```

Functions

- `kiss_fftr_cfg kiss_fftr_alloc` (int *nfft*, void **mem*, size_t **lenmem*)
- void `kiss_fftr_twiddles` (`kiss_fftr_cfg` *st*, `kiss_fft_scalar` **freqdata*)
- void `kiss_fftr` (`kiss_fftr_cfg` *st*, const `kiss_fft_scalar` **timedata*, `kiss_fft_scalar` **freqdata*)
- void `kiss_fftr_inplace` (`kiss_fftr_cfg` *st*, `kiss_fft_scalar` **X*)
- void `kiss_fftri` (`kiss_fftr_cfg` *st*, const `kiss_fft_scalar` **freqdata*, `kiss_fft_scalar` **timedata*)

6.32.1 Function Documentation

6.32.1.1 void kiss_fftr (kiss_fftr_cfg st, const kiss_fft_scalar * timedata, kiss_fft_scalar * freqdata)

Definition at line 123 of file `kiss_fftr.c`.

References `kiss_fft`, `kiss_fftr_twiddles`, and `kiss_fftr_state::substate`.

6.32.1.2 kiss_fftr_cfg kiss_fftr_alloc (int nfft, void * mem, size_t * lenmem)

Definition at line 31 of file `kiss_fftr.c`.

References `DIV32`, `kf_cexp`, `kf_cexp2`, `kiss_fft_alloc`, `KISS_FFT_MALLOC`, `kiss_fft_state::scale`, `SHL32`, `kiss_fftr_state::substate`, and `kiss_fftr_state::super_twiddles`.

6.32.1.3 void kiss_fftr_inplace (kiss_fftr_cfg st, kiss_fft_scalar * X)

Definition at line 131 of file `kiss_fftr.c`.

References `kiss_fft_state::factors`, `kf_work`, `kiss_fftr_twiddles`, and `kiss_fftr_state::substate`.

6.32.1.4 void kiss_fftr_twiddles (kiss_fftr_cfg st, kiss_fft_scalar * freqdata)

Definition at line 79 of file `kiss_fftr.c`.

References `ADD32`, `C_FIXDIV`, `C_MULC`, `CHECK_OVERFLOW_OP`, `EXT32`, `HALF_OF`, `kiss_fft_cpx::i`, `kiss_fft_state::nfft`, `PSHR32`, `kiss_fft_cpx::r`, `SHR32`, `SUB32`, `kiss_fftr_state::substate`, and `kiss_fftr_state::super_twiddles`.

6.32.1.5 void kiss_fftri (kiss_fftr_cfg st, const kiss_fft_scalar * freqdata, kiss_fft_scalar * timedata)

Definition at line 137 of file `kiss_fftr.c`.

References `kiss_fft_state::bitrev`, `C_ADD`, `C_MUL`, `C_SUB`, `kiss_fft_state::factors`, `kiss_fft_cpx::i`, `ki_work`, `kiss_fft_state::nfft`, `kiss_fft_cpx::r`, `kiss_fftr_state::substate`, and `kiss_fftr_state::super_twiddles`.

6.33 libcelt/kiss_fftr.h File Reference

```
#include "kiss_fft.h"
```

Data Structures

- struct [kiss_fftr_state](#)

Defines

- #define [kiss_fftr_alloc](#) SUF(kiss_fftr_alloc,KF_SUFFIX)
- #define [kiss_fftr_inplace](#) SUF(kiss_fftr_inplace,KF_SUFFIX)
- #define [kiss_fftr_alloc](#) SUF(kiss_fftr_alloc,KF_SUFFIX)
- #define [kiss_fftr_twiddles](#) SUF(kiss_fftr_twiddles,KF_SUFFIX)
- #define [kiss_fftr](#) SUF(kiss_fftr,KF_SUFFIX)
- #define [kiss_fftri](#) SUF(kiss_fftri,KF_SUFFIX)
- #define [kiss_fftr_free](#) speex_free

Typedefs

- typedef struct [kiss_fftr_state](#) * [kiss_fftr_cfg](#)

Functions

- [kiss_fftr_cfg](#) [kiss_fftr_alloc](#) (int nfft, void *mem, size_t *lenmem)
- void [kiss_fftr_twiddles](#) ([kiss_fftr_cfg](#) st, kiss_fft_scalar *freqdata)
- void [kiss_fftr](#) ([kiss_fftr_cfg](#) st, const kiss_fft_scalar *timedata, kiss_fft_scalar *freqdata)
- void [kiss_fftr_inplace](#) ([kiss_fftr_cfg](#) st, kiss_fft_scalar *X)
- void [kiss_fftri](#) ([kiss_fftr_cfg](#) st, const kiss_fft_scalar *freqdata, kiss_fft_scalar *timedata)

6.33.1 Define Documentation

6.33.1.1 #define [kiss_fftr](#) SUF(kiss_fftr,KF_SUFFIX)

Definition at line 31 of file [kiss_fftr.h](#).

6.33.1.2 #define [kiss_fftr_alloc](#) SUF(kiss_fftr_alloc,KF_SUFFIX)

Definition at line 29 of file [kiss_fftr.h](#).

6.33.1.3 #define [kiss_fftr_alloc](#) SUF(kiss_fftr_alloc,KF_SUFFIX)

Definition at line 29 of file [kiss_fftr.h](#).

6.33.1.4 #define [kiss_fftr_free](#) [speex_free](#)

Definition at line 77 of file [kiss_fftr.h](#).

6.33.1.5 #define kiss_fftr_inplace *SUF(kiss_fftr_inplace, KF_SUFFIX)*

Definition at line 28 of file kiss_fftr.h.

6.33.1.6 #define kiss_fftr_twiddles *SUF(kiss_fftr_twiddles, KF_SUFFIX)*

Definition at line 30 of file kiss_fftr.h.

Referenced by kiss_fftr(), and kiss_fftr_inplace().

6.33.1.7 #define kiss_fftri *SUF(kiss_fftri, KF_SUFFIX)*

Definition at line 32 of file kiss_fftr.h.

6.33.2 Typedef Documentation**6.33.2.1 typedef struct kiss_fftr_state* kiss_fftr_cfg**

Definition at line 50 of file kiss_fftr.h.

6.33.3 Function Documentation**6.33.3.1 void kiss_fftr** (*kiss_fftr_cfg st, const kiss_fft_scalar * timedata, kiss_fft_scalar * freqdata*)

Definition at line 123 of file kiss_fftr.c.

References kiss_fft, kiss_fftr_twiddles, and kiss_fftr_state::substate.

6.33.3.2 kiss_fftr_cfg kiss_fftr_alloc (*int nfft, void * mem, size_t * lenmem*)

Definition at line 31 of file kiss_fftr.c.

References DIV32, kf_cexp, kf_cexp2, kiss_fft_alloc, KISS_FFT_MALLOC, kiss_fft_state::scale, SHL32, kiss_fftr_state::substate, and kiss_fftr_state::super_twiddles.

6.33.3.3 void kiss_fftr_inplace (*kiss_fftr_cfg st, kiss_fft_scalar * X*)

Definition at line 131 of file kiss_fftr.c.

References kiss_fft_state::factors, kf_work, kiss_fftr_twiddles, and kiss_fftr_state::substate.

6.33.3.4 void kiss_fftr_twiddles (*kiss_fftr_cfg st, kiss_fft_scalar * freqdata*)

Definition at line 79 of file kiss_fftr.c.

References ADD32, C_FIXDIV, C_MULC, CHECK_OVERFLOW_OP, EXT32, HALF_OF, kiss_fft_cpx::i, kiss_fft_state::nfft, PSHR32, kiss_fft_cpx::r, SHR32, SUB32, kiss_fftr_state::substate, and kiss_fftr_state::super_twiddles.

6.33.3.5 void kiss_fftri (kiss_fftr_cfg st, const kiss_fft_scalar *freqdata, kiss_fft_scalar *timedata)

Definition at line 137 of file kiss_fftr.c.

References kiss_fft_state::bitrev, C_ADD, C_MUL, C_SUB, kiss_fft_state::factors, kiss_fft_cpx::i, ki_work, kiss_fft_state::nfft, kiss_fft_cpx::r, kiss_fftr_state::substate, and kiss_fftr_state::super_twiddles.

6.34 libcelt/laplace.c File Reference

```
#include "laplace.h"
```

Functions

- int [ec_laplace_get_start_freq](#) (int decay)
- void [ec_laplace_encode_start](#) (ec_enc *enc, int *value, int decay, int fs)
- void [ec_laplace_encode](#) (ec_enc *enc, int *value, int decay)
- int [ec_laplace_decode_start](#) (ec_dec *dec, int decay, int fs)
- int [ec_laplace_decode](#) (ec_dec *dec, int decay)

6.34.1 Function Documentation

6.34.1.1 int [ec_laplace_decode](#) (ec_dec * *dec*, int *decay*)

Decode a value that is assumed to be the realisation of a Laplace-distributed random process

Parameters:

dec Entropy decoder state

decay Probability of the value +/- 1, multiplied by 16384

Returns:

Value decoded

Definition at line 136 of file laplace.c.

References [ec_laplace_decode_start](#)(), and [ec_laplace_get_start_freq](#)().

6.34.1.2 int [ec_laplace_decode_start](#) (ec_dec * *dec*, int *decay*, int *fs*)

Definition at line 99 of file laplace.c.

References [ec_dec_update](#)(), and [ec_decode_bin](#)().

Referenced by [ec_laplace_decode](#)(), and [unquant_coarse_energy](#)().

6.34.1.3 void [ec_laplace_encode](#) (ec_enc * *enc*, int * *value*, int *decay*)

Encode a value that is assumed to be the realisation of a Laplace-distributed random process

Parameters:

enc Entropy encoder state

value Value to encode

decay Probability of the value +/- 1, multiplied by 16384

Definition at line 92 of file laplace.c.

References [ec_laplace_encode_start](#)(), and [ec_laplace_get_start_freq](#)().

6.34.1.4 void ec_laplace_encode_start (ec_enc * enc, int * value, int decay, int fs)

Definition at line 47 of file laplace.c.

References ec_encode_bin().

Referenced by ec_laplace_encode(), and quant_coarse_energy().

6.34.1.5 int ec_laplace_get_start_freq (int decay)

Definition at line 39 of file laplace.c.

Referenced by ec_laplace_decode(), ec_laplace_encode(), and quant_prob_alloc().

6.35 libcelt/laplace.h File Reference

```
#include "entenc.h"  
#include "entdec.h"
```

Functions

- int [ec_laplace_get_start_freq](#) (int decay)
- void [ec_laplace_encode](#) (ec_enc *enc, int *value, int decay)
- void [ec_laplace_encode_start](#) (ec_enc *enc, int *value, int decay, int fs)
- int [ec_laplace_decode](#) (ec_dec *dec, int decay)
- int [ec_laplace_decode_start](#) (ec_dec *dec, int decay, int fs)

6.35.1 Function Documentation

6.35.1.1 int [ec_laplace_decode](#) (ec_dec *dec, int decay)

Decode a value that is assumed to be the realisation of a Laplace-distributed random process

Parameters:

- dec* Entropy decoder state
- decay* Probability of the value +/- 1, multiplied by 16384

Returns:

Value decoded

Definition at line 136 of file laplace.c.

References [ec_laplace_decode_start\(\)](#), and [ec_laplace_get_start_freq\(\)](#).

6.35.1.2 int [ec_laplace_decode_start](#) (ec_dec *dec, int decay, int fs)

Definition at line 99 of file laplace.c.

References [ec_dec_update\(\)](#), and [ec_decode_bin\(\)](#).

Referenced by [ec_laplace_decode\(\)](#), and [unquant_coarse_energy\(\)](#).

6.35.1.3 void [ec_laplace_encode](#) (ec_enc *enc, int *value, int decay)

Encode a value that is assumed to be the realisation of a Laplace-distributed random process

Parameters:

- enc* Entropy encoder state
- value* Value to encode
- decay* Probability of the value +/- 1, multiplied by 16384

Definition at line 92 of file laplace.c.

References [ec_laplace_encode_start\(\)](#), and [ec_laplace_get_start_freq\(\)](#).

6.35.1.4 void ec_laplace_encode_start (ec_enc * enc, int * value, int decay, int fs)

Definition at line 47 of file laplace.c.

References ec_encode_bin().

Referenced by ec_laplace_encode(), and quant_coarse_energy().

6.35.1.5 int ec_laplace_get_start_freq (int decay)

Definition at line 39 of file laplace.c.

Referenced by ec_laplace_decode(), ec_laplace_encode(), and quant_prob_alloc().

6.36 libcelt/mathops.h File Reference

Various math functions. #include "arch.h"

```
#include "entcode.h"
```

```
#include "os_support.h"
```

Defines

- #define `FRAC_MUL16(a, b)` $((16384+((\text{celt_int32})(\text{celt_int16})(a)*(\text{celt_int16})(b)))>>15)$
- #define `celt_sqrt(x)` $((\text{float})\text{sqrt}(x))$
- #define `celt_psqrt(x)` $((\text{float})\text{sqrt}(x))$
- #define `celt_rsqrt(x)` $(1./\text{celt_sqrt}(x))$
- #define `celt_acos` `acos`
- #define `celt_exp` `exp`
- #define `celt_cos_norm(x)` $(\cos((.5f*M_PI)*(x)))$
- #define `celt_atan` `atan`
- #define `celt_rcp(x)` $(1./(\text{float})(x))$
- #define `celt_div(a, b)` $((a)/(\text{float})(b))$
- #define `celt_log2(x)` $(1.442695040888963387*\log(x))$
- #define `celt_exp2(x)` $(\exp(0.6931471805599453094*(\text{float})(x)))$

6.36.1 Detailed Description

Various math functions.

Definition in file [mathops.h](#).

6.36.2 Define Documentation

6.36.2.1 #define `celt_acos` `acos`

Definition at line 109 of file [mathops.h](#).

6.36.2.2 #define `celt_atan` `atan`

Definition at line 112 of file [mathops.h](#).

6.36.2.3 #define `celt_cos_norm(x)` $(\cos((.5f*M_PI)*(x)))$

Definition at line 111 of file [mathops.h](#).

Referenced by `mdct_init()`.

6.36.2.4 #define `celt_div(a, b)` $((a)/(\text{float})(b))$

Definition at line 114 of file [mathops.h](#).

6.36.2.5 #define celt_exp exp

Definition at line 110 of file mathops.h.

6.36.2.6 #define celt_exp2(x) (exp(0.6931471805599453094*(x)))

Definition at line 161 of file mathops.h.

6.36.2.7 #define celt_log2(x) (1.442695040888963387*log(x))

Definition at line 160 of file mathops.h.

6.36.2.8 #define celt_psqrt(x) ((float)sqrt(x))

Definition at line 107 of file mathops.h.

6.36.2.9 #define celt_rcp(x) (1.f/(x))

Definition at line 113 of file mathops.h.

Referenced by `alg_quant()`, `celt_encode_float()`, and `renormalise_vector()`.

6.36.2.10 #define celt_rsqrt(x) (1.f/celt_sqrt(x))

Definition at line 108 of file mathops.h.

Referenced by `find_spectral_pitch()`.

6.36.2.11 #define celt_sqrt(x) ((float)sqrt(x))

Definition at line 106 of file mathops.h.

Referenced by `compute_pitch_gain()`, `folding_decision()`, `quant_bands()`, `quant_bands_stereo()`, `renormalise_vector()`, and `unquant_bands_stereo()`.

6.36.2.12 #define FRAC_MUL16(a, b) ((16384+((celt_int32)(celt_int16)(a)*(celt_int16)(b)))>>15)

Definition at line 88 of file mathops.h.

6.37 libcelt/mdct.c File Reference

```
#include "mdct.h"
#include "kfft_double.h"
#include <math.h>
#include "os_support.h"
#include "mathops.h"
#include "stack_alloc.h"
```

Defines

- #define `M_PI` 3.141592653

Functions

- void `mdct_init` (`mdct_lookup` **l*, int *N*)
- void `mdct_clear` (`mdct_lookup` **l*)
- void `mdct_forward` (const `mdct_lookup` **l*, `kiss_fft_scalar` **in*, `kiss_fft_scalar` **restrict out*, const `celt_word16` **window*, int *overlap*)
- void `mdct_backward` (const `mdct_lookup` **l*, `kiss_fft_scalar` **in*, `kiss_fft_scalar` **restrict out*, const `celt_word16` **restrict window*, int *overlap*)

6.37.1 Define Documentation

6.37.1.1 #define `M_PI` 3.141592653

Definition at line 58 of file `mdct.c`.

Referenced by `celt_mode_create()`, and `mdct_init()`.

6.37.2 Function Documentation

6.37.2.1 void `mdct_backward` (const `mdct_lookup` * *l*, `kiss_fft_scalar` * *in*, `kiss_fft_scalar` **restrict out*, const `celt_word16` **restrict window*, int *overlap*)

Definition at line 187 of file `mdct.c`.

References `ALLOC`, `cpx32_ifft`, `mdct_lookup::kfft`, `kiss_fft_scalar`, `MULT16_32_Q15`, `mdct_lookup::n`, `RESTORE_STACK`, `S_MUL`, `SAVE_STACK`, `mdct_lookup::trig`, and `VARDECL`.

6.37.2.2 void `mdct_clear` (`mdct_lookup` * *l*)

Definition at line 90 of file `mdct.c`.

References `cpx32_fft_free`, `mdct_lookup::kfft`, and `mdct_lookup::trig`.

Referenced by `celt_mode_destroy()`.

6.37.2.3 void mdct_forward (const mdct_lookup * *l*, kiss_fft_scalar * *in*, kiss_fft_scalar *restrict *out*, const celt_word16 * *window*, int *overlap*)

Definition at line 96 of file mdct.c.

References ALLOC, cpx32_fft, mdct_lookup::kfft, kiss_fft_scalar, MULT16_32_Q15, mdct_lookup::n, RESTORE_STACK, S_MUL, SAVE_STACK, mdct_lookup::trig, and VARDECL.

6.37.2.4 void mdct_init (mdct_lookup * *l*, int *N*)

Definition at line 61 of file mdct.c.

References ADD32, celt_cos_norm, cpx32_fft_alloc, DIV32, EXTEND32, mdct_lookup::kfft, kiss_twiddle_scalar, M_PI, mdct_lookup::n, SHL32, and mdct_lookup::trig.

Referenced by celt_mode_create().

6.38 libcelt/mdct.h File Reference

```
#include "kiss_fft.h"
#include "arch.h"
```

Data Structures

- struct [mdct_lookup](#)

Functions

- void [mdct_init](#) ([mdct_lookup](#) *l, int N)
- void [mdct_clear](#) ([mdct_lookup](#) *l)
- void [mdct_forward](#) (const [mdct_lookup](#) *l, [kiss_fft_scalar](#) *in, [kiss_fft_scalar](#) *out, const [celt_word16](#) *window, int overlap)
- void [mdct_backward](#) (const [mdct_lookup](#) *l, [kiss_fft_scalar](#) *in, [kiss_fft_scalar](#) *out, const [celt_word16](#) *restrict window, int overlap)

6.38.1 Function Documentation

6.38.1.1 void [mdct_backward](#) (const [mdct_lookup](#) *l, [kiss_fft_scalar](#) *in, [kiss_fft_scalar](#) *out, const [celt_word16](#) *restrict window, int overlap)

Compute a backward MDCT (no scaling) and performs weighted overlap-add (scales implicitly by 1/2)

6.38.1.2 void [mdct_clear](#) ([mdct_lookup](#) *l)

Definition at line 90 of file [mdct.c](#).

References [cpx32_fft_free](#), [mdct_lookup::kfft](#), and [mdct_lookup::trig](#).

Referenced by [celt_mode_destroy](#)().

6.38.1.3 void [mdct_forward](#) (const [mdct_lookup](#) *l, [kiss_fft_scalar](#) *in, [kiss_fft_scalar](#) *out, const [celt_word16](#) *window, int overlap)

Compute a forward MDCT and scale by 4/N

6.38.1.4 void [mdct_init](#) ([mdct_lookup](#) *l, int N)

Definition at line 61 of file [mdct.c](#).

References [ADD32](#), [celt_cos_norm](#), [cpx32_fft_alloc](#), [DIV32](#), [EXTEND32](#), [mdct_lookup::kfft](#), [kiss_twiddle_scalar](#), [M_PI](#), [mdct_lookup::n](#), [SHL32](#), and [mdct_lookup::trig](#).

Referenced by [celt_mode_create](#)().

6.39 libcelt/mfrngcod.h File Reference

```
#include "entcode.h"
```

Defines

- `#define _mfrngcode_H (1)`
- `#define EC_SYM_BITS (8)`
- `#define EC_CODE_BITS (32)`
- `#define EC_SYM_MAX ((1U<<EC_SYM_BITS)-1)`
- `#define EC_CODE_SHIFT (EC_CODE_BITS-EC_SYM_BITS-1)`
- `#define EC_CODE_TOP (((ec_uint32)1U)<<EC_CODE_BITS-1)`
- `#define EC_CODE_BOT (EC_CODE_TOP>>EC_SYM_BITS)`
- `#define EC_CODE_CARRY (((ec_uint32)EC_SYM_MAX)<<EC_CODE_SHIFT)`
- `#define EC_CODE_EXTRA ((EC_CODE_BITS-2)%EC_SYM_BITS+1)`
- `#define EC_CODE_MASK (((ec_uint32)1U)<<EC_CODE_BITS-1)-1<<1|1)`

6.39.1 Define Documentation

6.39.1.1 `#define _mfrngcode_H (1)`

Definition at line 33 of file mfrngcod.h.

6.39.1.2 `#define EC_CODE_BITS (32)`

Definition at line 41 of file mfrngcod.h.

Referenced by `ec_dec_tell()`, `ec_enc_done()`, and `ec_enc_tell()`.

6.39.1.3 `#define EC_CODE_BOT (EC_CODE_TOP>>EC_SYM_BITS)`

Definition at line 49 of file mfrngcod.h.

6.39.1.4 `#define EC_CODE_CARRY (((ec_uint32)EC_SYM_MAX)<<EC_CODE_SHIFT)`

Definition at line 51 of file mfrngcod.h.

6.39.1.5 `#define EC_CODE_EXTRA ((EC_CODE_BITS-2)%EC_SYM_BITS+1)`

Definition at line 53 of file mfrngcod.h.

Referenced by `ec_dec_init()`.

6.39.1.6 `#define EC_CODE_MASK (((ec_uint32)1U)<<EC_CODE_BITS-1)-1<<1|1)`

Definition at line 58 of file mfrngcod.h.

6.39.1.7 #define EC_CODE_SHIFT (EC_CODE_BITS-EC_SYM_BITS-1)

Definition at line 45 of file mfrngcod.h.

Referenced by ec_enc_done().

6.39.1.8 #define EC_CODE_TOP (((ec_uint32)1U)<<EC_CODE_BITS-1)

Definition at line 47 of file mfrngcod.h.

Referenced by ec_enc_done(), and ec_enc_init().

6.39.1.9 #define EC_SYM_BITS (8)

Definition at line 39 of file mfrngcod.h.

Referenced by ec_dec_init(), and ec_dec_tell().

6.39.1.10 #define EC_SYM_MAX ((1U<<EC_SYM_BITS)-1)

Definition at line 43 of file mfrngcod.h.

6.40 libcelt/mfrngdec.c File Reference

```
#include "arch.h"
#include "entdec.h"
#include "mfrngcod.h"
```

Functions

- void `ec_dec_init` (`ec_dec * _this`, `ec_byte_buffer * _buf`)
- unsigned `ec_decode` (`ec_dec * _this`, unsigned `_ft`)
- unsigned `ec_decode_bin` (`ec_dec * _this`, unsigned `bits`)
- void `ec_dec_update` (`ec_dec * _this`, unsigned `_fl`, unsigned `_fh`, unsigned `_ft`)
- long `ec_dec_tell` (`ec_dec * _this`, int `_b`)

6.40.1 Function Documentation

6.40.1.1 void `ec_dec_init` (`ec_dec * _this`, `ec_byte_buffer * _buf`)

Definition at line 155 of file `mfrngdec.c`.

References `ec_dec::buf`, `ec_dec::dif`, `EC_CODE_EXTRA`, `EC_SYM_BITS`, `ec_dec::end_bits_left`, `ec_dec::nb_end_bits`, `ec_dec::rem`, and `ec_dec::rng`.

6.40.1.2 long `ec_dec_tell` (`ec_dec * _this`, int `_b`)

Definition at line 227 of file `mfrngdec.c`.

References `ec_dec::buf`, `EC_CODE_BITS`, `EC_ILOG`, `EC_SYM_BITS`, `ec_dec::nb_end_bits`, and `ec_dec::rng`.

6.40.1.3 void `ec_dec_update` (`ec_dec * _this`, unsigned `_fl`, unsigned `_fh`, unsigned `_ft`)

Definition at line 206 of file `mfrngdec.c`.

References `ec_dec::dif`, `EC_MINI`, `ec_dec::nrm`, and `ec_dec::rng`.

6.40.1.4 unsigned `ec_decode` (`ec_dec * _this`, unsigned `_ft`)

Definition at line 167 of file `mfrngdec.c`.

References `ec_dec::dif`, `EC_ILOG`, `EC_MAXI`, `ec_dec::nrm`, and `ec_dec::rng`.

6.40.1.5 unsigned `ec_decode_bin` (`ec_dec * _this`, unsigned `bits`)

Definition at line 185 of file `mfrngdec.c`.

References `ec_dec::buf`, `ec_byte_look_at_end()`, `ec_decode()`, `ec_dec::end_bits_left`, `ec_dec::end_byte`, and `ec_dec::nb_end_bits`.

6.41 libcelt/mfrngenc.c File Reference

```
#include "arch.h"
#include "entenc.h"
#include "mfrngcod.h"
```

Functions

- void `ec_enc_init` (`ec_enc *_this`, `ec_byte_buffer *_buf`)
- void `ec_encode` (`ec_enc *_this`, `unsigned _fl`, `unsigned _fh`, `unsigned _ft`)
- void `ec_encode_bin` (`ec_enc *_this`, `unsigned _fl`, `unsigned _fh`, `unsigned bits`)
- long `ec_enc_tell` (`ec_enc *_this`, `int _b`)
- void `ec_enc_done` (`ec_enc *_this`)

6.41.1 Function Documentation

6.41.1.1 void ec_enc_done (ec_enc * *this*)

Definition at line 205 of file mfrngenc.c.

References `ec_enc::buf`, `EC_CODE_BITS`, `EC_CODE_SHIFT`, `EC_CODE_TOP`, `EC_ILOG`, `ec_enc::end_bits_left`, `ec_enc::end_byte`, `ec_byte_buffer::end_ptr`, `ec_enc::ext`, `ec_enc::low`, `ec_byte_buffer::ptr`, `ec_enc::rem`, and `ec_enc::rng`.

6.41.1.2 void ec_enc_init (ec_enc * *this*, ec_byte_buffer * *buf*)

Definition at line 117 of file mfrngenc.c.

References `ec_enc::buf`, `EC_CODE_TOP`, `ec_enc::end_bits_left`, `ec_enc::end_byte`, `ec_enc::ext`, `ec_enc::low`, `ec_enc::nb_end_bits`, `ec_enc::rem`, and `ec_enc::rng`.

6.41.1.3 long ec_enc_tell (ec_enc * *this*, int *b*)

Definition at line 183 of file mfrngenc.c.

References `ec_enc::buf`, `EC_CODE_BITS`, `EC_ILOG`, `ec_enc::ext`, `ec_enc::nb_end_bits`, `ec_enc::rem`, and `ec_enc::rng`.

6.41.1.4 void ec_encode (ec_enc * *this*, unsigned *fl*, unsigned *fh*, unsigned *ft*)

Definition at line 128 of file mfrngenc.c.

References `EC_ILOG`, `EC_MINI`, `ec_enc::low`, and `ec_enc::rng`.

6.41.1.5 void ec_encode_bin (ec_enc * *this*, unsigned *fl*, unsigned *fh*, unsigned *bits*)

Definition at line 164 of file mfrngenc.c.

References `ec_enc::buf`, `ec_byte_write_at_end()`, `ec_encode()`, `ec_enc::end_bits_left`, `ec_enc::end_byte`, and `ec_enc::nb_end_bits`.

6.42 libcelt/modes.c File Reference

```
#include "celt.h"
#include "modes.h"
#include "rate.h"
#include "os_support.h"
#include "stack_alloc.h"
#include "quant_bands.h"
```

Defines

- #define [MODEVALID](#) 0xa110ca7e
- #define [MODEPARTIAL](#) 0x7eca10a1
- #define [MODEFREED](#) 0xb10cf8ee
- #define [BARK_BANDS](#) 25
- #define [BITALLOC_SIZE](#) 12

Functions

- int [celt_mode_info](#) (const [CELTMode](#) *mode, int request, [celt_int32](#) *value)
- [CELTMode](#) * [celt_mode_create](#) ([celt_int32](#) Fs, int frame_size, int *error)
- void [celt_mode_destroy](#) ([CELTMode](#) *mode)
- int [check_mode](#) (const [CELTMode](#) *mode)

6.42.1 Define Documentation

6.42.1.1 #define BARK_BANDS 25

Definition at line 86 of file modes.c.

6.42.1.2 #define BITALLOC_SIZE 12

Definition at line 102 of file modes.c.

6.42.1.3 #define MODEFREED 0xb10cf8ee

Definition at line 51 of file modes.c.

Referenced by [celt_mode_destroy\(\)](#), and [check_mode\(\)](#).

6.42.1.4 #define MODEPARTIAL 0x7eca10a1

Definition at line 50 of file modes.c.

Referenced by [celt_mode_create\(\)](#), and [celt_mode_destroy\(\)](#).

6.42.1.5 #define MODEVALID 0xa110ca7e

Definition at line 49 of file modes.c.

Referenced by `celt_mode_create()`, `celt_mode_destroy()`, and `check_mode()`.

6.42.2 Function Documentation

6.42.2.1 int check_mode (const CELTMode * mode)

Definition at line 457 of file modes.c.

References `CELT_INVALID_MODE`, `CELT_OK`, `CELTMode::marker_end`, `CELTMode::marker_start`, `MODEFREED`, and `MODEVALID`.

Referenced by `celt_decode()`, `celt_decode_float()`, `celt_decoder_create()`, `celt_decoder_ctl()`, `celt_decoder_destroy()`, `celt_encode()`, `celt_encode_float()`, `celt_encoder_create()`, `celt_encoder_ctl()`, `celt_encoder_destroy()`, `celt_header_init()`, and `celt_mode_info()`.

6.43 libcelt/modes.h File Reference

```
#include "celt_types.h"
#include "celt.h"
#include "arch.h"
#include "mdct.h"
#include "psy.h"
#include "pitch.h"
```

Data Structures

- struct [CELTMode](#)
Mode definition.

Defines

- #define [CELT_BITSTREAM_VERSION](#) 0x8000000b
- #define [MAX_PERIOD](#) 1024
- #define [MCHANNELS](#)(mode) ((mode)->nbChannels)
- #define [CHANNELS](#)(_C) (_C)
- #define [MDCT](#)(mode) (&(mode)->mdct)
- #define [OVERLAP](#)(mode) ((mode)->overlap)
- #define [FRAMESIZE](#)(mode) ((mode)->mdctSize)

Functions

- int [check_mode](#) (const [CELTMode](#) *mode)

6.43.1 Define Documentation

6.43.1.1 #define [CELT_BITSTREAM_VERSION](#) 0x8000000b

Definition at line 44 of file modes.h.

Referenced by [celt_mode_info](#)().

6.43.1.2 #define [CHANNELS](#)(_C) (_C)

Definition at line 64 of file modes.h.

Referenced by [apply_pitch](#)(), [celt_decode](#)(), [celt_decode_float](#)(), [celt_decoder_create](#)(), [celt_encode](#)(), [celt_encode_float](#)(), [compute_allocation](#)(), [compute_band_energies](#)(), [compute_pitch_gain](#)(), [denormalise_bands](#)(), [find_spectral_pitch](#)(), [folding_decision](#)(), [normalise_bands](#)(), [quant_coarse_energy](#)(), [quant_energy_finalise](#)(), [quant_fine_energy](#)(), [renormalise_bands](#)(), [unquant_coarse_energy](#)(), [unquant_energy_finalise](#)(), and [unquant_fine_energy](#)().

6.43.1.3 #define FRAMESIZE(mode) ((mode)->mdctSize)

Definition at line 75 of file modes.h.

Referenced by `apply_pitch()`, `compute_band_energies()`, `compute_pitch_gain()`, `denormalise_bands()`, `folding_decision()`, and `normalise_bands()`.

6.43.1.4 #define MAX_PERIOD 1024

Definition at line 50 of file modes.h.

Referenced by `celt_decode_float()`, `celt_decoder_create()`, `celt_encode_float()`, `celt_encoder_create()`, `celt_encoder_ctl()`, `celt_mode_create()`, `dump_modes()`, and `find_spectral_pitch()`.

6.43.1.5 #define MCHANNELS(mode) ((mode)->nbChannels)

Definition at line 56 of file modes.h.

6.43.1.6 #define MDCT(mode) (&(mode)->mdct)

Definition at line 68 of file modes.h.

6.43.1.7 #define OVERLAP(mode) ((mode)->overlap)

Definition at line 71 of file modes.h.

Referenced by `find_spectral_pitch()`.

6.43.2 Function Documentation

6.43.2.1 int check_mode (const CELTMode * mode)

Definition at line 457 of file modes.c.

References `CELT_INVALID_MODE`, `CELT_OK`, `CELTMode::marker_end`, `CELTMode::marker_start`, `MODEFREED`, and `MODEVALID`.

Referenced by `celt_decode()`, `celt_decode_float()`, `celt_decoder_create()`, `celt_decoder_ctl()`, `celt_decoder_destroy()`, `celt_encode()`, `celt_encode_float()`, `celt_encoder_create()`, `celt_encoder_ctl()`, `celt_encoder_destroy()`, `celt_header_init()`, and `celt_mode_info()`.

6.44 libcelt/os_support.h File Reference

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
```

Defines

- #define [CELT_COPY](#)(dst, src, n) (memcpy((dst), (src), (n)*sizeof(*(dst)) + 0*((dst)-(src))))
- #define [CELT_MOVE](#)(dst, src, n) (memmove((dst), (src), (n)*sizeof(*(dst)) + 0*((dst)-(src))))
- #define [CELT_MEMSET](#)(dst, c, n) (memset((dst), (c), (n)*sizeof(*(dst))))

6.44.1 Define Documentation

6.44.1.1 #define CELT_COPY(dst, src, n) (memcpy((dst), (src), (n)*sizeof(*(dst)) + 0*((dst)-(src))))

Copy n bytes of memory from src to dst. The 0* term provides compile-time type checking

Definition at line 92 of file os_support.h.

Referenced by `celt_encode_float()`, `celt_header_from_packet()`, `celt_header_init()`, `celt_header_to_packet()`, and `celt_mode_create()`.

6.44.1.2 #define CELT_MEMSET(dst, c, n) (memset((dst), (c), (n)*sizeof(*(dst))))

Set n bytes of memory to value of c, starting at address s

Definition at line 103 of file os_support.h.

Referenced by `celt_decoder_ctl()`, `celt_encode_float()`, `celt_encoder_ctl()`, `celt_header_from_packet()`, `celt_header_to_packet()`, and `find_spectral_pitch()`.

6.44.1.3 #define CELT_MOVE(dst, src, n) (memmove((dst), (src), (n)*sizeof(*(dst)) + 0*((dst)-(src))))

Copy n bytes of memory from src to dst, allowing overlapping regions. The 0* term provides compile-time type checking

Definition at line 98 of file os_support.h.

Referenced by `celt_decode_float()`, and `celt_encode_float()`.

6.45 libcelt/pitch.c File Reference

```
Pitch analysis. #include "kfft_single.h"
#include "pitch.h"
#include "psy.h"
#include "os_support.h"
#include "mathops.h"
#include "modes.h"
#include "stack_alloc.h"
```

Defines

- #define [normalise16](#)(x, len, val)
- #define [INPUT_SHIFT](#) 15

Functions

- [kiss_fftr_cfg pitch_state_alloc](#) (int max_lag)
- void [pitch_state_free](#) (kiss_fftr_cfg st)
- void [find_spectral_pitch](#) (const [CELTMode](#) *m, [kiss_fftr_cfg](#) fft, const struct [PsyDecay](#) *decay, const [celt_sig](#) *restrict x, const [celt_sig](#) *restrict y, const [celt_word16](#) *restrict window, [celt_word16](#) *restrict spectrum, int len, int max_pitch, int *pitch, int _C)

6.45.1 Detailed Description

Pitch analysis.

Definition in file [pitch.c](#).

6.45.2 Define Documentation

6.45.2.1 #define [INPUT_SHIFT](#) 15

Definition at line 106 of file [pitch.c](#).

Referenced by [find_spectral_pitch](#)().

6.45.2.2 #define [normalise16](#)(x, len, val)

Definition at line 103 of file [pitch.c](#).

Referenced by [find_spectral_pitch](#)().

6.45.3 Function Documentation

6.45.3.1 `void find_spectral_pitch (const CELTMode * m, kiss_fftr_cfg fft, const struct PsyDecay * decay, const celt_sig *restrict x, const celt_sig *restrict y, const celt_word16 *restrict window, celt_word16 *restrict spectrum, int len, int max_pitch, int * pitch, int C)`

Definition at line 108 of file `pitch.c`.

References `ADD16`, `ALLOC`, `BITREV`, `celt_ilog2`, `CELT_MEMSET`, `celt_rsqrt`, `CHANNELS`, `compute_masking()`, `EPSILON`, `EXTRACT16`, `INPUT_SHIFT`, `MAX_PERIOD`, `MULT16_16`, `MULT16_16_16`, `MULT16_16_Q15`, `normalise16`, `OVERLAP`, `real16_fft_inplace`, `real16_ifft`, `RESTORE_STACK`, `SAVE_STACK`, `SHR32`, `SUB16`, and `VARDECL`.

Referenced by `celt_encode_float()`.

6.45.3.2 `kiss_fftr_cfg pitch_state_alloc (int max_lag)`

Definition at line 54 of file `pitch.c`.

References `real16_fft_alloc`.

Referenced by `celt_mode_create()`.

6.45.3.3 `void pitch_state_free (kiss_fftr_cfg st)`

Definition at line 59 of file `pitch.c`.

References `real16_fft_free`.

Referenced by `celt_mode_destroy()`.

6.46 libcelt/pitch.h File Reference

Pitch analysis. `#include "kiss_fftr.h"`

`#include "psy.h"`

`#include "modes.h"`

Functions

- `kiss_fftr_cfg pitch_state_alloc` (int max_lag)
- void `pitch_state_free` (`kiss_fftr_cfg` st)
- void `find_spectral_pitch` (const `CELTMode` *m, `kiss_fftr_cfg` fft, const struct `PsyDecay` *decay, const `celt_sig` *x, const `celt_sig` *y, const `celt_word16` *window, `celt_word16` *restrict X, int len, int max_pitch, int *pitch, int _C)

6.46.1 Detailed Description

Pitch analysis.

Definition in file [pitch.h](#).

6.46.2 Function Documentation

6.46.2.1 void `find_spectral_pitch` (const `CELTMode` * m, `kiss_fftr_cfg` *fft*, const struct `PsyDecay` * *decay*, const `celt_sig` * x, const `celt_sig` * y, const `celt_word16` * *window*, `celt_word16` *restrict X, int *len*, int *max_pitch*, int * *pitch*, int _C)

Find the optimal delay for the pitch prediction. Computation is done in the frequency domain, both to save time and to make it easier to apply psychoacoustic weighting

6.46.2.2 `kiss_fftr_cfg pitch_state_alloc` (int *max_lag*)

Definition at line 54 of file `pitch.c`.

References `real16_fft_alloc`.

Referenced by `celt_mode_create()`.

6.46.2.3 void `pitch_state_free` (`kiss_fftr_cfg` *st*)

Definition at line 59 of file `pitch.c`.

References `real16_fft_free`.

Referenced by `celt_mode_destroy()`.

6.47 libcelt/psy.c File Reference

```
#include "psy.h"
#include <math.h>
#include "os_support.h"
#include "arch.h"
#include "stack_alloc.h"
#include "mathops.h"
```

Defines

- #define [toBARK](#)(n) (13.1f*atan(.00074f*(n))+2.24f*atan((n)*(n)*1.85e-8f)+1e-4f*(n))
- #define [fromBARK](#)(z) (102.f*(z)-2.f*pow(z,2.f)+.4f*pow(z,3.f)+pow(1.46f,z)-1.f)

Functions

- void [psydecay_init](#) (struct [PsyDecay](#) *decay, int len, [celt_int32](#) Fs)
- void [psydecay_clear](#) (struct [PsyDecay](#) *decay)
- void [compute_masking](#) (const struct [PsyDecay](#) *decay, [celt_word16](#) *X, [celt_mask](#) *restrict mask, int len)

6.47.1 Define Documentation

6.47.1.1 #define [fromBARK](#)(z) (102.f*(z)-2.f*pow(z,2.f)+.4f*pow(z,3.f)+pow(1.46f,z)-1.f)

Definition at line 46 of file psy.c.

6.47.1.2 #define [toBARK](#)(n) (13.1f*atan(.00074f*(n))+2.24f*atan((n)*(n)*1.85e-8f)+1e-4f*(n))

Definition at line 45 of file psy.c.

6.47.2 Function Documentation

6.47.2.1 void [compute_masking](#) (const struct [PsyDecay](#) * decay, [celt_word16](#) * X, [celt_mask](#) *restrict mask, int len)

Definition at line 134 of file psy.c.

References [ADD32](#), and [MULT16_16](#).

Referenced by [find_spectral_pitch](#)().

6.47.2.2 void [psydecay_clear](#) (struct [PsyDecay](#) * decay)

Free the memory allocated for the spreading function

Definition at line 78 of file psy.c.

References PsyDecay::decayR.

Referenced by celt_encoder_destroy(), and celt_mode_destroy().

6.47.2.3 void psydecay_init (struct PsyDecay * *decay*, int *len*, celt_int32 *Fs*)

Pre-compute the decay of the psycho-acoustic spreading function

Definition at line 53 of file psy.c.

References PsyDecay::decayR, and Q15ONE.

Referenced by celt_encoder_create(), and celt_mode_create().

6.48 libcelt/psy.h File Reference

```
#include "arch.h"
#include "celt.h"
```

Data Structures

- struct [PsyDecay](#)

Functions

- void [psydecay_init](#) (struct [PsyDecay](#) *decay, int len, [celt_int32](#) Fs)
- void [psydecay_clear](#) (struct [PsyDecay](#) *decay)
- void [compute_masking](#) (const struct [PsyDecay](#) *decay, [celt_word16](#) *X, [celt_mask](#) *mask, int len)
- void [compute_mdct_masking](#) (const struct [PsyDecay](#) *decay, [celt_word32](#) *X, [celt_word16](#) *tonality, [celt_word16](#) *long_window, [celt_mask](#) *mask, int len)
- void [compute_tonality](#) (const [CELTMode](#) *m, [celt_word16](#) *restrict X, [celt_word16](#) *mem, int len, [celt_word16](#) *tonality, int mdct_size)

6.48.1 Function Documentation

6.48.1.1 void [compute_masking](#) (const struct [PsyDecay](#) * decay, [celt_word16](#) * X, [celt_mask](#) * mask, int len)

Compute the masking curve for an input (DFT) spectrum X

6.48.1.2 void [compute_mdct_masking](#) (const struct [PsyDecay](#) * decay, [celt_word32](#) * X, [celt_word16](#) * tonality, [celt_word16](#) * long_window, [celt_mask](#) * mask, int len)

Compute the masking curve for an input (MDCT) spectrum X

6.48.1.3 void [compute_tonality](#) (const [CELTMode](#) * m, [celt_word16](#) *restrict X, [celt_word16](#) * mem, int len, [celt_word16](#) * tonality, int mdct_size)

6.48.1.4 void [psydecay_clear](#) (struct [PsyDecay](#) * decay)

Free the memory allocated for the spreading function

Definition at line 78 of file psy.c.

References [PsyDecay::decayR](#).

Referenced by [celt_encoder_destroy\(\)](#), and [celt_mode_destroy\(\)](#).

6.48.1.5 void [psydecay_init](#) (struct [PsyDecay](#) * decay, int len, [celt_int32](#) Fs)

Pre-compute the decay of the psycho-acoustic spreading function

Definition at line 53 of file psy.c.

References [PsyDecay::decayR](#), and [Q15ONE](#).

Referenced by `celt_encoder_create()`, and `celt_mode_create()`.

6.49 libcelt/quant_bands.c File Reference

```
#include "quant_bands.h"
#include "laplace.h"
#include <math.h>
#include "os_support.h"
#include "arch.h"
#include "mathops.h"
#include "stack_alloc.h"
```

Functions

- int `intra_decision` (`celt_word16 *eBands`, `celt_word16 *oldEBands`, int `len`)
- int * `quant_prob_alloc` (const `CELTMode *m`)
- void `quant_prob_free` (int *`freq`)
- unsigned `quant_coarse_energy` (const `CELTMode *m`, `celt_word16 *eBands`, `celt_word16 *oldEBands`, int `budget`, int `intra`, int *`prob`, `celt_word16 *error`, `ec_enc *enc`, int `_C`)
- void `quant_fine_energy` (const `CELTMode *m`, `celt_ener *eBands`, `celt_word16 *oldEBands`, `celt_word16 *error`, int *`fine_quant`, `ec_enc *enc`, int `_C`)
- void `quant_energy_finalise` (const `CELTMode *m`, `celt_ener *eBands`, `celt_word16 *oldEBands`, `celt_word16 *error`, int *`fine_quant`, int *`fine_priority`, int `bits_left`, `ec_enc *enc`, int `_C`)
- void `unquant_coarse_energy` (const `CELTMode *m`, `celt_ener *eBands`, `celt_word16 *oldEBands`, int `budget`, int `intra`, int *`prob`, `ec_dec *dec`, int `_C`)
- void `unquant_fine_energy` (const `CELTMode *m`, `celt_ener *eBands`, `celt_word16 *oldEBands`, int *`fine_quant`, `ec_dec *dec`, int `_C`)
- void `unquant_energy_finalise` (const `CELTMode *m`, `celt_ener *eBands`, `celt_word16 *oldEBands`, int *`fine_quant`, int *`fine_priority`, int `bits_left`, `ec_dec *dec`, int `_C`)

Variables

- const `celt_word16 eMeans` [24] = {7.5f, -1.33f, -2.f, -0.42f, 0.17f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f}

6.49.1 Function Documentation

6.49.1.1 int `intra_decision` (`celt_word16 *eBands`, `celt_word16 *oldEBands`, int `len`)

Definition at line 52 of file `quant_bands.c`.

References `MAC16_16`, `SHR32`, and `SUB16`.

Referenced by `celt_encode_float()`.

6.49.1.2 unsigned `quant_coarse_energy` (const `CELTMode *m`, `celt_word16 *eBands`, `celt_word16 *oldEBands`, int `budget`, int `intra`, int *`prob`, `celt_word16 *error`, `ec_enc *enc`, int `_C`)

Definition at line 89 of file `quant_bands.c`.

References CHANNELS, ec_enc_tell(), ec_laplace_encode_start(), eMeans, CELTMode::ePredCoef, MULT16_16_Q15, CELTMode::nbEBands, Q15ONE, QCONST16, and SHL16.

Referenced by celt_encode_float().

6.49.1.3 void quant_energy_finalise (const CELTMode * m, celt_ener * eBands, celt_word16 * oldEBands, celt_word16 * error, int * fine_quant, int * fine_priority, int bits_left, ec_enc * enc, int _C)

Definition at line 185 of file quant_bands.c.

References CHANNELS, ec_enc_bits(), CELTMode::nbEBands, QCONST16, SHL16, and SHR16.

Referenced by celt_encode_float().

6.49.1.4 void quant_fine_energy (const CELTMode * m, celt_ener * eBands, celt_word16 * oldEBands, celt_word16 * error, int * fine_quant, ec_enc * enc, int _C)

Definition at line 146 of file quant_bands.c.

References CHANNELS, ec_enc_bits(), CELTMode::nbEBands, QCONST16, SHL16, SHR16, and SUB16.

Referenced by celt_encode_float().

6.49.1.5 int* quant_prob_alloc (const CELTMode * m)

Definition at line 64 of file quant_bands.c.

References ec_laplace_get_start_freq(), and CELTMode::nbEBands.

Referenced by celt_mode_create().

6.49.1.6 void quant_prob_free (int * freq)

Definition at line 84 of file quant_bands.c.

Referenced by celt_mode_destroy().

6.49.1.7 void unquant_coarse_energy (const CELTMode * m, celt_ener * eBands, celt_word16 * oldEBands, int budget, int intra, int * prob, ec_dec * dec, int _C)

Definition at line 221 of file quant_bands.c.

References CHANNELS, ec_dec_tell(), ec_laplace_decode_start(), eMeans, CELTMode::ePredCoef, MULT16_16_Q15, CELTMode::nbEBands, Q15ONE, QCONST16, and SHL16.

Referenced by celt_decode_float().

6.49.1.8 void unquant_energy_finalise (const CELTMode * m, celt_ener * eBands, celt_word16 * oldEBands, int * fine_quant, int * fine_priority, int bits_left, ec_dec * dec, int _C)

Definition at line 285 of file quant_bands.c.

References CHANNELS, ec_dec_bits(), CELTMode::nbEBands, QCONST16, SHL16, and SHR16.

Referenced by `celt_decode_float()`.

6.49.1.9 `void unquant_fine_energy (const CELTMode * m, celt_ener * eBands, celt_word16 * oldEBands, int * fine_quant, ec_dec * dec, int C)`

Definition at line 259 of file `quant_bands.c`.

References `CHANNELS`, `ec_dec_bits()`, `CELTMode::nbEBands`, `QCONST16`, `SHL16`, `SHR16`, and `SUB16`.

Referenced by `celt_decode_float()`.

6.49.2 Variable Documentation

6.49.2.1 `const celt_word16 eMeans[24] = {7.5f, -1.33f, -2.f, -0.42f, 0.17f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f, 0.f}`

Definition at line 48 of file `quant_bands.c`.

Referenced by `quant_coarse_energy()`, and `unquant_coarse_energy()`.

6.50 libcelt/quant_bands.h File Reference

```
#include "arch.h"
#include "modes.h"
#include "entenc.h"
#include "entdec.h"
#include "mathops.h"
```

Functions

- int * [quant_prob_alloc](#) (const [CELTMode](#) *m)
- void [quant_prob_free](#) (int *freq)
- void [compute_fine_allocation](#) (const [CELTMode](#) *m, int *bits, int budget)
- int [intra_decision](#) (celt_word16 *eBands, celt_word16 *oldEBands, int len)
- unsigned [quant_coarse_energy](#) (const [CELTMode](#) *m, celt_word16 *eBands, celt_word16 *oldEBands, int budget, int intra, int *prob, celt_word16 *error, ec_enc *enc, int _C)
- void [quant_fine_energy](#) (const [CELTMode](#) *m, celt_ener *eBands, celt_word16 *oldEBands, celt_word16 *error, int *fine_quant, ec_enc *enc, int _C)
- void [quant_energy_finalise](#) (const [CELTMode](#) *m, celt_ener *eBands, celt_word16 *oldEBands, celt_word16 *error, int *fine_quant, int *fine_priority, int bits_left, ec_enc *enc, int _C)
- void [unquant_coarse_energy](#) (const [CELTMode](#) *m, celt_ener *eBands, celt_word16 *oldEBands, int budget, int intra, int *prob, ec_dec *dec, int _C)
- void [unquant_fine_energy](#) (const [CELTMode](#) *m, celt_ener *eBands, celt_word16 *oldEBands, int *fine_quant, ec_dec *dec, int _C)
- void [unquant_energy_finalise](#) (const [CELTMode](#) *m, celt_ener *eBands, celt_word16 *oldEBands, int *fine_quant, int *fine_priority, int bits_left, ec_dec *dec, int _C)

6.50.1 Function Documentation

6.50.1.1 void [compute_fine_allocation](#) (const [CELTMode](#) * m, int * bits, int budget)

6.50.1.2 int [intra_decision](#) (celt_word16 * eBands, celt_word16 * oldEBands, int len)

Definition at line 52 of file quant_bands.c.

References [MAC16_16](#), [SHR32](#), and [SUB16](#).

Referenced by [celt_encode_float](#)().

6.50.1.3 unsigned [quant_coarse_energy](#) (const [CELTMode](#) * m, celt_word16 * eBands, celt_word16 * oldEBands, int budget, int intra, int * prob, celt_word16 * error, ec_enc * enc, int _C)

Definition at line 89 of file quant_bands.c.

References [CHANNELS](#), [ec_enc_tell](#)(), [ec_laplace_encode_start](#)(), [eMeans](#), [CELTMode::ePredCoef](#), [MULT16_16_Q15](#), [CELTMode::nbEBands](#), [Q15ONE](#), [QCONST16](#), and [SHL16](#).

Referenced by [celt_encode_float](#)().

6.50.1.4 `void quant_energy_finalise (const CELTMode * m, celt_ener * eBands, celt_word16 * oldEBands, celt_word16 * error, int * fine_quant, int * fine_priority, int bits_left, ec_enc * enc, int _C)`

Definition at line 185 of file quant_bands.c.

References CHANNELS, ec_enc_bits(), CELTMode::nbEBands, QCONST16, SHL16, and SHR16.

Referenced by celt_encode_float().

6.50.1.5 `void quant_fine_energy (const CELTMode * m, celt_ener * eBands, celt_word16 * oldEBands, celt_word16 * error, int * fine_quant, ec_enc * enc, int _C)`

Definition at line 146 of file quant_bands.c.

References CHANNELS, ec_enc_bits(), CELTMode::nbEBands, QCONST16, SHL16, SHR16, and SUB16.

Referenced by celt_encode_float().

6.50.1.6 `int* quant_prob_alloc (const CELTMode * m)`

Definition at line 64 of file quant_bands.c.

References ec_laplace_get_start_freq(), and CELTMode::nbEBands.

Referenced by celt_mode_create().

6.50.1.7 `void quant_prob_free (int * freq)`

Definition at line 84 of file quant_bands.c.

Referenced by celt_mode_destroy().

6.50.1.8 `void unquant_coarse_energy (const CELTMode * m, celt_ener * eBands, celt_word16 * oldEBands, int budget, int intra, int * prob, ec_dec * dec, int _C)`

Definition at line 221 of file quant_bands.c.

References CHANNELS, ec_dec_tell(), ec_laplace_decode_start(), eMeans, CELTMode::ePredCoef, MULT16_16_Q15, CELTMode::nbEBands, Q15ONE, QCONST16, and SHL16.

Referenced by celt_decode_float().

6.50.1.9 `void unquant_energy_finalise (const CELTMode * m, celt_ener * eBands, celt_word16 * oldEBands, int * fine_quant, int * fine_priority, int bits_left, ec_dec * dec, int _C)`

Definition at line 285 of file quant_bands.c.

References CHANNELS, ec_dec_bits(), CELTMode::nbEBands, QCONST16, SHL16, and SHR16.

Referenced by celt_decode_float().

6.50.1.10 void unquant_fine_energy (const CELTMode * *m*, celt_ener * *eBands*, celt_word16 * *oldEBands*, int * *fine_quant*, ec_dec * *dec*, int *_C*)

Definition at line 259 of file quant_bands.c.

References CHANNELS, ec_dec_bits(), CELTMode::nbEBands, QCONST16, SHL16, SHR16, and SUB16.

Referenced by celt_decode_float().

6.51 libcelt/rangedec.c File Reference

```
#include "arch.h"
#include "entdec.h"
#include "mfrngcod.h"
```

Functions

- void `ec_dec_init` (`ec_dec *_this`, `ec_byte_buffer *_buf`)
- unsigned `ec_decode` (`ec_dec *_this`, unsigned `_ft`)
- unsigned `ec_decode_bin` (`ec_dec *_this`, unsigned `_bits`)
- unsigned `ec_decode_raw` (`ec_dec *_this`, unsigned `bits`)
- void `ec_dec_update` (`ec_dec *_this`, unsigned `_fl`, unsigned `_fh`, unsigned `_ft`)
- long `ec_dec_tell` (`ec_dec *_this`, int `_b`)

6.51.1 Function Documentation

6.51.1.1 void `ec_dec_init` (`ec_dec *_this`, `ec_byte_buffer *_buf`)

Definition at line 137 of file rangedec.c.

References `ec_dec::buf`, `ec_dec::dif`, `EC_CODE_EXTRA`, `EC_SYM_BITS`, `ec_dec::end_bits_left`, `ec_dec::nb_end_bits`, `ec_dec::rem`, and `ec_dec::rng`.

Referenced by `celt_decode_float()`.

6.51.1.2 long `ec_dec_tell` (`ec_dec *_this`, int `_b`)

Definition at line 189 of file rangedec.c.

References `ec_dec::buf`, `EC_CODE_BITS`, `EC_ILOG`, `EC_SYM_BITS`, `ec_dec::nb_end_bits`, and `ec_dec::rng`.

Referenced by `celt_decode_float()`, `quant_bands()`, `unquant_bands_stereo()`, and `unquant_coarse_energy()`.

6.51.1.3 void `ec_dec_update` (`ec_dec *_this`, unsigned `_fl`, unsigned `_fh`, unsigned `_ft`)

Definition at line 181 of file rangedec.c.

References `ec_dec::dif`, `IMUL32`, `ec_dec::nrm`, and `ec_dec::rng`.

Referenced by `ec_dec_uint()`, and `ec_laplace_decode_start()`.

6.51.1.4 unsigned `ec_decode` (`ec_dec *_this`, unsigned `_ft`)

Definition at line 150 of file rangedec.c.

References `ec_dec::dif`, `EC_MINI`, `ec_dec::nrm`, and `ec_dec::rng`.

Referenced by `ec_dec_uint()`, and `ec_decode_bin()`.

6.51.1.5 unsigned ec_decode_bin (ec_dec * *this*, unsigned *bits*)

Definition at line 157 of file rangedec.c.

References `ec_dec::dif`, `EC_MINI`, `ec_dec::nrm`, and `ec_dec::rng`.

Referenced by `ec_laplace_decode_start()`.

6.51.1.6 unsigned ec_decode_raw (ec_dec * *this*, unsigned *bits*)

Definition at line 164 of file rangedec.c.

References `ec_dec::buf`, `ec_byte_look_at_end()`, `ec_dec::end_bits_left`, `ec_dec::end_byte`, and `ec_dec::nb_end_bits`.

Referenced by `ec_dec_bits()`.

6.52 libcelt/rangeenc.c File Reference

```
#include "arch.h"
#include "entenc.h"
#include "mfrngcod.h"
```

Functions

- void `ec_enc_init` (`ec_enc *_this`, `ec_byte_buffer *_buf`)
- void `ec_encode` (`ec_enc *_this`, unsigned `_fl`, unsigned `_fh`, unsigned `_ft`)
- void `ec_encode_bin` (`ec_enc *_this`, unsigned `_fl`, unsigned `_fh`, unsigned `_bits`)
- void `ec_encode_raw` (`ec_enc *_this`, unsigned `_fl`, unsigned `_fh`, unsigned `bits`)
- long `ec_enc_tell` (`ec_enc *_this`, int `_b`)
- void `ec_enc_done` (`ec_enc *_this`)

6.52.1 Function Documentation

6.52.1.1 void ec_enc_done (ec_enc * *this*)

Definition at line 178 of file rangeenc.c.

References `ec_enc::buf`, `EC_CODE_BITS`, `EC_CODE_SHIFT`, `EC_CODE_TOP`, `EC_ILOG`, `ec_enc::end_bits_left`, `ec_enc::end_byte`, `ec_byte_buffer::end_ptr`, `ec_enc::ext`, `ec_enc::low`, `ec_byte_buffer::ptr`, `ec_enc::rem`, and `ec_enc::rng`.

Referenced by `celt_encode_float()`.

6.52.1.2 void ec_enc_init (ec_enc * *this*, ec_byte_buffer * *buf*)

Definition at line 108 of file rangeenc.c.

References `ec_enc::buf`, `EC_CODE_TOP`, `ec_enc::end_bits_left`, `ec_enc::end_byte`, `ec_enc::ext`, `ec_enc::low`, `ec_enc::nb_end_bits`, `ec_enc::rem`, and `ec_enc::rng`.

Referenced by `celt_encode_float()`.

6.52.1.3 long ec_enc_tell (ec_enc * *this*, int *b*)

Definition at line 156 of file rangeenc.c.

References `ec_enc::buf`, `EC_CODE_BITS`, `EC_ILOG`, `ec_enc::ext`, `ec_enc::nb_end_bits`, `ec_enc::rem`, and `ec_enc::rng`.

Referenced by `celt_encode_float()`, `quant_bands()`, `quant_bands_stereo()`, and `quant_coarse_energy()`.

6.52.1.4 void ec_encode (ec_enc * *this*, unsigned *fl*, unsigned *fh*, unsigned *ft*)

Definition at line 119 of file rangeenc.c.

References `IMUL32`, `ec_enc::low`, and `ec_enc::rng`.

Referenced by `ec_enc_uint()`, and `ec_encode_bin()`.

6.52.1.5 void ec_encode_bin (ec_enc * *this*, unsigned *fl*, unsigned *fh*, unsigned *bits*)

Definition at line 130 of file rangeenc.c.

References IMUL32, ec_enc::low, and ec_enc::rng.

Referenced by ec_laplace_encode_start().

6.52.1.6 void ec_encode_raw (ec_enc * *this*, unsigned *fl*, unsigned *fh*, unsigned *bits*)

Definition at line 141 of file rangeenc.c.

References ec_enc::buf, ec_byte_write_at_end(), ec_enc::end_bits_left, ec_enc::end_byte, and ec_enc::nb_end_bits.

Referenced by ec_enc_bits().

6.53 libcelt/rate.c File Reference

```
#include <math.h>
#include "modes.h"
#include "cwrs.h"
#include "arch.h"
#include "os_support.h"
#include "entcode.h"
#include "rate.h"
```

Functions

- `celt_int16** compute_alloc_cache` (CELTMode *m, int C)
- void `compute_allocation` (const CELTMode *m, int *offsets, int total, int *pulses, int *ebits, int *fine_priority, int _C)

6.53.1 Function Documentation

6.53.1.1 `celt_int16** compute_alloc_cache` (CELTMode *m, int C)

Computes a cache of the pulses->bits mapping in each band

Definition at line 49 of file rate.c.

References BITRES, CELTMode::eBands, get_required_bits(), MAX_PSEUDO, MAX_PULSES, CELTMode::nbEBands, and CELTMode::pitchEnd.

Referenced by celt_mode_create().

6.53.1.2 void `compute_allocation` (const CELTMode *m, int *offsets, int total, int *pulses, int *ebits, int *fine_priority, int _C)

Compute the pulse allocation, i.e. how many pulses will go in each band.

Parameters:

- m* mode
- offsets* Requested increase or decrease in the number of bits for each band
- total* Number of bands
- pulses* Number of pulses per band (returned)

Returns:

Total number of bits allocated

Definition at line 176 of file rate.c.

References ALLOC, CELTMode::allocVectors, BITRES, CHANNELS, CELTMode::nbAllocVectors, CELTMode::nbEBands, RESTORE_STACK, SAVE_STACK, and VARDECL.

Referenced by celt_decode_float(), and celt_encode_float().

6.54 libcelt/rate.h File Reference

```
#include "cwrs.h"
```

Defines

- #define [MAX_PSEUDO](#) 40
- #define [LOG_MAX_PSEUDO](#) 6
- #define [MAX_PULSES](#) 128
- #define [LOG_MAX_PULSES](#) 7
- #define [BITRES](#) 4
- #define [FINE_OFFSET](#) 50
- #define [QTHETA_OFFSET](#) 40
- #define [BITOVERFLOW](#) 30000

Functions

- [celt_int16 ** compute_alloc_cache](#) ([CELTMode](#) *m, int C)
- void [compute_allocation](#) (const [CELTMode](#) *m, int *offsets, int total, int *pulses, int *ebits, int *fine_priority, int _C)

6.54.1 Define Documentation

6.54.1.1 #define BITOVERFLOW 30000

Definition at line 46 of file rate.h.

6.54.1.2 #define BITRES 4

Definition at line 42 of file rate.h.

Referenced by [celt_encode_float\(\)](#), [compute_alloc_cache\(\)](#), [compute_allocation\(\)](#), [quant_bands\(\)](#), [quant_bands_stereo\(\)](#), and [unquant_bands_stereo\(\)](#).

6.54.1.3 #define FINE_OFFSET 50

Definition at line 43 of file rate.h.

6.54.1.4 #define LOG_MAX_PSEUDO 6

Definition at line 37 of file rate.h.

6.54.1.5 #define LOG_MAX_PULSES 7

Definition at line 40 of file rate.h.

6.54.1.6 #define MAX_PSEUDO 40

Definition at line 36 of file rate.h.

Referenced by compute_alloc_cache().

6.54.1.7 #define MAX_PULSES 128

Definition at line 39 of file rate.h.

Referenced by compute_alloc_cache(), and dump_modes().

6.54.1.8 #define QTHETA_OFFSET 40

Definition at line 44 of file rate.h.

Referenced by quant_bands_stereo(), and unquant_bands_stereo().

6.54.2 Function Documentation**6.54.2.1 celt_int16** compute_alloc_cache (CELTMode * m, int C)**

Computes a cache of the pulses->bits mapping in each band

Definition at line 49 of file rate.c.

References BITRES, CELTMode::eBands, get_required_bits(), MAX_PSEUDO, MAX_PULSES, CELTMode::nbEBands, and CELTMode::pitchEnd.

Referenced by celt_mode_create().

6.54.2.2 void compute_allocation (const CELTMode * m, int * offsets, int total, int * pulses, int * ebits, int * fine_priority, int C)

Compute the pulse allocation, i.e. how many pulses will go in each band.

Parameters:

m mode

offsets Requested increase or decrease in the number of bits for each band

total Number of bands

pulses Number of pulses per band (returned)

Returns:

Total number of bits allocated

Definition at line 176 of file rate.c.

References ALLOC, CELTMode::allocVectors, BITRES, CHANNELS, CELTMode::nbAllocVectors, CELTMode::nbEBands, RESTORE_STACK, SAVE_STACK, and VARDECL.

Referenced by celt_decode_float(), and celt_encode_float().

6.55 libcelt/stack_alloc.h File Reference

Temporary memory allocation on stack. `#include "os_support.h"`

Defines

- `#define ALIGN(stack, size) ((stack) += ((size) - (long)(stack)) & ((size) - 1))`
- `#define PUSH(stack, size, type) (ALIGN((stack),sizeof(type)/sizeof(char)),(stack)+=(size)*(sizeof(type)/sizeof(char)),(type)*(sizeof(type)/sizeof(char)))`
- `#define RESTORE_STACK (global_stack = _saved_stack)`
- `#define ALLOC_STACK (global_stack = (global_stack==0) ? celt_alloc_scratch(GLOBAL_STACK_SIZE) : global_stack)`
- `#define VARDECL(type, var) type *var`
- `#define ALLOC(var, size, type) var = PUSH(global_stack, size, type)`
- `#define SAVE_STACK char *_saved_stack = global_stack;`

Variables

- `char * global_stack = 0`

6.55.1 Detailed Description

Temporary memory allocation on stack.

Definition in file [stack_alloc.h](#).

6.55.2 Define Documentation

6.55.2.1 `#define ALIGN(stack, size) ((stack) += ((size) - (long)(stack)) & ((size) - 1))`

Aligns the stack to a 'size' boundary

Parameters:

stack Stack

size New size boundary

Definition at line 130 of file [stack_alloc.h](#).

6.55.2.2 `#define ALLOC(var, size, type) var = PUSH(global_stack, size, type)`

Allocate 'size' elements of 'type' on stack

Parameters:

var Name of variable to allocate

size Number of elements

type Type of element

Definition at line 139 of file stack_alloc.h.

Referenced by alg_quant(), alg_unquant(), c64_fft16_inplace(), c64_fft32(), c64_ifft16(), c64_ifft32(), celt_decode(), celt_decode_float(), celt_encode(), celt_encode_float(), compute_allocation(), find_spectral_pitch(), get_required_bits(), mdct_backward(), mdct_forward(), quant_bands(), quant_bands_stereo(), and unquant_bands_stereo().

6.55.2.3 #define ALLOC_STACK (global_stack = (global_stack==0) ? celt_alloc_scratch(GLOBAL_STACK_SIZE) : global_stack)

Definition at line 133 of file stack_alloc.h.

Referenced by celt_mode_create().

6.55.2.4 #define PUSH(stack, size, type) (ALIGN((stack),sizeof(type)/sizeof(char)),(stack)+=(size)*(sizeof(type)/sizeof(char)),(type*)((stack)-(size)*(sizeof(type)/sizeof(char))))

Allocates 'size' elements of type 'type' on the stack

Parameters:

stack Stack

size Number of elements

type Type of element

Definition at line 131 of file stack_alloc.h.

6.55.2.5 #define RESTORE_STACK (global_stack = _saved_stack)

Definition at line 132 of file stack_alloc.h.

Referenced by alg_quant(), alg_unquant(), c64_fft16_inplace(), c64_fft32(), c64_ifft16(), c64_ifft32(), celt_decode(), celt_decode_float(), celt_encode(), celt_encode_float(), compute_allocation(), find_spectral_pitch(), get_required_bits(), mdct_backward(), mdct_forward(), quant_bands(), quant_bands_stereo(), and unquant_bands_stereo().

6.55.2.6 #define SAVE_STACK char *_saved_stack = global_stack;

Definition at line 140 of file stack_alloc.h.

Referenced by alg_quant(), alg_unquant(), c64_fft16_inplace(), c64_fft32(), c64_ifft16(), c64_ifft32(), celt_decode(), celt_decode_float(), celt_encode(), celt_encode_float(), compute_allocation(), find_spectral_pitch(), get_required_bits(), mdct_backward(), mdct_forward(), quant_bands(), quant_bands_stereo(), and unquant_bands_stereo().

6.55.2.7 #define VARDECL(type, var) type *var

Declare variable on stack

Parameters:

var Variable to declare

Definition at line 138 of file stack_alloc.h.

Referenced by alg_quant(), alg_unquant(), c64_fft16_inplace(), c64_fft32(), c64_ifft16(), c64_ifft32(), celt_decode(), celt_decode_float(), celt_encode(), celt_encode_float(), compute_allocation(), find_spectral_pitch(), get_required_bits(), mdct_backward(), mdct_forward(), quant_bands(), quant_bands_stereo(), and unquant_bands_stereo().

6.55.3 Variable Documentation

6.55.3.1 char* global_stack = 0

Definition at line 108 of file stack_alloc.h.

Referenced by celt_mode_create().

6.56 libcelt/testcelt.c File Reference

```
#include "celt.h"  
#include "arch.h"  
#include <stdio.h>  
#include <stdlib.h>  
#include <math.h>  
#include <string.h>
```

Defines

- #define [MAX_PACKET](#) 1024

Functions

- int [main](#) (int argc, char *argv[])

6.56.1 Define Documentation

6.56.1.1 #define MAX_PACKET 1024

Definition at line 44 of file testcelt.c.

Referenced by [main\(\)](#).

6.56.2 Function Documentation

6.56.2.1 int main (int argc, char * argv[])

Definition at line 46 of file testcelt.c.

References [celt_decode\(\)](#), [celt_decoder_create\(\)](#), [celt_decoder_destroy\(\)](#), [celt_encode\(\)](#), [celt_encoder_create\(\)](#), [celt_encoder_ctl\(\)](#), [celt_encoder_destroy\(\)](#), [CELT_GET_FRAME_SIZE](#), [CELT_GET_LOOKAHEAD](#), [celt_mode_create\(\)](#), [celt_mode_destroy\(\)](#), [celt_mode_info\(\)](#), [CELT_SET_COMPLEXITY](#), [celt_strerror\(\)](#), [MAX_PACKET](#), and [PRINT_MIPS](#).

6.57 libcelt/vq.c File Reference

```
#include "mathops.h"
#include "cwrs.h"
#include "vq.h"
#include "arch.h"
#include "os_support.h"
#include "rate.h"
```

Functions

- void `alg_quant` (`celt_norm *X`, int `N`, int `K`, int `spread`, `ec_enc *enc`)
- void `alg_unquant` (`celt_norm *X`, int `N`, int `K`, int `spread`, `ec_dec *dec`)
- `celt_word16 renormalise_vector` (`celt_norm *X`, `celt_word16` value, int `N`, int `stride`)
- void `intra_fold` (const `CELTMode *m`, int `N`, const `celt_norm *restrict Y`, `celt_norm *restrict P`, int `N0`, int `B`)

6.57.1 Function Documentation

6.57.1.1 void `alg_quant` (`celt_norm *X`, int `N`, int `K`, int `spread`, `ec_enc *enc`)

Algebraic pulse-vector quantiser. The signal `x` is replaced by the sum of the pitch and a combination of pulses such that its norm is still equal to 1. This is the function that will typically require the most CPU.

Parameters:

- `x` Residual signal to quantise/encode (returns quantised version)
- `W` Perceptual weight to use when optimising (currently unused)
- `N` Number of samples to encode
- `K` Number of pulses to use
- `p` Pitch vector (it is assumed that `p+x` is a unit vector)
- `enc` Entropy encoder state

Definition at line 116 of file `vq.c`.

References `ALLOC`, `celt_assert2`, `celt_ilog2`, `celt_rcp`, `encode_pulses()`, `EPSILON`, `EXTRACT16`, `MAC16_16`, `MULT16_16`, `MULT16_16_Q15`, `MULT16_32_Q16`, `QCONST16`, `RESTORE_STACK`, `SAVE_STACK`, `SHL16`, `SHR32`, `VARDECL`, and `VERY_LARGE16`.

Referenced by `quant_bands()`, and `quant_bands_stereo()`.

6.57.1.2 void `alg_unquant` (`celt_norm *X`, int `N`, int `K`, int `spread`, `ec_dec *dec`)

Decode pulse vector and combine the result with the pitch vector to produce the final normalised signal in the current band.

Definition at line 282 of file `vq.c`.

References `ALLOC`, `decode_pulses()`, `MAC16_16`, `RESTORE_STACK`, `SAVE_STACK`, and `VARDECL`.

Referenced by `quant_bands()`, and `unquant_bands_stereo()`.

6.57.1.3 void intra_fold (const CELTMode * *m*, int *N*, const celt_norm *restrict *Y*, celt_norm *restrict *P*, int *N0*, int *B*)

Intra-frame predictor that matches a section of the current frame (at lower frequencies) to encode the current band.

Parameters:

- x* Residual signal to quantise/encode (returns quantised version)
- W* Perceptual weight
- N* Number of samples to encode
- K* Number of pulses to use
- Y* Lower frequency spectrum to use, normalised to the same standard deviation
- P* Pitch vector (it is assumed that $p+x$ is a unit vector)
- B* Stride (number of channels multiplied by the number of MDCTs per frame)
- N0* Number of valid offsets

Definition at line 356 of file vq.c.

References Q15ONE, and renormalise_vector().

Referenced by quant_bands(), quant_bands_stereo(), and unquant_bands_stereo().

6.57.1.4 celt_word16 renormalise_vector (celt_norm * *X*, celt_word16 *value*, int *N*, int *stride*)

Definition at line 302 of file vq.c.

References celt_rcp, celt_sqrt, EPSILON, MAC16_16, MULT16_16, MULT16_16_Q15, PSHR32, Q15ONE, and SHL32.

Referenced by intra_fold(), quant_bands_stereo(), renormalise_bands(), and unquant_bands_stereo().

6.58 libcelt/vq.h File Reference

Vector quantisation of the residual. `#include "entenc.h"`

`#include "entdec.h"`

`#include "modes.h"`

Functions

- void `alg_quant` (`celt_norm *X`, int `N`, int `K`, int `spread`, `ec_enc *enc`)
- void `alg_unquant` (`celt_norm *X`, int `N`, int `K`, int `spread`, `ec_dec *dec`)
- `celt_word16` `renormalise_vector` (`celt_norm *X`, `celt_word16` `value`, int `N`, int `stride`)
- void `intra_fold` (const `CELTMode *m`, int `N`, const `celt_norm *restrict Y`, `celt_norm *restrict P`, int `N0`, int `B`)

6.58.1 Detailed Description

Vector quantisation of the residual.

Definition in file `vq.h`.

6.58.2 Function Documentation

6.58.2.1 void `alg_quant` (`celt_norm * X`, int `N`, int `K`, int `spread`, `ec_enc * enc`)

Algebraic pulse-vector quantiser. The signal `x` is replaced by the sum of the pitch and a combination of pulses such that its norm is still equal to 1. This is the function that will typically require the most CPU.

Parameters:

- `x` Residual signal to quantise/encode (returns quantised version)
- `W` Perceptual weight to use when optimising (currently unused)
- `N` Number of samples to encode
- `K` Number of pulses to use
- `p` Pitch vector (it is assumed that `p+x` is a unit vector)
- `enc` Entropy encoder state

Definition at line 116 of file `vq.c`.

References `ALLOC`, `celt_assert2`, `celt_ilog2`, `celt_rcp`, `encode_pulses()`, `EPSILON`, `EXTRACT16`, `MAC16_16`, `MULT16_16`, `MULT16_16_Q15`, `MULT16_32_Q16`, `QCONST16`, `RESTORE_STACK`, `SAVE_STACK`, `SHL16`, `SHR32`, `VARDECL`, and `VERY_LARGE16`.

Referenced by `quant_bands()`, and `quant_bands_stereo()`.

6.58.2.2 void `alg_unquant` (`celt_norm * X`, int `N`, int `K`, int `spread`, `ec_dec * dec`)

Algebraic pulse decoder

Parameters:

- `x` Decoded normalised spectrum (returned)

N Number of samples to decode
K Number of pulses to use
p Pitch vector (automatically added to *x*)
dec Entropy decoder state

Decode pulse vector and combine the result with the pitch vector to produce the final normalised signal in the current band.

Definition at line 282 of file `vq.c`.

References `ALLOC`, `decode_pulses()`, `MAC16_16`, `RESTORE_STACK`, `SAVE_STACK`, and `VARDECL`.

Referenced by `quant_bands()`, and `unquant_bands_stereo()`.

6.58.2.3 `void intra_fold (const CELTMode * m, int N, const celt_norm *restrict Y, celt_norm *restrict P, int N0, int B)`

Intra-frame predictor that matches a section of the current frame (at lower frequencies) to encode the current band.

Parameters:

x Residual signal to quantise/encode (returns quantised version)
W Perceptual weight
N Number of samples to encode
K Number of pulses to use
Y Lower frequency spectrum to use, normalised to the same standard deviation
P Pitch vector (it is assumed that *p+x* is a unit vector)
B Stride (number of channels multiplied by the number of MDCTs per frame)
N0 Number of valid offsets

Definition at line 356 of file `vq.c`.

References `Q15ONE`, and `renormalise_vector()`.

Referenced by `quant_bands()`, `quant_bands_stereo()`, and `unquant_bands_stereo()`.

6.58.2.4 `celt_word16 renormalise_vector (celt_norm * X, celt_word16 value, int N, int stride)`

Definition at line 302 of file `vq.c`.

References `celt_rcp`, `celt_sqrt`, `EPSILON`, `MAC16_16`, `MULT16_16`, `MULT16_16_Q15`, `PSHR32`, `Q15ONE`, and `SHL32`.

Referenced by `intra_fold()`, `quant_bands_stereo()`, `renormalise_bands()`, and `unquant_bands_stereo()`.

- EPSILON, 48
- EXTEND32, 49
- EXTRACT16, 49
- GLOBAL_STACK_SIZE, 49
- HALF32, 49
- IMAX, 49
- IMIN, 49
- IMUL32, 49
- LOG2_BITS_PER_CHAR, 49
- MAC16_16, 50
- MAC16_16_P13, 50
- MAC16_16_Q11, 50
- MAC16_16_Q13, 50
- MAC16_32_Q11, 50
- MAC16_32_Q15, 50
- MAX16, 50
- MAX32, 50
- MIN16, 50
- MIN32, 51
- MULT16_16, 51
- MULT16_16_16, 51
- MULT16_16_P13, 51
- MULT16_16_P14, 51
- MULT16_16_P15, 51
- MULT16_16_Q11_32, 51
- MULT16_16_Q13, 51
- MULT16_16_Q14, 51
- MULT16_16_Q15, 52
- MULT16_32_P15, 52
- MULT16_32_Q11, 52
- MULT16_32_Q13, 52
- MULT16_32_Q14, 52
- MULT16_32_Q15, 52
- MULT16_32_Q16, 52
- MULT32_32_Q31, 52
- NEG16, 52
- NEG32, 52
- NORM_SCALING, 53
- NORM_SCALING_1, 53
- PDIV32, 53
- PDIV32_16, 53
- PGAIN_SCALING, 53
- PGAIN_SCALING_1, 53
- PRINT_MIPS, 53
- PSHR, 53
- PSHR16, 53
- PSHR32, 53
- Q15_ONE, 54
- Q15_ONE_1, 54
- Q15ONE, 54
- Q30ONE, 54
- QCONST16, 54
- QCONST32, 54
- ROUND16, 54
- SATURATE, 54
- SATURATE16, 54
- SATURATE32, 55
- SCALEIN, 55
- SCALEOUT, 55
- SHL, 55
- SHL16, 55
- SHL32, 55
- SHR, 55
- SHR16, 55
- SHR32, 55
- SUB16, 56
- SUB32, 56
- UADD32, 56
- UMUL16_16, 56
- UMUL32, 56
- USUB32, 56
- VERY_LARGE16, 56
- VERY_LARGE32, 56
- VERY_SMALL, 56
- VSHR32, 56
- bands.c
 - apply_pitch, 58
 - compute_band_energies, 58
 - compute_pitch_gain, 59
 - denormalise_bands, 59
 - folding_decision, 59
 - normalise_bands, 60
 - quant_bands, 60
 - quant_bands_stereo, 60
 - renormalise_bands, 61
 - unquant_bands_stereo, 61
- bands.h
 - apply_pitch, 62
 - compute_band_energies, 62
 - compute_pitch_gain, 63
 - denormalise_bands, 63
 - folding_decision, 63
 - normalise_bands, 64
 - quant_bands, 64
 - quant_bands_stereo, 64
 - renormalise_bands, 64
 - stereo_decision, 64
 - unquant_bands, 65
 - unquant_bands_stereo, 65
- BARK_BANDS
 - modes.c, 152
- BITALLOC_SIZE
 - modes.c, 152
- BITOVERFLOW
 - rate.h, 175
- BITRES
 - rate.h, 175

- BITREV
 - kfft_single.h, 128
- bitrev
 - kiss_fft_state, 35
- bits
 - CELTMode, 25
- BITS_PER_CHAR
 - arch.h, 47
- block_size
 - CELTDecoder, 15
 - CELTEncoder, 18
- buf
 - CELTDecoder, 15
 - ec_byte_buffer, 29
 - ec_dec, 30
 - ec_enc, 32
- BYTES_PER_CHAR
 - arch.h, 48
- bytes_per_packet
 - CELTHHeader, 22
- c64_fft.c
 - c64_fft16_alloc, 66
 - c64_fft16_free, 66
 - c64_fft16_inplace, 67
 - c64_fft32, 67
 - c64_fft32_alloc, 67
 - c64_fft32_free, 67
 - c64_iff16, 67
 - c64_iff32, 67
 - gen_twiddle16, 67
 - gen_twiddle32, 67
 - NBCACHE, 66
 - PI, 66
- c64_fft.h
 - c64_fft16_alloc, 69
 - c64_fft16_free, 69
 - c64_fft16_inplace, 69
 - c64_fft32, 69
 - c64_fft32_alloc, 69
 - c64_fft32_free, 69
 - c64_iff16, 70
 - c64_iff32, 70
- c64_fft16_alloc
 - c64_fft.c, 66
 - c64_fft.h, 69
- c64_fft16_free
 - c64_fft.c, 66
 - c64_fft.h, 69
- c64_fft16_inplace
 - c64_fft.c, 67
 - c64_fft.h, 69
- c64_fft32
 - c64_fft.c, 67
- c64_fft.h, 69
 - c64_fft32_alloc
 - c64_fft.c, 67
 - c64_fft.h, 69
 - c64_fft32_free
 - c64_fft.c, 67
 - c64_fft.h, 69
 - c64_fft_t, 13
 - itwiddle, 13
 - nfft, 13
 - shift, 13
 - twiddle, 13
- c64_iff16
 - c64_fft.c, 67
 - c64_fft.h, 70
- c64_iff32
 - c64_fft.c, 67
 - c64_fft.h, 70
- C_ADD
 - _kiss_fft_guts.h, 42
- C_ADDTO
 - _kiss_fft_guts.h, 42
- C_FIXDIV
 - _kiss_fft_guts.h, 42
- C_MUL
 - _kiss_fft_guts.h, 42
- C_MUL4
 - _kiss_fft_guts.h, 42
- C_MULBYSCALAR
 - _kiss_fft_guts.h, 42
- C_MULC
 - _kiss_fft_guts.h, 42
- C_SUB
 - _kiss_fft_guts.h, 43
- C_SUBFROM
 - _kiss_fft_guts.h, 43
- CAT_SUFFIX
 - kiss_fft.h, 131
- celt.c
 - CELT_C, 72
 - celt_decode, 73
 - celt_decode_float, 74
 - celt_decoder_ctl, 74
 - celt_encode, 74
 - celt_encode_float, 74
 - celt_encoder_ctl, 75
 - check_decoder, 75
 - DECODE_BUFFER_SIZE, 72
 - DECODERFREED, 72
 - DECODERPARTIAL, 72
 - DECODERINVALID, 72
 - ENCODERFREED, 72
 - ENCODERPARTIAL, 73
 - ENCODERINVALID, 73

- FLAG_FOLD, 73
- FLAG_INTRA, 73
- FLAG_MASK, 73
- FLAG_NONE, 73
- FLAG_PITCH, 73
- FLAG_SHORT, 73
- celt.h
 - _celt_check_int, 77
 - _celt_check_mode_ptr_ptr, 77
 - CELT_ALLOC_FAIL, 77
 - CELT_BAD_ARG, 77
 - CELT_CORRUPTED_DATA, 77
 - CELT_GET_BITSTREAM_VERSION, 77
 - CELT_GET_FRAME_SIZE, 78
 - CELT_GET_LOOKAHEAD, 78
 - CELT_GET_MODE, 78
 - CELT_GET_MODE_REQUEST, 78
 - CELT_GET_SAMPLE_RATE, 78
 - CELT_INTERNAL_ERROR, 78
 - CELT_INVALID_MODE, 78
 - CELT_INVALID_STATE, 79
 - CELT_OK, 79
 - CELT_RESET_STATE, 79
 - CELT_RESET_STATE_REQUEST, 79
 - CELT_SET_COMPLEXITY, 79
 - CELT_SET_COMPLEXITY_REQUEST, 79
 - CELT_SET_PREDICTION, 79
 - CELT_SET_PREDICTION_REQUEST, 80
 - CELT_SET_VBR_RATE, 80
 - CELT_SET_VBR_RATE_REQUEST, 80
 - CELT_UNIMPLEMENTED, 80
 - CELTDecoder, 80
 - CELTEncoder, 80
 - CELTMode, 81
 - EXPORT, 80
- celt_acos
 - mathops.h, 143
- CELT_ALLOC_FAIL
 - celt.h, 77
- celt_assert
 - arch.h, 48
- celt_assert2
 - arch.h, 48
- celt_atan
 - mathops.h, 143
- CELT_BAD_ARG
 - celt.h, 77
- CELT_BITSTREAM_VERSION
 - modes.h, 154
- CELT_C
 - celt.c, 72
- CELT_COPY
 - os_support.h, 156
- CELT_CORRUPTED_DATA
 - celt.h, 77
- celt_cos_norm
 - mathops.h, 143
- celt_decode
 - celt.c, 73
 - codec, 7
- celt_decode_float
 - celt.c, 74
 - codec, 8
- celt_decoder_create
 - codec, 8
- celt_decoder_ctl
 - celt.c, 74
 - codec, 8
- celt_decoder_destroy
 - codec, 9
- celt_div
 - mathops.h, 143
- celt_encode
 - celt.c, 74
 - codec, 9
- celt_encode_float
 - celt.c, 74
 - codec, 9
- celt_encoder_create
 - codec, 10
- celt_encoder_ctl
 - celt.c, 75
 - codec, 10
- celt_encoder_destroy
 - codec, 11
- celt_ener
 - arch.h, 57
- celt_exp
 - mathops.h, 143
- celt_exp2
 - mathops.h, 144
- celt_fatal
 - arch.h, 48
- CELT_GET_BITSTREAM_VERSION
 - celt.h, 77
- CELT_GET_FRAME_SIZE
 - celt.h, 78
- CELT_GET_LOOKAHEAD
 - celt.h, 78
- CELT_GET_MODE
 - celt.h, 78
- CELT_GET_MODE_REQUEST
 - celt.h, 78
- CELT_GET_SAMPLE_RATE
 - celt.h, 78
- celt_header.h
 - celt_header_from_packet, 82
 - celt_header_init, 82

- celt_header_to_packet, 82
- celt_header_from_packet
 - celt_header.h, 82
 - header.c, 125
- celt_header_init
 - celt_header.h, 82
 - header.c, 125
- celt_header_to_packet
 - celt_header.h, 82
 - header.c, 125
- celt_ilog2
 - fixed_c5x.h, 107
 - fixed_c6x.h, 109
- celt_int16
 - celt_types.h, 83
- celt_int32
 - celt_types.h, 83
- CELT_INTERNAL_ERROR
 - celt.h, 78
- CELT_INVALID_MODE
 - celt.h, 78
- CELT_INVALID_STATE
 - celt.h, 79
- celt_log2
 - mathops.h, 144
- celt_mask
 - arch.h, 57
- celt_maxabs16
 - fixed_c5x.h, 107
- CELT_MEMSET
 - os_support.h, 156
- celt_mips
 - fixed_debug.h, 115
- celt_mode_create
 - codec, 11
- celt_mode_destroy
 - codec, 12
- celt_mode_info
 - codec, 12
- CELT_MOVE
 - os_support.h, 156
- celt_norm
 - arch.h, 57
- CELT_OK
 - celt.h, 79
- celt_pgain
 - arch.h, 57
- celt_psqrt
 - mathops.h, 144
- celt_rcp
 - mathops.h, 144
- CELT_RESET_STATE
 - celt.h, 79
- CELT_RESET_STATE_REQUEST
 - celt.h, 79
- celt_rsqrt
 - mathops.h, 144
- CELT_SET_COMPLEXITY
 - celt.h, 79
- CELT_SET_COMPLEXITY_REQUEST
 - celt.h, 79
- CELT_SET_PREDICTION
 - celt.h, 79
- CELT_SET_PREDICTION_REQUEST
 - celt.h, 80
- CELT_SET_VBR_RATE
 - celt.h, 80
- CELT_SET_VBR_RATE_REQUEST
 - celt.h, 80
- celt_sig
 - arch.h, 57
- CELT_SIG_SCALE
 - arch.h, 48
- celt_sqrt
 - mathops.h, 144
- celt_strerror
 - codec, 12
- celt_types.h
 - celt_int16, 83
 - celt_int32, 83
 - celt_uint16, 83
 - celt_uint32, 83
- celt_uint16
 - celt_types.h, 83
- celt_uint32
 - celt_types.h, 83
- CELT_UNIMPLEMENTED
 - celt.h, 80
- celt_word16
 - arch.h, 57
- celt_word32
 - arch.h, 57
- CELTDecoder, 15
 - block_size, 15
 - buf, 15
 - celt.h, 80
 - channels, 15
 - decode_mem, 15
 - enc, 15
 - frame_size, 16
 - last_pitch_index, 16
 - loss_count, 16
 - marker, 16
 - mode, 16
 - oldBandE, 16
 - out_mem, 16
 - overlap, 16
 - preemph_memD, 16

- CELTEncoder, 18
 - block_size, 18
 - celt.h, 80
 - channels, 18
 - delayedIntra, 18
 - fold_decision, 19
 - force_intra, 19
 - frame_size, 19
 - gain_prod, 19
 - in_mem, 19
 - marker, 19
 - mode, 19
 - oldBandE, 19
 - out_mem, 19
 - overlap, 20
 - pitch_available, 20
 - pitch_enabled, 20
 - pitch_permitted, 20
 - preemph_memD, 20
 - preemph_memE, 20
 - tonal_average, 20
 - vbr_count, 20
 - vbr_drift, 20
 - vbr_offset, 21
 - vbr_rate, 21
 - vbr_reservoir, 21
- CELTHHeader, 22
 - bytes_per_packet, 22
 - codec_id, 22
 - codec_version, 22
 - extra_headers, 22
 - frame_size, 23
 - header_size, 23
 - nb_channels, 23
 - overlap, 23
 - sample_rate, 23
 - version_id, 23
- CELTMode, 25
 - allocVectors, 25
 - bits, 25
 - celt.h, 81
 - eBands, 26
 - ePredCoef, 26
 - fft, 26
 - Fs, 26
 - marker_end, 26
 - marker_start, 26
 - mdct, 26
 - mdctSize, 27
 - nbAllocVectors, 27
 - nbEBands, 27
 - nbShortMdcts, 27
 - overlap, 27
 - pitchEnd, 27
 - prob, 27
 - psy, 28
 - shortMdct, 28
 - shortMdctSize, 28
 - shortWindow, 28
 - window, 28
- CHANNELS
 - modes.h, 154
- channels
 - CELTDecoder, 15
 - CELTEncoder, 18
- check_decoder
 - celt.c, 75
- check_mode
 - modes.c, 153
 - modes.h, 155
- CHECK_OVERFLOW_OP
 - _kiss_fft_guts.h, 43
- codec
 - celt_decode, 7
 - celt_decode_float, 8
 - celt_decoder_create, 8
 - celt_decoder_ctl, 8
 - celt_decoder_destroy, 9
 - celt_encode, 9
 - celt_encode_float, 9
 - celt_encoder_create, 10
 - celt_encoder_ctl, 10
 - celt_encoder_destroy, 11
 - celt_mode_create, 11
 - celt_mode_destroy, 12
 - celt_mode_info, 12
 - celt_strerror, 12
- codec_id
 - CELTHHeader, 22
- codec_version
 - CELTHHeader, 22
- compute_alloc_cache
 - rate.c, 174
 - rate.h, 176
- compute_allocation
 - rate.c, 174
 - rate.h, 176
- compute_band_energies
 - bands.c, 58
 - bands.h, 62
- compute_fine_allocation
 - quant_bands.h, 167
- compute_masking
 - psy.c, 160
 - psy.h, 162
- compute_mdct_masking
 - psy.h, 162
- compute_pitch_gain

- bands.c, 59
- bands.h, 63
- compute_tonality
 - psy.h, 162
- cpx32_fft
 - kfft_double.h, 126
- cpx32_fft_alloc
 - kfft_double.h, 126
- cpx32_fft_free
 - kfft_double.h, 126
- cpx32_ifft
 - kfft_double.h, 126
- cwrs.c
 - decode_pulses, 84
 - encode_pulses, 84
 - fits_in32, 84
 - get_required_bits, 85
 - icwrs, 85
 - log2_frac, 85
 - MASK32, 84
- cwrs.h
 - decode_pulses, 86
 - encode_pulses, 86
 - fits_in32, 86
 - get_required_bits, 86
 - log2_frac, 86
- decayR
 - PsyDecay, 39
- DECODE_BUFFER_SIZE
 - celt.c, 72
- decode_mem
 - CELTDecoder, 15
- decode_pulses
 - cwrs.c, 84
 - cwrs.h, 86
- DECODERFREED
 - celt.c, 72
- DECODERPARTIAL
 - celt.c, 72
- DECODERVALID
 - celt.c, 72
- delayedIntra
 - CELTEncoder, 18
- denormalise_bands
 - bands.c, 59
 - bands.h, 63
- dif
 - ec_dec, 30
- DIV32
 - arch.h, 48
 - fixed_debug.h, 111
 - fixed_generic.h, 117
- DIV32_16
 - arch.h, 48
 - fixed_debug.h, 111
 - fixed_generic.h, 117
- dump_header
 - dump_modes.c, 89
- dump_modes
 - dump_modes.c, 89
- dump_modes.c
 - dump_header, 89
 - dump_modes, 89
 - FLOAT, 88
 - INT16, 88
 - INT32, 88
 - main, 89
 - WORD16, 88
 - WORD32, 88
- eBands
 - CELTMode, 26
- EC_BUFFER_INCREMENT
 - entenc.c, 102
- ec_byte_adv1
 - entcode.h, 94
 - entdec.c, 97
- ec_byte_adv4
 - entcode.h, 94
- ec_byte_buffer, 29
 - buf, 29
 - end_ptr, 29
 - entcode.h, 94
 - ptr, 29
 - storage, 29
- ec_byte_look1
 - entcode.h, 94
- ec_byte_look4
 - entcode.h, 94
- ec_byte_look_at_end
 - entcode.h, 94
 - entdec.c, 97
- ec_byte_read1
 - entcode.h, 94
 - entdec.c, 97
- ec_byte_read4
 - entcode.h, 95
- ec_byte_readinit
 - entcode.h, 95
 - entdec.c, 97
- ec_byte_shrink
 - entcode.h, 95
 - entenc.c, 102
- ec_byte_write1
 - entcode.h, 95
 - entenc.c, 102
- ec_byte_write4

- entcode.h, 95
- ec_byte_write_at_end
 - entcode.h, 95
 - entenc.c, 102
- ec_byte_writeclear
 - entcode.h, 95
- ec_byte_writecopy
 - entcode.h, 95
- ec_byte_writeinit
 - entcode.h, 95
- ec_byte_writeinit_buffer
 - entcode.h, 95
 - entenc.c, 102
- ec_byte_writetrunc
 - entcode.h, 95
- EC_CLAMPI
 - ecintrin.h, 90
- EC_CODE_BITS
 - mfrngcod.h, 148
- EC_CODE_BOT
 - mfrngcod.h, 148
- EC_CODE_CARRY
 - mfrngcod.h, 148
- EC_CODE_EXTRA
 - mfrngcod.h, 148
- EC_CODE_MASK
 - mfrngcod.h, 148
- EC_CODE_SHIFT
 - mfrngcod.h, 148
- EC_CODE_TOP
 - mfrngcod.h, 149
- ec_dec, 30
 - buf, 30
 - dif, 30
 - end_bits_left, 30
 - end_byte, 30
 - entdec.h, 99
 - nb_end_bits, 30
 - nm, 30
 - rem, 31
 - rng, 31
- ec_dec_bits
 - entdec.c, 97
 - entdec.h, 99
- ec_dec_init
 - entdec.h, 99
 - mfrngdec.c, 150
 - rangedec.c, 170
- ec_dec_tell
 - entdec.h, 100
 - mfrngdec.c, 150
 - rangedec.c, 170
- ec_dec_uint
 - entdec.c, 98
- entdec.h, 100
- ec_dec_update
 - entdec.h, 100
 - mfrngdec.c, 150
 - rangedec.c, 170
- ec_decode
 - entdec.h, 100
 - mfrngdec.c, 150
 - rangedec.c, 170
- ec_decode_bin
 - entdec.h, 100
 - mfrngdec.c, 150
 - rangedec.c, 170
- ec_decode_raw
 - entdec.h, 100
 - rangedec.c, 171
- ec_enc, 32
 - buf, 32
 - end_bits_left, 32
 - end_byte, 32
 - entenc.h, 104
 - ext, 32
 - low, 32
 - nb_end_bits, 32
 - rem, 33
 - rng, 33
- ec_enc_bits
 - entenc.c, 103
 - entenc.h, 104
- ec_enc_done
 - entenc.h, 104
 - mfrngenc.c, 151
 - rangeenc.c, 172
- ec_enc_init
 - entenc.h, 105
 - mfrngenc.c, 151
 - rangeenc.c, 172
- ec_enc_tell
 - entenc.h, 105
 - mfrngenc.c, 151
 - rangeenc.c, 172
- ec_enc_uint
 - entenc.c, 103
 - entenc.h, 105
- ec_encode
 - entenc.h, 105
 - mfrngenc.c, 151
 - rangeenc.c, 172
- ec_encode_bin
 - entenc.h, 105
 - mfrngenc.c, 151
 - rangeenc.c, 172
- ec_encode_raw
 - entenc.h, 105

- rangeenc.c, 173
- EC_ILOG
 - ecintrin.h, 90
- ec_ilog
 - entcode.c, 92
 - entcode.h, 96
- EC_ILOG64
 - ecintrin.h, 90
- ec_int32
 - entcode.h, 94
- ec_laplace_decode
 - laplace.c, 139
 - laplace.h, 141
- ec_laplace_decode_start
 - laplace.c, 139
 - laplace.h, 141
- ec_laplace_encode
 - laplace.c, 139
 - laplace.h, 141
- ec_laplace_encode_start
 - laplace.c, 139
 - laplace.h, 141
- ec_laplace_get_start_freq
 - laplace.c, 140
 - laplace.h, 142
- EC_MAXI
 - ecintrin.h, 90
- EC_MINI
 - ecintrin.h, 90
- EC_SIGNI
 - ecintrin.h, 90
- EC_SIGNMASK
 - ecintrin.h, 91
- EC_SYM_BITS
 - mfrngcod.h, 149
- EC_SYM_MAX
 - mfrngcod.h, 149
- ec_uint32
 - entcode.h, 94
- EC_UNIT_BITS
 - entcode.h, 93
- EC_UNIT_MASK
 - entcode.h, 94
- ecintrin.h
 - _ecintrin_H, 90
 - EC_CLAMPI, 90
 - EC_ILOG, 90
 - EC_ILOG64, 90
 - EC_MAXI, 90
 - EC_MINI, 90
 - EC_SIGNI, 90
 - EC_SIGNMASK, 91
- eMeans
 - quant_bands.c, 166
- enc
 - CELTDecoder, 15
- encode_pulses
 - cwrs.c, 84
 - cwrs.h, 86
- ENCODERFREED
 - celt.c, 72
- ENCODERPARTIAL
 - celt.c, 73
- ENCODERVALID
 - celt.c, 73
- Encoding and decoding, 7
- end_bits_left
 - ec_dec, 30
 - ec_enc, 32
- end_byte
 - ec_dec, 30
 - ec_enc, 32
- end_ptr
 - ec_byte_buffer, 29
- ENER_SCALING
 - arch.h, 48
- ENER_SCALING_1
 - arch.h, 48
- entcode.c
 - ec_ilog, 92
- entcode.h
 - _entcode_H, 93
 - ec_byte_adv1, 94
 - ec_byte_adv4, 94
 - ec_byte_buffer, 94
 - ec_byte_look1, 94
 - ec_byte_look4, 94
 - ec_byte_look_at_end, 94
 - ec_byte_read1, 94
 - ec_byte_read4, 95
 - ec_byte_readinit, 95
 - ec_byte_shrink, 95
 - ec_byte_write1, 95
 - ec_byte_write4, 95
 - ec_byte_write_at_end, 95
 - ec_byte_writeclear, 95
 - ec_byte_writecopy, 95
 - ec_byte_writeinit, 95
 - ec_byte_writeinit_buffer, 95
 - ec_byte_writetrunc, 95
 - ec_ilog, 96
 - ec_int32, 94
 - ec_uint32, 94
 - EC_UNIT_BITS, 93
 - EC_UNIT_MASK, 94
- entdec.c
 - ec_byte_adv1, 97
 - ec_byte_look_at_end, 97

- ec_byte_read1, 97
- ec_byte_readinit, 97
- ec_dec_bits, 97
- ec_dec_uint, 98
- entdec.h
 - _entdec_H, 99
 - ec_dec, 99
 - ec_dec_bits, 99
 - ec_dec_init, 99
 - ec_dec_tell, 100
 - ec_dec_uint, 100
 - ec_dec_update, 100
 - ec_decode, 100
 - ec_decode_bin, 100
 - ec_decode_raw, 100
- entenc.c
 - EC_BUFFER_INCREMENT, 102
 - ec_byte_shrink, 102
 - ec_byte_write1, 102
 - ec_byte_write_at_end, 102
 - ec_byte_writeinit_buffer, 102
 - ec_enc_bits, 103
 - ec_enc_uint, 103
- entenc.h
 - _entenc_H, 104
 - ec_enc, 104
 - ec_enc_bits, 104
 - ec_enc_done, 104
 - ec_enc_init, 105
 - ec_enc_tell, 105
 - ec_enc_uint, 105
 - ec_encode, 105
 - ec_encode_bin, 105
 - ec_encode_raw, 105
- ePredCoef
 - CELTMode, 26
- EPSILON
 - arch.h, 48
- EXPORT
 - celt.h, 80
- ext
 - ec_enc, 32
- EXT32
 - _kiss_fft_guts.h, 43
- EXTEND32
 - arch.h, 49
 - fixed_debug.h, 111
 - fixed_generic.h, 118
- extra_headers
 - CELTHHeader, 22
- EXTRACT16
 - arch.h, 49
 - fixed_debug.h, 111
 - fixed_generic.h, 118
- factors
 - kiss_fft_state, 35
- fft
 - CELTMode, 26
- find_spectral_pitch
 - pitch.c, 158
 - pitch.h, 159
- FINE_OFFSET
 - rate.h, 175
- fits_in32
 - cwrs.c, 84
 - cwrs.h, 86
- fixed_c5x.h
 - celt_ilog2, 107
 - celt_maxabs16, 107
 - MAX16, 107
 - MAX32, 107
 - MIN16, 107
 - MIN32, 108
 - MULT16_16_Q15, 108
 - MULT16_16SU, 108
 - MULT16_32_Q15, 108
 - MULT_16_16, 108
 - OVERRIDE_CELT_ILOG2, 108
 - OVERRIDE_CELT_MAXABS16, 108
 - OVERRIDE_FIND_MAX16, 108
 - VSHR32, 108
- fixed_c6x.h
 - celt_ilog2, 109
 - MULT16_16SU, 109
 - MULT16_32_Q15, 109
 - MULT_16_16, 109
 - OVERRIDE_CELT_ILOG2, 109
- fixed_debug.h
 - ADD16, 111
 - ADD32, 111
 - celt_mips, 115
 - DIV32, 111
 - DIV32_16, 111
 - EXTEND32, 111
 - EXTRACT16, 111
 - HALF32, 111
 - MAC16_16, 112
 - MAC16_16_P13, 112
 - MAC16_16_Q11, 112
 - MAC16_16_Q13, 112
 - MAC16_32_Q11, 112
 - MAC16_32_Q15, 112
 - MIPS_INC, 112
 - MULT16_16, 112
 - MULT16_16_Q15, 112
 - MULT16_16SU, 112
 - MULT16_32_P15, 112
 - MULT16_32_Q11, 113

- MULT16_32_Q12, [113](#)
- MULT16_32_Q13, [113](#)
- MULT16_32_Q14, [113](#)
- MULT16_32_Q15, [113](#)
- MULT16_32_Q16, [113](#)
- MULT16_32_QX, [113](#)
- MULT32_32_Q31, [113](#)
- PDIV32, [113](#)
- PDIV32_16, [113](#)
- PRINT_MIPS, [114](#)
- PSHR, [114](#)
- PSHR16, [114](#)
- PSHR32, [114](#)
- QCONST16, [114](#)
- QCONST32, [114](#)
- ROUND16, [114](#)
- SATURATE16, [114](#)
- SATURATE32, [114](#)
- SHL16, [114](#)
- SHR, [114](#)
- SHR16, [115](#)
- SUB16, [115](#)
- SUB32, [115](#)
- UADD32, [115](#)
- USUB32, [115](#)
- VERIFY_INT, [115](#)
- VERIFY_SHORT, [115](#)
- VERIFY_UINT, [115](#)
- VSHR32, [115](#)
- fixed_generic.h
 - ADD16, [117](#)
 - ADD32, [117](#)
 - DIV32, [117](#)
 - DIV32_16, [117](#)
 - EXTEND32, [118](#)
 - EXTRACT16, [118](#)
 - HALF32, [118](#)
 - MAC16_16, [118](#)
 - MAC16_16_P13, [118](#)
 - MAC16_16_Q11, [118](#)
 - MAC16_16_Q13, [118](#)
 - MAC16_32_Q11, [118](#)
 - MAC16_32_Q15, [118](#)
 - MULT16_16, [119](#)
 - MULT16_16_16, [119](#)
 - MULT16_16_P13, [119](#)
 - MULT16_16_P14, [119](#)
 - MULT16_16_P15, [119](#)
 - MULT16_16_Q11_32, [119](#)
 - MULT16_16_Q13, [119](#)
 - MULT16_16_Q14, [119](#)
 - MULT16_16_Q15, [119](#)
 - MULT16_16SU, [119](#)
 - MULT16_32_P15, [120](#)
 - MULT16_32_Q11, [120](#)
 - MULT16_32_Q12, [120](#)
 - MULT16_32_Q13, [120](#)
 - MULT16_32_Q14, [120](#)
 - MULT16_32_Q15, [120](#)
 - MULT16_32_Q16, [120](#)
 - MULT32_32_Q31, [121](#)
 - MULT32_32_Q32, [121](#)
 - NEG16, [121](#)
 - NEG32, [121](#)
 - PDIV32, [121](#)
 - PDIV32_16, [121](#)
 - PSHR, [121](#)
 - PSHR16, [121](#)
 - PSHR32, [122](#)
 - QCONST16, [122](#)
 - QCONST32, [122](#)
 - ROUND16, [122](#)
 - SATURATE, [122](#)
 - SATURATE16, [122](#)
 - SATURATE32, [122](#)
 - SHL, [122](#)
 - SHL16, [122](#)
 - SHL32, [122](#)
 - SHR, [123](#)
 - SHR16, [123](#)
 - SHR32, [123](#)
 - SUB16, [123](#)
 - SUB32, [123](#)
 - VSHR32, [123](#)
- FLAG_FOLD
 - celt.c, [73](#)
- FLAG_INTRA
 - celt.c, [73](#)
- FLAG_MASK
 - celt.c, [73](#)
- FLAG_NONE
 - celt.c, [73](#)
- FLAG_PITCH
 - celt.c, [73](#)
- FLAG_SHORT
 - celt.c, [73](#)
- FLOAT
 - dump_modes.c, [88](#)
- float2int
 - float_cast.h, [124](#)
- float_cast.h
 - float2int, [124](#)
- fold_decision
 - CELTEncoder, [19](#)
- folding_decision
 - bands.c, [59](#)
 - bands.h, [63](#)
- force_intra

- CELTEncoder, 19
- FRAC_MUL16
 - mathops.h, 144
- frame_size
 - CELTDecoder, 16
 - CELTEncoder, 19
 - CELTHHeader, 23
- FRAMESIZE
 - modes.h, 154
- fromBARK
 - psy.c, 160
- Fs
 - CELTMode, 26
- gain_prod
 - CELTEncoder, 19
- gen_twiddle16
 - c64_fft.c, 67
- gen_twiddle32
 - c64_fft.c, 67
- get_required_bits
 - cwrs.c, 85
 - cwrs.h, 86
- global_stack
 - stack_alloc.h, 179
- GLOBAL_STACK_SIZE
 - arch.h, 49
- HALF32
 - arch.h, 49
 - fixed_debug.h, 111
 - fixed_generic.h, 118
- HALF_OF
 - _kiss_fft_guts.h, 43
- header.c
 - celt_header_from_packet, 125
 - celt_header_init, 125
 - celt_header_to_packet, 125
- header_size
 - CELTHHeader, 23
- i
 - kiss_fft_cpx, 34
 - kiss_twiddle_cpx, 37
- icwrs
 - cwrs.c, 85
- IMAX
 - arch.h, 49
- IMIN
 - arch.h, 49
- IMUL32
 - arch.h, 49
- in_mem
 - CELTEncoder, 19
- INPUT_SHIFT
 - pitch.c, 157
- INT16
 - dump_modes.c, 88
- INT32
 - dump_modes.c, 88
- intra_decision
 - quant_bands.c, 164
 - quant_bands.h, 167
- intra_fold
 - vq.c, 181
 - vq.h, 184
- itwiddle
 - c64_fft_t, 13
- kf_cexp
 - _kiss_fft_guts.h, 43
- kf_cexp2
 - _kiss_fft_guts.h, 44
- KF_SUFFIX
 - kiss_fft.h, 131
- kf_work
 - kiss_fft.c, 129
 - kiss_fft.h, 132, 133
- kfft
 - mdct_lookup, 38
- kfft_double.h
 - cpx32_fft, 126
 - cpx32_fft_alloc, 126
 - cpx32_fft_free, 126
 - cpx32_ifft, 126
- kfft_single.h
 - BITREV, 128
 - real16_fft_alloc, 128
 - real16_fft_free, 128
 - real16_fft_inplace, 128
 - real16_ifft, 128
- ki_work
 - kiss_fft.c, 129
 - kiss_fft.h, 132, 133
- kiss_fft
 - kiss_fft.c, 129
 - kiss_fft.h, 132, 133
- kiss_fft.c
 - kf_work, 129
 - ki_work, 129
 - kiss_fft, 129
 - kiss_fft_alloc, 129
 - kiss_fft_stride, 130
 - kiss_ifft, 130
 - kiss_ifft_stride, 130
- kiss_fft.h
 - CAT_SUFFIX, 131
 - KF_SUFFIX, 131

- kf_work, 132, 133
- ki_work, 132, 133
- kiss_fft, 132, 133
- kiss_fft_alloc, 132, 134
- kiss_fft_cfg, 133
- kiss_fft_free, 132
- KISS_FFT_MALLOC, 132
- kiss_fft_scalar, 132
- kiss_fft_stride, 132, 134
- kiss_ifft, 132, 134
- kiss_ifft_stride, 133, 134
- kiss_twiddle_scalar, 133
- SUF, 133
- kiss_fft_alloc
 - kiss_fft.c, 129
 - kiss_fft.h, 132, 134
- kiss_fft_cfg
 - kiss_fft.h, 133
- KISS_FFT_COS
 - _kiss_fft_guts.h, 44
- kiss_fft_cpx, 34
 - i, 34
 - r, 34
- kiss_fft_free
 - kiss_fft.h, 132
- KISS_FFT_MALLOC
 - kiss_fft.h, 132
- kiss_fft_scalar
 - kiss_fft.h, 132
- KISS_FFT_SIN
 - _kiss_fft_guts.h, 44
- kiss_fft_state, 35
 - bitrev, 35
 - factors, 35
 - nfft, 35
 - scale, 35
 - twiddles, 35
- kiss_fft_stride
 - kiss_fft.c, 130
 - kiss_fft.h, 132, 134
- kiss_fftr
 - kiss_fftr.c, 135
 - kiss_fftr.h, 136, 137
- kiss_fftr.c
 - kiss_fftr, 135
 - kiss_fftr_alloc, 135
 - kiss_fftr_inplace, 135
 - kiss_fftr_twiddles, 135
 - kiss_fftri, 135
- kiss_fftr.h
 - kiss_fftr, 136, 137
 - kiss_fftr_alloc, 136, 137
 - kiss_fftr_cfg, 137
 - kiss_fftr_free, 136
 - kiss_fftr_inplace, 136, 137
 - kiss_fftr_twiddles, 137
 - kiss_fftri, 137
- kiss_fftr_alloc
 - kiss_fftr.c, 135
 - kiss_fftr.h, 136, 137
- kiss_fftr_cfg
 - kiss_fftr.h, 137
- kiss_fftr_free
 - kiss_fftr.h, 136
- kiss_fftr_inplace
 - kiss_fftr.c, 135
 - kiss_fftr.h, 136, 137
- kiss_fftr_state, 36
 - substate, 36
 - super_twiddles, 36
- kiss_fftr_twiddles
 - kiss_fftr.c, 135
 - kiss_fftr.h, 137
- kiss_fftri
 - kiss_fftr.c, 135
 - kiss_fftr.h, 137
- kiss_ifft
 - kiss_fft.c, 130
 - kiss_fft.h, 132, 134
- kiss_ifft_stride
 - kiss_fft.c, 130
 - kiss_fft.h, 133, 134
- kiss_twiddle_cpx, 37
 - i, 37
 - r, 37
- kiss_twiddle_scalar
 - kiss_fft.h, 133
- laplace.c
 - ec_laplace_decode, 139
 - ec_laplace_decode_start, 139
 - ec_laplace_encode, 139
 - ec_laplace_encode_start, 139
 - ec_laplace_get_start_freq, 140
- laplace.h
 - ec_laplace_decode, 141
 - ec_laplace_decode_start, 141
 - ec_laplace_encode, 141
 - ec_laplace_encode_start, 141
 - ec_laplace_get_start_freq, 142
- last_pitch_index
 - CELTDecoder, 16
- libcelt/_kiss_fft_guts.h, 41
- libcelt/arch.h, 45
- libcelt/bands.c, 58
- libcelt/bands.h, 62
- libcelt/c64_fft.c, 66
- libcelt/c64_fft.h, 69

- libcelt/celt.c, 71
- libcelt/celt.h, 76
- libcelt/celt_header.h, 82
- libcelt/celt_types.h, 83
- libcelt/cwrs.c, 84
- libcelt/cwrs.h, 86
- libcelt/dump_modes.c, 88
- libcelt/ecintrin.h, 90
- libcelt/encode.c, 92
- libcelt/encode.h, 93
- libcelt/entdec.c, 97
- libcelt/entdec.h, 99
- libcelt/entenc.c, 102
- libcelt/entenc.h, 104
- libcelt/fixed_c5x.h, 107
- libcelt/fixed_c6x.h, 109
- libcelt/fixed_debug.h, 110
- libcelt/fixed_generic.h, 116
- libcelt/float_cast.h, 124
- libcelt/header.c, 125
- libcelt/kfft_double.h, 126
- libcelt/kfft_single.c, 127
- libcelt/kfft_single.h, 128
- libcelt/kiss_fft.c, 129
- libcelt/kiss_fft.h, 131
- libcelt/kiss_fftr.c, 135
- libcelt/kiss_fftr.h, 136
- libcelt/laplace.c, 139
- libcelt/laplace.h, 141
- libcelt/mathops.h, 143
- libcelt/mdct.c, 145
- libcelt/mdct.h, 147
- libcelt/mfrngcod.h, 148
- libcelt/mfrngdec.c, 150
- libcelt/mfrngenc.c, 151
- libcelt/modes.c, 152
- libcelt/modes.h, 154
- libcelt/os_support.h, 156
- libcelt/pitch.c, 157
- libcelt/pitch.h, 159
- libcelt/psy.c, 160
- libcelt/psy.h, 162
- libcelt/quant_bands.c, 164
- libcelt/quant_bands.h, 167
- libcelt/rangedec.c, 170
- libcelt/rangeenc.c, 172
- libcelt/rate.c, 174
- libcelt/rate.h, 175
- libcelt/stack_alloc.h, 177
- libcelt/testcelt.c, 180
- libcelt/vq.c, 181
- libcelt/vq.h, 183
- LOG2_BITS_PER_CHAR
 - arch.h, 49
- log2_frac
 - cwrs.c, 85
 - cwrs.h, 86
- LOG_MAX_PSEUDO
 - rate.h, 175
- LOG_MAX_PULSES
 - rate.h, 175
- loss_count
 - CELTDecoder, 16
- low
 - ec_enc, 32
- M_PI
 - mdct.c, 145
- MAC16_16
 - arch.h, 50
 - fixed_debug.h, 112
 - fixed_generic.h, 118
- MAC16_16_P13
 - arch.h, 50
 - fixed_debug.h, 112
 - fixed_generic.h, 118
- MAC16_16_Q11
 - arch.h, 50
 - fixed_debug.h, 112
 - fixed_generic.h, 118
- MAC16_16_Q13
 - arch.h, 50
 - fixed_debug.h, 112
 - fixed_generic.h, 118
- MAC16_32_Q11
 - arch.h, 50
 - fixed_debug.h, 112
 - fixed_generic.h, 118
- MAC16_32_Q15
 - arch.h, 50
 - fixed_debug.h, 112
 - fixed_generic.h, 118
- main
 - dump_modes.c, 89
 - testcelt.c, 180
- marker
 - CELTDecoder, 16
 - CELTEncoder, 19
- marker_end
 - CELTMode, 26
- marker_start
 - CELTMode, 26
- MASK32
 - cwrs.c, 84
- mathops.h
 - celt_acos, 143
 - celt_atan, 143
 - celt_cos_norm, 143

- celt_div, 143
- celt_exp, 143
- celt_exp2, 144
- celt_log2, 144
- celt_psqrt, 144
- celt_rcp, 144
- celt_rsqrt, 144
- celt_sqrt, 144
- FRAC_MUL16, 144
- MAX
 - _kiss_fft_guts.h, 44
- MAX16
 - arch.h, 50
 - fixed_c5x.h, 107
- MAX32
 - arch.h, 50
 - fixed_c5x.h, 107
- MAX_PACKET
 - testcelt.c, 180
- MAX_PERIOD
 - modes.h, 155
- MAX_PSEUDO
 - rate.h, 175
- MAX_PULSES
 - rate.h, 176
- MAXFACTORS
 - _kiss_fft_guts.h, 44
- MCHANNELS
 - modes.h, 155
- MDCT
 - modes.h, 155
- mdct
 - CELTMode, 26
- mdct.c
 - M_PI, 145
 - mdct_backward, 145
 - mdct_clear, 145
 - mdct_forward, 145
 - mdct_init, 146
- mdct.h
 - mdct_backward, 147
 - mdct_clear, 147
 - mdct_forward, 147
 - mdct_init, 147
- mdct_backward
 - mdct.c, 145
 - mdct.h, 147
- mdct_clear
 - mdct.c, 145
 - mdct.h, 147
- mdct_forward
 - mdct.c, 145
 - mdct.h, 147
- mdct_init
 - mdct.c, 146
 - mdct.h, 147
- mdct_lookup, 38
 - kfft, 38
 - n, 38
 - trig, 38
- mdctSize
 - CELTMode, 27
- mfrngcod.h
 - _mfrngcode_H, 148
 - EC_CODE_BITS, 148
 - EC_CODE_BOT, 148
 - EC_CODE_CARRY, 148
 - EC_CODE_EXTRA, 148
 - EC_CODE_MASK, 148
 - EC_CODE_SHIFT, 148
 - EC_CODE_TOP, 149
 - EC_SYM_BITS, 149
 - EC_SYM_MAX, 149
- mfrngdec.c
 - ec_dec_init, 150
 - ec_dec_tell, 150
 - ec_dec_update, 150
 - ec_decode, 150
 - ec_decode_bin, 150
- mfrngenc.c
 - ec_enc_done, 151
 - ec_enc_init, 151
 - ec_enc_tell, 151
 - ec_encode, 151
 - ec_encode_bin, 151
- MIN
 - _kiss_fft_guts.h, 44
- MIN16
 - arch.h, 50
 - fixed_c5x.h, 107
- MIN32
 - arch.h, 51
 - fixed_c5x.h, 108
- MIPS_INC
 - fixed_debug.h, 112
- mode
 - CELTDecoder, 16
 - CELTEncoder, 19
- MODEFREED
 - modes.c, 152
- MODEPARTIAL
 - modes.c, 152
- modes.c
 - BARK_BANDS, 152
 - BITALLOC_SIZE, 152
 - check_mode, 153
 - MODEFREED, 152
 - MODEPARTIAL, 152

- MODEVALID, 152
- modes.h
 - CELT_BITSTREAM_VERSION, 154
 - CHANNELS, 154
 - check_mode, 155
 - FRAMESIZE, 154
 - MAX_PERIOD, 155
 - MCHANNELS, 155
 - MDCT, 155
 - OVERLAP, 155
- MODEVALID
 - modes.c, 152
- MULT16_16
 - arch.h, 51
 - fixed_debug.h, 112
 - fixed_generic.h, 119
- MULT16_16_16
 - arch.h, 51
 - fixed_generic.h, 119
- MULT16_16_P13
 - arch.h, 51
 - fixed_generic.h, 119
- MULT16_16_P14
 - arch.h, 51
 - fixed_generic.h, 119
- MULT16_16_P15
 - arch.h, 51
 - fixed_generic.h, 119
- MULT16_16_Q11_32
 - arch.h, 51
 - fixed_generic.h, 119
- MULT16_16_Q13
 - arch.h, 51
 - fixed_generic.h, 119
- MULT16_16_Q14
 - arch.h, 51
 - fixed_generic.h, 119
- MULT16_16_Q15
 - arch.h, 52
 - fixed_c5x.h, 108
 - fixed_debug.h, 112
 - fixed_generic.h, 119
- MULT16_16SU
 - fixed_c5x.h, 108
 - fixed_c6x.h, 109
 - fixed_debug.h, 112
 - fixed_generic.h, 119
- MULT16_32_P15
 - arch.h, 52
 - fixed_debug.h, 112
 - fixed_generic.h, 120
- MULT16_32_Q11
 - arch.h, 52
 - fixed_debug.h, 113
 - fixed_generic.h, 120
- MULT16_32_Q12
 - fixed_debug.h, 113
 - fixed_generic.h, 120
- MULT16_32_Q13
 - arch.h, 52
 - fixed_debug.h, 113
 - fixed_generic.h, 120
- MULT16_32_Q14
 - arch.h, 52
 - fixed_debug.h, 113
 - fixed_generic.h, 120
- MULT16_32_Q15
 - arch.h, 52
 - fixed_c5x.h, 108
 - fixed_c6x.h, 109
 - fixed_debug.h, 113
 - fixed_generic.h, 120
- MULT16_32_Q16
 - arch.h, 52
 - fixed_debug.h, 113
 - fixed_generic.h, 120
- MULT16_32_QX
 - fixed_debug.h, 113
- MULT32_32_Q31
 - arch.h, 52
 - fixed_debug.h, 113
 - fixed_generic.h, 121
- MULT32_32_Q32
 - fixed_generic.h, 121
- MULT_16_16
 - fixed_c5x.h, 108
 - fixed_c6x.h, 109
- n
 - mdct_lookup, 38
- nb_channels
 - CELTHHeader, 23
- nb_end_bits
 - ec_dec, 30
 - ec_enc, 32
- nbAllocVectors
 - CELTMode, 27
- NBCACHE
 - c64_fft.c, 66
- nbEBands
 - CELTMode, 27
- nbShortMdcts
 - CELTMode, 27
- NEG16
 - arch.h, 52
 - fixed_generic.h, 121
- NEG32
 - arch.h, 52

- fixed_generic.h, 121
- nfft
 - c64_fft_t, 13
 - kiss_fft_state, 35
- NORM_SCALING
 - arch.h, 53
- NORM_SCALING_1
 - arch.h, 53
- normalise16
 - pitch.c, 157
- normalise_bands
 - bands.c, 60
 - bands.h, 64
- nrm
 - ec_dec, 30
- oldBandE
 - CELTDecoder, 16
 - CELTEncoder, 19
- os_support.h
 - CELT_COPY, 156
 - CELT_MEMSET, 156
 - CELT_MOVE, 156
- out_mem
 - CELTDecoder, 16
 - CELTEncoder, 19
- OVERLAP
 - modes.h, 155
- overlap
 - CELTDecoder, 16
 - CELTEncoder, 20
 - CELTHHeader, 23
 - CELTMode, 27
- OVERRIDE_CELT_ILOG2
 - fixed_c5x.h, 108
 - fixed_c6x.h, 109
- OVERRIDE_CELT_MAXABS16
 - fixed_c5x.h, 108
- OVERRIDE_FIND_MAX16
 - fixed_c5x.h, 108
- PDIV32
 - arch.h, 53
 - fixed_debug.h, 113
 - fixed_generic.h, 121
- PDIV32_16
 - arch.h, 53
 - fixed_debug.h, 113
 - fixed_generic.h, 121
- PGAIN_SCALING
 - arch.h, 53
- PGAIN_SCALING_1
 - arch.h, 53
- PI
 - c64_fft.c, 66
- pitch.c
 - find_spectral_pitch, 158
 - INPUT_SHIFT, 157
 - normalise16, 157
 - pitch_state_alloc, 158
 - pitch_state_free, 158
- pitch.h
 - find_spectral_pitch, 159
 - pitch_state_alloc, 159
 - pitch_state_free, 159
- pitch_available
 - CELTEncoder, 20
- pitch_enabled
 - CELTEncoder, 20
- pitch_permitted
 - CELTEncoder, 20
- pitch_state_alloc
 - pitch.c, 158
 - pitch.h, 159
- pitch_state_free
 - pitch.c, 158
 - pitch.h, 159
- pitchEnd
 - CELTMode, 27
- preemph_memD
 - CELTDecoder, 16
 - CELTEncoder, 20
- preemph_memE
 - CELTEncoder, 20
- PRINT_MIPS
 - arch.h, 53
 - fixed_debug.h, 114
- prob
 - CELTMode, 27
- PSHR
 - arch.h, 53
 - fixed_debug.h, 114
 - fixed_generic.h, 121
- PSHR16
 - arch.h, 53
 - fixed_debug.h, 114
 - fixed_generic.h, 121
- PSHR32
 - arch.h, 53
 - fixed_debug.h, 114
 - fixed_generic.h, 122
- psy
 - CELTMode, 28
- psy.c
 - compute_masking, 160
 - fromBARK, 160
 - psydecay_clear, 160
 - psydecay_init, 161

- toBARK, 160
- psy.h
 - compute_masking, 162
 - compute_mdct_masking, 162
 - compute_tonality, 162
 - psydecay_clear, 162
 - psydecay_init, 162
- PsyDecay, 39
 - decayR, 39
- psydecay_clear
 - psy.c, 160
 - psy.h, 162
- psydecay_init
 - psy.c, 161
 - psy.h, 162
- ptr
 - ec_byte_buffer, 29
- PUSH
 - stack_alloc.h, 178
- Q15_ONE
 - arch.h, 54
- Q15_ONE_1
 - arch.h, 54
- Q15ONE
 - arch.h, 54
- Q30ONE
 - arch.h, 54
- QCONST16
 - arch.h, 54
 - fixed_debug.h, 114
 - fixed_generic.h, 122
- QCONST32
 - arch.h, 54
 - fixed_debug.h, 114
 - fixed_generic.h, 122
- QTHETA_OFFSET
 - rate.h, 176
- quant_bands
 - bands.c, 60
 - bands.h, 64
- quant_bands.c
 - eMeans, 166
 - intra_decision, 164
 - quant_coarse_energy, 164
 - quant_energy_finalise, 165
 - quant_fine_energy, 165
 - quant_prob_alloc, 165
 - quant_prob_free, 165
 - unquant_coarse_energy, 165
 - unquant_energy_finalise, 165
 - unquant_fine_energy, 166
- quant_bands.h
 - compute_fine_allocation, 167
 - intra_decision, 167
 - quant_coarse_energy, 167
 - quant_energy_finalise, 167
 - quant_fine_energy, 168
 - quant_prob_alloc, 168
 - quant_prob_free, 168
 - unquant_coarse_energy, 168
 - unquant_energy_finalise, 168
 - unquant_fine_energy, 168
- quant_bands_stereo
 - bands.c, 60
 - bands.h, 64
- quant_coarse_energy
 - quant_bands.c, 164
 - quant_bands.h, 167
- quant_energy_finalise
 - quant_bands.c, 165
 - quant_bands.h, 167
- quant_fine_energy
 - quant_bands.c, 165
 - quant_bands.h, 168
- quant_prob_alloc
 - quant_bands.c, 165
 - quant_bands.h, 168
- quant_prob_free
 - quant_bands.c, 165
 - quant_bands.h, 168
- r
 - kiss_fft_cpx, 34
 - kiss_twiddle_cpx, 37
- rangedec.c
 - ec_dec_init, 170
 - ec_dec_tell, 170
 - ec_dec_update, 170
 - ec_decode, 170
 - ec_decode_bin, 170
 - ec_decode_raw, 171
- rangeenc.c
 - ec_enc_done, 172
 - ec_enc_init, 172
 - ec_enc_tell, 172
 - ec_encode, 172
 - ec_encode_bin, 172
 - ec_encode_raw, 173
- rate.c
 - compute_alloc_cache, 174
 - compute_allocation, 174
- rate.h
 - BITOVERFLOW, 175
 - BITRES, 175
 - compute_alloc_cache, 176
 - compute_allocation, 176
 - FINE_OFFSET, 175

- LOG_MAX_PSEUDO, 175
- LOG_MAX_PULSES, 175
- MAX_PSEUDO, 175
- MAX_PULSES, 176
- QTHETA_OFFSET, 176
- real16_fft_alloc
 - kfft_single.h, 128
- real16_fft_free
 - kfft_single.h, 128
- real16_fft_inplace
 - kfft_single.h, 128
- real16_ifft
 - kfft_single.h, 128
- rem
 - ec_dec, 31
 - ec_enc, 33
- renormalise_bands
 - bands.c, 61
 - bands.h, 64
- renormalise_vector
 - vq.c, 182
 - vq.h, 184
- RESTORE_STACK
 - stack_alloc.h, 178
- rng
 - ec_dec, 31
 - ec_enc, 33
- ROUND16
 - arch.h, 54
 - fixed_debug.h, 114
 - fixed_generic.h, 122
- S_MUL
 - _kiss_fft_guts.h, 44
- sample_rate
 - CELTHHeader, 23
- SATURATE
 - arch.h, 54
 - fixed_generic.h, 122
- SATURATE16
 - arch.h, 54
 - fixed_debug.h, 114
 - fixed_generic.h, 122
- SATURATE32
 - arch.h, 55
 - fixed_debug.h, 114
 - fixed_generic.h, 122
- SAVE_STACK
 - stack_alloc.h, 178
- scale
 - kiss_fft_state, 35
- SCALEIN
 - arch.h, 55
- SCALEOUT
 - arch.h, 55
- shift
 - c64_fft_t, 13
- SHL
 - arch.h, 55
 - fixed_generic.h, 122
- SHL16
 - arch.h, 55
 - fixed_debug.h, 114
 - fixed_generic.h, 122
- SHL32
 - arch.h, 55
 - fixed_generic.h, 122
- shortMdct
 - CELTMode, 28
- shortMdctSize
 - CELTMode, 28
- shortWindow
 - CELTMode, 28
- SHR
 - arch.h, 55
 - fixed_debug.h, 114
 - fixed_generic.h, 123
- SHR16
 - arch.h, 55
 - fixed_debug.h, 115
 - fixed_generic.h, 123
- SHR32
 - arch.h, 55
 - fixed_generic.h, 123
- stack_alloc.h
 - ALIGN, 177
 - ALLOC, 177
 - ALLOC_STACK, 178
 - global_stack, 179
 - PUSH, 178
 - RESTORE_STACK, 178
 - SAVE_STACK, 178
 - VARDECL, 178
- stereo_decision
 - bands.h, 64
- storage
 - ec_byte_buffer, 29
- SUB16
 - arch.h, 56
 - fixed_debug.h, 115
 - fixed_generic.h, 123
- SUB32
 - arch.h, 56
 - fixed_debug.h, 115
 - fixed_generic.h, 123
- substate
 - kiss_fftr_state, 36
- SUF

- kiss_fft.h, 133
- super_twiddles
 - kiss_fftr_state, 36
- testcelt.c
 - main, 180
 - MAX_PACKET, 180
- toBARK
 - psy.c, 160
- tonal_average
 - CELTEncoder, 20
- trig
 - mdct_lookup, 38
- twiddle
 - c64_fft_t, 13
- twiddles
 - kiss_fft_state, 35
- UADD32
 - arch.h, 56
 - fixed_debug.h, 115
- UMUL16_16
 - arch.h, 56
- UMUL32
 - arch.h, 56
- unquant_bands
 - bands.h, 65
- unquant_bands_stereo
 - bands.c, 61
 - bands.h, 65
- unquant_coarse_energy
 - quant_bands.c, 165
 - quant_bands.h, 168
- unquant_energy_finalise
 - quant_bands.c, 165
 - quant_bands.h, 168
- unquant_fine_energy
 - quant_bands.c, 166
 - quant_bands.h, 168
- USUB32
 - arch.h, 56
 - fixed_debug.h, 115
- VARDECL
 - stack_alloc.h, 178
- vbr_count
 - CELTEncoder, 20
- vbr_drift
 - CELTEncoder, 20
- vbr_offset
 - CELTEncoder, 21
- vbr_rate
 - CELTEncoder, 21
- vbr_reservoir
 - CELTEncoder, 21
- VERIFY_INT
 - fixed_debug.h, 115
- VERIFY_SHORT
 - fixed_debug.h, 115
- VERIFY_UINT
 - fixed_debug.h, 115
- version_id
 - CELHeader, 23
- VERY_LARGE16
 - arch.h, 56
- VERY_LARGE32
 - arch.h, 56
- VERY_SMALL
 - arch.h, 56
- vq.c
 - alg_quant, 181
 - alg_unquant, 181
 - intra_fold, 181
 - renormalise_vector, 182
- vq.h
 - alg_quant, 183
 - alg_unquant, 183
 - intra_fold, 184
 - renormalise_vector, 184
- VSHR32
 - arch.h, 56
 - fixed_c5x.h, 108
 - fixed_debug.h, 115
 - fixed_generic.h, 123
- window
 - CELTMode, 28
- WORD16
 - dump_modes.c, 88
- WORD32
 - dump_modes.c, 88