

Source: TSG_SA WG4
Title: ANSI-C code for the Adaptive Multi Rate Wideband speech codec (Release 4)
Document for: Approval

Presentation of Specification to TSG-SA

Presentation to: TSG SA Meeting #11
Document for presentation: TS 26.173 Version 2.0.0
Presented for: Approval

Abstract of document:

The present document contains an electronic copy of the ANSI-C code for the Adaptive Multi-Rate Wideband codec. The ANSI-C code is necessary for a bit exact implementation of the Adaptive Multi Rate Wideband speech transcoder (3GPP TS 26.190), Voice Activity Detection (3GPP TS 26.194), Comfort Noise (3GPP TS 26.192), Source Controlled Rate Operation (3GPP TS 26.193) and example solutions for Substituting and Muting of Lost Frames (3GPP TS 26.191)

Changes since last presentation to TSG Meeting #10:
[Editorial changes.](#)

Outstanding Issues:
[None.](#)

Contentious Issues:
[None.](#)

3GPP TS 26.173 V2.0.0 (2001-03)

Technical Specification

**3rd Generation Partnership Project;
Technical Specification Group Services and System Aspects;
ANSI-C code for the Adaptive Multi Rate Wideband speech
codec;
(Release 4)**



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, CODEC, Adaptive Multi-Rate, Wideband
speech coder

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.

© 2001, 3GPP Organizational Partners (ARIB, CWTS, ETSI, T1, 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	10
4.5.1 Description of constants used in the C-code	10
4.5.2 Description of fixed tables used in the C-code.....	10
4.5.3 Static variables used in the C-code.....	12
5 Homing procedure.....	15
6 File formats	15
6.1 Speech file (encoder input / decoder output)	15
6.2 Mode control file (encoder input)	15
6.3 Parameter bitstream file (encoder output / decoder input).....	16
Annex A (informative): Change history	17

Foreword

This Technical Specification (TS) 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 Adaptive Multi-Rate Wideband codec. The ANSI-C code is necessary for a bit exact implementation of the Adaptive Multi Rate Wideband speech transcoder (3GPP TS 26.190 [2]), Voice Activity Detection (3GPP TS 26.194 [6]), comfort noise (3GPP TS 26.192 [4]), source controlled rate operation (3GPP TS 26.193 [5]) and example solutions for substituting and muting of lost frames (3GPP TS 26.191 [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.

- [1] 3GPP TS 26.174: "AMR Wideband Speech Codec; Test sequences".
- [2] 3GPP TS 26.190: "AMR Wideband Speech Codec; Speech transcoding".
- [3] 3GPP TS 26.191: "AMR Wideband Speech Codec; Substitution and muting of lost frames".
- [4] 3GPP TS 26.192: "AMR Wideband Speech Codec; Comfort noise aspects".
- [5] 3GPP TS 26.193: "AMR Wideband Speech Codec; Source controlled rate operation".
- [6] 3GPP TS 26.194: "AMR Wideband Speech Codec; Voice Activity Detection".

3 Definitions and abbreviations

3.1 Definitions

Definition of terms used in the present document, can be found in 3GPP TS 26.190 [2], 3GPP TS 26.191 [3], 3GPP TS 26.192 [4], 3GPP TS 26.193 [5] and 3GPP TS 26.194 [6].

3.2 Abbreviations

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

AMR-WB	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 this document.

The C code has been verified on the following systems:

- Sun Microsystems workstations and GNU gcc compiler
- HP workstations and cc compiler
- IBM PC compatible computers with Windows NT4 operating system and GNU gcc 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 all files in the root level.

The distributed files with suffix "c" contain the source code and the files with suffix "h" are the header files. The ROM data is contained mostly in files with suffix "tab".

The C code distribution also contains one speech coder installation verification data file, "spch_dos.inp". The reference encoder output file is named "spch_dos.cod", the reference decoder input file is named "spch_dos.dec" and the reference decoder output file is named "spch_dos.out". These four files are formatted such that they are correct for an IBM PC/AT compatible computer. The same files with reversed byte order of the 16 bit words are named "spch_unx.inp", "spch_unx.cod", "spch_unx.dec" and "spch_unx.out", respectively.

Final verification is to be performed using the GSM Adaptive Multi-Rate Wideband test sequences described in 3GPP TS 26.174 [1].

Makefiles are provided for the platforms in which the C code has been verified (listed above). Once the software is installed, this directory will have a compiled version of *encoder* and *decoder* (the bit-exact C executables of the speech codec) and all the object files.

4.2 Program execution

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

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

The programs should be called like:

- encoder [encoder options] <speech input file> <parameter file>;
- decoder <parameter file> <speech output file>.

The speech files contain 16-bit linear encoded PCM speech samples and the parameter files contain encoded speech 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 to 3 are call graphs that show the functions used in the speech codec, including the functions of VAD, DTX, and comfort noise generation.

Each column represents a call level and each cell a function. The functions contain calls to the functions in rightwards neighboring 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()) 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.

The encoder call graph is broken down into two separate call graphs, Table 1 to 2.

Table 1: Speech encoder call structure

coder	Copy			
	Decim_12k8	Down_samp	Interpol (function)	
		Copy		
	Set_zero			
	HP50_12k8			
	Scale_sig			
	wb_vad	Filter_bank	Filter5	
			Filter3	
		vad_decision	Level_calculation	
			llog2	
		Noise_estimate_update	update_cntrl	
		hangover_addition		
		Estimate_Speech		
	tx_dtx_handler			
	Parm_serial			
	Autocorr			
	Lag_window			
	Levinson			
	Az_isp	Chebbs2		
	Int_isp	Isp_Az	Get_isp_pol	
	Isp_isf			
	Gp_clip_test_isf			
	Weight_a			
	Residu			
	Deemph2			
	LP_Decim2			
	Scale_mem_Hp_wsp			
	Pitch_med_ol	Hp_wsp		
		Isqrt_n		
	wb_vad_tone_detection			
	Med_olag	median5		
	dtx_buffer	Copy		
	dtx_enc	Find_frame_indices		
		Aver_isf_history		
		Qisf_ns	Sub_VQ	
			Disf_ns	Reorder_isf
		Parm_serial		
		Pow2		
		Random		
		Dot_product12		
		Isqrt_n		
		Isf_isp		
	Isp_Az	Get_isp_pol		
	Synthesis	Copy		
		Syn_filt_32		
		Deemph_32		
		HP50_12k8		
		Random		
		Scale_sig		
		Dot_product12		
Isqrt_n				
HP400_12k8				
Weight_a				
Syn_filt				
Filt_6k_7k				
Reset_encoder	Set_zero			
	Init_gp_clip			
	Init_Phase_dispersion	Set_zero		
Qpisf_2s_36b	VQ_stage1			
	Sub_VQ			
	Dpisf_2s_36b	Reorder_isf		
Qpisf_2s_46b	VQ_stage1			
	Sub_VQ			
	Dpisf_2s_46b	Reorder_isf		
Syn_filt				
Preemph2				
Pitch_fr4	Norm_Corr	Convolve		
	Interpol_4	Isqrt_n		
Gp_clip				
Pred_lt4				
Convolve				
G_pitch	Dot_product12			
Updt_tar				
Preemph				
Pit_shrp				
Cor_h_x				
ACELP_2t64_fx	Dot_product12			
	Isqrt_n			
ACELP_4t64_fx	See Table 2			
Q_gain2	Dot_product12			
	Pow2			
Gp_clip_test_gain_pit				
voice_factor	Dot_product12			

Table 2: ACELP_4t64_fx call structure

ACELP_4t64_fx	Dot_product12			
	lsqrt_n			
	cor_h_vec			
	search_ixiy			
	quant_1p_N1			
	quant_2p_2N1			
	quant_3p_3N1	quant_2p_2N1		
		quant_1p_N1		
	quant_4p_4N	quant_4p_4N1	Quant_2p_2N1	
		quant_1p_N1		
		quant_3p_3N1	Quant_2p_2N1	
			Quant_1p_N1	
		quant_2p_2N1		
	quant_5p_5N	quant_3p_3N1	Quant_2p_2N1	
			Quant_1p_N1	
		quant_2p_2N1		
	quant_6p_6N_2	quant_5p_5N	Quant_3p_3N1	quant_2p_2N1
			quant_2p_2N1	Quant_1p_N1
		quant_1p_N1		
		quant_4p_4N	quant_4p_4N1	quant_2p_2N1
		quant_1p_N1		
		quant_3p_3N1	quant_2p_2N1	
			quant_1p_N1	
	quant_2p_2N1	quant_2p_2N1		
	quant_3p_3N1	quant_2p_2N1		
		Quant_1p_N1		

Table 3: Speech decoder call structure

decoder	Rx_dtx_handler			
	Dtx_dec	Copy		
		Disf_ns	Reorder_isf	
		Serial_parm		
		Pow2		
		Random		
		Dot_product12		
		lsqrt_n		
		Serial_parm		
		Isf_isp		
		Isp_Az	Get_isp_pol	
		Copy		
		Synthesis	Copy	
			Syn_filt_32	
			Deemph_32	
			HP50_12k8	
			Oversamp_16k	Copy
				Up_samp
				Interpol
			Random	
			Scale_sig	
			Dot_product12	
			lsqrt_n	
			HP400_12k8	
			Isf_Extrapolation	Isf_isp
			Isp_Az	Get_isp_pol
			Weight_a	
			Syn_filt	
			Filt_6k_7k	Copy
			Filt_7k	Copy
		Reset_decoder	Set_zero	
			Init_Phase_dispersion	Set_zero
		Dpisyf_2s_36b	Reorder_isf	
		Dpisyf_2s_46b	Reorder_isf	
		Int_isp	Isp_Az	Get_isp_pol
		Lagconc	insertion_sort	Insert
			Random	
		Pred_lt4		
		Random		
		DEC_ACELP_2t64_fx		
	DEC_ACELP_4t64_fx	dec_1p_N1		
		add_pulses		
		dec_2p_2N1		
		dec_3p_3N1	Dec_2p_2N1	
			dec_1p_N1	
		dec_4p_4N	dec_4p_4N1	
			dec_2p_2N1	
			dec_1p_N1	
			Dec_3p_3N1	
			Dec_2p_2N1	
			Dec_1p_N1	
		dec_5p_5N	dec_3p_3N1	
			Dec_2p_2N1	
			Dec_1p_N1	
		dec_6p_6N_2	Dec_5p_5N	
			dec_3p_3N1	
			Dec_2p_2N1	
			Dec_1p_N1	
			dec_1p_N1	
			dec_4p_4N	
			dec_2p_2N1	
			dec_1p_N1	
			Dec_3p_3N1	
			Dec_2p_2N1	
			Dec_1p_N1	
			Dec_2p_2N1	
		dec_2p_2N1		

		dec_3p_3N1	Dec_2p_2N1
			Dec_1p_N1
Preemph			
Pit_shrp			
D_gain2	Dot_product12		
	lsqrt_n		
	Median5		
	Pow2		
Scale_sig			
voice_factor	Dot_product12		
Phase_dispersion	Set_zero		
Agc2	lsqrt	lsqrt_n	
Set_zero			
Dtx_dec_activity_update	Copy		

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 constants used in the C-code

This subclause contains a listing of all global constants defined in cnst.h.

Table 5: Global constants

Constant	Value	Description
L_TOTAL	384	total size of speech buffer.
L_WINDOW	384	window size in LP analysis
L_NEXT	64	Look-ahead size
L_FRAME	256	frame size in 12.8 kHz
L_FRAME16k	320	frame size in 16 kHz
L_SUBFR	64	Subframe size in 12.8 kHz
L_SUBFR16k	80	Subframe size in 16 kHz
NB_SUBFR	4	Number of subframes
M16k	20	order of LP filter in high-band synthesis in 6.60 mode
M	16	order of LP filter
L_FILT16k	15	Delay of down-sampling filter in 16 kHz
L_FILT	12	Delay of down-sampling filter in 12.8 kHz
GP_CLIP	15565	Pitch gain clipping
PIT_SHARP	27853	pitch sharpening factor
PIT_MIN	34	minimum pitch lag (all modes)
PIT_FR2	128	Minimum pitch lag with resolution $\frac{1}{2}$
PIT_FR1_9b	160	Minimum pitch lag with resolution for 9 bit quantization
PIT_FR1_8b	92	Minimum pitch lag with resolution for 8 bit quantization
PIT_MAX	231	maximum pitch lag
L_INTERPOL	(16+1)	length of filter for interpolation
OPL_DECIM	2	Decimation in open-loop pitch analysis
PREEMPH_FAC	22282	preemphasis factor
GAMMA1	30147	Weighting factor (numerator)
TILT_FAC	22282	tilt factor (denominator)
Q_MAX	8	scaling max for signal
RANDOM_INITSEED	21845	random init value
L_MEANBUF	3	Size of ISF buffer
ONE_PER_MEANBUF	10923	Inverse of L_MEANBUF

4.5.2 Description of fixed tables used in the C-code

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

Table 6: Fixed tables

File	Table name	Length	Description
C4t64fx.c	Tipos	36	starting points of iterations
Cod_main.c	HP_gain	16	High band gain table for 23.85 kbit/s mode
Cod_main.c	Interpol_frac	4	LPC interpolation coefficients
Cod_main.c	lsp_init	16	isp tables for initialization
Cod_main.c	lsf_init	16	isf tables for initialization
D_gain2.c	cdown_unusable	7	attenuation factors for codebook gain in lost frames
D_gain2.c	cdown_usable	7	attenuation factors for codebook gain in bad frames
D_gain2.c	pdown_unusable	7	attenuation factors for adaptive codebook gain in lost frames
D_gain2.c	pdown_usable	7	attenuation factors for adaptive codebook gain in bad frames
D_gain2.c	Pred	4	algebraic code book gain MA predictor coefficients
Dec_main.c	HP_gain	16	High band gain table for 23.85 kbit/s mode
Dec_main.c	Interpol_frac	4	LPC interpolation coefficients
Dec_main.c	lsp_init	16	isp tables for initialization
Dec_main.c	lsf_init	16	isf tables for initialization
Decim54.c	fir_down	120	Downsample FIR filter coefficients
Decim54.c	fir_up	120	Upsample FIR filter coefficients
Dtx.c	en_adjust	9	Energy scaling factor for each mode during comfort noise
Grid100.tab	grid	101	grid points at wich Chebyshev polynomials
Ham_wind.tab	Window	384	LP analysis window
Hp400.c	A	3	HP filter coefficients (denominator) in higher band energy estimation
Hp400.c	B	3	HP filter coefficients (numerator) in higher band energy estimation
Hp50.c	A	3	HP filter coefficients (denominator) in pre-filtering
Hp50.c	B	3	HP filter coefficients (numerator) in pre-filtering
Hp6k.c	Fir_6k_7k	31	Bandpass FIR filter coefficients for higher band generation
Hp7k.c	Fir_7k	31	Bandpass FIR filter coefficients for higher band in 23.85 kbit/s mode
Hp_wsp.c	A	3	HP filter coefficients (denominator) in open-loop lag gain computation
Hp_wsp.c	B	3	HP filter coefficients (numerator) in open-loop lag gain computation
lsp_isf.tab	slope	128	table to compute $\cos(x)$ in $Lsf_lsp()$
lsp_isf.tab	Table	129	table to compute $\arccos(x)$ in $Lsp_lsf()$
Lag_wind.tab	lag_h	16	high part of the lag window table
Lag_wind.tab	lag_l	16	low part of the lag window table
Lp_dec2.c	h_fir	5	HP FIR filter coefficients in open-loop lag search
Math_op.c	table_isqrt	49	table used in inverse square root computation
Math_op.c	table_pow2	33	table used in power of two computation
P_med_ol.tab	Corrweight	199	weighting of the correlation function in open loop LTP search
Ph_disp.c	ph_imp_low	64	phase dispersion impulse response
Ph_disp.c	ph_imp_mid	64	phase dispersion impulse response
Pitch_fr4.c	inter4_1	32	interpolation filter coefficients
Pred_lt4.c	inter4_2	128	interpolation filter coefficients
Q_gain2.c	pred	4	algebraic code book gain MA predictor coefficients
Q_gain2.tab	t_qua_gain6b	2*64	gain quantization table for 6-bit gain quantization
Q_gain2.tab	t_qua_gain7b	2*128	gain quantization table for 7-bit gain quantization
Qisf_ns.tab	dico1_isf_noise	2*64	1 st ISF quantizer for comfort noise
Qisf_ns.tab	dico2_isf_noise	3*64	2 nd ISF quantizer for comfort noise
Qisf_ns.tab	Dico3_isf_noise	3*64	3 rd LSF quantizer for comfort noise
Qisf_ns.tab	Dico4_isf_noise	4*32	4 th LSF quantizer for comfort noise
Qisf_ns.tab	Dico5_isf_noise	4*32	5 th LSF quantizer for comfort noise
Qisf_ns.tab	mean_isf_noise	16	ISF mean for comfort noise
Qpif_2s.tab	dico1_isf	9*256	1 st ISF quantizer of the 1 st stage
Qpif_2s.tab	Dico2_isf	7*256	2 nd ISF quantizer of the 1 st stage
Qpif_2s.tab	Dico21_isf	3*64	1 st ISF quantizer of the 2 nd stage (not the 6.60 kbit/s mode)
Qpif_2s.tab	Dico21_isf_36b	5*128	1 st ISF quantizer of the 2 nd stage (the 6.60 kbit/s mode)
Qpif_2s.tab	Dico22_isf	3*128	2 nd ISF quantizer of the 2 nd stage (not the 6.60 kbit/s mode)
Qpif_2s.tab	Dico22_isf_36b	4*128	2 nd ISF quantizer of the 2 nd stage (the 6.60 kbit/s mode)

(continued)

Table 6 (concluded): Fixed tables

File	Table name	Length	Description
Qpif_2s.tab	Dico23_isf	3*128	3 rd ISF quantizer of the 2 nd stage (not the 6.60 kbit/s mode)
Qpif_2s.tab	Dico23_isf_36b	7*64	3 rd ISF quantizer of the 2 nd stage (the 6.60 kbit/s mode)
Qpif_2s.tab	Dico24_isf	3*32	4 th ISF quantizer of the 2 nd stage (not the 6.60 kbit/s mode)
Qpif_2s.tab	Dico25_isf	4*32	5 th ISF quantizer of the 2 nd stage (not the 6.60 kbit/s mode)
Qpif_2s.tab	Mean_isf	16	ISF mean

4.5.3 Static variables used in the C-code

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

Table 7: Speech encoder static variables

Struct name	Variable	Type[Length]	Description
Coder_State	mem_decim	Word16[30]	Decimation filter memory
	mem_sig_in	Word16[6]	Prefilter memory
	mem_preemph	Word16	Preemphasis filter memory
	old_speech	Word16[128]	speech buffer
	old_wsp	Word16[115]	buffer holding spectral weighted speech
	old_exc	Word16[248]	excitation vector
	mem_levinson	Word16[18]	Levinson memories
	ispold	Word16[16]	Old ISP vector
	ispold_q	Word16[16]	Old quantized ISP vector
	past_isfq	Word16[16]	past quantized ISF prediction error
	mem_wsp	Word16	Open-loop LTP deemphasis filter memory
	mem_decim2	Word16[3]	Open-loop LTP decimation filter memory
	mem_w0	Word16	weighting filter memory (applied to error signal)
	mem_syn	Word16[16]	synthesis filter memory
	tilt_code	Word16	Preemphasis filter memory
	old_wsp_max	Word16	Open loop scaling factor
	old_wsp_shift	Word16	Maximum open loop scaling factor
	Q_old	Word16	Old scaling factor
	Q_max	Word16[2]	Maximum scaling factor
	gp_clip	Word16[2]	memory of pitch clipping
	qua_gain	Word16[4]	Gain quantization memory
	old_T0_med	Word16	weighted open loop pitch lag
	ol_gain	Word16	Open-loop gain
	ada_w	Word16	weighing level depending on open loop pitch gain
	ol_wght_flg	Word16	switches lag weighting on and off
	old_ol_lag	Word16[5]	Open loop lag history
	hp_wsp_mem	Word16[9]	Open-loop lag gain filter memory
	old_hp_wsp	Word16[243]	Open-loop lag
	vadSt	VadVars*	see below in this table
	dtx_encSt	dtx_encState*	see below in this table
	first_frame	Word16	First frame indicator
	lsfold	Word16[16]	Old ISF vector
	L_gc_thres	Word16	Noise enhancer threshold
	mem_syn_hi	Word16[16]	synthesis filter memory (most significant word)
	mem_syn_lo	Word16[16]	synthesis filter memory (least significant word)
	mem_deemph	Word16	Deemphasis filter memory
	mem_sig_out	Word16[6]	HP filter memory in the synthesis
mem_hp400	Word16[6]	HP filter memory	
mem_oversamp	Word16[2*12]	Oversampling filter memory	
mem_syn_hf	Word16[16]	Higher band synthesis filter memory	
mem_hf	Word16[30]	Estimated BP filter memory (23.85 kbit/s mode)	
mem_hf2	Word16[30]	Input BP filter memory (23.85 kbit/s mode)	
mem_hf3	Word16[30]	Input LP filter memory (23.85 kbit/s mode)	
seed2	Word16	Random generation seed	
disp_mem	Word16[8]	Phase dispersion memory	
vad_hist	Word16	VAD history	
Gain_alpha	Word16	Higher band gain weighting factor (23.85 kbit/s mode)	
dtx_encState	lsf_hist	Word16[128]	LSP history (8 frames)
	Log_en_hist	Word16[8]	logarithmic frame energy history (8 frames)
	Hist_ptr	Word16	pointer to the cyclic history vectors
	Log_en_index	Word16	Index for logarithmic energy
	Cng_seed	Word16	Comfort noise excitation seed
	D	Word16[28]	ISF history distance matrix
	sumD	Word16[8]	Sum of ISF history distances
	dtxHangoverCount	Word16	is decreased in DTX hangover period
decAnaElapsedCount	Word16	counter for elapsed speech frames in DTX	
vadState1	bckr_est	Word16[12]	background noise estimate
	ave_level	Word16[12]	averaged input components for stationary estimation
	old_level	Word16[12]	input levels of the previous frame
	sub_level	Word16[12]	input levels calculated at the end of a frame (lookahead)
	a_data5	Word16[5][2]	memory for the filter bank
	a_data3	Word16[6]	memory for the filter bank
burst_count	Word16	counts length of a speech burst	

Struct name	Variable	Type[Length]	Description
	Hang_count	Word16	hangover counter
	Stat_count	Word16	stationary counter
	Vadreg	Word16	15 flags for intermediate VAD decisions
	Tone_flag	Word16	15 flags for tone detection
	sp_est_cnt	Word16	Speech level estimation counter
	Sp_max	Word16	Maximum signal level
	sp_max_cnt	Word16	Maximum level estimation counter
	Speech_level	Word16	Speech level
	prev_pow_sum	Word16	Power of previous frame

Table 8: Speech decoder static variables

Struct name	Variable	Type[Length]	Description
Decoder_State	old_exc	Word16[248]	excitation vector
	ispold	Word16[16]	Old ISP vector
	isfold	Word16[16]	Old ISF vector
	isf_buf	Word16[48]	ISF vector history
	past_isfq	Word16[16]	past quantized ISF prediction error
	tilt_code	Word16	Preemphasis filter memory
	Q_old	Word16	Old scaling factor
	Qsubfr	Word16	Scaling factor history
	L_gc_thres	Word16	Noise enhancer threshold
	mem_syn_hi	Word16[16]	synthesis filter memory (most significant word)
	mem_syn_lo	Word16[16]	synthesis filter memory (least significant word)
	mem_deemph	Word16	Deemphasis filter memory
	mem_sig_out	Word16[6]	HP filter memory in the synthesis
	mem_oversamp	Word16[24]	Oversampling filter memory
	mem_syn_hf	Word16[20]	Higher band synthesis filter memory
	mem_hf	Word16[30]	Estimated BP filter memory (23.85 kbit/s mode)
	mem_hf2	Word16[30]	Input BP filter memory (23.85 kbit/s mode)
	mem_hf3	Word16[30]	Input LP filter memory (23.85 kbit/s mode)
	seed	Word16	Random code generation seed for bad frames
	seed2	Word16	Random generation seed for higher band
	old_T0	Word16	Old LTP lag (integer part)
	old_T0_frac	Word16	Old LTP lag (fraction part)
	lag_hist	Word16[5]	LTP lag history
	dec_gain	Word16[23]	Gain decoding memory
	seed3	Word16	Random LTP lag generation seed for bad frames
	disp_mem	Word16[8]	Phase dispersion memory
	mem_hp400	Word16[6]	HP filter memory
prev_bfi	Word16	Previous BFI	
state	Word16	BGH state machine memory	
first_frame	Word16	First frame indicator	
dtx_decSt	dtx_decState*	see below in this table	
Vad_hist	Word16	VAD history	
dtx_decState	Since_last_sid	Word16	number of frames since last SID frame
	true_sid_period_inv	Word16	inverse of true SID update rate
	log_en	Word16	logarithmic frame energy
	old_log_en	Word16	previous value of log_en
	isf	Word16[16]	ISF vector
	Isf_old	Word16[16]	Previous ISF vector
	Cng_seed	Word16	Comfort noise excitation seed
	Isf_hist	Word16[128]	ISF vector history (8 frames)
	Log_en_hist	Word16[8]	logarithmic frame energy history
	Hist_ptr	Word16	index to beginning of LSF history
	dtxHangoverCount	Word16	counts down in hangover period
	DecAnaElapsedCount	Word16	counts elapsed speech frames after DTX
	sid_frame	Word16	flags SID frames
	valid_data	Word16	flags SID frames containing valid data
	log_en_adjust	Word16	mode-dependent frame energy adjustment
	dtxHangoverAdded	Word16	flags hangover period at end of speech
	dtxGlobalState	Word16	DTX state flags
data_updated	Word16	flags CNI updates	

5 Homing procedure

The principles of the homing procedures are described in [2]. This specification only includes a detailed description of the 9 decoder homing frames. For each AMR-WB codec mode, the corresponding decoder homing frame has a fixed set of parameters. The parameters in serial format are packed into parameters in 15-bit-long format where the first serial bit is inserted into most significant bit in the 15-bit-long format. These 15-bit-long parameters do not represent real speech parameters, but they decrease memory consumption compared to the speech parameters. Table 9 shows the homing frame in 15-bit-long format for different modes. In the decoder, the received speech parameters in serial format are first converted into 15-bit-long format. Then the obtained parameters are compared against the homing frame table values (Table 9)..

Table 9: Table values for the decoder homing frame in 15-bit-long format for different modes

Mode	Value (MSB=b0)
0	25351, 4331, 515, 15620, 20992, 0, 0, 0, 0
1	25351, 14010, 26489, 30912, 5254, 3459, 0, 0, 0, 0, 0, 0
2	25351, 14010, 29177, 18070, 19971, 3968, 32492, 8430, 13280, 0, 0, 0, 0, 0, 0, 0
3	25351, 14010, 1912, 16326, 25140, 16384, 502, 15167, 1772, 11512, 0, 0, 0, 0, 0, 0, 0, 0
4	25351, 14010, 1912, 30593, 14594, 19990, 864, 4635, 20446, 27456, 21310, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
5	25351, 14010, 19995, 14446, 6159, 7329, 20752, 4228, 19488, 24383, 364, 20124, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
6	25351, 14010, 3567, 560, 32536, 20534, 5139, 16384, 26161, 18755, 20444, 22173, 12623, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
7	25351, 14010, 2912, 28827, 15347, 28610, 9853, 1316, 30720, 786, 32259, 13279, 14336, 29152, 23302, 20352, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
8	025351, 14010, 1601, 16734, 7923, 15017, 5450, 5477, 5760, 2187, 1534, 12142, 30894, 13419, 13141, 2376, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

6 File formats

This section describes the file formats used by the encoder and decoder programs. The test sequences defined in [1] also use the file formats described here.

6.1 Speech file (encoder input / decoder output)

Speech files read by the encoder and written by the decoder consist of 16-bit words where each word contains a 14-bit, left aligned speech sample. The byte order depends on the host architecture (e.g. MSByte first on SUN workstations, LSByte first on PCs etc.). Both the encoder and the decoder program process complete frames (of 320 samples) only.

This means that the encoder will only process n frames if the length of the input file is $n*320 + k$ words, while the files produced by the decoder will always have a length of $n*320$ words.

6.2 Mode control file (encoder input)

The encoder program can optionally read in a mode control file which specifies the encoding mode for each frame of speech processed. The file is a text file containing one number per speech frame. Each line contains one of the mode numbers 0-8.

6.3 Parameter bitstream file (encoder output / decoder input)

The files produced by the speech encoder/expected by the speech decoder contain an arbitrary number of frames in the following format.

TYPE_OF_FRAME_TYPE	FRAME_TYPE	MODE	B1	B2	...	Bnn
--------------------	------------	------	----	----	-----	-----

Each box corresponds to one Word16 value in the bitstream file, for a total of 3+nn words or 6+2nn bytes per frame, where nn is the number of encoded bits in the frame. The fields have the following meaning:

TYPE_OF_FRAME_TYPE transmit frame type, which is one of
 TX_TYPE (0x6b21)
 RX_TYPE (0x6b20)

If TYPE_OF_FRAME_TYPE is TX_TYPE,

FRAME_TYPE transmit frame type, which is one of
 TX_SPEECH (0x0000)
 TX_SID_FIRST (0x0001)
 TX_SID_UPDATE (0x0002)
 TX_NO_DATA (0x0003)

If TYPE_OF_FRAME_TYPE is RX_TYPE,

FRAME_TYPE transmit frame type, which is one of
 RX_SPEECH_GOOD (0x0000)
 RX_SPEECH_PROBABLY_DEGRADED (0x0001)
 RX_SPEECH_LOST (0x0002)
 RX_SPEECH_BAD (0x0003)
 RX_SID_FIRST (0x0004)
 RX_SID_UPDATE (0x0005)
 RX_SID_BAD (0x0006)
 RX_NO_DATA (0x0007)

B0...B2nn speech encoder parameter bits (i.e. the bitstream itself). Each B_x either has the value 0x0000 or 0x0001.

MODE_INFO encoding mode information, which is one of
 6.60 kbit/s mode (0x0000)
 8.85 kbit/s mode (0x0001)
 12.65 kbit/s mode (0x0002)
 14.25 kbit/s mode (0x0003)
 15.85 kbit/s mode (0x0004)
 18.25 kbit/s mode (0x0005)
 19.85 kbit/s mode (0x0006)
 23.05 kbit/s mode (0x0007)
 23.85 kbit/s mode (0x0008)

As indicated in section 6.1 above, the byte order depends on the host architecture.

Annex A (informative): Change history

Change history							
Date	TSG #	TSG Doc.	CR	Rev	Subject/Comment	Old	New
12-2000	10	SP-000557			Version 1.0.0 provided for information		
03-2001	11	SP-000083			Version 2.0.0 provided for approval		