Technical Specification Group Services and System Aspects *TSGS#27 (05)0083*
Meeting #27, Tokyo, Japan, 14-17 March 2005

Source: **TSG-SA WG4**

Title: 3GPP TS 26.273 version 2.0.0 "ANSI-C code for the Fixed-point Extended AMR Wideband codec" (Release 6)

Agenda Item: **7.4.3**

Presentation of Specification to TSG SA Plenary

Presentation to: **TSG SA Meeting #27**

Document for presentation: **TS 26.273 "ANSI-C code for the Fixed-point Extended AMR Wideband codec", Version 2.0.0 (Release 6)**

Presented for: **Approval**

Abstract of document:

The present document contains an electronic copy of the ANSI-C code for the Fixed-point Extended Adaptive Multi-Rate Wideband codec. Alternatively, floating-point ANSI-C code is specified in 3GPP TS 26.304. The fixed-point codec/encoder/decoder specified in this document or the floating-point codec/encoder/decoder may be used depending on if the implementation platform is better suited for a floating-point or a fixed-point implementation. It has been verified that the fixed-point and floating-point codecs interoperate with each other without any artifacts.

The fixed-point ANSI-C code in the present document defines, besides the floating-point c-code specified in 3GPP TS 26.304, one valid reference implementation of the Extended Adaptive Multi-Rate Wideband transcoder (3GPP TS 26.290). Standard conformance is enforced by meeting the conformance criteria defined in TS 26.274.

The present document includes information applicable to network operators, service providers and manufacturers.

Changes since last presentation:

This specification is presented to TSG SA Plenary for approval: version 1.0.0 of the specification was provided at TSG SA#24 for information. The specification is now considered stable.

Outstanding Issues:

None.

Contentious Issues:

None.

Comment(s):

None.

3GPP TS 26.273 V2.0.0 (2005-03)

Technical Specification

3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; ANSI-C code for the Fixed-point Extended AMR Wideband codec (Release 6)

The present document has been developed within the 3rd Generation Partnership Project (3GPP ™) and may be further elaborated for the purposes of 3GPP.

The present document has not been subject to any approval process by the 3GPP Organizational Partners and shall not be implemented.
This Specification is provided for future development work within 3GPP only. The Organizational Partners accept no liability for any use of this Specification.
Specifications and reports for implementation of the 3GPP ™ system should be obtained via the 3GPP Organizational Partners' Publications Offices.

Keywords

AMR-WB, AMR-WB+, audio CODEC, Extended AMR Wideband codec

3GPP

Postal address

3GPP support office address

650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Internet

http://www.3gpp.org

Copyright Notification

No part may be reproduced except as authorized by written permission.
The copyright and the foregoing restriction extend to reproduction in all media.

© 2005, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TTA, TTC).
All rights reserved.

Contents

Foreword....4
1 Scope....5
2 References....5
3 Definitions and abbreviations....5
3.1 Definitions....5
3.2 Abbreviations....5
4 C code structure....6
4.1 Contents of the C source code....6
4.2 Program execution....6
4.3 Code hierarchy....6
4.5 Variables, constants and tables....19
4.5.1 Description of fixed tables used in the C-code....19
4.5.2 Static variables used in the C-code....20
5 File formats....25
5.1 Audio file (encoder input/decoder output)....25
5.2 Parameter bitstream file (encoder output/decoder input)....25
Annex A (informative): Change history....26

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

1 Scope

The present document contains an electronic copy of the ANSI-C code for the Fixed-point Extended Adaptive Multi-Rate Wideband codec. Alternatively, floating-point ANSI-C code is specified in 3GPP TS 26.304 [1]. The floating-point codec/encoder/decoder specified in the present document or the floating-point codec/encoder/decoder specified in 3GPP TS 26.304 [1] may be used depending on if the implementation platform is better suited for a floating-point or a fixed-point implementation. It has been verified that the fixed-point and floating-point codecs interoperate with each other without any artifacts.

The fixed-point ANSI-C code in the present document defines, besides the floating-point c-code specified in 3GPP TS 26.304 [1], one valid reference implementation of the Extended Adaptive Multi-Rate Wideband transcoder (3GPP TS 26.290 [2]). Standard conformance it is enforced by meeting the conformance criteria defined in 3GPP TS 26.274 [3].

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TS 26.304: "Extended Adaptive Multi-Rate - Wideband (AMR-WB+) codec; Floating-point ANSI-C code".

[2] 3GPP TS 26.290: "Audio codec processing functions; Extended Adaptive Multi-Rate - Wideband (AMR-WB+) codec; Transcoding functions".

[3] 3GPP TS 26.274: "Speech codec speech processing functions; Extended Adaptive Multi-Rate - Wideband (AMR-WB+) speech codec; Conformance testing".

[4] 3GPP TS 26.244: "Transparent end-to-end packet switched streaming service (PSS); 3GPP file format (3GP)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions are given in 3GPP TS 26.290 [1].

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AMR-WB+	Extended Adaptive Multi-Rate WideBand
ANSI	American National Standards Institute
ETS	European Telecommunication Standard
GSM	Global System for Mobile communications
I/O	Input/Output
RAM	Random Access Memory

ROM Read Only Memory

4 C code structure

This clause gives an overview of the structure of the bit-exact C code and provides an overview of the contents and organization of the C code attached to the present document.

The C code has been verified on the following systems:

- IBM PC/AT compatible computers with Windows 2000 SP4 and Microsoft Visual C++ v.6.0 compiler.

ANSI-C was selected as the programming language because portability was desirable.

4.1 Contents of the C source code

The C code distribution has the files divided in six different directories, all present in the directory *c-code*. The directories are: *common_fx, decoder_fx, encoder_fx, lib_amr_fx include_fx* and *basic_op*. The distributed files with suffix "c" contain the source code and the files with suffix "h" are the header files.

Project and workspace files are provided in the directory *MSVC*.

4.2 Program execution

The Extended Adaptive Multi-Rate Wideband codec is implemented in two programs:

- (*encoder*) audio encoder;
- (*decoder*) audio decoder.

The programs should be called like:

- encoder [encoder options] -if <audio input file> -of <parameter file>;
- decoder [decoder options] -if <parameter file> -of <audio output file>.

The input files contain one or two channels of 16-bit linear encoded PCM audio samples stored in the *wav* file format and the parameter files contain encoded audio data and some additional flags.

The encoder and decoder options will be explained by running the applications without input arguments. See the file readme.txt for more information on how to run the *encoder* and *decoder* programs.

4.3 Code hierarchy

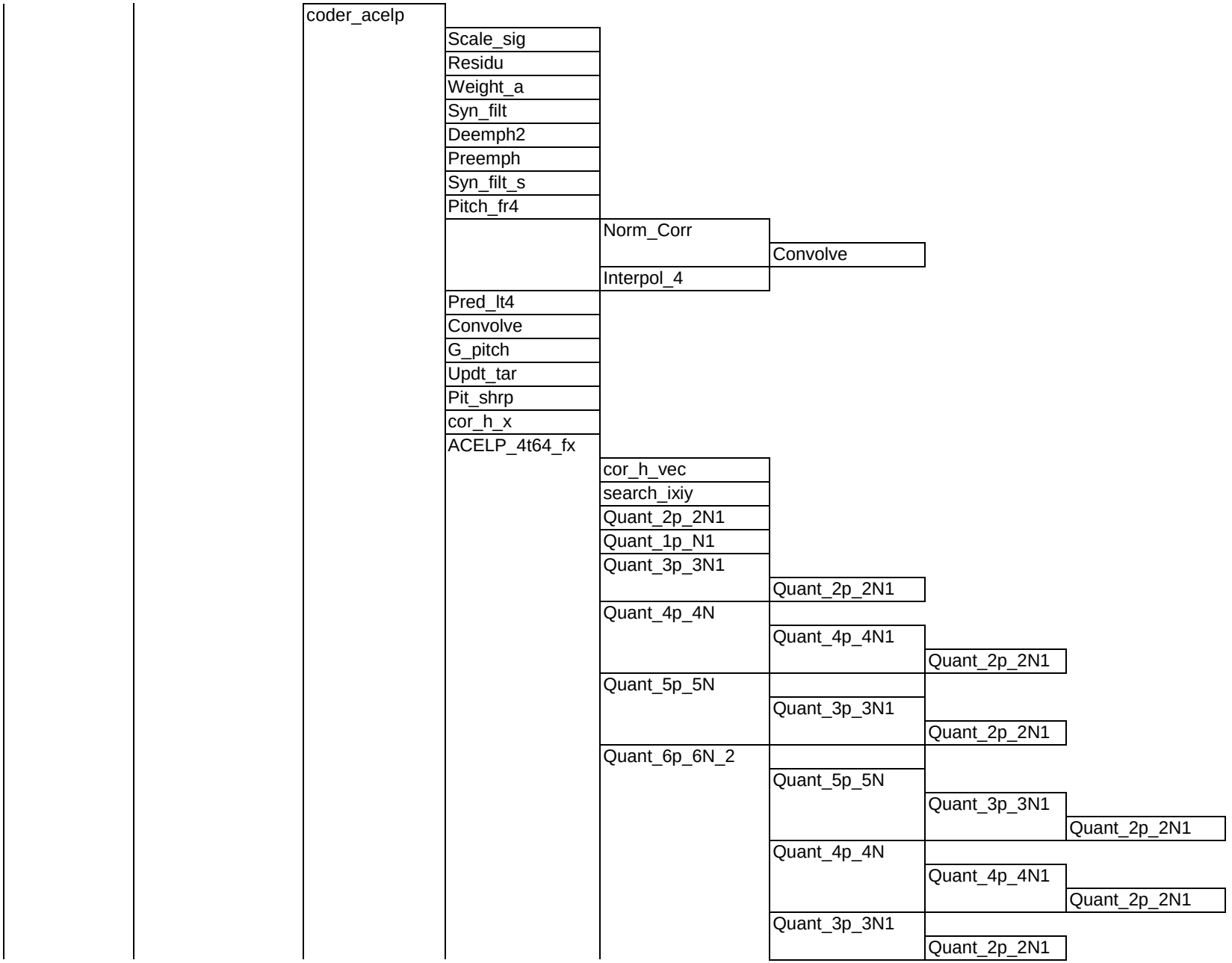
Tables 1 and 2 are call graphs that show the functions used in the audio codec.

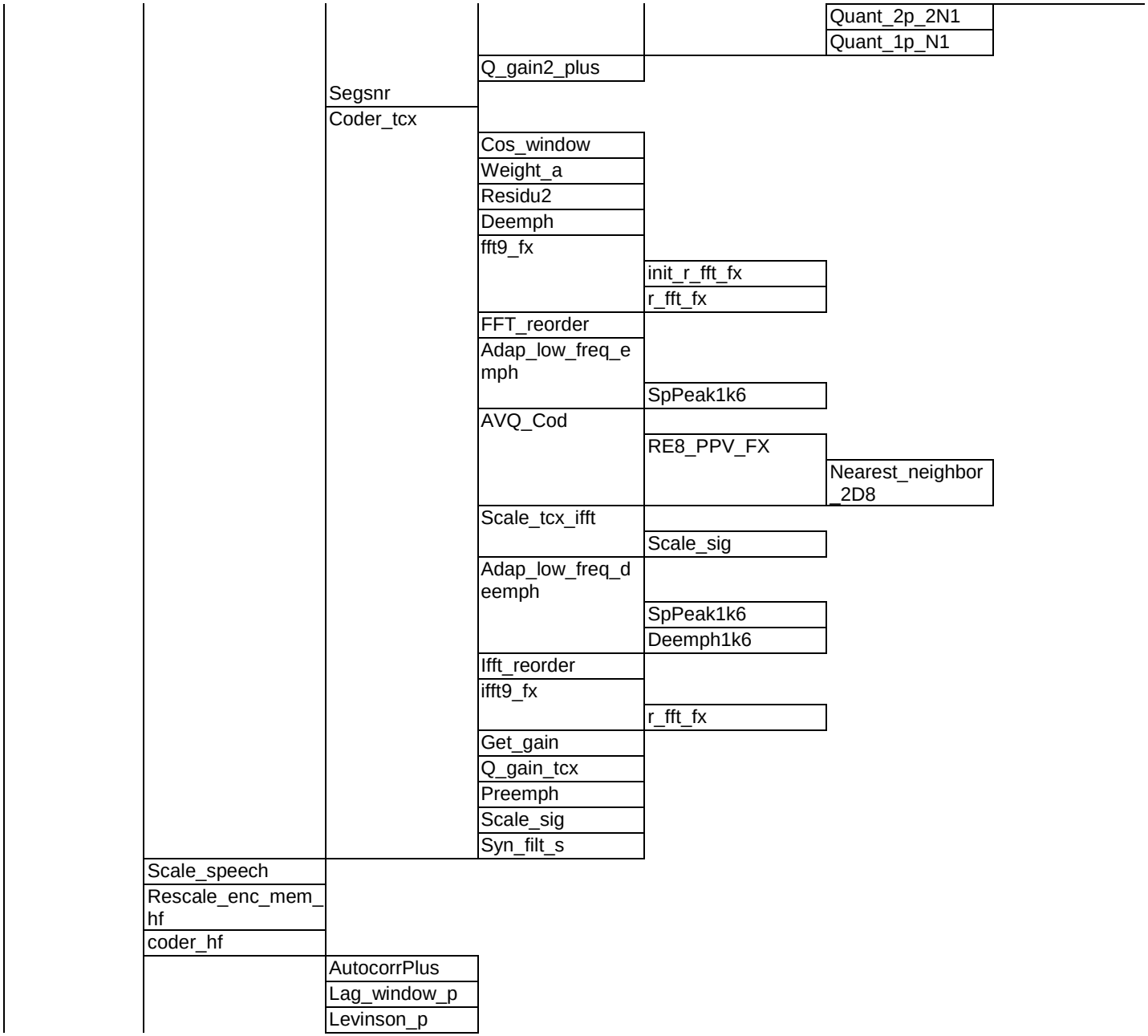
Each column represents a call level and each cell a function. The functions contain calls to the functions in rightwards neighbouring cells. The time order in the call graphs is from the top downwards as the processing of a frame advances. All standard C functions: printf(), fwrite(), etc. have been omitted. Also, no basic operations (add(), L_add(), mac(), etc.) or double precision extended operations (e.g. L_Extract(), Copy() or Set_zero()) appear in the graphs. The initialization of the static RAM (i.e. calling the _init functions) is also omitted.

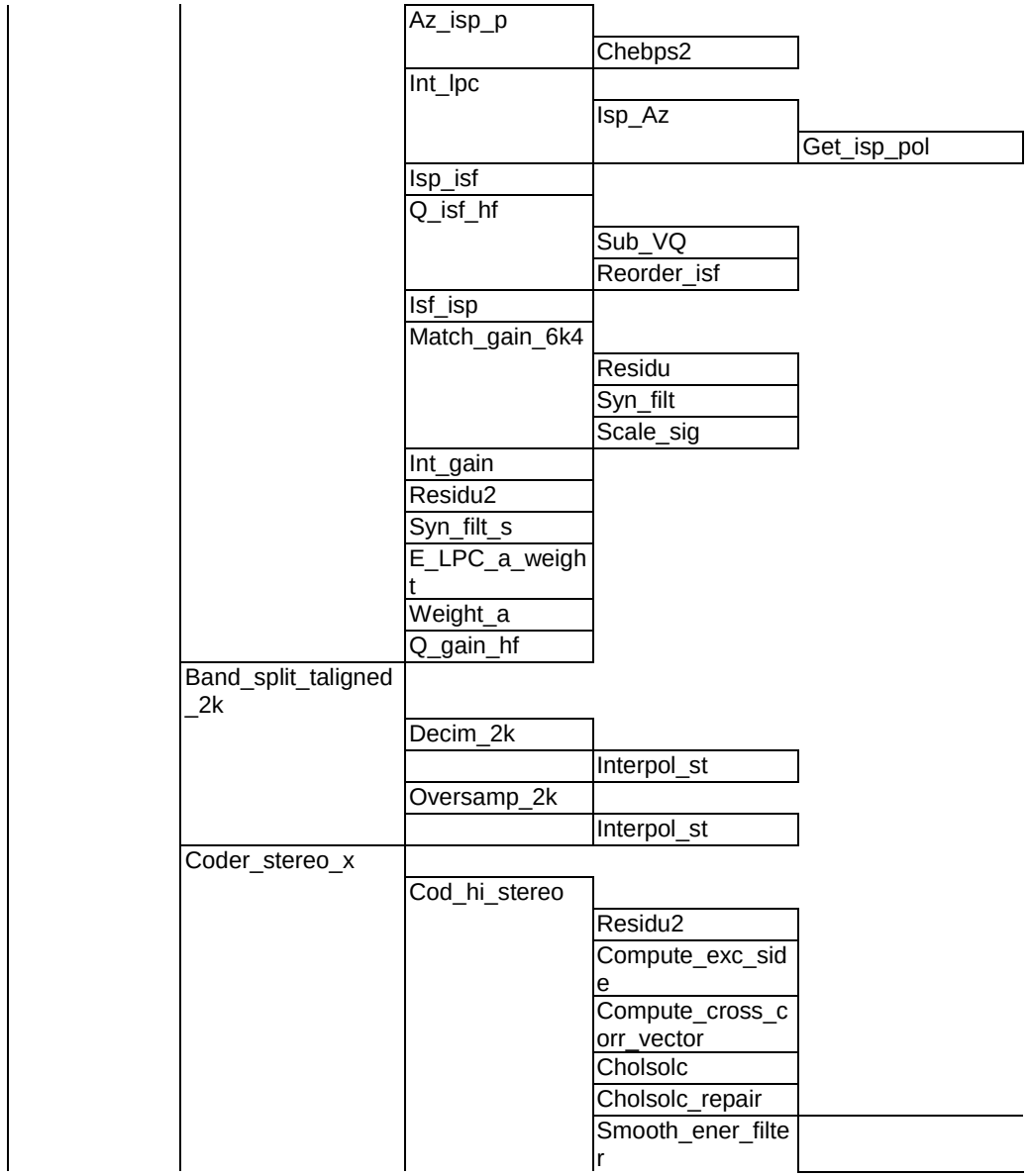
The basic operations are not counted as extending the depth, therefore the deepest level in this software is level 6.

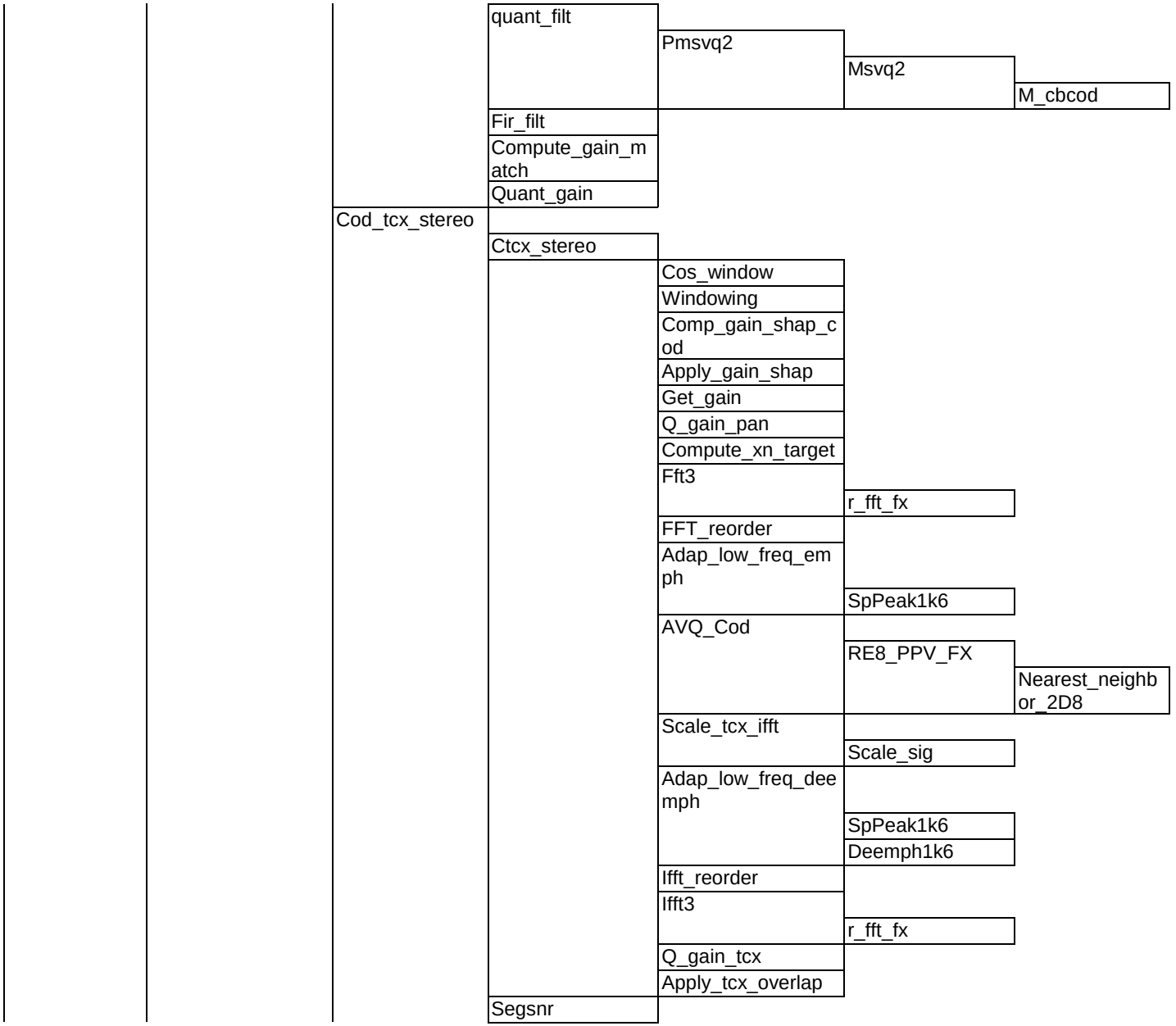
Table 1: Encoder call structure

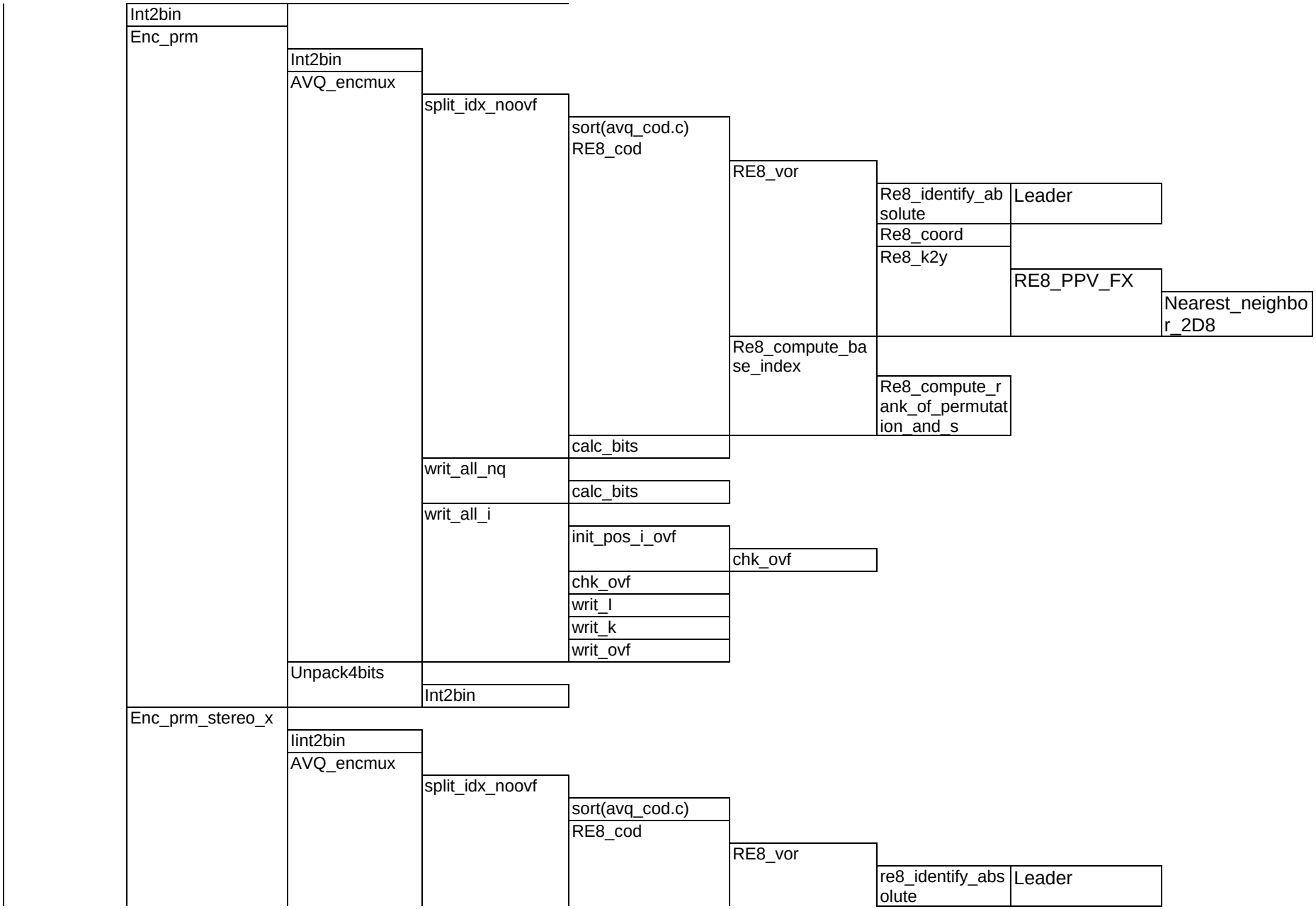
- coder_amrwb_plus_stereo
  - Scale_speech_st
  - Rescale_enc_mem_st
    - Scale_sig
  - Decim_12k8_p
    - Interpol
  - HP50_12k8_p
  - Mix_ch
  - Preemph_scaled
  - Rescale_enc_mem_st
  - Coder_lf
    - AutocorrPlus
    - Lag_window_p
    - Levinson
    - Az_isp
      - Chebps2
    - Int_lpc
      - Isp_Az
        - Get_isp_pol
    - Find_wsp
      - Weight_a
      - Residu2
      - Deemph
    - LP_Decim2
    - Scale_sig
    - scale_mem_Hp_wsp
    - Pitch_med_ol
      - Hp_wsp
    - Isp_isf
    - Qpisf_2s_46b
      - VQ_stage1
      - Sub_VQ
      - Dpisf_2s_46b
        - Reorder_isf


coder_acelp
Scale_sig
Residu
Weight_a
Syn_filt
Deemph2
Preemph
Syn_filt_s
Pitch_fr4
Norm_Corr
Convolve
Interpol_4
Pred_lt4
Convolve
G_pitch
Updt_tar
Pit_shrp
cor_h_x
ACELP_4t64_fx
cor_h_vec
search_ixiy
Quant_2p_2N1
Quant_1p_N1
Quant_3p_3N1
Quant_2p_2N1
Quant_4p_4N
Quant_4p_4N1
Quant_2p_2N1
Quant_5p_5N
Quant_3p_3N1
Quant_2p_2N1
Quant_6p_6N_2
Quant_5p_5N
Quant_3p_3N1
Quant_2p_2N1
Quant_4p_4N
Quant_4p_4N1
Quant_2p_2N1
Quant_3p_3N1
Quant_2p_2N1


Quant_2p_2N1
Quant_1p_N1
Q_gain2_plus
Segsnr
Coder_tcx
Cos_window
Weight_a
Residu2
Deemph
fft9_fx
init_r_fft_fx
r_fft_fx
FFT_reorder
Adap_low_freq_emph
SpPeak1k6
AVQ_Cod
RE8_PPV_FX
Nearest_neighbor_2D8
Scale_tcx_ifft
Scale_sig
Adap_low_freq_deemph
SpPeak1k6
Deemph1k6
Ifft_reorder
ifft9_fx
r_fft_fx
Get_gain
Q_gain_tcx
Preemph
Scale_sig
Syn_filt_s
Scale_speech
Rescale_enc_mem_hf
coder_hf
AutocorrPlus
Lag_window_p
Levinson_p


Az_isp_p
Chebps2
Int_lpc
Isp_Az
Get_isp_pol
Isp_isf
Q_isf_hf
Sub_VQ
Reorder_isf
Isf_isp
Match_gain_6k4
Residu
Syn_filt
Scale_sig
Int_gain
Residu2
Syn_filt_s
E_LPC_a_weigh t
Weight_a
Q_gain_hf
Band_split_taligned _2k
Decim_2k
Interpol_st
Oversamp_2k
Interpol_st
Coder_stereo_x
Cod_hi_stereo
Residu2
Compute_exc_sid e
Compute_cross_c orr_vector
Cholsolc
Cholsolc_repair
Smooth_ener_filte r


quant_filt
Pmsvq2
Msvq2
M_cbcod
Fir_filt
Compute_gain_match
Quant_gain
Cod_tcx_stereo
Ctcx_stereo
Cos_window
Windowing
Comp_gain_shap_cod
Apply_gain_shap
Get_gain
Q_gain_pan
Compute_xn_target
Fft3
r_fft_fx
FFT_reorder
Adap_low_freq_emph
SpPeak1k6
AVQ_Cod
RE8_PPV_FX
Nearest_neighbor_2D8
Scale_tcx_ifft
Scale_sig
Adap_low_freq_deemph
SpPeak1k6
Deemph1k6
Ifft_reorder
Ifft3
r_fft_fx
Q_gain_tcx
Apply_tcx_overlap
Segsnr


Int2bin
Enc_prm
Int2bin
AVQ_encmux
split_idx_noovf
sort(avq_cod.c)
RE8_cod
RE8_vor
Re8_identify_absolute
Leader
Re8_coord
Re8_k2y
RE8_PPV_FX
Nearest_neighbor_2D8
Re8_compute_base_index
Re8_compute_rank_of_permutation_and_s
calc_bits
writ_all_nq
calc_bits
writ_all_i
init_pos_i_ovf
chk_ovf
chk_ovf
writ_l
writ_k
writ_ovf
Unpack4bits
Int2bin
Enc_prm_stereo_x
Iint2bin
AVQ_encmux
split_idx_noovf
sort(avq_cod.c)
RE8_cod
RE8_vor
re8_identify_absolute
Leader

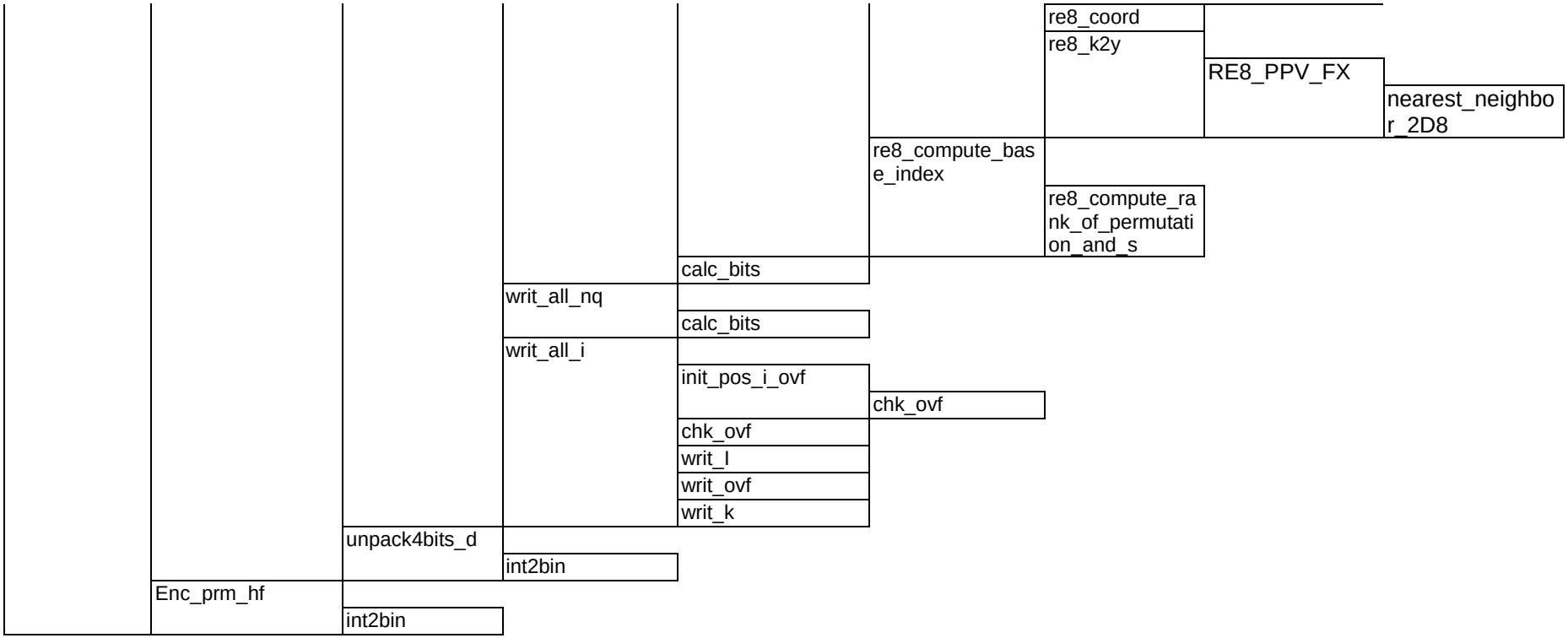
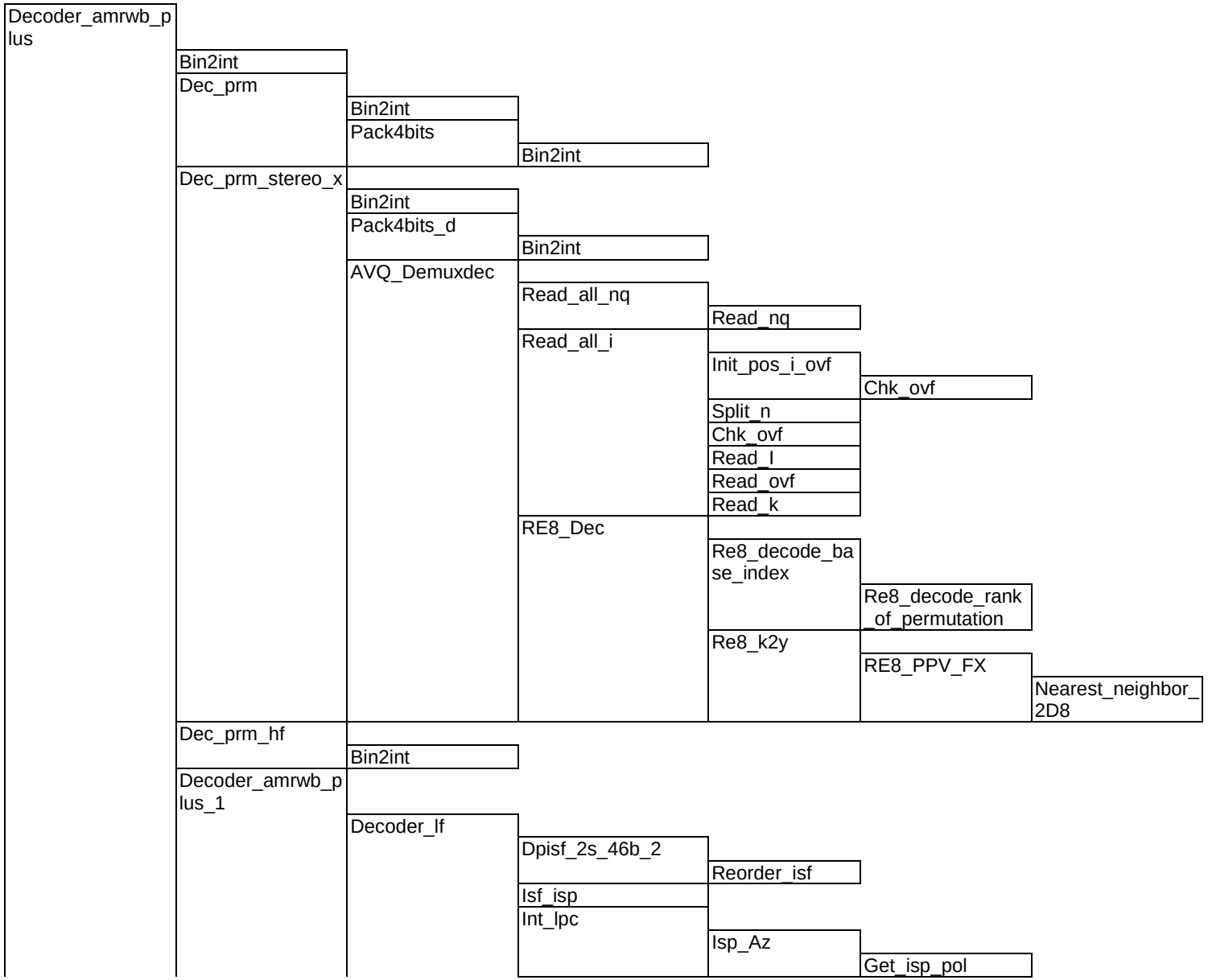
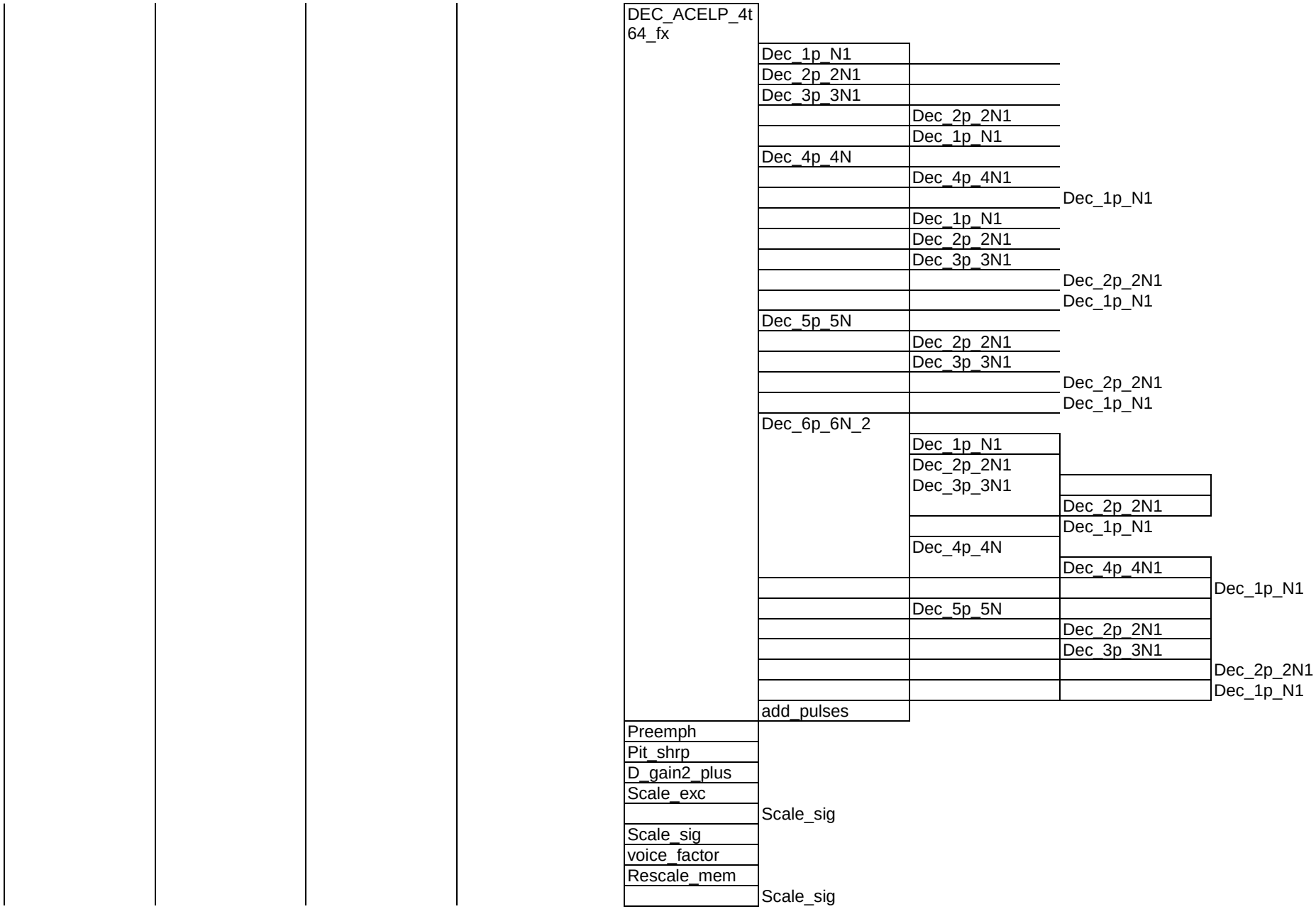
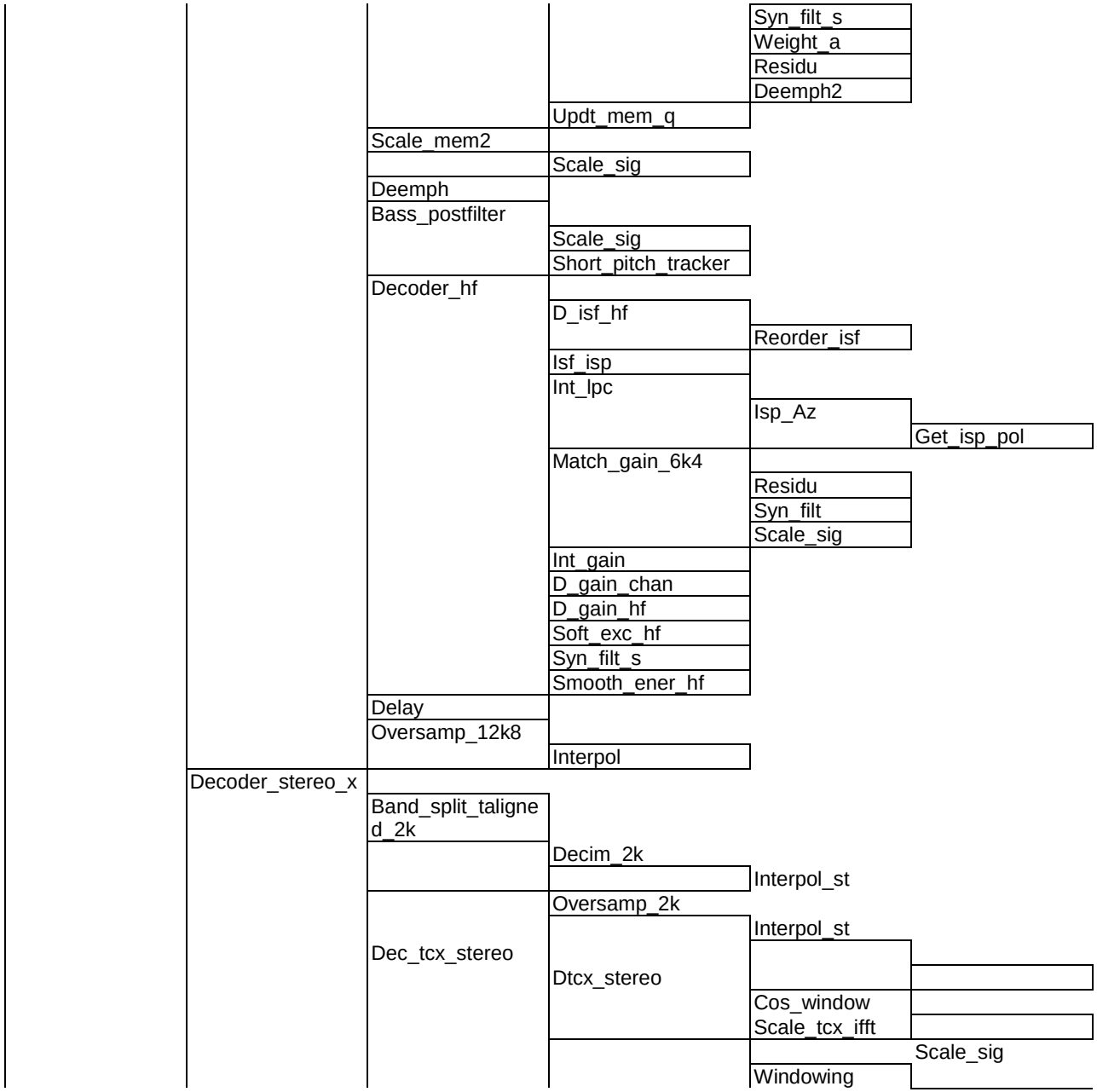

re8_coord
re8_k2y
RE8_PPV_FX
nearest_neighbor_2D8
re8_compute_base_index
re8_compute_rank_of_permutation_and_s
calc_bits
writ_all_nq
calc_bits
writ_all_i
init_pos_i_ovf
chk_ovf
chk_ovf
writ_l
writ_ovf
writ_k
unpack4bits_d
int2bin
Enc_prm_hf
int2bin

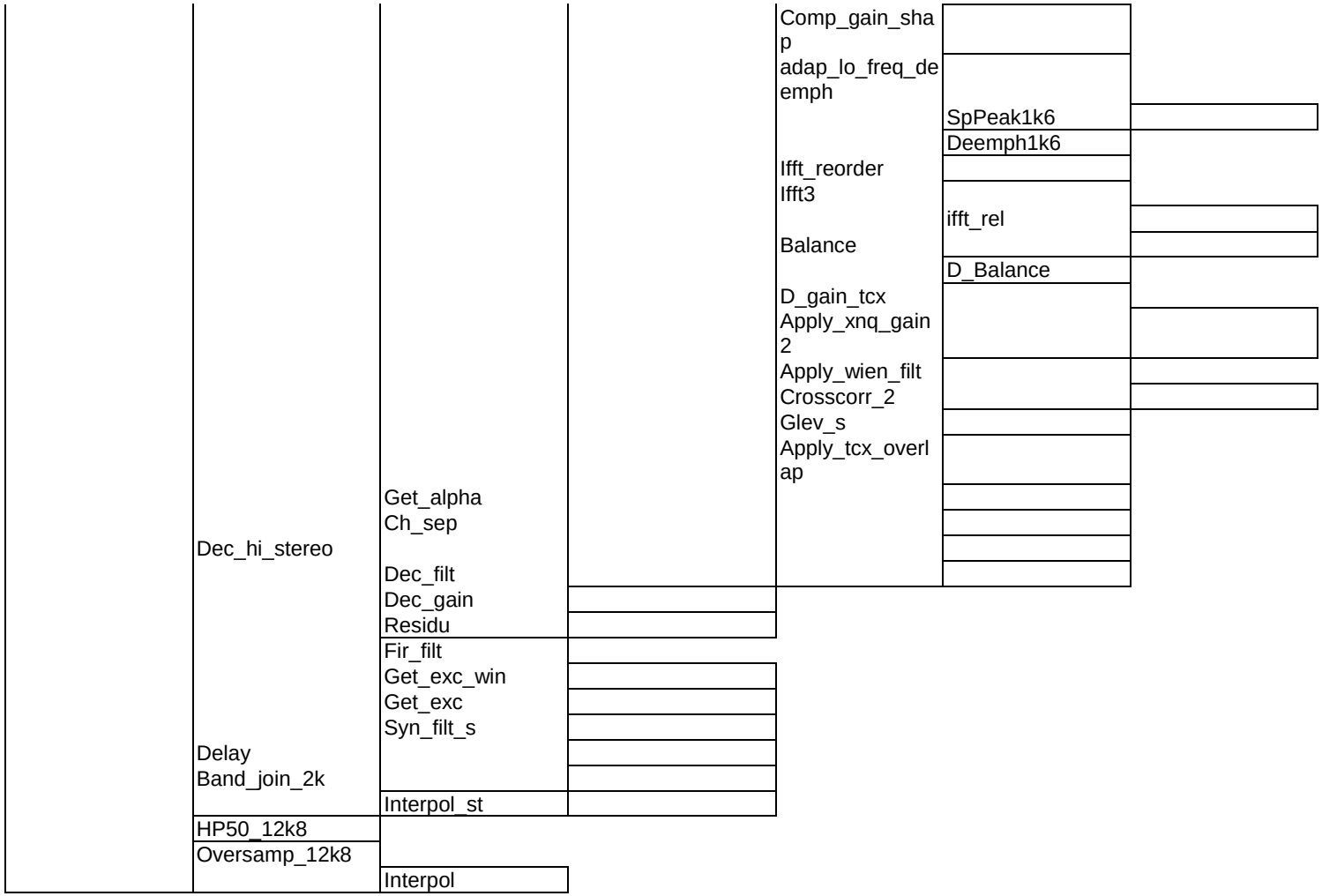
Table 2: Decoder call structure



- decoder_tcx
  - AVQ_Demuxdec
    - Read_all_nq
      - Read_nq
    - Read_all_i
      - Init_pos_i_ovf
        - Chk_ovf
      - Split_n
      - Chk_ovf
      - Read_I
      - Read_k
      - Read_ovf
    - RE8_dec
      - Re8_decode_base_index
        - Re8_decode_rank_of_permutation
      - Re8_k2y
        - RE8_PPV_FX
          - Nearest_neighbor_2D8
  - Scale_tcx_ifft
    - Scale_sig
  - NoiseFill
    - Rnd_ph16
      - Random
  - Adap_low_freq_deemph
    - SpPeak1k6
    - Deemph1k6
  - Find_mpitch
  - Ifft_reorder
  - ifft9_fx
    - r_fft_fx
  - Cos_window
  - D_gain_tcx
  - Scale_mem_tcx
    - Scale_sig
  - Preemph
  - Weight_a
  - Syn_filt_s
  - Residu2
- Decoder_acelp
  - Pred_lt4


DEC_ACELP_4t64_fx
Dec_1p_N1
Dec_2p_2N1
Dec_3p_3N1
Dec_2p_2N1
Dec_1p_N1
Dec_4p_4N
Dec_4p_4N1
Dec_1p_N1
Dec_1p_N1
Dec_2p_2N1
Dec_3p_3N1
Dec_2p_2N1
Dec_1p_N1
Dec_5p_5N
Dec_2p_2N1
Dec_3p_3N1
Dec_2p_2N1
Dec_1p_N1
Dec_6p_6N_2
Dec_1p_N1
Dec_2p_2N1
Dec_3p_3N1
Dec_2p_2N1
Dec_1p_N1
Dec_4p_4N
Dec_4p_4N1
Dec_1p_N1
Dec_5p_5N
Dec_2p_2N1
Dec_3p_3N1
Dec_2p_2N1
Dec_1p_N1
add_pulses
Preemph
Pit_shrp
D_gain2_plus
Scale_exc
Scale_sig
Scale_sig
voice_factor
Rescale_mem
Scale_sig


Syn_filt_s
Weight_a
Residu
Deemph2
Updt_mem_q
Scale_mem2
Scale_sig
Deemph
Bass_postfilter
Scale_sig
Short_pitch_tracker
Decoder_hf
D_isf_hf
Reorder_isf
Isf_isp
Int_lpc
Isp_Az
Get_isp_pol
Match_gain_6k4
Residu
Syn_filt
Scale_sig
Int_gain
D_gain_chan
D_gain_hf
Soft_exc_hf
Syn_filt_s
Smooth_ener_hf
Delay
Oversamp_12k8
Interpol
Decoder_stereo_x
Band_split_taligned_2k
Decim_2k
Interpol_st
Oversamp_2k
Interpol_st
Dec_tcx_stereo
Dtcx_stereo
Cos_window
Scale_tcx_ifft
Scale_sig
Windowing


Comp_gain_shap
adap_lo_freq_deemph
SpPeak1k6
Deemph1k6
Ifft_reorder
Ifft3
ifft_rel
Balance
D_Balance
D_gain_tcx
Apply_xnq_gain2
Apply_wien_filt
Crosscorr_2
Glev_s
Apply_tcx_overlap
Get_alpha
Ch_sep
Dec_hi_stereo
Dec_filt
Dec_gain
Residu
Fir_filt
Get_exc_win
Get_exc
Syn_filt_s
Delay
Band_join_2k
Interpol_st
HP50_12k8
Oversamp_12k8
Interpol

4.5 Variables, constants and tables

The data types of variables and tables used in the fixed point implementation are signed integers in 2's complement representation, defined by:

- **Word16** 16 bit variable;
- **Word32** 32 bit variable.

4.5.1 Description of fixed tables used in the C-code

This clause contains a listing of all fixed tables sorted by source file name and table name. All table data is declared as **Word16**.

Table 3: Encoder fixed tables

Format	Table name	Size	Description	Format
Word16	MonoRate	54	Predefined mono rate	Q6/Q0/Q0
Word16	StereoRate	81	Predefined stereo rate	Q6/Q0/Q0
Word16	NBITS_CORE_AMR_WB_FX	9 miMode_fx	AMR WB Core bits	Q0
Word16	miMode_fx	48	Stereo Mode Index	Q0
Word16	isfIndex_fx	14	Internal sampling Frequency	Q0
Word16	NBITS_CORE_FX	8	Core bit-rates	Q0
Word16	t_sinFxS4	1440	FFT Sine&Cos table	Q13
Word16	Filter_32k	61	FIR table for decimation/oversampling	Q13
Word16	Filter_32k_hf	61	FIR table for decimation/oversampling	Q13
Word16	Filter_32k_7k	61	FIR table for decimation/oversampling	Q13
Word16	Filter_48k	185	FIR table for decimation/oversampling	Q13
Word16	Filter_48k_hf	185	FIR table for decimation/oversampling	Q13
Word16	Filter_8k	61	FIR table for decimation/oversampling	Q13
Word16	Isf_init	16	Initial ISF memory	
Word16	Isp_init	16	Initial ISP memory	Q15
Word16	Isp_init_HF	8	HF Initial ISP memory	Q15
Word16	Mean_isf	16	Means of ISFs	
Word16	Dico1_isf	2304	1st stage codebook, isf0 to isf8	
Word16	Dico2_isf	1792	1st stage codebook, isf9 to isf15	
Word16	Dico21_isf	192	2nd stage codebook, isf2_0 to isf 2_2	
Word16	Dico22_isf	384	2nd stage codebook, isf2_3 to isf 2_5	
Word16	Dico23_isf	384	2nd stage codebook, isf2_6 to isf 2_8	
Word16	Dico24_isf	96	2nd stage codebook, isf2_9 to isf 2_11	
Word16	Dico25_isf	128	2nd stage codebook, isf2_12 to isf 2_15	
Word16	Dico21_isf_36b	640	1st stage codebook, (36b) split 1	
Word16	Dico22_isf_36b	512	1st stage codebook, (36b) split 2	
Word16	Dico23_isf_36b	448	1st stage codebook, (36b) split 3	
Word16	Dico_gain_hf	512	Quantization table for one-stage HF gain	Q8
Word16	Mean_isf_hf_12k8	8	Means of ISFs (full band)	
Word16	Dico1_isf_hf_12k8	32	1st stage isf codebook (full band)	
Word16	Mean_isf_hf_low_rate	8	Means of isfs	
Word16	Dico1_isf_hf_low_rate	32	1st stage isf codebook	
Word16	Dico2_isf_hf	1024	2nd stage isf codebook	
Word16	Filt_lp	13	Low-pass fir filter for bass post filter	Q15
Word16	Sin20	20	Random phase	Q15
Word16	Inter4_2	128	1/4 resolution interpolation filter	Q14
Word16	VadFiltBandFreqs	12	Open-loop classifier	Q0
Word16	Bw_inv	12	Open-loop classifier	Q22
Word16	Lwg	8	Open-loop claissifier	Q15
Word16	Gain_hf_ramp	64	HF gain ramp for wb->wb+ switiching	Q15
Word16	Inter2_coef	12	Filter coefficients for band join/split	Q13
Word16	Filter_LP180	2341	Filter for 48 kHz interpolation	Q14
Word16	StereoNbits_FX	18	Stereo bit-rates	Q0

Format	Table name	Size	Description	Format
Word16	Filter_2k_fxQ14_32	321	2k decimation filter	Q14
Word16	Filter_2k_fxQ14_5	321	2k decimation filter	Q14
Word16	cb_filt_hi_mean_fx	9	Average filter	Q14
Word16	Filt_hi_mscb_4a_fx	16*9		Q14
Word16	Filt_hi_mscb_7a_fx	16*9		Q14
Word16	Filt_hi_mscb_7b_fx	8*9		Q14
Word16	Cb_gain_hi_mean_fx	2	Average gain vector	Q10
Word16	Gain_hi_mscb_2a_fx	4*2		Q10
Word16	Gain_hi_mscb_5a_fx	32*2		Q10
Word16	Dico1_isf_hf_high_rate	32	1st stage isf codebook	
Word16	Mean_isf_hf_high_rate	8	Means of isfs	
Word16	Filter_LP45	586	Filter for 48 kHz interpolation	Q14
Word16	t_qua_gain6b	128	Gain pitch and gain code	Q14/Q11
Word16 t_qua_gain7b	t_qua_gain7b	256	Gain pitch and gain code	Q14/Q11
Word16	Overlap_wind	63	Overlap window	Q15
Word16	Cos_wind	128	Cos window	Q15
Word16	Cos_wind_LR	224	Cos _window (Low rate)	Q15
Word16	TXV	31	Arctan piece table	Q15
Word16	Len_tbl	6	Inverse length	Q15
Word16	interpol_frac4	4	Interpolation Window 4 sub-frame	Q15
Word16	interpol_frac8	8	Interpolation Window 8 sub-frame	Q15
Word16	interpol_frac16	16	Interpolation Window 16 sub-frame	Q15
Word16	size_filt_hi_msvq_4_fx	16	Stereo param	
Word16 PMSQ_fx	*cbs_filt_hi_msvq_4_fx filt_hi_pmsvq4_fx	16	Stereo param	
Word16	size_filt_hi_msvq_7_fx	16	Stereo param	
Word16 PMSQ_fx	*cbs_filt_hi_msvq_7_fx filt_hi_pmsvq7_fx	16	Stereo param	
Word16	Size_gain_hi_msvq_2_fx	16	Stereo param	
Word16 PMSQ_fx	*cbs_gain_hi_msvq_2_fx gain_hi_pmsvq2_fx	16	Stereo param	
Word16	size_gain_hi_msvq_5_fx	16	Stereo param	
Word16 PMSQ_fx	*cbs_gain_hi_msvq_5_fx gain_hi_pmsvq5_fx	16	Stereo param	

Table 4: Decoder fixed tables

Format	Table name	Size	Description
Same as encoder			

4.5.2 Static variables used in the C-code

In this clause two tables that specify the static variables for the encoder and decoder respectively are shown. All static variables are declared within a C **struct.**

Table 6: Encoder static variables

Struct name	Type	Variable	Type[Length]	Description
Coder_StState_fx				
	Word16	mem_decim	1608	speech decimated filter memory
	Word16	decim_frac	1	Fractional decimation factor
	Word16	mem_sig_in	6	hp filter memory
	Word16	mem_preemph	1	speech preemphasis filter mem
	Word16	mem_decim_hf	46	HF filter memory
	Word16	old_speech_hf	528	HF old speech vector
	Word16	past_q_isf_hf	8	HF past quantized isf
	Word16	ispold_hf	8	HF old isp
	Word16	ispold_q_hf	8	HF quantized old isp
	Word16	old_gain;	1	HF old gain match
	Word16	mem_hf1	8	HF memory for gain 1
	Word16	mem_hf2	8	HF memory for gain 2
	Word16	mem_hf3	8	HF memory for gain 3
	Word16	old_exc	375	old excitation
	Word16	Q_sp_hf	1	Scaling hf speech
	Word16	OldQ_sp_hf	2	old scaling hf speech
	Word16*	mean_isf_hf	1	isf codebook mean
	Word16*	dico1_isf_hf	1	isf codebook first stage
Coder_State_Plus_fx				
	Coder_StState_fx	Left	2617	state for left channel
	Coder_StState_fx	Right	2617	state for right channel
	Word16	old_chan	528	old left signal
	Word16	old_chan_2k	140	old left signal 2 kHz sampl. rate
	Word16	old_chan_hi	448	old left signal HB
	Word16	old_speech_2k	140	old mono signal 2 kHz sampl. rate
	Word16	old_speech_hi	448	old mono signal HB
	Word16	old_speech_pe	528	past pre-emphasised mono
	Word16	old_wh	9	past weighted filter
	Word16	old_wh_q	9	past quantized weighted filter
	Word16	old_gm_gain	2	past gain matching
	Word16	old_exc_mono	9	past mono excitation
	Word16	filt_energy_threshold	1	filter energy thershold
	Word16	w_window	64	weighting window
	PMSVQ_fx*	*filt_hi_pmsvq	1	MSVQ quantizer
	PMSVQ_fx*	*gain_hi_pmsvq	1	MSVQ quantizer
	Word16	mem_stereo_ovlp_size	1	past stereo overlap size
	Word16	mem_stereo_ovlp	32	past stereo overlap
	NCLASSDATA	*stClass	1	use case B classifier
	VadVars	*vadSt	1	VAD state
	Word16	vad_hist	1	VAD history
	Word16	old_speech	528	old speech
	Word16	old_synth	16	synthesis memory
	Word16	past_isfq	16	past isf quantizer
	Word16	old_wovlp	128	last tcx overlap
	Word16	old_d_wsp	187	Weighted speech vector
	Word16	old_exc	392	old excitation vector
	Word16	old_mem_wsyn	1	weighted synthesis memory
	Word16	old_mem_w0	1	weighted speech memory
	Word16	old_mem_xnq	1	quantized target memory
	Word16	old_ovlp_size	1	last tcx overlap size
	Word16	Isfold	16	old isf frequency domain
	Word16	Ispold	16	old isp
	Word16	ispold_q	16	quantized old isp
	Word16	mem_wsp	1	wsp vector mem
	Word16	mem_lp_decim2	3	wsp decimator filter mem

Struct name	Type	Variable	Type[Length]	Description
	Word16	ada_w	1	open loop LTP
	Word16	ol_gain	1	open loop LTP
	Word16	ol_wght_flg	1	open loop LTP
	Word16	old_ol_lag	5	past openloop lag
	Word16	old_T0_med	1	past pitch
	Word16	hp_old_wsp	699	past HP weighted speech
	Word16	hp_ol_ltp_mem	9	past HP openloop long term prediction
	Word16	window	512	LP analysis window
	Word16	SwitchFlagPlusToWB	1	flag for switching to AMR-WB
	Word16	mem_gain_code	4	past code gain
	Word16	prev_mod	1	past frame type
	Word16	Q_sp	1	Scaling of speech
	Word16	OldQ_sp	1	Old scaling of speech
	Word16	i_offset	1	
	Word16	pit_max	1	Mem of pit_max
	Word16	lev_mem	18	Levinson durbin memory
	Word16	old_wsp_max	4	Weight speech scaling memory
	Word16	old_wsp_shift	1	limit dynamic at 12 bits
	Word16	scale_fac	1	scaling factor (preemph speech)
	Word16	Q_new	1	scaling factor of speech
	Word16	Q_max	2	Q_new limitation
	Word16	OldQ_sp_deci	2	Q_new memory
	Word16	Q_exc	1	excitation scaling
	Word16	Old_Qexc	1	excitation scaling memory
	Word16	LastQMode	1	Last subfr mode (acelp/tcx)
Encoder Config				
	Word16	mode	1	AMR_WB core mode: 0..8
	Word16	extension	1	0=AMRWB, 1=WB+
	Word16	st_mode	1	stereo mode
	Word16	fscale	1	frequency scaling
	Word16	use_case_mode	1	use case (for AMRWB+ only)
	Word16	allow_dtx	1	dtx (for AMRWB only)
	Word16	FileFormat	1	3gp or raw
	Word16	mode_index	1	index of wb+ mode used
	Word16	fscale_index	1	index of internal frequency sampling
	Word16	bc	1	Backward compatible file format

Table 7: Decoder static variables

Struct name	Type	Variable	Type[Length]	Description
Decoder_StState				
	Word16	wmem_oversamp	72	Memory oversampling
	Word16	wover_frac	1	Fractional overcloking factor
	Word16	wmem_oversamp_hf	24	memory
	Word16	wpast_q_isf_hf	8	HF past quantized isf
	Word16	wpast_q_isf_hf_other	8	HF past quantized isf for the other channel when mono decoding stereo
	Word16	wpast_q_gain_hf	1	HF past quantized gain
	Word16	wpast_q_gain_hf_other	1	HF past quantized gain for the other channel when mono decoding stereo
	Word16	wold_gain	1	HF old gain match
	Word16	wispold_hf	8	HF old isp
	Word32	Lthreshold;	1	HF memory for smooth ener
	Word16	wmem_syn_hf	8	HF synthesis memory
	Word16	mem_d_tcx_fx	96	delay compensation memory
	Word16	wmem_d_nonc	64	Non causality delay
	Word16	mem_synth_hi	16	High band sunthesis memory
	Word16	wmem_sig_out	6	hp filter memory
	Word16	wold_synth_hf	512	synch delay memory
	Word32	Lp_amp	1	memory for soft exc
	Word16*	mean_isf_hf	1	isf codebook mean
	Word16*	dico1_isf_hf	1	isf codebook first stage
	Word16	Q_synHF	1	scaling of hf synth
Decoder_State_Plus				
	Decoder_StState	left	833	State for left channel
	Decoder_StState	right	833	State for right channel
	Word16	mem_left_2k_fx	20	2kHz memory on left chan
	Word16	mem_right_2k_fx	20	2kHz memory on right chan
	Word16	mem_left_hi_fx	64	HB memory left channel
	Word16	mem_right_hi_fx	64	HB memory right channel
	Word16	my_old_synth_2k_fx	35	old 2kHz synthesis
	Word16	my_old_synth_hi_fx	128	old HB synthesis
	Word16	my_old_synth_fx	148	old stereo synth
	Word16	old_AqLF_fx	85	old quantized LPC
	Word16	old_wh_fx	9	old decoded filter
	Word16	old_wh2_fx	9	old decoded filter 2
	Word16	old_exc_mono_fx	9	old mono excitation
	Word16	old_gain_left_fx	4	old gain on left chan
	Word16	old_gain_right_fx	4	old gain on right chan
	Word16	old_wh_q_fx	9	past quantized filter
	Word16	old_gm_gain_fx	2	past gain matching
	Word16	W_window	64	weighted synthesis window
	PMSVQ	*filt_hi_pmsvq_fx	1	past MSVQ filter
	PMSVQ	*gain_hi_pmsvq_fx	1	past MSVQ gain
	Word16	mem_stereo_ovlp_size_fx	1	past stereo overlap size
	Word16	mem_stereo_ovlp_fx	32	past stereo overlap
	Word16	last_stereo_mode	1	past stereo mode
	Word16	side_rms_fx	1	side signal RMS
	Word16	H_fx	9	current filter
	Word16	wold_xri	1 148	old spectral coefficeints
	Word16	last_mode	1	last mode in previous 80ms frame
	Word16	wmem_sig_out	6	hp50 filter memory for synthesis
	Word16	wmem_deemph	1	speech deemph filter memory
	Word16	prev_lpc_lost	1	previous lpc is lost when = 1
	Word16	wold_synth	16	synthesis memory
	Word16	wold_exc	392	old excitation vector
	Word16	wisfold	16	old isf (frequency domain)

Struct name	Type	Variable	Type[Length]	Description
	Word16	wispold	16	old isp (immittance spectral pairs)
	Word16	wpast_isfq	16	past isf quantizer
	Word16	wovlp	128	last weighted synthesis for overlap
	Word16	ovlp_size	1	overlap size
	Word16	wisf_buf	51	old isf (for frame recovery)
	Word16	wold_T0	1	old pitch value (for frame recovery)
	Word16	wold_T0_frac	1	old pitch value (for frame recovery)
	Word16	seed_ace	1	seed memory (for random function)
	Word16	wmem_wsyn	1	TCX synthesis memory
	Word16	seed_tcx	1	seed memory (for random function)
	Word16	wwsyn_rms	1	rms value of weighted synthesis
	Word16	wpast_gpit	1	past gain of pitch (for frame recovery)
	Word32	Lpast_gcode	1	past gain of code (for frame recovery)
	Word16	pitch_tcx	1	for bfi
	Word32	L_gc_threshold	1	GC threshold
	Word16	wold_synth_pf	503	Bass post-filter: old synthesis
	Word16	wold_noise_pf	24	bass post-filter: noise memory
	Word16	wold_T_pf	2	bass post-filter: old pitch
	Word16	wold_gain_pf	2	Bass post-filter: old pitch gain
	Word16	*mean_isf_hf	1	HF isf codebook in-use
	Word16	*dico1_isf_hf	1	HF isf codebook in-use
	Word16	wmem_gain_code	4	past code gain
	Word16	wmem_lpc_hf	9	past HF lpc filter
	Word16	wmem_gain_hf	1	past HF gain
	Word16	wramp_state	1	ramp state
	Word16	cp_old_synth	16	old synthesis switching memory
	Word16	Q_old	1	Old scaling
	Word16	Q_exc	1	excitation scaling
	Word16	Q_syn	1	synthesis scaling
	Word16	Old_Q_syn	1	Old synthesis scaling
	Word16	Old_Q_exc	1	Old excitation scaling
	Word16	prev_Q_syn	1	Limitation on synthesis scaling
	Word16	mem_syn2	16	Switching synthesis memory
	Word16	Old_Qxnq	1	Old xnq scaling
	Word16	Old_QxnqMax	1	Old maximum xnq scaling
	Word16	Old_Qxri	1	Old xri scaling
	Word16	Old_bpf_scale	1	noise buf scaling
	Word16	mem_subfr_q	7	subfr maximum excitation scaling
	Word16	old_subfr_q	16	subfr true excitation scaling
	Word16	old_syn_q	16	subfr true synthesis scaling
	Word16	i_offset	1	offset memory
DecoderConfig				
	Word16	mode	1	AMR_WB core modes
	Word16	extension	1	0 = AMR_WB 1 = WB+
	Word16	st_mode	1	Stereo modes
	Word16	fscale	1	Internal Frequency scaling
	Word32	fs	1	Sampling rate
	Word32	mono_dec_stereo	1	decode mono a stereo bitstream
	Word32	limiter_on	1	Limite clipping
	Word16	Fileformat	1	File format used (raw/3gp)
	Word16	fer_sim	1	Frame erasure simulation

5 File formats

This clause describes the file formats used by the encoder and decoder programs.

5.1 Audio file (encoder input/decoder output)

Audio files read by the encoder must be formatted as 16 bits PCM wave (*.wav) files. The decoder output is written as a 16 bit PCM wave file (*.wav).

Note that the decoder, with proper command line switch, can produce a mono file from a stereo bit-stream.

5.2 Parameter bitstream file (encoder output/decoder input)

For AMR-WB+ operation, the files produced by the audio encoder/expected by the audio decoder are either according to the raw format defined in 3GPP TS 26.290 [2], clause 8.2, or according to the 3GP file format (3GPP TS 26.244 [4]), whereby the storage sample definition is found in 3GPP TS 26.290 [2], clause 8.3.

Annex A (informative): Change history

Change history							
Date	**TSG SA#**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
2005-03	27	SP-050083			Presented to TSG SA#27 for approval	1.0.0	2.0.0