ETSI TS 126 173 V13.0.0 (2016-01)

TECHNICAL SPECIFICATION

**Digital cellular telecommunications system (Phase 2+);
Universal Mobile Telecommunications System (UMTS);
LTE;
ANSI-C code for the Adaptive Multi-Rate -
Wideband (AMR-WB) speech codec
(3GPP TS 26.173 version 13.0.0 Release 13)**

Reference
RTS/TSGS-0426173vd00

Keywords
GSM,LTE,UMTS

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

The present document can be downloaded from:
http://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the only prevailing document is the print of the Portable Document Format (PDF) version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, please send your comment to one of the following services:
https://portal.etsi.org/People/CommiteeSupportStaff.aspx

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

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

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp.

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Contents

Intellectual Property Rights 2

Foreword 2

Modal verbs terminology 2

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 11
4.5.2 Description of fixed tables used in the C-code 11
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
Default 3GPP format: 16
ITU format (activated with command line parameter -itu) 17
MIME/file storage format (activated with command line parameter -mime) 17

Annex A (informative): Change history 18

History 19

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. 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 26.174: "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".

[7] RFC 3267 'A Real-Time Transport Protocol (RTP) Payload Format and File Storage Format for Adaptive Multi-Rate (AMR) and Adaptive Multi-Rate Wideband (AMR-WB) Audio Codecs, June 2002.

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 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()) 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	
			Level_calculation	
		vad_decision	Ilog2	
			Noise_estimate_update	update_cntrl
			hangover_addition	
		Estimate_Speech		
	tx_dtx_handler			
	Parm_serial			
	Autocorr			
	Lag_window			
	Levinson			
	Az_isp	Chebps2		
	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	
			Isqrt_n	
		Interpol_4		
	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			
	Isqrt_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_1p_N1
			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_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			
		Isqrt_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			
		Isqrt_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		
	Dpisf_2s_36b	Reorder_isf			
	Dpisf_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_2p_2N1		
		dec_5p_5N	dec_3p_3N1	Dec_2p_2N1	
				Dec_1p_N1	
			Dec_2p_2N1		
		dec_6p_6N_2	Dec_5p_5N	dec_3p_3N1	Dec_2p_2N1
					Dec_1p_N1
				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_2p_2N1	
			dec_2p_2N1		
			dec_3p_3N1	Dec_2p_2N1	
				Dec_1p_N1	
	Preemph				
	Pit_shrp				
	D_gain2	Dot_product12			
		Isqrt_n			
		Median5			
		Pow2			
	Scale_sig				
	voice_factor	Dot_product12			
	Phase_dispersion	Set_zero			
	Agc2	Isqrt	Isqrt_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 ½
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	Isp_init	16	isp tables for initialization
Cod_main.c	Isf_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	Isp_init	16	isp tables for initialization
Dec_main.c	Isf_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
Isp_isf.tab	slope	128	table to compute cos(x) in Lsf_lsp()
Isp_isf.tab	Table	129	table to compute acos(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	1st ISF quantizer for comfort noise
Qisf_ns.tab	dico2_isf_noise	3*64	2nd ISF quantizer for comfort noise
Qisf_ns.tab	Dico3_isf_noise	3*64	3rd LSF quantizer for comfort noise
Qisf_ns.tab	Dico4_isf_noise	4*32	4th LSF quantizer for comfort noise
Qisf_ns.tab	Dico5_isf_noise	4*32	5th LSF quantizer for comfort noise
Qisf_ns.tab	mean_isf_noise	16	ISF mean for comfort noise
Qpisf_2s.tab	dico1_isf	9*256	1st ISF quantizer of the 1st stage
Qpisf_2s.tab	Dico2_isf	7*256	2nd ISF quantizer of the 1st stage
Qpisf_2s.tab	Dico21_isf	3*64	1st ISF quantizer of the 2nd stage (not the 6.60 kbit/s mode)
Qpisf_2s.tab	Dico21_isf_36b	5*128	1st ISF quantizer of the 2nd stage (the 6.60 kbit/s mode)
Qpisf_2s.tab	Dico22_isf	3*128	2nd ISF quantizer of the 2nd stage (not the 6.60 kbit/s mode)
Qpisf_2s.tab	Dico22_isf_36b	4*128	2nd ISF quantizer of the 2nd stage (the 6.60 kbit/s mode)
Qpisf_2s.tab	Dico23_isf	3*128	3rd ISF quantizer of the 2nd stage (not the 6.60 kbit/s mode)
Qpisf_2s.tab	Dico23_isf_36b	7*64	3rd ISF quantizer of the 2nd stage (the 6.60 kbit/s mode)
Qpisf_2s.tab	Dico24_isf	3*32	4th ISF quantizer of the 2nd stage (not the 6.60 kbit/s mode)
Qpisf_2s.tab	Dico25_isf	4*32	5th ISF quantizer of the 2nd stage (not the 6.60 kbit/s mode)
Qpisf_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	Preemhasis 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	weigthing level depeding 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
	Isfold	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	Isf_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	Preemhasis 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	3168, 29954, 29213, 16121, 64, 13440, 30624, 16430, 19008
1	3168, 31665, 9943, 9123, 15599, 4358, 20248, 2048, 17040, 27787, 16816, 13888
2	3168, 31665, 9943, 9128, 3647, 8129, 30930, 27926, 18880, 12319, 496, 1042, 4061, 20446, 25629, 28069, 13948
3	3168, 31665, 9943, 9131, 24815, 655, 26616, 26764, 7238, 19136, 6144, 88, 4158, 25733, 30567, 30494, 221, 20321, 17823
4	3168, 31665, 9943, 9131, 24815, 700, 3824, 7271, 26400, 9528, 6594, 26112, 108, 2068, 12867, 16317, 23035, 24632, 7528, 1752, 6759, 24576
5	3168, 31665, 9943, 9135, 14787, 14423, 30477, 24927, 25345, 30154, 916, 5728, 18978, 2048, 528, 16449, 2436, 3581, 23527, 29479, 8237, 16810, 27091, 19052, 0
6	3168, 31665, 9943, 9129, 8637, 31807, 24646, 736, 28643, 2977, 2566, 25564, 12930, 13960, 2048, 834, 3270, 4100, 26920, 16237, 31227, 17667, 15059, 20589, 30249, 29123, 0
7	3168, 31665, 9943, 9132, 16748, 3202, 28179, 16317, 30590, 15857, 19960, 8818, 21711, 21538, 4260, 16690, 20224, 3666, 4194, 9497, 16320, 15388, 5755, 31551, 14080, 3574, 15932, 50, 23392, 26053, 31216
8	3168, 31665, 9943, 9134, 24776, 5857, 18475, 28535, 29662, 14321, 16725, 4396, 29353, 10003, 17068, 20504, 720, 0, 8465, 12581, 28863, 24774, 9709, 26043, 7941, 27649, 13965, 15236, 18026, 22047, 16681, 3968

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 available formats.

NOTE ON DEFAULT 3GPP AND ITU BITSTREAM FORMATS:
ITU stream format gives very limited possibilities to distinguish NO_DATA and SID_FIRST frame types at the beginning of a stream. In some very limited cases for which some instance between encoder and decoder cuts of the first hangover period frames (e.g. handovers, editing of the stream), the output of the decoder is different depending on the stream format, ITU or default 3GPP.

Default 3GPP format:

This is the default format used in 3GPP. This format shall be used when the codec is tested against the test vectors.

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. Each encoded bit is represented as follows: Bit 0 = 0xff81, Bit 1 = 0x007f. 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 `Bx` either has the value `0x0081 (for bit 0)` or `0x007F (for bit 1)`.

`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.

ITU format (activated with command line parameter -itu)

SYNC_WORD	DATA_LENGTH	B1	B2	…	Bnn

Each box corresponds to one `Word16` value in the bitstream file, for a total of 2+nn words or 4+2nn bytes per frame, where nn is the number of encoded bits in the frame. Each encoded bit is represented as follows: Bit 0 = 0x007f, Bit 1 = 0x0081. The fields have the following meaning:

`SYNC_WORD` Word to ensure correct frame synchronization between the encoder and the decoder. It is also used to indicate the occurrences of bad frames.

```
In the encoder output:   (0x6b21)
In the decoder input:    Good frames (0x6b21)
                         Bad frames  (0x6b20)
```

`DATA_LENGTH` Length of the speech data. Codec mode and frame type is extracted in the decoder using this parameter:

DATA _LENGTH	PREVIOUS FRAME	CODEC MODE	FRAMETYPE
0	RX_SPEECH_GOOD/ RX_SPEECH_LOST	DTX	RX_SID_FIRST
0	OTHER THAN RX_SPEECH_GOOD/ RX_SPEECH_LOST	DTX	RX_NO_DATA
35	-	DTX	RX_SID_UPDATE
132	-	6.60 kbit/s	RX_SPEECH_GOOD/ RX_SPEECH_LOST
177	-	8.85 kbit/s	RX_SPEECH_GOOD/ RX_SPEECH_LOST
253	-	12.65 kbit/s	RX_SPEECH_GOOD/ RX_SPEECH_LOST
285	-	14.25 kbit/s	RX_SPEECH_GOOD/ RX_SPEECH_LOST
317	-	15.85 kbit/s	RX_SPEECH_GOOD/ RX_SPEECH_LOST
365	-	18.25 kbit/s	RX_SPEECH_GOOD/ RX_SPEECH_LOST
397	-	19.85 kbit/s	RX_SPEECH_GOOD/ RX_SPEECH_LOST
461	-	23.05 kbit/s	RX_SPEECH_GOOD/ RX_SPEECH_LOST
477	-	23.85 kbit/s	RX_SPEECH_GOOD/ RX_SPEECH_LOST

MIME/file storage format (activated with command line parameter -mime)

Detailed description of the AMR-WB single channel MIME/file storage format can be found in [7] (sections 5.1 and 5.3). This format is used e.g. by the Multimedia Messaging Service (MMS).

Annex A (informative):
Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
03-2001	11	SP-010083			Version 2.0.0 provided for approval		5.0.0
06-2001	12	SP-010307	001	1	Unnecessary printing in Az_isp-function	5.0.0	5.1.0
06-2001	12	SP-010307	002	1	Overflow in isp_az.c	5.0.0	5.1.0
06-2001	12	SP-010307	003	1	Error in the ISF extrapolation in 6.60 kbit/s mode	5.0.0	5.1.0
06-2001	12	SP-010307	004	1	14-bit masking to decoder	5.0.0	5.1.0
06-2001	12	SP-010307	005	1	Correction of the homing function	5.0.0	5.1.0
06-2001	12	SP-010307	006	1	Fixed codebook initialisation	5.0.0	5.1.0
06-2001					Minor editorial to cover page	5.1.0	5.1.1
09-2001	13	SP-010455	007		Error in the C-code of the encoder homing function	5.1.1	5.2.0
09-2001	13	SP-010455	008		Inconsistency in the file format description	5.1.1	5.2.0
12-2001	14	SP-010699	009		Incorrect mode usage during DTX	5.2.0	5.3.0
12-2001	14	SP-010699	010		Correction of decoder homing function for 23.85 kbit/s mode	5.2.0	5.3.0
03-2002	15	SP-020081	011	2	Correction of mode reading and memory usage	5.3.0	5.4.0
03-2002	15	SP-020081	012		Correction of pitch calculation of AMR-WB encoder	5.3.0	5.4.0
03-2002	15	SP-020081	013		Error concealment of high band gain in 23.85 kbit/s mode	5.3.0	5.4.0
12-2002	18	SP-020692	014		Correction of ambiguous expression in the AMR-WB C-Code	5.4.0	5.5.0
03-2003	19	SP-030089	015	2	Harmonization of 3GPP TS 26.173 and ITU-T G.722.2 C-codes	5.5.0	5.6.0
03-2003	19	SP-030089	016		Correction for handling of RX_NO_DATA frames	5.5.0	5.6.0
06-2003	20	SP-030216	017	1	MMS compatible input/output option for fixed-point AMR-WB source code	5.6.0	5.7.0
					Added file containing the C-code accidentally omitted from previous version	5.7.0	5.7.1
09-2003	21	SP-030446	019		Possible decoder LPC coefficients overflow	5.7.1	5.8.0
12-2004	26	SP-040844	020	1	Incorrect definition of vector nb_of_bits	5.8.0	6.0.0
12-2006	34	SP-060846	0023	1	Correction to bug in ITU-T bitstream format in the presence of frame erasures	6.0.0	6.1.0
03-2007	35	SP-070023	0025	1	Correct text specification to be aligned with the C-code	6.1.0	6.2.0
03-2007	35	SP-070029	0026		Correction in AMR decoder to avoid division by zero in RX-DTX Handling	6.2.0	7.0.0
09-2007	37	SP-070626	0029	1	Robust operation of AMRWB-decoder	7.0.0	7.1.0
12-2008	42				Version for Release 8	7.1.0	8.0.0
12-2009	46				Version for Release 9	8.0.0	9.0.0
03-2011	51				Version for Release 10	9.0.0	10.0.0
09-2012	57				Version for Release 11	10.0.0	11.0.0
09-2014	65				Version for Release 12	11.0.0	12.0.0
03-2015	67	SP-150094	0030	2	Correction on AMR-WB (noise energy initialization)	12.0.0	12.1.0
03-2015	67	SP-150094	0031	2	Correction on AMR-WB (out-of-bound memory access)	12.0.0	12.1.0
12-2015	70				Version for Release 13	12.1.0	13.0.0

History

Document history		
V13.0.0	January 2016	Publication