ETSI TS 126 410 V6.1.1 (2005-01)

Technical Specification

**Universal Mobile Telecommunications System (UMTS);
General audio codec audio processing functions;
Enhanced aacPlus general audio codec;
Floating-point ANSI-C code
(3GPP TS 26.410 version 6.1.1 Release 6)**

Reference

DTS/TSGS-0426410v611

Keywords

UMTS

Intellectual Property Rights

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 .

Contents

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 Floating-point Enhanced aacPlus codec [1].

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 26.401: "Enhanced aacPlus general audio codec; General Description".

[2] 3GPP TS 26.403: "Enhanced aacPlus general audio codec; Encoder Specification AAC part".

[3] 3GPP TS 26.404: "Enhanced aacPlus general audio codec; Encoder Specification SBR part".

[4] 3GPP TS 26.405: "Enhanced aacPlus general audio codec; Encoder Specification Parametric Stereo part".

[5] ISO/IEC 14496-3:2001: "Information technology - Coding of audio-visual objects - Part 3: Audio".

[6] ISO/IEC 14496-3:2001/Amd.1:2003: "Bandwidth Extension".

[7] ISO/IEC 14496-3:2001/Amd.1:2003/DCOR1".

[8] ISO/IEC 14496-3:2001/ Amd.2:2004: "Parametric Coding for High Quality Audio.

[9] 3GPP TS 26.402: Enhanced aacPlus general audio codec; Additional Decoder Tools".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TS 26.401 [1], TS 26.403 [2], TS 26.404 [3], TS 26.405 [4] and TS 26.402 [9] apply.

3.2 Abbreviations

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

AAC	Advanced Audio Coding
aacPlus	Combination of MPEG-4 AAC and MPEG-4 Bandwidth extension (SBR)
Enhanced aacPlus	Combination of MPEG-4 AAC, MPEG-4 Bandwidth extension (SBR) and MPEG-4 Parametric Stereo
MDCT	Modified Discrete Cosine Transform
QMF	Quadrature Mirror Filter
SBR	Spectral Band Replication

ANSI	American National Standards Institute
GSM	Global System for Mobile communications
I/O	Input/Output
RAM	Random Access Memory
ROM	Read Only Memory

4 Floating point ANSI-C code structure

This clause gives an overview of the structure of the floating point ANSI-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 XP, 2000 and Microsoft Visual C++ v.6.0 compiler.
- IBM PC/AT compatible computers with Linux OS and GCC v.3.3 compiler.

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

4.1 Contents of the floating point ANSI-C source code

The C code distribution is organised in two directories for encoder and decoder and further into several subdirectories, reflecting the major building blocks of the Enhanced aacPlus codec. The file descriptions on root level as well as the directory structure is given as follows:

Table 1: Source code directory structure for the encoder (FloatFR_aacPlusenc)

Directory	Description
README.txt	information on how to compile
Makefile	UNIX style encoder Makefile
FloatFR_aacPlusEnc.dsw	Win32 MSVC 6.0 encoder workspace
FloatFR_aacPlusEnc.dsp	Win32 MSVC 6.0 encoder makefile
src/	directory for the encoder frontend
FloatFR_fastaacenc/	AAC encoder library
FloatFR_resamplib/	resampler library
FloatFR_sbrenclib/	SBR encoder library

Table 2: Source code directory structure for the decoder (FloatFR_aacPlusdec)

Directory	Description
README.txt	information on how to compile
Makefile	UNIX style encoder Makefile
FloatFR_aacPlusdec_mpeg4.dsw	Win32 MSVC 6.0 decoder workspace
FloatFR_aacPlusdec_mpeg4.dsp	Win32 MSVC 6.0 decodec makefile
src/	directory for the decoder frontend
FloatFR_aacdec	AAC decoder library
FloatFR_sbrdeclib/	SBR decoder library

Table 3: Source code directory structure common for encoder and decoder

Directory	Description
FloatFR_bitbuflib/	bitstream reading/writing library
FloatFRlib/	general purpose functionalities
lib/	precompiled libraries for audio and bitstream file format handling

The distributed files with suffix "c" contain the source code and the files with suffix "h" are the header files. Within the respective libraries, the RAM data is contained in "xxx_ram" files with suffix "c", the ROM data is contained in "xxx_rom" files with suffix "c". Makefiles are provided for the platforms in which the C code has been verified (listed above).

Note that the FloatFRlib/, FloatFR_bitbuflib/ and lib/ directory are identical for encoder and decoder. A list of source code files with the respective lines of code (pure C instructions) is given below:

Table 4: Encoder source code files and lines of code

Directory	Module	Lines of code
src/	main.c	332
	mp4file.c	255
FloatFR_fastaacenclib/	qc_main.c	224
	aacenc.c	136
	ms_stereo.c	50
	spreading.c	10
	interface.c	44
	bit_cnt.c	588
	adj_thr.c	592
	quantize.c	56
	psy_configuration.c	175
	sf_estim.c	508
	tns_param.c	45
	grp_data.c	114
	pre_echo_control.c	22
	stprepro.c	149
	tns.c	358
	dyn_bits.c	281
	psy_main.c	232
	channel_map.c	52
	block_switch.c	201
	band_nrg.c	34
	transform.c	151
	bitenc.c	262
	line_pe.c	55
	stat_bits.c	107
FloatFR_sbrenclib/	qmf_enc.c	565
	ton_corr.c	287
	fram_gen.c	688
	env_bit.c	56
	env_est.c	630
	mh_det.c	515
	hybrid.c	139
	bit_sbr.c	375
	ps_bitenc.c	225
	sbr_main.c	355
	tran_det.c	183
	sbr_misc.c	49
	code_env.c	290
	nf_est.c	195
	freq_sca.c	309
	invf_est.c	140
	ps_enc.c	299
FloatFR_resamplib/	iir32resample.c	71
	resampler.c	68

Table 5: Decoder source code files and lines of code

Directory	Module	Lines of code
src/	main.c	299
	fileifc.c	173
	spline_resampler.c	172
FloatFR_aacdec/	aacdecoder.c	168
	streaminfo.c	10
	channelinfo.c	102
	stereo.c	78
	longblock.c	234
	shortblock.c	241
	pulsedata.c	24
	block.c	163
	pns.c	96
	imdct.c	50
	tns.c	137
	bitstream.c	15
	channel.c	92
	conceal.c	245
FloatFR_sbrdeclib/	env_dec.c	370
	FFR_aacPLUScheck.c	32
	sbr_bitb.c	37
	env_calc.c	775
	lpp_tran.c	504
	sbrdecoder.c	514
	sbr_dec.c	218
	sbr_crc.c	45
	sbr_fft.c	615
	hybrid.c	140
	ps_bitdec.c	223
	huff_dec.c	9
	env_extr.c	655
	freq_sca.c	337
	ps_dec.c	317
	qmf_dec.c	526

Table 6: Common source code files and lines of code

Directory	Module	Lines of code
FloatFR_bitbuflib/	bitbuffer.c	111
FloatFRlib/	cfftn.c	649
	transcendent.c	15

4.2 Program execution

The Enhanced aacPlus codec is implemented in two programs:

- enhAacPlusEnc.exe
- enhAacPlusDec.exe

The programs should be called like:

- enhAacPlusEnc.exe <wav_file> <bitstream_file> <bitrate> <(m)ono/(s)tereo>
- enhAacPlusDec.exe <bitstream_file> <wav_file> <mode> [error_pattern_file]

The audio files contain 16-bit linear encoded PCM samples with wav header, the bitstream files are of 3GPP type an the error patter file is a ASCII file, see section 5.

The encoder and decoder command line handling is also explained by running the applications without input arguments.

4.3 Memory requirements

The data types of variables and tables used in the floating-point implementation are plain ANSI-C data types, the following types are used:

- char
- unsigned char
- short
- int
- unsigned int
- float

4.3.1 Constants and tables

This clause contains a listing of all constants and tables contributing to the ROM requirements of the encoder and decoder.

Table 7: Encoder constants and tables

Name	Data type	Size [word]	Allocated in Source File	Description
LongWindowSine	float	1024	aac_rom.c	Window coefficients
ShortWindowSine	float	128	aac_rom.c	Window coefficients
LongWindowKBD	float	1024	aac_rom.c	Window coefficients
fftTwiddleTab	float	513	aac_rom.c	FFT twiddle coefficients
quantTableQ	float	16	aac_rom.c	Quantizer table, used for efficient pow () implementation
quantTableE	float	17	aac_rom.c	Quantizer table, used for efficient pow () implementation
invQuantTableQ	float	16	aac_rom.c	Quantizer table, used for efficient pow () implementation
invQuantTableE	float	17	aac_rom.c	Quantizer table, used for efficient pow () implementation
pow4_3_tab	float	64	aac_rom.c	Quantizer table, used for efficient pow () implementation
p_8000_mono_long	float	4	aac_rom.c	TNS tuning parameters
p_8000_stereo_long	float	4	aac_rom.c	TNS tuning parameters
p_8000_mono_short	float	4	aac_rom.c	TNS tuning parameters
p_8000_stereo_short	float	4	aac_rom.c	TNS tuning parameters
p_16000_mono_long	float	4	aac_rom.c	TNS tuning parameters
p_16000_stereo_long	float	4	aac_rom.c	TNS tuning parameters
p_16000_mono_short	float	4	aac_rom.c	TNS tuning parameters
p_16000_stereo_short	float	4	aac_rom.c	TNS tuning parameters
p_24000_mono_long	float	4	aac_rom.c	TNS tuning parameters
p_24000_stereo_long	float	4	aac_rom.c	TNS tuning parameters
p_24000_mono_short	float	4	aac_rom.c	TNS tuning parameters
p_24000_stereo_short	float	4	aac_rom.c	TNS tuning parameters
p_32000_mono_long	float	4	aac_rom.c	TNS tuning parameters
p_32000_stereo_long	float	4	aac_rom.c	TNS tuning parameters
p_32000_mono_short	float	4	aac_rom.c	TNS tuning parameters
p_32000_stereo_short	float	4	aac_rom.c	TNS tuning parameters
tnsCoeff3	float	8	aac_rom.c	TNS filter coefficients
tnsCoeff3Borders	float	8	aac_rom.c	TNS filter borders
tnsCoeff4	float	16	aac_rom.c	TNS filter coefficients
tnsCoeff4Borders	float	16	aac_rom.c	TNS filter borders
tnsInfoTab	int	24	aac_rom.c	TNS bitrate to tuning mapping table
tnsMaxBandsTab	int	27	aac_rom.c	max. TNS bands per sampling rate table
huff_ltab1_2	short	80	aac_rom.c	Huffman codeword table AAC
huff_ltab3_4	short	80	aac_rom.c	Huffman codeword table AAC
huff_ltab5_6	short	80	aac_rom.c	Huffman codeword table AAC
huff_ltab7_8	short	64	aac_rom.c	Huffman codeword table AAC
huff_ltab9_10	short	168	aac_rom.c	Huffman codeword table AAC
huff_ltab11	short	288	aac_rom.c	Huffman codeword table AAC
huff_ltabscf	short	120	aac_rom.c	Huffman codeword table AAC
huff_ctab1	short	80	aac_rom.c	Huffman codeword table AAC
huff_ctab2	short	80	aac_rom.c	Huffman codeword table AAC
huff_ctab3	short	80	aac_rom.c	Huffman codeword table AAC
huff_ctab4	short	80	aac_rom.c	Huffman codeword table AAC
huff_ctab5	short	80	aac_rom.c	Huffman codeword table AAC
huff_ctab6	short	80	aac_rom.c	Huffman codeword table AAC
huff_ctab7	short	64	aac_rom.c	Huffman codeword table AAC
huff_ctab8	short	64	aac_rom.c	Huffman codeword table AAC
huff_ctab9	short	168	aac_rom.c	Huffman codeword table AAC
huff_ctab10	short	168	aac_rom.c	Huffman codeword table AAC
huff_ctab11	short	288	aac_rom.c	Huffman codeword table AAC
huff_ctabscf	short	242	aac_rom.c	Huffman codeword table AAC
sfb_11025_long_1024	char	43	aac_rom.c	Scalefactor band table
sfb_11025_short_128	char	15	aac_rom.c	Scalefactor band table
sfb_12000_long_1024	char	43	aac_rom.c	Scalefactor band table
sfb_12000_short_128	char	15	aac_rom.c	Scalefactor band table
sfb_16000_long_1024	char	43	aac_rom.c	Scalefactor band table
sfb_16000_short_128	char	15	aac_rom.c	Scalefactor band table
sfb_22050_long_1024	char	47	aac_rom.c	Scalefactor band table

sfb_22050_short_128	char	15	aac_rom.c	Scalefactor band table
sfb_24000_long_1024	char	47	aac_rom.c	Scalefactor band table
sfb_24000_short_128	char	15	aac_rom.c	Scalefactor band table
panClass	float	7	sbr_rom.c	Parametric Stereo quantization table
saClass	float	7	sbr_rom.c	Parametric Stereo quantization table
p4_13	float	13	sbr_rom.c	Hybrid filterbank coefficients
p8_13	float	13	sbr_rom.c	Hybrid filterbank coefficients
sbr_cos_twiddle	float	16	sbr_rom.c	QMF filterbank twiddle table
sbr_sin_twiddle	float	16	sbr_rom.c	QMF filterbank twiddle table
sbr_alt_sin_twiddle	float	17	sbr_rom.c	QMF filterbank twiddle table
sbr_qmf_64_640	float	325	sbr_rom.c	QMF window coefficients
p_64_640_qmf	float	640	sbr_rom.c	QMF window coefficients (Note: could be made obsolete)
trigData_fct4_32	float	32	sbr_rom.c	FFT twiddle table
trigData_fct4_16	float	16	sbr_rom.c	FFT twiddle table
trigData_fct4_8	float	8	sbr_rom.c	FFT twiddle table
aBookPsIidTimeCode	int	29	sbr_rom.c	Huffman codeword table Parametric Stereo
aBookPsIidFreqCode	int	29	sbr_rom.c	Huffman codeword table Parametric Stereo
aHybridResolution	int	3	sbr_rom.c	Number of hybrid bands in each QMF band
hiResBandBorders	int	21	sbr_rom.c	Borders of Parametric Stereo bins
groupBordersMix	int	29	sbr_rom.c	Borders of Parametric Stereo groups
bins2groupMap	int	29	sbr_rom.c	Mapping of Parametric Stereo bins to Parametric Stereo groups
v_Huff_envelopeLevelC10T	int	121	sbr_rom.c	Huffman codeword table SBR
v_Huff_envelopeLevelC10F	int	121	sbr_rom.c	Huffman codeword table SBR
bookSbrEnvBalanceC10F	int	49	sbr_rom.c	Huffman codeword table SBR
bookSbrEnvBalanceC10T	int	49	sbr_rom.c	Huffman codeword table SBR
v_Huff_envelopeLevelC11T	int	63	sbr_rom.c	Huffman codeword table SBR
v_Huff_NoiseLevelC11T	int	63	sbr_rom.c	Huffman codeword table SBR
bookSbrEnvBalanceC11T	int	25	sbr_rom.c	Huffman codeword table SBR
bookSbrNoiseBalanceC11T	int	25	sbr_rom.c	Huffman codeword table SBR
v_Huff_envelopeLevelC11F	int	63	sbr_rom.c	Huffman codeword table SBR
bookSbrEnvBalanceC11F	int	25	sbr_rom.c	Huffman codeword table SBR
aBookPsIidTimeLength	char	29	sbr_rom.c	Huffman codeword table Parametric Stereo
aBookPsIidFreqLength	char	29	sbr_rom.c	Huffman codeword table Parametric Stereo
aBookPsIccFreqLength	char	15	sbr_rom.c	Huffman codeword table Parametric Stereo
aBookPsIccTimeLength	char	15	sbr_rom.c	Huffman codeword table Parametric Stereo
v_Huff_envelopeLevelL10T	char	121	sbr_rom.c	Huffman codeword table SBR
v_Huff_envelopeLevelL10F	char	121	sbr_rom.c	Huffman codeword table SBR
bookSbrEnvBalanceL10F	char	49	sbr_rom.c	Huffman codeword table SBR
bookSbrEnvBalanceL10T	char	49	sbr_rom.c	Huffman codeword table SBR
v_Huff_envelopeLevelL11T	char	63	sbr_rom.c	Huffman codeword table SBR
bookSbrEnvBalanceL11T	char	25	sbr_rom.c	Huffman codeword table SBR
v_Huff_NoiseLevelL11T	char	63	sbr_rom.c	Huffman codeword table SBR
bookSbrNoiseBalanceL11T	char	25	sbr_rom.c	Huffman codeword table SBR
v_Huff_envelopeLevelL11F	char	63	sbr_rom.c	Huffman codeword table SBR
bookSbrEnvBalanceL11F	char	25	sbr_rom.c	Huffman codeword table SBR
aBookPsIccFreqCode	short	15	sbr_rom.c	Huffman codeword table Parametric Stereo
aBookPsIccTimeCode;	short	15	sbr_rom.c	Huffman codeword table Parametric Stereo
logDualisTable	float	65	transcendent.c	Lookup table for efficient log() implementation
set1_a	float	14	resampler.c	IIR filter coefficients for 2:1 resampling
set1_b	float	14	resampler.c	IIR filter coefficients for 2:1 resampling
set1	float	5	resampler.c	IIR filter coefficients for 2:1 resampling
set2_a	float	21	resampler.c	IIR filter coefficients for 2:1 resampling
set2_b	float	21	resampler.c	IIR filter coefficients for 2:1 resampling
set2	float	5	resampler.c	IIR filter coefficients for 2:1 resampling
set3_a	float	18	resampler.c	IIR filter coefficients for 2:1 resampling
set3_b	float	18	resampler.c	IIR filter coefficients for 2:1 resampling
set3	float	5	resampler.c	IIR filter coefficients for 2:1 resampling
coeffNum	float	8	iir32resample.c	IIR filter coefficients for 3:2 resampling
coeffDen	float	8	iir32resample.c	IIR filter coefficients for 3:2 resampling
tuningTable	tuningTable	121	sbr_main.c	SBR tuning parameters
Sum		**8533**		

Table 8: Decoder constants and tables

Name	Data type	Size [word]	Allocated in Source File	Description
tnsCoeff3	float	8	aac_rom.c	TNS filter coefficients
tnsCoeff4	float	16	aac_rom.c	TNS filter coefficients
trigData	float	513	aac_rom.c	Sine table, used for efficient sin(), cos()
OnlyLongWindowKBD	float	1024	aac_rom.c	Window coefficients
OnlyShortWindowKBD	float	128	aac_rom.c	Window coefficients
OnlyLongWindowSine	float	1024	aac_rom.c	Window coefficients
OnlyShortWindowSine	float	128	aac_rom.c	Window coefficients
sfb_48_1024	short	50	aac_rom.c	Scalefactor band table
sfb_48_128	short	15	aac_rom.c	Scalefactor band table
sfb_32_1024	short	51	aac_rom.c	Scalefactor band table
sfb_24_1024	short	49	aac_rom.c	Scalefactor band table
sfb_24_128	short	16	aac_rom.c	Scalefactor band table
sfb_16_1024	short	44	aac_rom.c	Scalefactor band table
sfb_16_128	short	16	aac_rom.c	Scalefactor band table
sfb_8_1024	short	41	aac_rom.c	Scalefactor band table
sfb_8_128	short	16	aac_rom.c	Scalefactor band table
HuffmanCodeBook_1	short	204	aac_rom.c	Huffman codeword table AAC
HuffmanCodeBook_2	short	156	aac_rom.c	Huffman codeword table AAC
HuffmanCodeBook_3	short	156	aac_rom.c	Huffman codeword table AAC
HuffmanCodeBook_4	short	152	aac_rom.c	Huffman codeword table AAC
HuffmanCodeBook_5	short	164	aac_rom.c	Huffman codeword table AAC
HuffmanCodeBook_6	short	160	aac_rom.c	Huffman codeword table AAC
HuffmanCodeBook_7	short	124	aac_rom.c	Huffman codeword table AAC
HuffmanCodeBook_8	short	124	aac_rom.c	Huffman codeword table AAC
HuffmanCodeBook_9	short	336	aac_rom.c	Huffman codeword table AAC
HuffmanCodeBook_10	short	328	aac_rom.c	Huffman codeword table AAC
HuffmanCodeBook_11	short	544	aac_rom.c	Huffman codeword table AAC
HuffmanCodeBook_SCL	short	260	aac_rom.c	Huffman codeword table AAC
SamplingRateInfoTable	mixed	45	aac_rom.c	Sampling rate to scalefactor mapping table AAC
HuffmanCodeBooks	mixed	52	aac_rom.c	Huffman codeword table AAC
tns_max_bands_tbl	char	18	aac_rom.c	max. TNS bands per sampling rate table
sbr_limGains	float	4	sbr_rom.c	SBR limiter gain values
sbr_limiterBandsPerOctave	float	4	sbr_rom.c	Number of SBR limiter bands
sbr_smoothFilter	float	4	sbr_rom.c	Smoothing filter for gain values
sbr_invIntTable	float	55	sbr_rom.c	Table of 1/x function
sbr_randomPhase	float	1024	sbr_rom.c	Random numbers for SBR noise addition and PNS
sbr_qmf_64_640	float	325	sbr_rom.c	QMF window coefficients
sbr_cos_twiddle_L04	float	2	sbr_rom.c	FFT twiddle table
sbr_cos_twiddle_L08	float	4	sbr_rom.c	FFT twiddle table
sbr_cos_twiddle_L16	float	8	sbr_rom.c	FFT twiddle table
sbr_cos_twiddle_L32	float	16	sbr_rom.c	FFT twiddle table
sbr_sin_twiddle_L04	float	2	sbr_rom.c	FFT twiddle table
sbr_sin_twiddle_L08	float	4	sbr_rom.c	FFT twiddle table
sbr_sin_twiddle_L16	float	8	sbr_rom.c	FFT twiddle table
sbr_sin_twiddle_L32	float	16	sbr_rom.c	FFT twiddle table
sbr_alt_sin_twiddle_L04	float	3	sbr_rom.c	FFT twiddle table
sbr_alt_sin_twiddle_L08	float	5	sbr_rom.c	FFT twiddle table
sbr_alt_sin_twiddle_L16	float	9	sbr_rom.c	FFT twiddle table
sbr_alt_sin_twiddle_L32	float	17	sbr_rom.c	FFT twiddle table
sbr_cos_twiddle_ds_L32	float	32	sbr_rom.c	FFT twiddle table, obsolete for mono only decoder
sbr_sin_twiddle_ds_L32	float	32	sbr_rom.c	FFT twiddle table, obsolete for mono only decoder
sbr_cos_twiddle_L64	float	32	sbr_rom.c	FFT twiddle table, obsolete for mono only decoder
sbr_sin_twiddle_L64	float	32	sbr_rom.c	FFT twiddle table, obsolete for mono only decoder
sbr_alt_sin_twiddle_L64	float	33	sbr_rom.c	FFT twiddle table, obsolete for mono only decoder
sbr_t_cos_L32	float	32	sbr_rom.c	FFT twiddle table

sbr_t_sin_L32	float	32	sbr_rom.c	FFT twiddle table
aRevLinkDecaySer	float	3	sbr_rom.c	Parametric Stereo all-pass filter coefficients
aFractDelayPhaseFactorReQmf	float	20	sbr_rom.c	Parametric Stereo phase rotation factor
aFractDelayPhaseFactorImQmf	float	20	sbr_rom.c	Parametric Stereo phase rotation factor
aFractDelayPhaseFactorReSubQmf	float	10	sbr_rom.c	Parametric Stereo phase rotation factor
aFractDelayPhaseFactorImSubQmf	float	10	sbr_rom.c	Parametric Stereo phase rotation factor
aaFractDelayPhaseFactorSerReQmf	float	3	sbr_rom.c	Parametric Stereo phase rotation factor
aaFractDelayPhaseFactorSerImQmf	float	3	sbr_rom.c	Parametric Stereo phase rotation factor
aaFractDelayPhaseFactorSerReSubQmf	float	3	sbr_rom.c	Parametric Stereo phase rotation factor
aaFractDelayPhaseFactorSerImSubQmf	float	3	sbr_rom.c	Parametric Stereo phase rotation factor
scaleFactors	float	15	sbr_rom.c	Parametric Stereo quantization table
scaleFactorsFine	float	41	sbr_rom.c	Parametric Stereo quantization table
alphas	float	8	sbr_rom.c	Parametric Stereo quantization table
p2_6	float	6	sbr_rom.c	Hybrid filterbank coefficients
p8_13	float	14	sbr_rom.c	Hybrid filterbank coefficients
sbr_whFactorsTable	float	54	sbr_rom.c	Tuning parameters for inverse filtering
bins2groupMap	short	22	sbr_rom.c	Mapping of Parametric Stereo bins to Parametric Stereo groups
sbr_whFactorsIndex	short	9	sbr_rom.c	Tuning parameter index for inverse filtering
sbr_start_freq_16	char	16	sbr_rom.c	SBR frequency scale index
sbr_start_freq_22	char	16	sbr_rom.c	SBR frequency scale index
sbr_start_freq_24	char	16	sbr_rom.c	SBR frequency scale index
sbr_start_freq_32	char	16	sbr_rom.c	SBR frequency scale index
sbr_start_freq_44	char	16	sbr_rom.c	SBR frequency scale index
sbr_start_freq_48	char	16	sbr_rom.c	SBR frequency scale index
sbr_frame_info1_16	char	18	sbr_rom.c	SBR frequency scale index
sbr_frame_info2_16	char	18	sbr_rom.c	SBR frequency scale index
sbr_frame_info4_16	char	18	sbr_rom.c	SBR frequency scale index
sbr_huffBook_EnvLevel10T	char	240	sbr_rom.c	Huffman codeword table SBR
sbr_huffBook_EnvLevel10F	char	240	sbr_rom.c	Huffman codeword table SBR
sbr_huffBook_EnvBalance10T	char	96	sbr_rom.c	Huffman codeword table SBR
sbr_huffBook_EnvBalance10F	char	96	sbr_rom.c	Huffman codeword table SBR
sbr_huffBook_EnvLevel11T	char	124	sbr_rom.c	Huffman codeword table SBR
sbr_huffBook_EnvLevel11F	char	124	sbr_rom.c	Huffman codeword table SBR
sbr_huffBook_EnvBalance11T	char	48	sbr_rom.c	Huffman codeword table SBR
sbr_huffBook_EnvBalance11F	char	48	sbr_rom.c	Huffman codeword table SBR
sbr_huffBook_NoiseLevel11T	char	124	sbr_rom.c	Huffman codeword table SBR
sbr_huffBook_NoiseBalance11T	char	48	sbr_rom.c	Huffman codeword table SBR
aRevLinkDelaySer	char	3	sbr_rom.c	Parametric Stereo all-pass delay line lengths
groupBorders	char	23	sbr_rom.c	Borders of Parametric Stereo groups
aBookPsIidTimeDecode	char	56	sbr_rom.c	Huffman codeword table Parametric Stereo
aBookPsIidFreqDecode	char	56	sbr_rom.c	Huffman codeword table Parametric Stereo
aBookPsIccTimeDecode	char	28	sbr_rom.c	Huffman codeword table Parametric Stereo
aBookPsIccFreqDecode	char	28	sbr_rom.c	Huffman codeword table Parametric Stereo
aBookPsIidFineTimeDecode	char	120	sbr_rom.c	Huffman codeword table Parametric Stereo
aBookPsIidFineFreqDecode	char	120	sbr_rom.c	Huffman codeword table Parametric Stereo
sbr_defaultHeader	char	32	sbr_rom.c	Default SBR header data
logDualisTable	float	65	transcendent.c	Lookup table for efficient log() implementation
Sum		**9866**		

4.3.2 Static memory

This clause contains a listing of all static buffers contributing to the RAM requirements of the encoder and decoder.

Table 9: Encoder static memory

Name	Data type	Size [word]	Allocated in Source File	Description
mdctDelayBuffer	float	3200	aac_ram.c	Time domain input signal delay
sideInfoTabLong	int	52	aac_ram.c	Table lookup for side information, long blocks
sideInfoTabShort	int	16	aac_ram.c	Table lookup for side information, short blocks
aacEncoder	AAC_ENCODER	3554	aacenc.c	AAC encoder instance
sbr_QmfStatesAnalysis	float	1280	sbr_ram.c	QMF filterbank states buffer
sbr_envYBuffer	float	4096	sbr_ram.c	QMF band energy buffer
sbr_quotaMatrix	float	512	sbr_ram.c	Tonality values
sbr_thresholds	float	128	sbr_ram.c	Detector parameters
sbr_toncorrBuff	float	1256	sbr_ram.c	Detector value buffer
EnvChannel[nChan]	ENV_CHANNEL	1794	sbr_main.c	SBR channel instance, only half the size for mono only encoder
sbrEncoder	SBR_ENCODER	200	sbr_main.c	SBR encoder instance
SynthesisQmfBank	SBR_QMF_FILTER_BANK	7	sbr_main.c	QMF synthesis filterbank instance
psEncoder	PS_ENC	281	sbr_main.c	Parametric Stereo encoder instance
sbr_freqBandTableLO	char	14	sbr_ram.c	SBR frequency band table, low resolution
sbr_freqBandTableHI	char	28	sbr_ram.c	SBR frequency band table, high resolution
sbr_v_k_master	char	28	sbr_ram.c	SBR frequency band table index
sbr_guideScfb	char	54	sbr_ram.c	Additional sine detection parameter
sbr_detectionVectors	char	216	sbr_ram.c	Additional sine detection parameter
sbr_prevEnvelopeCompensation	char	54	sbr_ram.c	Additional sine detection parameter
sbr_guideVectorDetected	char	216	sbr_ram.c	Additional sine detection parameter
outputBuffer	int	384	main.c	Bitstream output buffer
inputBuffer[nChan]	float	7202	main.c	Time domain input signal buffer, only half the size for mono only encoder
IIR21_resampler[nChan]	float	144	main.c	2:1 IIR resampler instance (includes states) , only half the size for mono only encoder
statesIIR	float	16	iir32resample.c	3:2 IIR resampler states buffer
Sum		**24732**		

Table 10: Decoder static memory

Name	Data type	Size [word]	Allocated in Source File	Description
OverlapBuffer[nChan]	float	1024	aac_ram.c	Delay buffer for overlap and add, only half the size for mono only decoder
AacDecoderInstance	AAC_DECODER_INSTANCE	11	aacdecoder.c	AAC decoder instance
StreamInfo	CStreamInfo	7	aac_ram.c	Bitstream information
AacDecoderStaticChannelInfo[nChan]	CaacDecoderStaticChannelInfo	14	aac_ram.c	Channel information, only half the size for mono only decoder
sbr_CodecQmfStatesAnalysis	float	640	sbr_ram.c	QMF analysis filter bank states
sbr_GainSmooth	float	96	sbr_ram.c	Gain smoothing filter states
sbr_NoiseSmooth	float	96	sbr_ram.c	Noise level smoothing filter states
sbr_QmfStatesSynthesis	float	1280	sbr_ram.c	QMF synthesis filter bank states
sbr_OverlapBuffer	float	1536	sbr_ram.c	SBR delay buffer, only half the size for mono only decoder
sbr_LpcFilterStatesReal	float	128	sbr_ram.c	LPC filter states
sbr_LpcFilterStatesImag	float	128	sbr_ram.c	LPC filter states, obsolete for mono only decoder
sbr_TransposerSettings	float	18	sbr_ram.c	Transposer configuration parameters
FreqBandData	FREQ_BAND_DATA	164	sbr_ram.c	SBR Frequency band information
PrevFrameData[nChan]	SBR_PREV_FRAME_DATA	120	sbr_ram.c	SBR previous frame data, only half the size for mono only decoder
sbr_PrevBitstream	SBRBITSTREAM	584	sbr_ram.c	SBR previous frame bitstream
sbrDecoderInstance	SBR_DECODER_INSTANCE	797	sbrdecoder.c	SBR decoder instance
TimeDataFloat[nChan]	float	4096	main.c	Output buffer for time-domain signal, only half the size for mono only decoder
inBuffer	int	384	main.c	Input buffer for bitstream
splineResamplerInstance	SPLINE_RESAMPLER	21	spline_resampler.c	Spline resampler instance
Sum		**11161**		

4.3.3 Dynamic memory

This clause contains a listing of all dynamic buffers contributing to the RAM requirements of the encoder and decoder. Dynamic memory can be re-used outside of the encoder or decoder application.

Table 11: Encoder dynamic memory

Name	Data type	Size [word]	Allocated in Source File	Description
PsBuf3	float	1024	sbr_ram.c	Note: reused in AAC encoder
sbr_envRBuffer	float	4096	sbr_ram.c	Note: reused in AAC encoder
sbr_envIBuffer	float	4096	sbr_ram.c	Note: reused in AAC encoder
sbr_transients	float	192	sbr_ram.c	Note: reused in AAC encoder
Sum		**9408**		

Table 12: Decoder dynamic memory

Name	Data type	Size [word]	Allocated in Source File	Description
WorkBufferCore	float	2048	aac_ram.c	Note: reused in SBR decoder
InterimResult	float	1024	sbr_ram.c	
Sum		**3072**		

4.3.4 Maximum stack size

This clause contains tables for the encoder and the decoder which describe the call stack that results in the maximum stack size usage.

Table 13: Encoder call stack

Function	Local variables	Stack used [bytes]
main	struct config;	20
	int error;	4
	int bEncodeMono;	4
	int bitrate;	4
	int nChannelsAAC, nChannelsSBR;	8
	int sampleRateAAC;	4
	int bandwidth;	4
	unsigned int numAncDataBytes;	4
	unsigned char ancDataBytes[256];	256
	unsigned int ancDataLength;	4
	int numSamplesRead;	4
	int bDoIIR2Downsample;	4
	int bDingleRate;	4
	int useParametricStereo;	4
	int coreWriteOffset;	4
	int coreReadOffset;	4
	int envWriteOffset;	4
	int envReadOffset;	4
	int writeOffset;	4
	struct *aacEnc;	4
	int bDoUpsample;	4
	int upsampleReadOffset;	4
	int inSamples;	4
	int bDoIIR32Resample;	4
	int nSamplesPerChannel;	4
	const int nRuns;	4
	float *resamplerScratch;	4
	struct *hEnvEnc;	4
	int i, ch, outSamples, numOutBytes;	16
		= 400
EnvEncodeFrame	struct *hEnvEncoder;	4
	float *samples;	4
	float *pCoreBuffer;	4
	unsigned int timeInStride;	4
	unsigned int *numAncBytes;	4
	unsigned char *ancData;	4
	struct *sbrBitstreamData;	4
		= 28
extractSbrEnvelope	float *timeInPtr;	4
	float *pCoreBuffer;	4
	unsigned int timeInStride;	4
	struct *h_con;	4
	struct *sbrHeaderData;	4
	struct *sbrBitstreamData;	4
	struct *h_envChan[];	4
	struct *h_ps_e;	4
	struct *hSynthesisQmfBank;	4
	struct *hCmonData;	4
	int ch, i, j, c;	16
	int nEnvelopes[2];	8
	int transient_info[2][2];	16
	const struct *frame_info[2];	8
	int nChannels, nInChannels;	8
	enum stereoMode;	4
	enum res[10];	40
	int v_tuning[6];	24
	int sfb_nrg [2][135];	1080
	float noiseFloor[2][10];	80

	int noise_level[2][10];
	int sfb_nrg_coupling[2][135];
	int noise_level_coupling[2][10];
	int maxQuantError;
		80
		1080
		80
		4
		= 2568

EncodePsFrame	struct *pms; float **iBufferLeft, float **rBufferLeft, float **iBufferRight, float **rBufferRight int env, i, bin, subband, maxSubband, startSample, stopSample; float **hybrLeftImag, **hybrLeftReal, **hybrRightImag, **hybrRightReal;	4 4 4 4 4 28 16 = 64

HybridAnalysis	const float **mQmfReal;
	const float **mQmfImag;
	float **mHybridReal;
	float **mHybridImag;
	struct *hHybrid;
	int n, band;
	enum hybridRes;
	int chOffset;
		4
		4
		4
		4
		4
		8
		4
		4
		= 36

eightChannelFiltering	const float *pQmfReal;
	const float *pQmfImag;
	float **mHybridReal;
	float **mHybridImag;
	int i, n;
	float real, imag;
	int midTap;
	float cum[16];
		4
		4
		4
		4
		8
		8
		4
		64
		= 100

CFFTN	float *afftData;
	int len;
	int isign;
		4
		4
		4
		= 12

cfftn	float Re[];
	float Im[];
	int nTotal;
	int nPass;
	int nSpan;
	int iSign;
	int ii, mfactor, kspan, ispan, inc, j, jc, jf, jj, k, k1, k2, k3, k4, kk, kt, nn, ns, nt;
	double radf, c1, c2, c3, cd, s1, s2, s3, sd;
	float ak, bk, akp, bkp, ajp, bjp, ajm, bjm, akm, bkm, aj, bj, aa, bb;
	float Rtmp[23], Itmp[23];
	double Cos[23], Sin[23];
	int Perm[209];
	int factor [11];
	double s60, c72, s72, pi2;
		4
		4
		4
		4
		4
		4
		76
		72
		56
		184
		368
		836
		44
		32
		= 1692

	Sum	**4900**

Table 14: Decoder call stack

Function	Local variables	Stack used [bytes]
main()	int endOfFile;	4
	char frameOk;	1
	int i;	4
	int written16;	4
	char channelMode;	1
	struct *hBitBuf;	4
	struct *aacDecoderInfo;	4
	struct *streamSBR;	4
	struct *sbrDecoderInfo;	4
	struct * splineResampler;	4
	int frameSize;	4
	int sampleRate, outputSampleRate;	8
	int numChannels;	4
	int numOutSamples;	4
	int bDownSample;	4
	int fosr16, fosr8;	8
	int bBitstreamDownMix;	4
	int bValidMode;	4
		= 74
applySBR()	struct *self;	4
	struct *Bitstr;	4
	float *timeData;	4
	int *numChannels;	4
	int SbrFrameOK;	4
	int bDownSample;	4
	int bBitstreamDownMix;	4
	unsigned char i, dualMono;	2
	int stereo, CRCLen, crcEnable, readHeader, err;	20
	struct *SbrChannel;	4
	struct bitBuf;	16
	struct *hHeaderData;	4
	enum headerStatus;	4
	int codecFrameSize;	4
	enum initialSyncState;	4
	struct *hConcealData;	4
	float *pWorkBuffer1;	4
	struct *hFrameDataLeft;	4
	struct *hFrameDataRight;	4
		= 102
sbr_dec()	struct *hSbrDec;	4
	float *timeIn;	4
	float *timeOut;	4
	float *interimResult;	4
	struct *hHeaderData;	4
	struct *hFrameData;	4
	struct *hPrevFrameData;	4
	int applyProcessing;	4
	struct *h_ps_d;	4
	struct *hSynthesisQmfBankRight;	4
	int nChannels;	4
	int i, k, slot, ov_len, bUseLP;	20
	float *QmfBufferReal[38];	152
	float *QmfBufferImag[38];	152
	float *ptr;	4
	int noCols, halflen, islots;	12
		= 384

cplxSynthesisQmfFiltering()	float **qmfReal;
	float **qmfImag;
	float *timeout;
	struct *synQmf;
	int bUseLP;
	struct *h_ps_dec;
	int active;
	int i, j;
	float *ptr_time_out, *filterStates;
	float accu;
	int p;
	float qmfReal2[64];
	float *imagSlot;
	int no_synthesis_channels;
	int qmf_filter_state_syn_size;
	float mfRealTmp[64];
	float qmfImagTmp[64];
	int env;
	const float *p_filter;
		4
		4
		4
		4
		4
		4
		4
		8
		8
		4
		4
		256
		4
		4
		4
		256
		256
		4
		4
		= 840

ApplyPsSlot()	struct *h_ps_dec;
	float **rIntBufferLeft;
	float **iIntBufferLeft;
	float *rIntBufferRight;
	float *iIntBufferRight;
		4
		4
		4
		4
		4
		= 20

HybridAnalysis()	const float **mQmfReal;
	const float **mQmfImag;
	float **mHybridReal;
	float **mHybridImag;
	struct *hHybrid;
	int n, band;
	enum hybridRes;
	int chOffset;
		4
		4
		4
		4
		4
		8
		4
		4
		= 36

eightChannelFiltering()	const float *pQmfReal;
	const float *pQmfImag;
	float **mHybridReal;
	float **mHybridImag;
	int i, n;
	float real, imag;
	int midTap;
	float cum[16];
		4
		4
		4
		4
		8
		8
		4
		64
		= 100

CFFTN()	float *afftData;
	int len;
	int isign;
		4
		4
		4
		= 12

cfftn()	float Re[];
	float Im[];
	int nTotal;
	int nPass;
	int nSpan;
	int iSign;
	int ii, mfactor, kspan, ispan, inc, j, jc, jf, jj, k, k1, k2, k3, k4, kk, kt, nn, ns, nt;
	double radf, c1, c2, c3, cd, s1, s2, s3, sd;
	float ak, bk, akp, bkp, ajp, bjp, ajm, bjm, akm, bkm, aj, bj, aa, bb;
	float Rtmp[23], Itmp[23];
	double Cos[23], Sin[23];
	int Perm[209];
	int factor [11];
	double s60, c72, s72, pi2;
		4
		4
		4
		4
		4
		4
		76
		72
		56
		184
		368
		836
		44
		32
		= 1692

	Sum	**3260**

4.4 Weighted MOPS and PROM

The complexity numbers for the Enhanced aacPlus audio codec can be found in the following table, the numbers have been derived using the "allcat.wav" item, which holds all the material from the selection test concatenated in one single item. For every test case the average and worst frame weighted MOPS figure has been derived. The worst case wMOPS figure over all test cases has been marked in **blue**.

Table 15: Weighted MOPS and PROM figures

	Test Case	Mono Encoder	Stereo Encoder	Decoder	Decoder, mono only
wMOPS [average / worst frame]	14m	15.23 / 16.98	15.36 / 17.21	9.38 / 10.07	8.07 / 8.78
	18s	---	25.79 / 28.36	19.48 / 20.35	8.31 / 9.17
	24m	**16.72 / 18.93**	16.86 / 19.14	10.30 / 11.39	8.89 / 9.94
	24s	---	27.01 / 29.85	20.45 / 21.63	8.82 / 9.93
	32s	---	27.49 / 29.97	**21.08 / 22.42**	9.28 / 10.58
	48s	---	**35.22 / 42.22**	17.96 / 20.26	**12.42 / 14.32**
	14m, 16 kHz	15.42 / 18.41	15.47 / 18.46	7.85 / 8.61	7.85 / 8.60
	14m, 3% FER	---	---	9.38 / 10.07	8.07 / 8.78
	24s, 3% FER	---	---	20.45 / 21.63	8.81 / 9.93
	32s, 1%FER	---	---	**21.08 / 22.42**	9.28 / 10.58
	32s, 3%FER	---	---	21.08 / 22.38	9.27 / 10.58
Program ROM [ops]	**---**	**12540**	**14365**	**8048**	**6209**

5 File formats

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

5.1 Audio input file (encoder input/decoder output)

The audio input files read by the encoder and written by the decoder are 16-bit PCM wave files. For convenient handling of wave files a precompiled audio-fileformat library is used.

5.2 Bitstream file format (encoder output/decoder input)

The encoder program writes and the decoder program reads raw frames packetized in access units as described by 3GPP TS 26.244. For packetization the ISO media library is used. A precompiled library is used.

5.3 Error pattern file (decoder input)

The decoder program can optionally process an additional input file which describes an error pattern. The format of the error pattern file is 1 character per line. Each line corresponds to one frame, where a "0" indicates that the respective frame has been transmitted without errors, while a "1" indicates that the corresponding frame has been lost and error concealment shall be applied by the decoder.

Annex A (informative): Change history

Change history							
Date	**TSG SA#**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
2004-09	25	SP-040638			Approved at SA#25	2.0.0	6.0.0
2004-12	26	SP-040840	001		Correction to C-code to increase error robustness	6.0.0	6.1.0
2004-12	26	SP-040840	002		Correction to C-code: Missing memory re-initialization	6.0.0	6.1.0
2004-12	26	SP-040840	003		Correction to C-code: Memory initialization added	6.0.0	6.1.0
2004-12	26	SP-040840	004		Correction to C-code: Wrong calculation of sine levels	6.0.0	6.1.0
2004-12	26	SP-040840	005		Correction to C-code: Prevent multiple reading of bitstream elements	6.0.0	6.1.0
2004-12	26	SP-040840	006	2	Correction to C-code: Corrected wrong table values	6.0.0	6.1.0
2004-12	26	SP-040840	007		Correction to C-code: Modify instrumentation	6.0.0	6.1.0
2004-12	26	SP-040840	008	1	Correction of C-code: Output data was copied into wrong array	6.0.0	6.1.0
2004-12	26	SP-040840	009	1	Correction to C-code: Bug in resampler	6.0.0	6.1.0
2004-12	26	SP-040840	010	1	Correction to C-code: Modify data types for FFT	6.0.0	6.1.0
2004-12	26	SP-040840	011	1	Correction to decoder C-Code: Alignment with MPEG specification	6.0.0	6.1.0
2004-12	26	SP-040840	012		Correction to C-code: Reset of Missing Harmonics flags during concealment added	6.0.0	6.1.0
2004-12	26	SP-040840	013		Removal of Complexity counters	6.0.0	6.1.0
2005-01					File "env_calc.c" replaced in the attached ANSI-C code	6.1.0	6.1.1

History

Document history		
V6.1.1	January 2005	Publication