ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р 53556.8— 2013

Звуковое вещание цифровое

КОДИРОВАНИЕ СИГНАЛОВ ЗВУКОВОГО ВЕЩАНИЯ С СОКРАЩЕНИЕМ ИЗБЫТОЧНОСТИ ДЛЯ ПЕРЕДАЧИ ПО ЦИФРОВЫМ КАНАЛАМ СВЯЗИ. ЧАСТЬ III (MPEG-4 AUDIO)

Параметрическое кодирование высококачественных звуковых сигналов (SSC)

(ISO/IEC 14496-3:2009, NEQ)

Издание официальное

Москва
Стандартинформ
2020

gost.e-ecolog.ru

Предисловие

1 РАЗРАБОТАН Санкт-Петербургским филиалом Центрального научно-исследовательского института связи «Ленинградское отделение» (ФГУП ЛО ЦНИИС)

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 480 «Связь»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 25 октября 2013 г. № 1212-ст

4 Настоящий стандарт разработан с учетом основных нормативных положений международного стандарта ИСО/МЭК 14496-3:2009 «Информационные технологии. Кодирование аудиовизуальных объектов. Часть 3. Аудио» (ISO/IEC 14496-3:2009 «Information technology — Coding of audio-visual objects — Part 3: Audio») [1]

5 ВВЕДЕН ВПЕРВЫЕ

6 ПЕРЕИЗДАНИЕ. Август 2020 г.

*Правила применения настоящего стандарта установлены в статье 26 Федерального закона от 29 июня 2015 г. № 162-ФЗ «О стандартизации в Российской Федерации». Информация об изменениях к настоящему стандарту публикуется в ежегодном (по состоянию на 1 января текущего года) информационном указателе «Национальные стандарты», а официальный текст изменений и поправок — в ежемесячном информационном указателе «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ближайшем выпуске ежемесячного информационного указателя «Национальные стандарты». Соответствующая информация, уведомление и тексты размещаются также в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет (*gost.e-ecolog.ru

© Стандартинформ, оформление, 2014, 2020

Настоящий стандарт не может быть полностью или частично воспроизведен, тиражирован и распространен в качестве официального издания без разрешения Федерального агентства по техническому регулированию и метрологии

Содержание

1 Область применения 1
2 Термины и определения 2
3 Символы и сокращения 2
3.1 Арифметические операторы 2
3.2 Операторы отношения 2
3.3 Мнемоники 2
3.4 Диапазоны 3
3.5 Численное представление 3
3.6 Определения 3
4 Полезные нагрузки для *SSC* аудио объектного типа 3
4.1 Конфигурация декодера (*SSCSpecificConfig*) 3
4.2 Полезная нагрузка потока битов *SSC* 3
5 Семантика 13
5.1 *SSCSpecificConfig* 13
5.2 Декодирование полезной нагрузки потока битов *SSC* 14
5.3 Индексация подфреймов 24
6 Процесс декодирования 24
6.1 Переходные процессы 25
6.2 Синусоиды 26
6.3 Шум 30
6.4 Параметрическое стерео 33
6.5 Ситуации запуска/остановки для декодирования 46
Приложение А (обязательное) Комбинация инструмента *SBR* с параметрическим инструментом стерео 47
Приложение Б (обязательное) Нормативные таблицы 49
Приложение В (справочное) Темп и подача, масштабируемые в декодере 77
Библиография 78

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Звуковое вещание цифровое

КОДИРОВАНИЕ СИГНАЛОВ ЗВУКОВОГО ВЕЩАНИЯ С СОКРАЩЕНИЕМ ИЗБЫТОЧНОСТИ ДЛЯ ПЕРЕДАЧИ ПО ЦИФРОВЫМ КАНАЛАМ СВЯЗИ. ЧАСТЬ III (MPEG-4 AUDIO)

Параметрическое кодирование высококачественных звуковых сигналов (SSC)

Sound broadcasting digital. Coding of signals of sound broadcasting with reduction of redundancy for transfer on digital communication channels. A part III (MPEG-4 audio). Technical description of parametric coding for high quality audio

Дата введения — 2014—09—01

1 Область применения

Стандарт описывает схему параметрического кодирования для сжатия высококачественного аудио *MPEG-4 audio*. Краткое наименование — *SSC* (*SinuSoidal Coding*). При скоростях передачи приблизительно 24 Кбит/с стерео и на частоте дискретизации 44,1 кГц схема кодирования *SSC* предлагает качество, которое необходимо для многих приложений.

SSC использует четыре различных инструмента, которые вместе параметризуют аудиосигнал. Эти инструменты состоят из моделирования переходного процесса, моделирования синусоид, моделирования шума и моделирования образа стерео. Одна из отличительных особенностей *SSC* состоит в том, что оно оказывает поддержку декодера для независимого темпа и масштабирования шага при почти любой дополнительной сложности.

Инструмент переходного процесса

Инструмент переходного процесса захватывает чрезвычайно динамичные события входного аудиосигнала. Эти события эффективно моделируются посредством ограниченного количества синусоид, которые формируются посредством огибающей.

Инструмент синусоид речевого выхода для инструментов *FA* и для дублирования *MP* с информацией о форме губ.

Инструмент синусоид захватывает детерминированные события входного аудиосигнала. Природа медленного изменения синусоидальных компонентов для типичных аудиосигналов используется, соединяя синусоиды по последовательным фреймам. Посредством дифференциального кодирования могут быть эффективно представлены параметры частоты, амплитуды и фазы.

Шумовой инструмент

Шумовой инструмент захватывает стохастические или недетерминированные события входного аудиосигнала. В декодере в качестве возбуждения используется генератор белого шума. Чтобы управлять временны́ми и спектральными свойствами шума в аудиосигнале применяется временна́я и спектральная огибающая.

Инструмент параметрического кодирования стерео

Инструмент параметрического кодирования стерео в состоянии захватывать стереообраз входного аудиосигнала в ограниченном количестве параметров, требуя только небольшого служебного сигнала в пределах от нескольких Кбит/с для среднего качества до приблизительно 9 Кбит/с для более высокого качества. Вместе с монофоническим смешиванием входного сигнала стерео, сгенерированного параметрическим инструментом кодирования стерео, инструмент параметрического декодирования стерео в состоянии регенерировать стереосигнал. Это — универсальный инструмент, который может работать в комбинации с любым монофоническим кодером. В приложении А настоящего стандарта при-

Издание официальное

водится нормативное описание комбинации *HE—AAC* с инструментом параметрического кодирования стерео. *SSC* также может работать в двойном режиме моно. В этом случае инструмент параметрического кодирования стерео не используется. Инструмент параметрического стерео предназначается для низких скоростей передачи.

2 Термины и определения

2.1 **фрейм:** Основная единица, которая может декодироваться сама по себе (требуется информация заголовка файла для общих настроек декодера).

2.2 **фильтр Лагерра:** Структура фильтра, используемого в анализе шумов и синтезе.

2.3 **аудио фрейм:** Содержит все данные, чтобы декодировать *SSC*-кодированный фрейм как автономный модуль (требуется информация заголовка файла для общих настроек декодера). Для аудио фреймов с *refresh_sinusoids* = = %1 и *refresh_noise* = = %1 полный фрейм всегда может быть восстановлен, иначе в случае произвольного доступа возможно, что части сигнала не могут быть восстановлены (например, синусоидальные продолжения, шум).

2.4 **подфрейм:** Мелкоструктурированность в пределах фрейма.

2.5 f_s**:** Частота дискретизации в герцах.

2.6 **сегмент:** Интервал выборок, которые могут быть синтезированы на основе параметров, которые соответствуют подфрейму. Размер сегмента 2**S*.

2.7 **окно:** Функция, которая используется, чтобы взвешивать синтезируемые выборки в пределах сегмента так, чтобы получить достоверный синтез.

2.8 ***LSF*:** Спектральная частота линии.

2.9 **наложение и дробление:** Аддитивный метод объединения перекрывающихся интервалов во время синтеза сигнала.

2.10 **процесс связывания:** Метод отслеживания синусоидальных компонент в течение продолжительного времени.

2.11 **рождение:** Первый компонент синусоидальной дорожки.

2.12 **продолжение:** Компонент синусоидальной дорожки, который находится не в начале или конце дорожки.

2.13 **смерть:** Последний компонент синусоидальной дорожки.

2.14 ***SMR*:** Отношение сигнала к маскировке.

2.15 **частичный:** Синусоида ограниченной продолжительности.

2.16 ***IID*:** Межканальные различия в интенсивности.

2.17 ***IPD*:** Межканальная разность фаз.

2.18 ***OPD*:** Полная Разность фаз.

2.19 ***ICC*:** Межканальная когерентность.

3 Символы и сокращения

3.1 Арифметические операторы

$\lfloor x \rfloor$ Округление x к минус бесконечность

$\lceil x \rceil$ Округление x к плюс бесконечность.

mod Оператор модуля: $\mathrm{mod}(x,y) \triangleq x - \left\lfloor \frac{x}{y} \right\rfloor y$. Определенный только для положительных значений *x* и *y*.

$\Gamma(\alpha)$ Гамма функция распределения, определенная как $\Gamma(\alpha) \triangleq \int_0^{\infty} e^{-t} \times t^{\alpha-1} dt$.

3.2 Операторы отношения

x? y:z Если *x* является истиной, тогда *y*, иначе *z*.

3.3 Мнемоники

Чтобы описать различные типы данных, используемые в кодированном потоке битов, определяются следующие мнемоники.

uimsbf Целое число без знака, старший значащий бит в начале.
simsbf Целое число со знаком, старший значащий бит в начале.
bslbf Левый бит потока битов в начале

3.4 Диапазоны

[0, 10] Число в диапазоне от 0 до, и включая 10.
[0, 10> Число в диапазоне от 0 до, но исключая 10.

3.5 Численное представление

%*X* Представление двоичного числа (например, %01111100).
$*X* Представление шестнадцатеричного числа (например, 7$C).
X Числа без префикса используют представление десятичного числа (например, 124).

3.6 Определения

S Число выборок в подфрейме.
L Число выборок в сегменте; *L* = 2**S*.
numQMFSlots Число выборок поддиапазона *QMF* на элемент *ps_data* (). Для *SSC* этот параметр фиксируется в 24.

4 Полезные нагрузки для *SSC* аудио объектного типа

4.1 Конфигурация декодера (*SSCSpecificConfig*)

Синтаксис *SSCSpecificConfig* (), см. таблицу 1.

Таблица 1 — Синтаксис *SSCSpecificConfig* ()

Синтаксис	Количество битов	Мнемоника
SSCSpecificConfig (channelConfiguration)		
{		
decoder_level	2	*uimsbf*
update rate	4	*uimsbf*
synthesis_method	2	*uimsbf*
if (*channelConfiguration != 1*)		
{		
mode_ext	2	*uimsbf*
if ((*channelConfiguration* == 2) && (*mode_ext* == 1))		
{		
reserved	2	*uimsbf*
}		
}		
}		

4.2 Полезная нагрузка потока битов *SSC*

Полезная нагрузка потока битов *SSC* приведена в таблицах 2—14.

Таблица 2 — Синтаксис *ssc_audio_frame* ()

Синтаксис	Количество битов	Мнемоника
ssc_audio_frame ()		
{		
ssc_audio_frame_header()		

Окончание таблицы 2

Синтаксис	Количество битов	Мнемоника
ssc_audio_frame_data() }		

Таблица 3 — Синтаксис *ssc_audio_frame_header* ()

Синтаксис	Количество битов	Мнемоника

ssc_audio_frame_header ()
{
refresh_sinusoids
refresh_sinusoids_next_frame
refresh_noise
for (*ch* = 0; *ch* < *nrof_channels*; *ch*++)
{
s_nrof_continuations[0][*ch*]
}
n_nrof_den
n_nrof_lsf
freq_granularity
amp granularity
phasejitter_present
if (*phase_jitter_present* = = 1)
{
phasejitter_percentage
phasejitter_band
}
}
	
	1
	1
	1
	
	Примечание
	
	5
	Примечание
	2
	2
	1
	
	2
	2
	
		uimsbf
		uimsbf
		uimsbf
		
		uimsbf
		
		uimsbf
		uimsbf
		uimsbf
		uimsbf
		uimsbf
		
		uimsbf
		uimsbf

Примечание — См. описание *s_nrof_continuations* и *n_nrof_lsf* в 5.2.		

Таблица 4 — Синтаксис *ssc_audio_frame_data* ()

Синтаксис	Количество битов	Мнемоника

ssc_audio_frame_data()
{
for (*sf* = **0**; *sf* < *nrof_subframes*; *sf*++)
{
for (*ch* = 0; *ch* < *nrof_channels*; *ch*++)
{
ssc_mono_subframe(*sf*,*ch*)
if ((*channelConfiguration* = = 2) && (*mode_ext* = = 1)
&& (*mod*(*sf*+1,4)= =0))
{
ps_data()
}
}
}
}		

Таблица 5 — Синтаксис *ssc_mono_subframe* ()

Синтаксис	Количество битов	Мнемоника

ssc_mono_subframe (*sf,ch*)
{
subframe_transients(*sf, ch*)
subframe_sinusoids(*sf, ch*)
subframe_noise(*sf, ch*)
}		

Таблица 6 — Синтаксис *subframe_transients* ()

Синтаксис	Количество битов	Мнемоника
subframe_transients (*sf, ch*) {		
t_transient_present[*sf*][*ch*]	1	*uimsbf*
if (*t_transient_present*[*sf*][*ch*] == 1)		
{		
t_loc[*sf*][*ch*]	Примечание	*uimsbf*
if (*t_type*[*sf*][*ch*]==1)	2	*uimsbf*
{		
t_b_par[*sf*][*ch*]	3	*uimsbf*
t_chi_par[*sf*][*ch*]	3	*uimsbf*
t_nrof_sin[*sf*][*ch*]	3	*uimsbf*
t_nrof_sin[*sf*][*ch*]++		
for (*i* = 0; *i* < *t_nrof_sin*[*sf*][*ch*]; *i*++)		
{		
t_freq[*sf*][*ch*][*i*]	9	*uimsbf*
t_amp[*sf*][*ch*][*i*]	5	*uimsbf*
t_phi[*sf*][*ch*][*i*]	5	*simsbf*
}		
}		
}		
}		
Примечание — См. описание *t_loc* в 5.2.		

Таблица 7 — Синтаксис *subframe_sinusoids*()

Синтаксис	Количество битов	Мнемоника
subframe_sinusoids(*sf, ch*) {		
n = 0; *p*=0; *q*=0;		
/* *Continuations* */		
if (*sf* > 0)		
{		
noc = 0;		

Продолжение таблицы 7

Синтаксис	Количество битов	Мнемоника
`while (tmp_cont[ch][noc] > 0) { noc++;}`		
`s_nrof_continuations[sf][ch] = noc;`		
`}`		
`if ((refresh_sinusoids == 1) && (sf == 0))`		
`{`		
`for (i = 0; i < s_nrof_continuations[sf][ch]; i++, n++)`		
`{`		
`s_cont[sf][ch][n] = ssc_huff_dec(huff_scont,bs_codeword);`	2..5	*bslbf*
`s_freq_coarse[sf][ch][n] = ssc_huff_dec(huff_sfreqc,bs_codeword);`	7..25	*bslbf*
`s_freq_fine[sf][ch][n]`	0..3	*simsbf*
`s_amp_coarse[sf][ch][n] = ssc_huff_dec(huff_sampca,bs_codeword);`	3..16	*bslbf*
`s_amp_fine[sf][ch][n]`	0..3	*simsbf*
`s_phi[sf][ch][n]`	5	*simsbf*
`if (s_cont[sf][ch][n] > 0)`		
`{`		
`s_adpcm_grid[sf][ch][n] = ssc_huff_dec(huff_sgrid,bs_codeword);`	3..7	*bslbf*
`s_delta_cont_freq_pha[sf+1][ch][p]`	2	*uimsbf*
`p++;`		
`}`		
`if (s_cont[sf][ch][n] > 1)`		
`{`		
`s_delta_cont_freq_pha[sf+2][ch][q]`	2	*uimsbf*
`q++;`		
`}`		
`}`		
`}`		
`else {`		
`for (i = 0; i < s_nrof_continuations[sf][ch]; i++, n++)`		
`{`		
`if (sf == 0)`		
`{`		
`s_cont[sf][ch][n] = ssc_huff_dec(huff_scont,bs_codeword);`	2..5	*bslbf*
`}`		
`else {`		
`s_cont[sf][ch][n] = tmp_cont[ch][n] - 1;`		
`}`		
`if (s_cont[sf][ch][n] > 0)`		
`}`		
`p++;`		
`}`		
`if (s_cont[sf][ch][n] > 1)`		
`{`		
`if ((refresh_sinusoids_next_frame == 0) || (nrof_subframes-sf > 2))`		

Продолжение таблицы 7

Синтаксис	Количество битов	Мнемоника
{		
s_delta_cont_freq_pha[*sf*+2][ch][*q*]	2	*uimsbf*
}		
q++;		
}		
s_delta_cont_amp[sf][ch][n] =	1..15	*bslbf*
ssc_huff_dec(*huff_sampcr*[*amp_granularity*],*bs_codeword*);		
}		
}		
/* Births */		
s_nrof_births[*sf*][*ch*] = *ssc_huff_dec*(*huff_nrofbirths*,*bs_codeword*);	3..15	*bslbf*
if (*s_nrof_births*[*sf*][*ch*] > 0)		
{		
s_cont[*sf*][*ch*][*n*] = *ssc_huff_dec*(*huff_scont*,*bs_codeword*);	2..5	*bslbf*
s_freq_coarse[*sf*][*ch*][*n*] = *ssc_huff_dec*(*huff_sfreqba*,*bs_codeword*);	7..21	*bslbf*
s_freq_fine[*sf*][*ch*][*n*]	0..3	*simsbf*
s_amp_coarse[*sf*][*ch*][*n*] = *ssc_huff_dec*(*huff_sampba*,*bs_codeword*);	3..15	*bslbf*
s_amp_fine[*sf*][*ch*][*n*]	0..3	*simsbf*
s_phi[*sf*][*ch*][*n*]	5	*simsbf*
if (*s_cont*[*sf*][*ch*][*n*] > 0)		
{		
if ((*refresh_sinusoids_next_frame* == 0) || (*nrof_subframes-sf* > 1))		
{		
s_delta_cont_freq_pha[*sf*+1][*ch*][*p*]	2	*uimsbf*
}		
p++;		
}		
if (*s_cont*[*sf*][*ch*][*n*] > 1)		
}		
if ((*refresh_sinusoids_next_frame* == 0) || (*nrof_subframes-sf* > 2))		
{		
s_delta_cont_freq_pha[*sf*+2][*ch*][*q*]	2	*uimsbf*
}		
q++;		
}		
n++;		
for (*i* = 1; *i* < *s_nrof_births*[*sf*][*ch*]; *i*++, *n*++)		
{		
s_cont[*sf*][*ch*][*n*] = *ssc_huff_dec*(*huff_scont*,*bs_codeword*);	2..5	*bslbf*
s_delta_birth_freq_coarse[*sf*][*ch*][*n*] =	5..23	*bslbf*
ssc_huff_dec(*huff_sfreqbr*,*bs_codeword*);		
s_delta_birth_freq_fine[*sf*][*ch*][*n*]	0..3	*simsbf*
s_delta_birth_amp_coarse[*sf*][*ch*][*n*] =	2..21	*bslbf*

Окончание таблицы 7

Синтаксис	Количество битов	Мнемоника
ssc_huff_dec(huff_sampbr,bs_codeword);		
s_delta_birth_amp_fine[sf][ch][n]	0..3	*simsbf*
s_phi[sf][ch][n]	5	*simsbf*
if (s_cont[sf][ch][n] > 0)		
{		
if ((refresh_sinusoids_next_frame == 0) || (nrof_subframes-sf > 1))		
{		
s_delta_cont_freq_pha[sf+1][ch][p]	2	*uimsbf*
}		
p++;		
}		
if (s_cont[sf][ch][n] > 1)		
{		
if ((refresh_sinusoids_next_frame == 0) || (nrof_subframes-sf > 2))		
{		
s_delta_cont_freq_pha[sf+2][ch][q]	2	*uimsbf*
}		
q++;		
}		
}		
}		
/* Keep track of sinusoids that continue in next sub-frame(s) */		
for (i = 0, k = 0; i < n; i++)		
{		
if (s_cont[sf][ch][i] > 0)		
{		
tmp_cont[ch][k] = s_cont[sf][ch][i];		
k++;		
}		
}		
}		

Примечание — Переменные *p*, *q* используются в качестве индексов позиции для *subframe*+1 и *subframe*+2, соответственно.

Таблица 8 — Синтаксис *subframe_noise*()

Синтаксис	Количество битов	Мнемоника
subframe_noise (sf, ch)		
{		
if ((refresh_noise == 1) && (sf == 0))		
{		
n_laguerre[ch]	2	*uimsbf*
n_laguerre_granularity[sf][ch]	1	*uimsbf*

Продолжение таблицы 8

Синтаксис	Количество битов	Мнемоника
for (i = 0; i < n_nrof_den; I++)		
{		
n_lar_den_coarse[sf][ch][i] =	1..18	bslbf
ssc_huff_dec(huff_nlag,bs_codeword);		
If (n_laguerre_granularity[sf][ch]==1)		
{		
n_lar_den_fine[sf][ch][i]	2	simsbf
}		
}		
n_gain[sf][ch]	7	uimsbf
n_lsf[sf][ch][0] = ssc_huff_dec(huff_nlsf,bs_codeword);	2..9	bslbf
for (i = 1; i < n_nrof_lsf; i++)		
{		
n_delta_lsf[sf][ch][i] = ssc_huff_dec(huff_nlsf,bs_codeword);	2..9	bslbf
}		
}		
else {		
if (mod(sf,2) = = 0)		
{		
n_laguerre_granularity[sf][ch]	1	uimsbf
for (i = 0; i < n_nrof_den; i++)		
{		
n_delta_lar_den_coarse[sf][ch][i] =	1..18	bslbf
ssc_huff_dec(huff_nlag,bs_codeword);		
if(n_laguerre_granularity[sf][ch]==1)		
{		
n_delta_lar_den_fine[sf][ch][i]	2	simsbf
}		
}		
}		
if (mod(sf,4) == 0)		
{		
n_delta_gain[sf][ch] = ssc_huff_dec(huff_ngain,bs_codeword);	1..12	bslbf
if (n_overlap_lsf == 1)	1	uimsbf
{		
for (i = n_nrof_overlap_lsf; i < n_nrof_lsf; i++)		
{		
n_delta_lsf[sf][ch][i] =	2..9	bslbf
ssc_huff_dec(huff_nlsf,bs_codeword);		
}		
}		

Окончание таблицы 8

Синтаксис	Количество битов	Мнемоника
`else {`		
`n_lsf[sf][ch][0] = ssc_huff_dec(huff_nlsf,bs_codeword);`	2..9	*bslbf*
`for ( i = 1; i < n_nr_of_lsf; i++)`		
`{`		
`n_delta_lsf[sf][ch][i] = ssc_huff_dec(huff_nlsf,bs_codeword);`	2..9	*bslbf*
`}`		
`}`		
`}`		
`}`		
`}`		

Таблица 9 — Синтаксис *ps_data*()

Синтаксис	Количество битов	Мнемоника
ps_data()		
{		
if (*enable_ps_header*) {	1	*uimsbf*
if (*enable_iid*) {	1	*uimsbf*
iid_mode	3	*uimsbf*
nr_iid_par = *nr_iid_par_tab*[*iid_mode*]		
nr_ipdopd_par = *nr_ipdopd_par_tab*[*iid_mode*]		
}		
if (*enable_icc*) {	1	*uimsbf*
icc_mode	3	*uimsbf*
nr_icc_par = *nr_icc_par_tab*[*icc_mode*]		
}		
enable_ext	1	*uimsbf*
}		
frame_class	1	*uimsbf*
num_env_idx	2	*uimsbf*
num_env = *num_env_tab*[*frame_class*][*num_env_idx*]		
if (*frame_class*) {		
for (*e*=0 ; *e*<*num_env* ; *e*++) {		
border_position[*e*]	5	*uimsbf*
}		
}		
for (*e*=0 ; *e*<*num_env* ; *e*++) {		
if (*enable_iid*) {		
iid_dt[*e*]	1	*uimsbf*
iid_data()		
}		
}		
for (*e*=0 ; *e*<*num_env* ; *e*++) {		
if (*enable_icc*) {		
icc_dt[*e*]	1	*uimsbf*

Окончание таблицы 9

Синтаксис	Количество битов	Мнемоника
icc_data()		
}		
}		
if (*enable_ext*) {		
cnt = *ps_extension_size*	4	*uimsbf*
if (*cnt* == 15)		
cnt += *esc_count*	8	*uimsbf*
num_bits_left = 8 * *cnt*		
while (*num_bits_left* > 7) {		
ps_extension_id	2	*uimsbf*
num_bits_left = 2		
ps_extension(*ps_extension_id*, *num_bits_left*)		
}		
fill_bits	*num_bits_left*	
}		
}		

Таблица 10 — Синтаксис *ps_extension*()

Синтаксис	Количество битов	Мнемоника
ps_extension(*ps_extension_id*, *num_bits_left*)		
{		
if (*ps_extension_id* == 0) {		
if (*enable_ipdopd*) {	1	*uimsbf*
for (*e*=0 ; *e*<*num_env* ; *e*++) {		
ipd_dt[*e*]	1	*uimsbf*
ipd_data()		
opd_dt[*e*]	1	*uimsbf*
opd_data()		
num_bits_left -= *ipd_bits* + *opd_bits* + 2		
}		
}		
reserved_ps	1	*uimsbf*
num_bits_left -= 2		
}		
}		

Таблица 11 — Синтаксис *iid_data*()

Синтаксис	Количество битов	Мнемоника
iid_data()		
if (*iid_dt*[*e*]) {		
for (*b*=0 ; *b*<*nr_iid_par*; *b*++) {		

Окончание таблицы 11

Синтаксис	Количество битов	Мнемоника

iid_par_df[e][b] = ssc_huff_dec(huff_iid_df[iid_quant], bs_codeword);
}
}
else {
for (b=0 ; b<nr_iid_par; b++) {
iid_par_df[e][b] = ssc_huff_dec(huff_iid_df[iid_quant], bs_codeword);
}
}
}
	1…20
	
	1...18	

Таблица 12 — Синтаксис *icc_data*()

Синтаксис	Количество битов	Мнемоника

icc_data()
{
if (icc_df[e]) {
for (b=0 ; b<nr_icc_par; b++) {
icc_par_df[e][b] = ssc_huff_dec(huff_icc_dt,bs_codeword);
}
}
else {
for (b=0 ; b<nr_icc_par; b++) {
icc_par_df[e][b] = ssc_huff_dec(huff_icc_df,bs_codeword);
}
}
}
	
	1...14
	
	1...13
		
		bslbf
		
		bslbf

Таблица 13 — Синтаксис *ipd_data*()

Синтаксис	Количество битов	Мнемоника

ipd_data()
{
if (ipd_df[e]) {
for (b=0 ; b<nr_ipdopd_par; b++) {
ipd_par_df[e][b] = ssc_huff_dec(huff_ipd_dt,
bs_codeword);
}
}
else {
for (b=0 ; b<nr_ipdopd_par; b++) {
ipd_par_df[e][b] = ssc_huff_dec(huff_ipd_df,bs_codeword);
}
}
}
	
	1...5
	
	1...4
		
		bslbf
		
		bslbf

Таблица 14 — Синтаксис *opd_data*()

Синтаксис	Количество битов	Мнемоника

opd_data()
{
if (*opd_df*[*e*]) {
for (*b*=0 ; *b*<*nr_ipdopd_par*; *b*++) {
opd_par_df[*e*][*b*] = *ssc_huff_dec*(*huff_opd_df*,*bs_codeword*);
}
}
else {
for (*b*=0 ; *b*<*nr_ipdopd_par*; *b*++) {
opd_par_df[*e*][*b*] = *ssc_huff_dec*(*huff_opd_df*,*bs_codeword*);
}
}
}
	
	1...5
	
	1...5
		
		bslbf
		
		bslbf

5 Семантика

5.1 SSCSpecificConfig

decoder_level — пределы сложности для настроек декодера. Декодер, который поддерживает определенный уровень сложности, не в состоянии декодировать поток битов, который кодируется согласно более высокому уровню сложности. Этот декодер в состоянии декодировать поток битов, который кодируется согласно более низкому уровню сложности (таблица 15).

Таблица 15 — Уровень декодера

decoder_level	Уровень сложности	*max_nrof_sinusoids*	*max_nrof_den*	Число битов для *s_nrof_continuation*	Число битов для *n_nrof_lsf*
00	Зарезервировано	*Na*	*Na*	*Na*	*Na*
01	Носитель	60	24	6	4
10	Зарезервировано	*Na*	*Na*	*Na*	*Na*
11	Зарезервировано	*Na*	*Na*	*Na*	*Na*

max_nrof_sinusoids — максимальное количество синусоид, которое допускается.
max_nrof_den — максимальное значение для *n_nrof_den*.
update_rate — четыре бита, указывающие размер подфрейма *S*.
Таблица 16 показывает соотношение между *update_rate* и размером подфрейма *S* в выборках.

Таблица 16 — Уровень обновления

update_rate	*S*	*update_rate*	*S*
0000	Зарезервировано	1000	Зарезервировано
0001	Зарезервировано	1001	Зарезервировано
0010	Зарезервировано	1010	Зарезервировано
0011	Зарезервировано	1011	Зарезервировано
0100	384	1100	Зарезервировано
0101	Зарезервировано	1101	Зарезервировано

Окончание таблицы 16

update_rate	*S*	*update_rate*	*S*
0110	Зарезервировано	1110	Зарезервировано
0111	Зарезервировано	1111	Зарезервировано

synthesis_method — два бита, предоставляющие информацию о предпочтительном синтезе для специально закодированной программы (таблица 17).

Таблица 17 — Метод синтеза

Synthesis_method	Оптимальный синтез
00	Наложение и добавление
01	Зарезервировано
10	Зарезервировано
11	Зарезервировано

mode_ext — в сочетании с *channelConfiguration* биты *mode_ext* обеспечивают полную конфигурацию канала. Число битов зависит от *channelConfiguration* (таблица 18).

Таблица 18 — Конфигурация канала

channelConfiguration	Количество битов для *mode_ext*	*nrof_channels*
1	0	1
2	2	Согласно *mode_ext*
0, 3 ... 15	*Na*	*Na*

Для *channelConfiguration* == 2, применяется таблица 19:

Таблица 19 — Конфигурация канала в случае, если *channelConfiguration* == 2

mode_ext	Полная конфигурация канала	*nrof_channels*
00	Двойной моно (*ch0=left, ch1=right*)	2
01	Параметрическое Стерео	1
10	Зарезервировано	*Na*
11	Зарезервировано	*Na*

Зарезервировано — два зарезервированных бита; должно быть установлено в %0.

5.2 Декодирование полезной нагрузки потока битов SSC

ssc_audio_frame () — синтаксический элемент, который содержит единственный фрейм *SSC*

ssc_audio_frame_header () — синтаксический элемент, который содержит данные заголовка для единственного фрейма *SSC*

ssc_audio_frame_data () — синтаксический элемент, который содержит данные для единственного фрейма *SSC*

ssc_huff_dec () — процедура декодирования Хаффмана.

refresh_sinusoids — один бит, указывающий, как кодируются синусоидальные продолжения первого подфрейма во фрейме. Если этот бит равняется %0, данные продолжающейся дорожки дифференцированно кодируются относительно последнего подфрейма предыдущего фрейма. Если этот бит равняется %1, данные продолжающейся дорожки в первом подфрейме фрейма кодируются как абсолютные значения.

refresh_sinusoids_next_frame — один бит, обеспечивающий дополнительный предпросмотр фрейма для декодирования *ADPCM* синусоидальных параметров. Если этот бит устанавливается в %1, следующий фрейм является фреймом обновления. В этом случае бит refresh_sinusoids должен быть установлен в %1 в следующем фрейме. Если этот бит устанавливается в %0, следующий фрейм не является фреймом обновления.

refresh_noise — один бит, указывающий, как кодируются шумовые параметры первого подфрейма во фрейме. Если этот бит равняется %0, шумовые параметры дифференцированно кодируются относительно последнего подфрейма предыдущего фрейма. Если этот бит равняется %1, шумовые параметры в первом подфрейме фрейма кодируются как абсолютные значения.

s_nrof_continuations [*sf*] [*ch*] — для подфрейма *sf* и канала *ch*, это значение представляет число продолжений. В случае *sf* == 0 значение *s_nrof_continuations* обеспечивается в потоке битов. Для остальных значений *sf* значение *s_nrof_continuations* получается неявно. Число битов, требуемых для *s_nrof_continuations* [*0*] [*ch*], зависит от максимального количества разрешенных синусоид, которое зависит от сложности декодера, указанной в *decoder_level*.

n_nrof_den — число коэффициентов *LAR* знаменателя фильтра *FIR* для генерации шума.

n_nrof_lsf — число коэффициентов *LSF*, используемых для генерации огибающей для генерации шума. Число битов, требующихся для *n_nrof_lsf*, зависит от сложности декодера, указанной *decoder_level*.

freq_granularity — гранулярность дифференцированно или абсолютно кодированных частотных параметров, используемых в *subframe_sinusoids* (). Этот параметр определяет число битов, которые будут считаны из тонкоструктурной части параметров частоты.

amp_granularity — гранулярность дифференцированно или абсолютно кодированных амплитудных параметров, используемых в *subframe_sinusoids* (). Этот параметр определяет таблицу Хаффмана, которая будет использоваться, или число битов, которые будут считаны из тонкоструктурной части амплитудных параметров.

phase_jitter_present — один бит, чтобы указать на присутствие параметров дрожания фазы. Если этот бит равняется %0, никакого дрожания фазы нет. Если этот бит равняется %1, дрожание фазы присутствует.

phase_jitter_percentage — это двухбитовое целое число без знака, указывающее процент расстояния. Полное расстояние равняется половине шага квантования. Максимальное дрожание, применяемое к частотным компонентам

$$\max_jitter = 2^{freq_granularity-1}\frac{phase_jitter_percentage+1}{2^2}.$$

phase_jitter_band — два бита, указывающие уровень представления частоты, из которого должно быть применено дрожание фазы, таблица 20.

Таблица 20 — Полоса дрожания фазы, выраженная в уровнях представления

phase_jitter_band	Уровень представления частоты $f_{jitter,min}$
00	0
01	800
10	1600
11	2400

nrof_subframes — число подфреймов в одном фрейме. Это значение фиксируется в 8.

ssc_mono_subframe () — синтаксический элемент, который содержит данные для одного подфрейма *SSC*.

ps_data () — синтаксический элемент, который содержит параметрические данные стерео.

subframe_transients () — синтаксический элемент, который содержит переходные данные для одного подфрейма *SSC*.

subframe_sinusoids () — синтаксический элемент, который содержит данные синусоид для одного подфрейма *SSC*.

subframe_noise () — синтаксический элемент, который содержит данные о шуме для одного подфрейма *SSC*.

t_transient_present [*sf*] [*ch*] — один бит, указывающий, присутствует ли переходное состояние в подфрейме *sf*, канала *ch*. Если *t_transient_present* [*sf*] [*ch*] == %1, переходное состояние присутствует. Если *t_transient_present* [sf] [ch] == %0, переходное состояние отсутствует.

t_loc [*sf*] [*ch*] — указание расположения переходного состояния в подфрейме *sf* канала *ch*, выраженное в числе выборок от начала подфрейма. Допустимый диапазон для *t_loc* равен [0, *S*>. Число битов, которое используется, чтобы представить *t_loc*, вычисляется согласно $\lceil \log_2 (S) \rceil$, где *S* представляет размер подфрейма в выборках.

t_type [*sf*] [*ch*] — два бита для указания типа переходного состояния в подфрейме *sf* канала *ch* (таблица 21).

Таблица 21 — Типы переходного состояния

t_type	Тип
00	Шаг
01	Мейкснера (*Meixner*)
10	Зарезервировано
11	Зарезервировано

t_b_par [*sf*] [*ch*] — для переходного состояния типа Мейкснера в подфрейме *sf* канала *ch*, эти 3 бита содержат значение для атаки огибающей перехода, обозначенное как '*b*-параметр'. Допустимые значения для *t_b_par* [0, 1, 2, 3]. Остальные значения резервируются. Значение *b* вычисляется как *b* = *t_b_par* + 2.

t_chi_par [*sf*] [*ch*] — для переходного состояния типа Мейкснера в подфрейме *sf* канала *ch*, эти 3 бита хранят значение для спада огибающей переходного состояния, обозначенное как 'ξ-параметр'. Допустимые значения для *t_chi_par* равны [0, 1, 2, 3]. Остающиеся значения резервируются. Значения ξ сведены в таблицу 22.

Таблица 22 — Квантованные значения для b и ξ

ξ		*t_b_par*			
		0	1	2	3
t_chi_par	0	0,9688	0,9685	0,9683	0,9681
	1	0,9763	0,9756	0,9750	0,9744
	2	0,9839	0,9827	0,9817	0,9807
	3	0,9914	0,9898	0,9884	0,9870

t_nrof_sin [*sf*] [*ch*] — для переходного состояния типа Мейкснера в подфрейме *sf* канала *ch*, эти 3 бита представляют число синусоид, которые присутствуют под огибающей. Число синусоид под огибающей Мейкснера равно значению в потоке плюс один.

t_freq [*sf*] [*ch*] [*i*] — для переходного состояния типа Мейкснера в подфрейме *sf* канала *ch* эти биты представляют частоту в радианах *i*-й синусоиды под огибающей переходного состояния.

$$tf[i] \approx \frac{2\pi}{f_s} \frac{10^{\frac{t_freq[sf][sf][ch][i]}{11,4 \cdot 21,4}} - 1}{0,00437},$$

где tf_q представляет деквантизованную абсолютную частоту в радианах.

t_amp [*sf*] [*ch*] [*i*] — для переходного состояния типа Мейкснера в подфрейме *sf* канала *ch* эти биты представляют амплитуду *i*-й синусоиды под огибающей перехода.

$$ta_q[i] \approx ta_b^{2\,t_amp[sf][ch][i]},$$

где ta_b представляет логарифмическую базу квантования, ta_b = 1885,1. ta_b представляет деквантованную абсолютную амплитуду.

t_phi [*sf*] [*ch*] [*i*] — для переходного состояния типа Мейкснера в подфрейме *sf* канала *ch* эти биты представляют фазу *i*-й синусоиды под огибающей переходного состояния. Декодированное значение преобразовывается в значение фазы в радианах в диапазоне [–π, π> и определяется для начала переходного состояния.

$$tp_q[i] = 2 \cdot tp_e \cdot t_phi[sf][nh][i],$$

где tp_e представляет абсолютную ошибку фазы $\left(tp_e = \frac{\pi}{32}\right)$, и tp_q представляет деквантизованную абсолютную фазу (в радианах). Допустимый диапазон для *t_phi* равен [–16, 15].

noc — локальная переменная, которая считает число продолжений в предыдущем подфрейме.

tmp_cont [*ch*] [*noc*] — локальный массив, который содержит копию *s_cont*-параметров предыдущего подфрейма, необходимую для того, чтобы правильно проанализировать поток (извлечь число продолжений и отслеживать сколько подфреймов должна продолжаться синусоидальная дорожка в текущем фрейме).

s_cont [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch*, эта величина указывает, сколько компонент подфреймов *n* будет продолжаться в текущем фрейме (если компонент будет продолжаться также в следующем фрейме к числу подкадров, которые он продолжается в текущем фрейме, должна быть добавлена единица. Если значение равно 0, это указывает на прекращение компонента *n* в подкадре *sf*, что называют смертью). Допустимый диапазон для *s_cont* составляет [0, 9].

s_freq_coarse [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch* это значение представляет грубый параметр частоты *n*-й синусоиды.

s_freq_fine [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch* это целое число со знаком представляет более высокий уровень детализации дополнительно к грубому параметру частоты. Число битов, которые будут считаны достигает (3 – *freq_granularity*). Уровень представления частоты f_{rl} является суммой грубой частоты, точной частоты, масштабируемой по сетке гранулярности.

$$f_{rl}[n] = s_freq_coarse\ [sf]\ [ch]\ [n] + s_freq_fine\ [sf]\ [ch]\ [n]\ \ 2^{freq_granularity}.$$

Дрожание фазы применяется только в комбинации с масштабированием шага и темпа. Если *phase_jitter_present* == %1 и $f_{rl} > f_{jitter,min}$ параметр дрожания фазы будет

$$f_{jitter}\lfloor \max_jitter \cdot (2x - 1) + 0{,}5 \rfloor,$$

где *x* содержит случайное число, равномерно распределенное между 0 и 1, сгенерированное для каждого параметра частоты в подфрейме, соответствуя вышеуказанным требованиям. Декодируемое значение преобразовывается в деквантованное абсолютное значение частоты f_q в радианах, используя следующее уравнение:

$$f_q[n] = \frac{2\pi}{f_s} \frac{10^{\frac{f_{rl}[n]}{91{,}2 \cdot 21{,}4}} - 1}{0{,}00437}$$

s_amp_coarse [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch* это значение представляет грубый амплитудный параметр *n*-й синусоиды.

s_amp_fine [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch* этот параметр представляет более высокий уровень детализации дополнительно к грубому амплитудному параметру. Число битов, которые будут считаны, насчитывает до (3 – *amp_granularity*). Уровень представления амплитуды sa_{rl} является суммой грубой амплитуды, точной амплитуды, масштабируемой по сетке гранулярности

$$sa_{rl}[n] = s_amp_coarse\ [sf]\ [ch]\ [n] + s_amp_fine\ [sf]\ [ch]\ [n] \cdot 2^{amp_granularity}.$$

Декодируемое значение преобразовывается в деквантованное линейное значение амплитуды sa_q в диапазоне [1, 2^{15}–1] в соответствии с

$$sa_q[n] = sa_b^{2\,sa_{rl}[n]},$$

где sa_b = 1,0218 является базой логарифмического квантования. Его значение соответствует максимальной ошибке 0,1875 дБ.

s_phi [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch* это представляет фазовый параметр *n*-й синусоиды. Это значение преобразовывается в значение фазы в радианах в диапазоне [–π, π > в соответствии с

$$sp_q[n] = 2 \cdot sp_e \cdot s_phi[sf]\ [ch]\ [n],$$

где sp_e представляет ошибку абсолютной фазы $\left(sp_e = \frac{\pi}{32}\right)$, а sp_q представляет деквантизованную абсолютную фазу (в радианах). Допустимый диапазон для *s_phi* равен [–16, 15].

s_adpcm_grid [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch* это значение представляет начальный индекс в таблице 35 как использующийся в декодере *ADPCM* для *n*-й синусоиды. Эта таблица используется, чтобы декодировать информацию о синусоидах.

s_delta_cont_freq_pha [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch* это значение представляет уровни представления для *n*-й синусоиды, которые служат вводом в декодер *ADPCM*. Чтобы компенсировать эту задержку декодера на 2 подфрейма, уровни представления передаются заранее на 2 подфрейма. В синтаксисе потока битов будущие уровни представления обозначаются индексами *sf* +1 и *sf* +2, указывающими уровни представления двух последующих подфреймов, соответственно. В случае, если *sf* +1 или *sf* +2 превышает *nrof_subframes*, уровень представления присваивается следующему фрейму. В этом случае новое число подфреймов в следующем фрейме будет (*sf* +1) – *nrof_subframes* или (*sf* +2) – *nrof_subframes* соответственно.

s_delta_cont_amp [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch* это представляет дифференциальный амплитудный параметр *n*-й синусоиды. Это значение преобразовывается в линейное амплитудное значение в диапазоне [1, 2^{15}–1] в соответствии с

$$sa_{rl}[n] = sa_{rl,psf} + s_delta_cont_amp\ [sf]\ [ch]\ [n]\ ,$$

где sa_{rl} представляет уровень представления амплитуды, а $sa_{rl,psf}$ представляет уровень представления амплитуды в предыдущем подфрейме. Для деквантизации sa_{rl} в sa_q смотрите *s_amp_fine*. В случае, если амплитудная гранулярность, *amp_granularity* текущего фрейма отличается от гранулярности предыдущего фрейма, перед применением дифференциально кодированных значений гранулярность значения предыдущего фрейма преобразовывается в гранулярность текущего фрейма согласно

$$sa_{rl,psf} = 2^{amp_granularity} \left\lfloor \frac{sa'_{rl,psf}}{2^{amp_granularity}} + 0{,}5 \right\rfloor,$$

где $sa'_{rl,psf}$ представляет уровень представления амплитуды предыдущего подфрейма, и *amp_granularity* представляет гранулярность текущего подфрейма.

s_nrof_births [*sf*] [*ch*] — для подфрейма *sf* и канала *ch* это значение представляет число рождений. Позволенный диапазон [0, *max_nrof_sinusoids-s_nrof_continuations* [*sf*] [*ch*]].

s_delta_birth_freq_coarse [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch* это значение представляет дифференциальный, грубый параметр частоты *n*-й синусоиды.

s_delta_birth_freq_fine [*sf*] [*ch*] [*n*]— для подфрейма *sf* и канала *ch* это представляет более высокий уровень детализации в сравнении с грубым дифференциальным параметром частоты. Число битов, которые будут считаны, равно (3 – *freq_granularity*). Уровень представления дельта-частоты df_{rl} будет

$$df_{rl} = s_delta_birth_freq_coarse\ [sf]\ [ch]\ [n] + s_delta_birth_freq_fine\ [sf]\ [ch]\ [n] \cdot 2^{freq_granularity}.$$

Декодированное значение *n*-й синусоиды преобразовывается в значение частоты в герцах, используя уровень представления частоты предыдущего рождения f_{rl} подкадра *sf* ((*n*–1)-ая синусоида)

$$f_{rl}[n] = f_{rl}[n–1] + df_{rl}[n],$$

где f_{rl} представляет уровень представления частоты. Модификация f_{rl} из-за дрожания фазы использует те же самые правила, как установленные под *s_freq_fine*. О деквантизации f_{rl} в f_q смотрите также *s_freq_fine*.

s_delta_birth_amp_coarse [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch* это представляет дифференциальный, грубый амплитудный параметр *n*-й синусоиды.

s_delta_birth_amp_fine [*sf*] [*ch*] [*n*] — для подфрейма *sf* и канала *ch* это представляет более высокий уровень детализации грубого амплитудного параметра. Число битов, которые будут считаны, составляет (3 – *amp_granularity*). Уровень дельта-представления амплитуды sda_{rl}

$sda_{rl}[n] = s_delta_birth_amp_coarse[sf][ch][n] + s_delta_birth_amp_fine[sf][ch][n] \cdot 2^{amp_granularity}$.

Декодированное значение для *n*-й синусоиды преобразовывается в линейное амплитудное значение, используя уровень представления амплитуды предыдущего рождения sa_{rl} ((*n*–1)-ая синусоида):

$$sa_{rl}[n] = sa_{rl}[n-1] + sd_{rl}[n],$$

где sa_{rl} представляет уровень представления амплитуды.

n_laguerre [*ch*] — λ коэффициент фильтра Лагерра для синтеза шума, см. таблицу 23.

Таблица 23 — Возможные значения для λ

n_laguerre	λ
00	0
01	0,5
10	0,7
11	Зарезервировано

n_laguerre_granularity [*sf*] [*ch*] — 1 бит, обозначающий точность квантования коэффициентов Лагерра.

n_lar_den_coarse [*sf*] [*ch*] [*i*] — для подфрейма *sf* и канала *ch* это представляет номер коэффициента *LAR* знаменателя *i*.

n_lar_den_fine [*sf*] [*ch*] [*i*] — для подфрейма *sf* и канала *ch* это представляет более высокий уровень детализации относительно грубого параметра коэффициента *LAR* знаменателя. Уровень представления $nlar_{rl}$ является суммой грубого *LAR* знаменателя и точного *LAR* знаменателя:

$$nlarrl[i] = n_lar_den_coarse[sf][ch][i] + n_lar_den_fine[sf][ch][i]$$

и преобразовывается в коэффициент *LAR* согласно:

$$nlar_q[i] = nlar_{rl}[i] * \Delta_{LAR}.$$

n_gain [*sf*] [*ch*] — для подфрейма *sf* и канала *ch* это значение представляет коэффициент усиления. Уровень представления усиления $ngain_{rl}$ получается как:

$$ngain_{rl} = n_gain[sf][ch]$$

n_lsf[*sf*][*ch*][*i*] — для подфрейма *sf* и канала *ch* это значение представляет номер коэффициента *LSF i*. Разрешенный диапазон для *n_lsf* составляет [0,255]. Параметры деквантизованного *LSF* $nlsf_q$ получают так:

$$nlsfq[i] = n_lsf[sf][ch][i] * \pi / 256 + \pi / 512.$$

n_delta_lsf [*sf*] [*ch*] [*i*] — для подфрейма *sf* и канала *ch* это значение представляет номер дифференциального коэффициента *LSF i*. Они получаются, используя следующий алгоритм

для $i > 0$:

$$nlsf_q[i] = (nlsf_q[i\text{-}1] + n_delta_lsf[sf][ch][i]) * \pi / 256 + \pi / 512.$$

n_delta_lar_den_coarse [*sf*] [*ch*] [*i*] — для подфрейма *sf* и канала *ch* это представляет номер дифференциального коэффициента *LAR* знаменателя *i*.

n_delta_lar_den_fine [*sf*] [*ch*] [*i*] — для подфрейма *sf* и канала *ch* это представляет более высокий уровень детализации для грубого параметра коэффициента *LAR* знаменателя. Уровень представления $ndlar_{rl}$ является суммой дифференциального грубого *LAR* знаменателя и дифференциального точного *LAR* знаменателя:

$$ndlar_{rl}[i] = n_delta_lar_den_coarse[sf][ch][i] + n_delta_lar_den_fine[sf][ch][i]$$

и преобразовывается в коэффициент *LAR* согласно.

$$nlar_{rl}[i] = nlar'_{rl,\ psf}[i] + ndlarrl[i],$$

где $nlar_{rl}$ [*i*] и $nlar_{rl,psf}$ [*i*] представляют уровень представления *LAR* текущего и предыдущего подфреймов, соответственно. В случае, когда *n_laguerre_granularity* изменяется от %1 до %0 проходя от подфрейма *sf*–1 к *sf*, значение $nlar_{rl,\ psf}$ [*i*] сначала преобразовывается в наиболее грубую сетку квантования согласно:

$$rl,\ psf\ [i] = 4 * \lfloor nlar'_{rl,\ psf}\ [i]\ /4 + 0{,}5 \rfloor,$$

где $nlar'_{rl,psf}$ [*i*] представляет уровень представления *LAR* предыдущего подфрейма.

n_delta_gain [*sf*] [*ch*] — для подфрейма *sf* и канала *ch* это значение представляет дифференцированный коэффициент усиления и преобразовывается в уровень представления $ngain_{rl}$ согласно:

$$ngain_{rl} = ngain_{rl,\ p4sf} + n_delta_gain\ [sf]\ [ch],$$

где $ngain_{rl,p4sf}$ представляет уровень представления усиления для подфрейма *sf*-4.

n_overlap_lsf — один бит, указывающий, перекрываются ли коэффициенты *LSF* из предыдущего определения в канале *ch*.

enable_ps_header — один бит, указывающий, присутствует ли информация заголовка *PS*. Если установлен в %1, данные заголовка *PS*, конфигурирующие декодер *PS*, передаются. Иначе, сохраняется последняя конфигурация.

enable_iid — один бит, обозначающий присутствие параметров *IID*. Если *enable_iid* устанавливается в %1, параметры межканального различия интенсивности (*IID*) будут отправлены с этого момента в потоке битов. Если *enable_iid*==%0, никакие параметры *IID* не будут отправлены с этого момента в потоке битов.

iid_mode — конфигурация параметров *IID* (число полос и сетка квантования, *iid_quant*) определяется *iid_mode*. Поддерживаются восемь различных конфигураций для параметров *IID* (таблица 24).

Таблица 24 — Конфигурации режима *IID*

iid_mode	*nr_iid_par_tab*	*nr_ipdopd_par_tab*	*iid_quant*	Диапазон индекса
0 (000)	10	5	0	-7 ... 7
1 (001)	20	11		-7 ... 7
2 (010)	34	17		-7 ... 7
3 (011)	10	5	1	-15 ... 15
4 (100)	20	11		-15 ... 15
5 (101)	34	17		-15 ... 15
6 (110)	Зарезервировано			
7 (111)	Зарезервировано			

Если данные *IID* не отправляются в потоке битов, все параметры *IID* сбрасываются в 0 (то есть *index*=0).

Сетки значения по умолчанию и тонкого квантования для *IID*, *iid_quant* = %0 и *iid_quant* = %1 назначаются, как предусмотрено в таблице 25 и таблице 26, соответственно.

Таблица 25 — Сетка квантования по умолчанию для *IID*

Индекс	–7	–6	–5	–4	–3	–2	–1	0
IID [дБ]	–25	–18	–14	–10	–7	–4	–2	0
Индекс	1	2	3	4	5	6	7	
IID [дБ]	2	4	7	10	14	18	25	

Таблица 26 — Сетка тонкого квантования для *IID*.

Индекс	–15	–14	–13	–12	–11	–10	–9	–8
IID [дБ]	–50	–45	–40	–35	–30	–25	–22	–19
Индекс	–7	–6	–5	–4	–3	–2	–1	0

Окончание таблицы 26

IID [дБ]	–16	–13	–10	–8	–6	–4	–2	0
Индекс	1	2	3	4	5	6	7	8
IID [дБ]	2	4	6	8	10	13	16	19
Индекс	9	10	11	12	13	14	15	
IID [дБ]	22	25	30	35	40	45	50	

Конфигурация параметров межканальной разности фаз (*IPD*) / полной разности фаз (*OPD*), строго связывается с конфигурацией *IID*. Это также иллюстрируется в таблице 24.

enable_icc — один бит, обозначающий присутствие параметров *ICC*. Если *enable_icc* будет установлен в %1, то параметры межканальной когерентности (*ICC*) будут отправлены с этого момента в потоке битов. Если *enable_icc* = = %0, никакие параметры *ICC* не будут отправлены с этого момента в потоке битов.

icc_mode — конфигурация параметров межканальной когерентности (число полос и сетка квантования) определяется *icc_mode*. Для параметров *ICC* поддерживаются восемь различных конфигураций (таблица 27).

Таблица 27 — Конфигурация режима *ICC*

icc_mode	*nr_icc_par_tab*	Диапазон индекса	Процедуры смешивания
0 (000)	10	0 ... 7	R_a
1 (001)	20	0 ... 7	
2 (010)	34	0 ... 7	
3 (011)	10	0 ... 7	R_b
4 (100)	20	0 ... 7	
5 (101)	34	0 ... 7	
6 (110)	Зарезервировано		
7 (111)	Зарезервировано		

Если никакие данные *ICC* не отправляются в потоке битов, все параметры *ICC* сбрасываются в 1 (то есть *index*=0). Сетка квантования по умолчанию для *ICC* приводится в таблице 28.

Таблица 28 — Сетка квантования для *ICC*

Индекс	0	1	2	3	4	5	6	7
ρ	1	0,937	0,84118	0,60092	0,36764	0	–0,589	–1

enable_ext — уровень расширения *PS* задействуется, используя бит *enable_ext*. Если он устанавливается в %1, параметры *IPD* и *OPD* отправляются. Если это отключается, то есть %0, уровень расширения пропускается, и *enable_ipdopd* устанавливается в %0.

frame_class — бит *frame_class* определяет, располагаются ли позиции параметров текущего фрейма равномерно на протяжении фрейма (*FIX_BORDERS: frame_class* == %0) или они определяются, используя позиции, описанные *border_position* (*VAR_BORDERS: frame_class* == %1).

num_env_idx — число (наборов) параметров (огибающих) на фрейм определяется, используя *num_env_idx*. В случае фиксированного интервала параметров (*frame_class* == %0) и переменного интервала параметров (*frame_class* = = %1) это соотношение показано в таблице 29.

num_env — локальная переменная, обозначающая число огибающих стерео (наборов параметров). *num_env* = = 0 сигнализирует о том, что никакие новые параметры стерео не передаются, и что последние параметры в предыдущем элементе *ps_data* () должны сохраняться неизменными и применяться к текущему элементу *ps_data* ().

Таблица 29 — Число наборов параметров *num_env* как функция *num_env_idx* в случае фиксированного и переменного интервала

num_env_idx	*num_env_tab [frame_class] [num_env_idx]*	
	frame_class == 0	*frame_class* == 1
0	0	1
1	1	2
2	2	3
3	4	4

border_position [*e*] — в случае расположения параметров с переменными интервалами позиции параметров определяются *border_position* [*e*]. Он содержит индекс выборки *QMF* n_e для набора параметров *e* текущего элемента *ps_data* ().

iid_dt [*e*] — этот флаг описывает для индекса огибающей *e*, кодируются ли параметры *IID* дифференцированно в течение времени (*iid_dt* == %1) или по частоте (*iid_dt* == %0). В случае, когда *iid_mode* текущего элемента *ps_data* () отличается от *iid_mode* предыдущего элемента *ps_data* (), *iid_dt* [0] должно иметь значение 0%, принуждая к дифференцированному кодированию частоты.

iid_data () — синтаксический элемент, содержащий данные *IID*.

icc_dt [*e*] — этот флаг описывает для индекса огибающей *e*, кодируются ли параметры *ICC* дифференцированно в течение времени (*icc_dt* == %1) или по частоте (*icc_dt* == %0). В случае, когда *icc_mode* текущего элемента *ps_data* () отличается от *icc_mode* предыдущего элемента, *ps_data* (), *icc_dt* [0] должно иметь значение 0%, принуждая к дифференцированному кодированию частоты.

icc_data () — синтаксический элемент, содержащий данные *ICC*.

cnt — локальная переменная, обозначающая число байтов, используемых для элемента *ps_extension* ().

ps_extension_size — длиной уровня расширения *PS* является *ps_extension_size*, измеренный в байтах. Если размер расширения приводит к использованию кода *escape* (*ps_extension_size* == 15), длина уровня расширения увеличивается на дополнительное количество байтов.

esc_count — в случае, если используется код *escape* (*ps_extension_size* == 15), *esc_count* описывает дополнительную длину уровня расширения *PS*, измеренную в байтах.

num_bits_left — локальная переменная, описывающая число битов, остающихся для чтения в элементе *ps_extension* ().

ps_extension_id — идентификационная бирка (версия) уровня расширения *PS* дается *ps_extension_id*. В данный момент поддерживается только одна версия (таблица 30).

Таблица 30 — Описание *ps_extension_id*

ps_extension_id	Версия
00 (0)	*v*0
01 (1)	Зарезервировано
10 (2)	Зарезервировано
11 (3)	Зарезервировано

fill_bits — эти *fill_bits* выполняют побайтное выравнивание данных *ps_extension* ().

enable_ipdopd — применение параметров *IPD* и *OPD* в потоке битов обозначается *enable_ipdopd*. Если установлено (*enable_ipdopd* == %1), параметры *IPD* и *OPD* отправляются, если отключено (*enable_ipdopd* == %0), параметры *IPD* и *OPD* для текущего фрейма в потоке битов не отправляются. В случае *enable_iid* == %0, *enable_ipdopd* не должен быть установлен в %1. Сетка квантования и для *IPD*, и для *OPD* обеспечивается в таблице 31. Если данные *IPD* или *OPD* не отправляются в потоке битов, все параметры *IPD* и *OPD* устанавливаются в 0 (то есть *index*=0).

ipd_dt [*e*] — этот флаг описывает для индекса огибающей *e*, кодируются ли параметры *IPD* дифференцированно в течение времени (*ipd_dt* == %1) или по частоте (*ipd_dt* == %0). В случае, когда *iid_mode* текущего элемента *ps_data* () отличается от *iid_mode* предыдущего элемента *ps_data* (), у *ipd_dt* [0] должно быть значение 0%, принуждающее к измененному кодированию частоты.

ipd_data () — синтаксический элемент, содержащий данные *IPD*.

opd_dt [*e*] — этот флаг описывает для индекса огибающей *e*, кодируются ли параметры *OPD* дифференцированно в течение времени (*opd_dt* == %1) или по частоте (*opd_dt* == %0). В случае, когда *iid_mode* текущего элемента *ps_data* () отличается от *iid_mode* предыдущего элемента *ps_data* (), у *opd_dt* [*0*] должно быть значение 0%, принуждающее к иному кодированию частоты.

opd_data () — синтаксический элемент, содержащий данные *OPD*.

reserved_ps — этот бит зарезервирован и имеет значение %0.

iid_par_dt [*e*] [*b*] — в случае дифференцированного кодирования параметров *IID* в течение времени (*iid_dt* [*e*] == %1), *iid_par_dt* [*e*] [*b*] описывает различие индекса *IID* относительно позиции параметра $b^{ый}$ для огибающей *e* – 1. Если никакой предыдущий параметр не доступен, *iid_par_dt* [*e*] [*b*] представляет различие индекса *IID* относительно декодируемого значения 0 (т.е *index*=0). Индекс *IID iid_par* [*e*] [*b*], определяется как:

$$iid_par\ [e]\ [b] = iid_par\ [e - 1[b] + iid_par_dt\ [b],$$

где *iid_par* [*e*–1] [*b*] представляет индекс *IID* предыдущей огибающей *e*–1. Значение *IID*, *iid* [*b*], получается при использовании *iid_par* [*e*] [*b*], как индекса к таблице 25 или таблице 26, в зависимости от *iid_mode*.

iid_par_df [*e*] [*b*] — в случае дифференцированного кодирования параметров *IID* по частоте (*iid_dt* [*e*] == %0), *iid_par_df* [*e*] [*b*] описывает отличие *IID* относительно (*b*–1)-го параметра в огибающей *e*. Если никакие предыдущие параметры не доступны, *iid_par_df* [*e*] [*b*] представляет отличие *IID* относительно декодируемого значения 0 (то есть *index*=0). Индекс *IID*, *iid_par* [*e*] [*b*], определяется как:

$$iid_par\ [e]\ [0] = iid_par_df\ [e]\ [0],$$

$$iid_par\ [e]\ [b] = iid_par\ [e]\ [b - 1] + iid_par_df\ [e]\ [b] \text{ для } b > 0,$$

где *iid_par* [*e*] [*b*–1] представляет индекс *IID* предыдущего значения *IID* для огибающей *e*. Величина *IID*, *iid* [*b*], получается при использовании *iid_par* [*e*] [*b*], как индекса к таблице 25 или таблице 26, в зависимости от *iid_mode*.

icc_par_dt [*e*] [*b*] — в случае дифференцированного кодирования параметров *ICC* в течение времени (*icc_dt* [*e*] == %1), *icc_par_dt* [*e*] [*b*] описывает отличие относительно $b^{ой}$ позиции параметра для огибающей *e*–1. Если никакой предыдущий параметр не доступен, *icc_par_dt* [*e*] [*b*] представляет различие в *ICC* относительно декодируемого значения 1 (то есть *index*=0). Индекс *ICC*, *icc_par* [*e*] [*b*] определяется как:

$$icc_par\ [e]\ [b] = icc_par\ [e - 1]\ [b] + icc_par_dt\ [e]\ [b],$$

где *icc_par* [*e*–1] [*b*] представляет индекс *ICC* предыдущей огибающей *e*–1. Значение *ICC*, ρ [*b*] получается при использовании *icc_par* [*e*] [*b*], как индекса к таблице 28.

icc_par_df [*e*] [*b*] — в случае дифференцированного кодирования параметров *ICC* по частоте (*icc_dt* [*e*] == %0), *icc_par_df* [*e*] [*b*] описывает отличие *ICC* относительно $(b–1)^{го}$ параметра для огибающей *e*. Если предыдущие параметры не доступны, *icc_par_df* [*e*] [*b*] представляет отличие *ICC* относительно декодируемого значения 1 (то есть *index*=0). Индекс *ICC*, *icc_par* [*e*] [*b*] определяется как:

$$icc_par\ [e]\ [0] = icc_par_df\ [e]\ [0],$$

$$icc_par\ [e]\ [b] = icc_par_[e]\ [b–1] + icc_par_df\ [e]\ [b] \text{ для } b > 0,$$

где *icc_par* [*e*] [*b*–1] представляет индекс *ICC* предыдущего значения *ICC* для огибающей *e*. Значение *ICC* ρ [*b*] получается при использовании *icc_par* [*e*] [*b*], как индекса к таблице 28.

ipd_par_dt [*e*] [*b*] — в случае дифференцированного кодирования параметров *IPD* в течение времени (*ipd_dt* [*e*] == %1), *ipd_par_dt* [*e*] [*b*] описывает отличие *IPD* относительно $b^{ой}$ позиции параметра для огибающей *e*. Если никакой предыдущий параметр не доступен, *ipd_par_dt* [*e*] [*b*] представляет отличие *IPD* относительно декодируемого значения 0 (то есть *index*=0). Для параметров *IPD* по модулю 8 применяется дифференцированное кодирование. Индекс *IPD*, *ipd_par* [*e*] [*b*], определяется как:

$$ipd_par\ [e]\ [b] = mod(ipd_par[e–1][b]+ipd_par_dt]\ [e]\ [b]\ ,\ 8)$$

где *ipd_par* [*e*–1] [*b*] представляет индекс *IPD* предыдущей огибающей, *e*–1. Величина *IPD*, *ipd* [*b*], получается при использовании *ipd_par* [*e*] [*b*], как индекса к таблице 31.

Таблица 31 — Сетка квантования для *IPD/OPD*

Индекс	0	1	2	3	4	5	6	7
Уровень представления	0	$\frac{\pi}{4}$	$\frac{\pi}{2}$	$\frac{3\pi}{4}$	π	$\frac{5}{4}\pi$	$\frac{3}{2}\pi$	$\frac{7}{4}\pi$

ipd_par_df [*e*] [*b*] — в случае дифференцированного кодирования параметров *IPD* по частоте (*ipd_dt* [*e*] == %0), *ipd_par_df* [*e*] [*b*] описывает различие *IPD* относительно (*b*–1)го параметра для огибающей *e*. Если предыдущие параметры не доступны, *ipd_par_df* [*e*] [*b*] представляет отличие *IPD* относительно декодируемого значения 0 (то есть *index*=0). Для параметров *IPD* по модулю 8 применяется дифференцированное кодирование. Индекс *IPD*, *ipd_par* [*e*] [*b*], определяется как:

$$ipd_par\,[e]\,[0] = ipd_par_df\,[e]\,[0]\,[e]\,[0]\,,$$

$$ipd_par\,[e]\,[b] = mod(ipd_par[e][b-1] + ipd_par_df)\,[e]\,[b]\,,\ 8)\ \text{для}\ b > 0,$$

где *ipd_par* [*e*] [*b*–1] представляет индекс *IPD* предыдущего значения *IPD* для огибающей *e*. Значение *IPD*, *ipd* [*b*] получается при использовании *ipd_par* [*e*] [*b*], как индекса к таблице 31.

opd_par_dt [*e*] [*b*] — в случае дифференцированного кодирования параметров *OPD* в течение времени (*opd_dt* [*e*] == %1), *opd_par_dt* [*e*] [*b*] описывает отличие *OPD* относительно *b*ой позиции параметра для огибающей (*e*–1). Если предыдущие параметры не доступны, *opd_par_dt* [*e*] [*b*] представляет отличие *OPD* относительно декодируемого значения 0 (то есть *index*=0). Замечание: для параметров *OPD* по модулю 8 применяется дифференцированное кодирование. Индекс *OPD*, *opd_par* [*e*] [*b*], определяется как:

$$opd_par\,[e]\,[b] = mod(opd_par[e-1][b] + opd_par_dt[e][b],8)\,,$$

где *opd_par* [*e*–1] [*b*] представляет индекс *OPD* предыдущей огибающей, *e*–1. Значение *OPD*, *opd* [*b*], получается при использовании *opd_par* [*e*] [*b*], как индекса к таблице 31.

opd_par_df [*e*] [*b*] — в случае дифференцированного кодирования параметров *OPD* в течение времени (*opd_dt* [*e*] == %0), *opd_par_dt* [*e*] [*b*] описывает отличие *OPD* относительно (*b*–1)ой позиции параметра для огибающей *e*. Если предыдущие параметры не доступны, *opd_par_df* [*e*] [*b*] представляет отличие *OPD* относительно декодируемого значения 0 (то есть *index*=0). Для параметров *OPD* по модулю 8 применяется дифференцированное кодирование. Индекс *OPD*, *opd_par* [*e*] [*b*], определяется как:

$$opd_par\,[e]\,[0] = opd_par_df\,[e]\,[0]$$

$$opd_par\,[e]\,[b] = mod\,(opd_par\,[e]\,[b-1] + opd_par_df\,[e]\,[b],\ 8\ )\ \text{для}\ b > 0,$$

где *opd_par* [*e*] [*b*–1] представляет индекс *OPD* предыдущего значения *OPD* для огибающей *e*. Величина *OPD*, *opd* [*b*], получается при использовании *opd_par* [*e*] [*b*], как индекса к таблице 31.

5.3 Индексация подфреймов

В случае, когда дифференцированное кодирование применяется от одного подфрейма до следующего, может быть назначен отрицательный индекс подфрейма *sf* фрейма *k*. В этом случае отрицательный подфрейм должен быть скорректирован согласно *sf* = *sf* + *nrof_subframes*.

Полученный таким образом индекс подфрейма находится во фрейме *k* – 1. Точно так же в случае, когда *sf* больше, чем *nrof_subframes*, подфрейм должен быть скорректирован согласно *sf* = *sf* – *nrof_subframes*.

Индекс подфрейма, полученный таким образом, находится во фрейме *k*+1.

6 Процесс декодирования

После деформатирования потока битов монофонический сигнал *M* восстанавливается как комбинация переходных процессов, синусоид и шума. Впоследствии используются параметры стерео, чтобы восстановить левый и правый сигналы из монофонического кодированного сигнала. Для двойного моно и моно параметрический декодер стерео не используется.

Параметрический декодер состоит из трех декодеров: декодера переходов, синусоидального декодера и декодера шума. Декодируемый сигнал получается суммированием выходов этих трех декодеров. В описании декодеров в параметрическом декодере индексация подфрейма *sf* и канала *ch* иногда для ясности опускается.

6.1 Переходные процессы

Определяются два типа переходных процессов, переходный процесс шага и переходный процесс типа Мейкслера. Декодирование переходного процесса шага включает только интерпретацию позиции. Для типа Мейкслера должны декодироваться параметризованная огибающая $g[n]$ и ряд синусоид.

6.1.1 Переходный процесс шага

Переходный процесс шага не генерирует собственный сигнал, но он используется, чтобы изменить форму окна для того, чтобы синтезировать синусоидальные и шумовые компоненты.

6.1.2 Переходный процесс Мейкслера

Для декодирования переходного процесса Мейкслера сначала должна быть сгенерирована огибающая. Для огибающей требуются следующие параметры: стартовая позиция t_{loc}, начальный наклон, обозначенный t_b_par (параметр b) и наклон спада, представленный t_chi_par (параметр ξ) (рисунок 1).

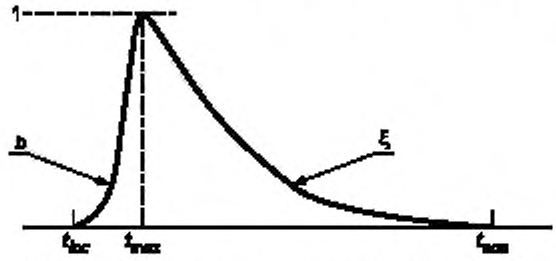

Рисунок 1 — Огибающая Мейкслера описывается функцией $g[n]$. Наклоном атаки управляет параметр b. Спадом управляет параметр ξ

Время старта огибающей t_{loc} для простоты объяснения принимается как $n = 0$. Эта огибающая $g[n]$ сгенерирована согласно

$$g[0] = \frac{(1-\xi^2)^{\frac{b}{2}}}{a_{\max}},$$

$$g[n] = g[n-1] \cdot \xi \cdot \sqrt{\frac{b+n-1}{n}},$$

для n =1, включая t_{end}, конечная позиция переходного окна t_{end} определяется ниже. Максимум $a_{\max}$ находится в позиции $t_{\max}$ и дается приближениями

$$t_{\max} = \frac{b-1}{-2 \cdot \log_e(\xi)},$$

$$a_{\max} = \sqrt{\frac{-2 \cdot \log_e(\xi)}{\Gamma(b)}} \cdot \left(\frac{b-1}{e}\right)^{\frac{b \cdot 1}{2}}.$$

Эти сложные выражения, особенно выражение для $a_{\max}$, были оценены для допустимых значений t_b_par и t_chi_par и сводятся в таблицу 32.

Таблица 32 — $t_{\max}$ для всех возможных значений t_b_par и t_chi_par

$t_{\max}$		t_b_par			
		0	1	2	3
t_chi_par	0	15	30	45	59
	1	20	39	57	75
	2	30	56	79	100
	3	57	96	126	150

Таблица 33 — a_{max} для всех возможных значений *t_b_par* и *t_chi_par*

a_{max}		*t_b_par*			
		0	1	2	3
t_chi_par	0	0,152713500109658	0,131630525645664	0,120142673294398	0,112550174511598
	1	0,132843681407528	0,115639700421076	0,106510539071702	0,100663024431527
	2	0,109279971016712	0,0971964875412947	0,0909719057150294	0,0872632874594248
	3	0,0797175749717262	0,0744985442180281	0,0723059623257423	0,0715041477354716

Переходный процесс должен быть синтезирован до t_{end}. Позиция t_{end} определяется как конец второго полного последующего подфрейма после позиции переходного процесса t_{loc}.

$$t_{end} = 3 \cdot S - t_{loc} - 1.$$

Подсчет выборок в подфрейме начинается с 0. Если переходный процесс стартует точно в начале подфрейма ($t_{loc} = 0$), то $t_{end} = 3^*S–1$.

6.1.3 Синусоиды под огибающей

Получающееся представление переходного процесса получается путем комбинирования огибающей и синусоид согласно

$$t(n) = g[n] \cdot \sum_{i=1}^{t_rof_sin} ta_q[i] \cdot \cos\left(tf_q[i] \cdot n + tp_q[i]\right),$$

для n вплоть до n=0 и включая t_{end}.

6.2 Синусоиды

6.2.1 Соединение

Фрейм обновления, обозначенный *refresh_sinusoids* == %1, используется, чтобы указать, стартует ли фрейм с абсолютных значений для всех продолжений или стартует с дифференциально кодированных продолжений. Для рождения каждой синусоидальной дорожки во фрейме в потоке битов предоставляется *s_cont*, чтобы сигнализировать о числе подфреймов, в которых продолжается дорожка после текущего подфрейма в этом фрейме. В случае, если дорожка продолжается в первом подфрейме следующего фрейма, к этому числу добавляется 1. Если дорожка продолжается после первого подфрейма следующего фрейма, к этому числу добавляется 2. Основываясь на этой информации, декодер в состоянии соединить параметры, которые принадлежат дорожке.

s_cont заполняется в следующем порядке:

1) продолжения;

2) рождения (отсортированные по частоте в порядке возрастания).

При прохождении от одного подфрейма к следующему декодер отслеживает число продолжений, *s_nrof_continuations* [*sf*]. Число продолжений, существующих в подфрейме *sf*+1, может быть непосредственно получено из числа записей в *s_cont* [*sf*], не равных нулю. Для первого подфрейма в фрейме *s_nrof_continuations* читается из потока битов, чтобы задействовать произвольный доступ.

Общее количество синусоидальных компонентов в подфрейме *sf*, *s_nrof_sin* [*sf*], вычисляется так:

sf == 0:

s_nrof_sin[0]=*s_nrof_continuations*[0][*ch*]+*s_nrof_births*[0][*ch*],

sf>0

$$s_nrof_\sin[sf] = \sum_{i=1}^{\max_nrof_sinusoids \cdot 1} (s_cont[sf-1][ch][i] > 0) + s_nrof_birts[sf][ch]$$

6.2.2 Декодирование синусоидальных параметров

В описании ниже мы принимаем, что имеет место синусоидальная дорожка длины к, в подфреймах *sf* = [*K*, K_{+k} + 1]. Для рождений дорожки (*sf* = *K*) частота и фаза индекса синусоиды *n* представляются f_q [*K*] [*ch*] [*n*] и sp_q [*K*] [*ch*] [*n*], соответственно. Для продолжений, чтобы получить информацию о частоте и фазе для подфрейма, требуются уровни представления вместе с отслеживанием информации.

Для продолжения ($sf = [K + 1, K_{+k} - 1]$) уровни представления *s_delta_cont_freq_pha* [*sf*] [*ch*] [*n*] преобразовываются в квантованную ошибку прогноза Δ [*sf*] [*ch*] [*n*], используя таблицу 35 с индексом == 2. Квантованная ошибка прогноза Δ [*sf*] [*ch*] [*n*] затем умножается на масштабный коэффициент *c* [*sf*] [*ch*] [*n*]. Для первого продолжения *c* [*K*+1] [*ch*] [*p*], где *p* представляет индекс синусоиды в подфрейме *K*+1, зависит от частоты рождения, то есть f_q [*K*]. Таблица 34 показывает значение масштабного коэффициента *c* для возможных частотных диапазонов f_q [*K*].

Таблица 34 — Таблица масштабных коэффициентов

Частотный диапазон, Гц	Масштабный коэффициент *c*[*K*+1]
[0—500]	1/8
<500—1000]	1/4
<1000—4000]	1/2
<4000—22050]	1

Для других продолжений $sf = [K + 2, K_{+k} - 1]$ *c* изменяется согласно полученным уровням представления вдоль дорожки посредством блока *"Backward Adaptation"*. Если *s_delta_cont_freq_pha* [*sf*] [*ch*] [*n*] равно 1 или 2 (внутренний уровень) для подфрейма *sf*, то *c* для подфрейма *sf*+1 устанавливается в

$$c[sf + 1]\,[ch]\,[p] = c[sf]\,[ch]\,[n] \times 2^{-1/4}.$$

Если *s_delta_cont_freq_pha* [*sf*] [*ch*] [*n*] равняется 0 или 3 (внешний уровень), то *c* для подфрейма *sf*+1 устанавливается в

$$c[sf + 1]\,[ch]\,[p] = c[sf]\,[ch]\,[n] \times 2^{1/2}.$$

Чтобы избежать очень маленьких или очень больших записей по ошибке прогноза, адаптация выполняется только если абсолютное значение внутреннего уровня 0,75*c* [*sf*+1] [*ch*] [*p*] находится между $\pi/128$ и $3\pi/8$.

При получении квантованной ошибки прогноза Δ вывод фильтра прогноза добавляется к ней, приводя к развернутой фазе ψ_r:

$$\psi_r\,[sf]\,[ch]\,[n] = \psi_p\,[sf]\,[ch]\,[n] + \Delta\,[sf]\,[ch]\,[n] \cdot c\,[sf]\,[ch]\,[n].$$

Используется прогнозирующее устройство второго порядка. Поведение ввода — вывода фильтра таково

$$\psi_p\,[sf + 1]\,[ch]\,[p] = 2 \cdot \psi_r\,[sf]\,[ch]\,[n] - \psi_r\,[sf - 1]\,[ch]\,[q],$$

где *q* является индексом синусоиды в подфрейме *sf*–1, ψ_r является вводом и ψ_p является выводом фильтра прогноза.

Чтобы инициализировать фильтр прогноза, нужно одно значение из предыдущих значений для ввода ψ_r [*K*–1], где *K* является индексом подфрейма рождения дорожки. Так как это значение не доступно, предполагаем, что частота является постоянной в подфрейме (*K*–1). Для рождения дорожки имеем информацию о частоте и фазе, таким образом, можем вычислить ввод в (*K*–1) и *K* согласно:

$$\psi_r\,[K - 1]\,[ch]\,[n] = sp_q\,[K]\,[ch]\,[n] - f_q\,[K]\,[ch]\,[n]\,S,$$

$$\psi_r\,[K]\,[ch]\,[n] = sp_q\,[K]\,[ch]\,[n],$$

где *S* представляет интервал обновления. Развернутые фазы являются отфильтрованными по низким частотам *LPF-phase* блока. Это делается следующим образом:

$$\psi\,[sf]\,[ch]\,[n] = 0{,}25\psi_r\,[sf + 1]\,[ch]\,[p] + 0{,}5\psi_r\,[sf]\,[ch]\,[n] + 0{,}25\psi_r\,[sf - 1]\,[ch]\,[q],$$

где *sf* является индексом подфрейма вдоль дорожки. В конце дорожки ($sf = K_{+k} - 1$), применяется следующее правило:

$$\psi\,[sf]\,[ch]\,[n] = \psi_r\,[sf]\,[ch]\,[n].$$

Восстановленные фазы получаются из сглаженных развернутых фаз следующим образом:

$$sp_q\,[sf]\,[ch]\,[n] = mod\,(\psi\,[sf]\,[ch]\,[n] + \pi,\, 2) - \pi.$$

Чтобы получить частоту, развернутые фазы должны дифференцироваться вдоль дорожки. Дифференциация реализуется приближением. Частота получается с помощью.

$$v[sf][ch][n] = \left(\psi[sf][ch][n] - \psi[sf-1][ch][q]\right)\frac{2}{S} - v[sf-1][ch][q],$$

где S представляет интервал обновления и $\varpi\,[K]\,[ch]\,[n] = f_q\,[K]\,[ch]\,[n]$. Когда фаза рождения и частота рождения известны в декодере, вычисляются частоты ϖ последующих фреймов. Чтобы ослабить сигнал с шумами, которые вносятся этой дифференциацией, применяется фильтр низких частот на частоты (*LPF-freq*):

$$f_q\,[sf]\,[ch]\,[n] = 0{,}25\varpi\,[sf+1]\,[ch]\,[p] + 0{,}5\varpi\,[sf]\,[ch]\,[n] + 0{,}25\varpi\,[sf-1]\,[ch]\,[q].$$

Для первого продолжения дорожки ($sf=K+1$) определение изменяется на:

$$f_q\,[K+1]\,[ch]\,[p] = 0{,}5 \cdot \varpi\,[K+1]\,[ch]\,[p] + 0{,}5 \cdot \varpi\,[K]\,[ch]\,[n].$$

Также последняя частота в дорожке ($sf = K_{+k}-1$) получается другим способом:

$$f_q\,[sf]\,[ch]\,[n] = 0{,}5 \cdot \varpi\,[sf]\,[ch]\,[n] + 0{,}5 \cdot \varpi\,[sf-1]\,[ch]\,[q].$$

Для дорожек длиной $\kappa = 2$ продолжение вычисляется согласно

$$f_q\,[sf]\,[ch]\,[n] = 0{,}5 \cdot \varpi\,[sf-1]\,[1\ ch]\,[q] + 0{,}5 \cdot \varpi\,[sf]\,[ch]\,[n].$$

Таким образом получаются фазы и частоты из уровней представления *s_delta_cont_freq_pha*.

Во фреймах обновлений применяется следующая процедура. Если $sf = [K, \ldots, K+R, \ldots, K_{+k}-1]$, подфрейм $K+R$ является первым подфреймом фрейма с *refresh_sinusoids* == %1. Подфрейм ($K+R-1$) является последним подфреймом фрейма с *refresh_sinusoids_next_frame* == %1. Значения фазы и частоты для подфрейма с подфрейма K до и включая подфрейм $K+R-1$ получаются как описано выше, как будто дорожка заканчивается в подфрейме ($K+R-1$). Значения фазы и частоты для подфрейма $K+R$ до подфрейма ($K_{+k}-1$) получаются как описано выше, как будто подфрейм $K+R$ является рожденным. Для инициализации квантованной ошибки прогноза Δ *s_adpcm_grid* используется в качестве индекса к таблице 35 и $c\,[K+R]\,[ch]\,[n] = 1$.

Таблица 35 — Квантованная ошибка прогноза Δ

Δ	*s_delta_cont_freq_pha*			
Индекс	0	1	2	3
0	–4,2426	–1,0607	1,0607	4,2426
1	–3,5676	–0,8919	0,8919	3,5676
2	–3,0000	–0,7500	0,7500	3,0000
3	–2,5227	–0,6307	0,6307	2,5227
4	–2,1213	–0,5303	0,5303	2,1213
5	–1,7838	–0,4460	0,4460	1,7838
6	–1,5000	–0,3750	0,3750	1,5000
7	–1,2613	–0,3153	0,3153	1,2613
8	–1,0607	–0,2652	0,2652	1,0607
9	–0,8919	–0,2230	0,2230	0,8919
10	–0,7500	–0,1875	0,1875	0,7500
11	–0,6307	–0,1577	0,1577	0,6307
12	–0,5303	–0,1326	0,1326	0,5303
13	–0,4460	–0,1115	0,1115	0,4460
14	–0,3750	–0,0938	0,0938	0,3750
15	–0,3153	–0,0788	0,0788	0,3153

Окончание таблицы 35

Δ	*s_delta_cont_freq_pha*			
Индекс	0	1	2	3
16	–0,2652	–0,0663	0,0663	0,2652
17	–0,2230	–0,0557	0,0557	0,2230
18	–0,1875	–0,0469	0,0469	0,1875
19	–0,1577	–0,0394	0,0394	0,1577
20	–0,1326	–0,0331	0,0331	0,1326
21	–0,1115	–0,0279	0,0279	0,1115

В случае, когда *phase_jitter_present* устанавливается в %1, и фаза и частота получаются посредством декодирования *ADPCM*, никакое дрожание фазы не применяется.

6.2.3 Синтез синусоид для сегментов без переходного процесса

Параметры синусоид используются, чтобы синтезировать синусоидальные компоненты. Это делается на основе сегмента, состоящего из выборок *L*.

Синтез использует стратегию 50% перекрытия и добавления. Чтобы синтезировать подфрейм, в начале нового фрейма должны быть доступными параметры предыдущего подфрейма. Это означает, что должны быть доступными параметры последнего подфрейма в предыдущем фрейме.

Фактический синтез синусоиды выполняется согласно

$$s_i[n] = sa_q[i] \cdot \cos\left(f_q[i] \cdot \left(n - \frac{L-1}{2}\right) + sp_q[i]\right), \text{ для } n = [0, L-1].$$

Фазовая информация sp_q определяется для середины сегмента (=(*L*–1)/2). Фазовая информация исходной фазы sp_q вычисляется из *s_phi* (см. 5.2). Для продолжений фаза вычисляется как описано в 6.2.2.

В перекрытии и добавлении выбирается следующее дополнительное симметричное окно амплитуды

$$w_s[n] = 1/2 - 1/2\cos\left(\frac{\pi(2n+1)}{L}\right), \text{ для } n = [0, L-1].$$

Сегмент длины *L* получается из

$$\bar{s}_{sf}[n] = w_s[n] \sum_{i=0}^{s_nrof_sin[sf]} s_i[n], \text{ для } n = [0, L-1].$$

Синусоидальное добавление для подфрейма *sf* тогда вычисляется, используя добавление из предыдущего подфрейма согласно:

$$s_{sf}[n] = \bar{s}_{sf-1}[S+n] + \bar{s}_{sf}[n], \text{ для } n = [0, S-1].$$

6.2.4 Синтез синусоид для сегментов без переходного процесса

Для синусоид с частотой ниже 400 Гц должно использоваться только окно $w_s[n]$.

Для других синусоид для синтеза сегмента применяются другие формы окна, в зависимости от компонента (продолжение, рождение, или отмирание).

Следующее выражение точно описывает окно $w_s[n]$

$$w_s[n] = \begin{cases} 0, & 0 \le n < d-10 \\ \dfrac{n-d+11}{22}, & d-10 \le n \le d+10 \\ 1, & d+10 < n < S \\ 1/2 - 1/2\cos\left(\pi \cdot \dfrac{2n+1}{L}\right), & S \le n < L \end{cases}.$$

6.2.4.1 Отмирание и переходный процесс во второй половине сегмента

Следующее выражение описывает окно $w_s[n]$

$$w_s[n] = \begin{cases} 1/2 - 1/2\cos\left(\pi \cdot \dfrac{2n+1}{L}\right), & 0 \le n < S \\ 1, & S \le n < d - 10 \\ \dfrac{d - n + 11}{22}, & d - 10 \le n \le d + 10 \\ 0, & d + 10 < n < L \end{cases}$$

6.3 Шум

Шум синтезируется в интервалах 4 подфреймов или 2*L* выборок. Модель для синтеза шума состоит из генератора псевдослучайных чисел, временного регулятора огибающей, механизма окна для перекрытия и добавления, и фильтра *IIR*.

Временная огибающая *H*[*n*] представляется при использовании единственного усиления и ряда линейных спектральных частот (*LSFs*), представляющих коэффициенты *LPC*. Как усиление, так и *LSFs* обновляются один раз каждые 4 подфрейма (2*L* выборок). Коэффициенты Лагерра представляются параметрами *LAR*. Они обновляются раз за 2 подфрейма (*L* выборок), то есть вдвое чаще малой частоты обновления, которая используется для временной огибающей.

Чтобы предотвратить разрывы, интервалы, которые изменяются временной огибающей, имеют перекрытие 25 %. В области перекрытия используется окно Ханнинга. Первые 4 сгенерированные подфрейма 2*L* выборок стартуют с постепенного появления, используя окно Ханнинга.

Окно w_n определяется так

$$w_s[n] = \begin{cases} 1/2 - 1/2\cos\left(\dfrac{3\pi(2n+1)}{8S}\right), & n = \left[0, \dfrac{4S}{3} - 1\right], \\ 1, & n = \left[\dfrac{4S}{3}, -4S - 1\right], \\ w_n\left[\dfrac{16S}{3} - 1 - n\right], & n = \left[4S, \dfrac{16S}{3} - 1\right]. \end{cases}$$

6.3.1 Генерация шума

Шум генерируется с помощью генератора псевдослучайных чисел, определенный линейной конгруэнтной последовательностью *U*

$$U[n+1] = \text{mod}\,(a*U[n] + c,\ m),$$

где *U* [0] является начальным значением, *a* множитель, *c* инкремент и *m* модуль (с $m=2^{32}$). В начале декодирования начальное значение устанавливается в *channel_number* (0 = левый, 1 = правый), приводя к независимым источникам шума для каждого канала. Для генерации каждого следующего интервала шума начальное значение устанавливается в конечное значение предыдущего интервала. Алгоритм приводится ниже.

```
#define RAND_SCALE  (1/4294967296,0)
# define RAND_FACTOR        1664525L
# define RAND_OFFSET        1013904223L
Double noiseUDN (unsigned long *lp_seed)
{
*lp_seed = (*lp_seed * RAND_FACTOR+RAND_OFFSET) & 0xFFFFFFFF;
return *lp_seed * RAND_SCALE;
}
```

Этот алгоритм возвращает значение *U* = [0,1>. Нормальное распределение *X* получается добавлением 12 последовательных выборок распределения (*U*–0,5). Для следующей выборки *X* используются 12 новых последовательных выборок. Используя это нормальное распределение *X*, посредством фильтрации шума генерируется спектральный шум.

Чтобы избежать разрывов в генерации шума, перекрывающиеся временные интервалы берутся от генератора шума холостого хода. Это реализуется, копируя начальное число из предыдущего интервала в текущий интервал во время перекрытия.

6.3.2 Временная огибающая

Временная огибающая применяется к интервалу 2*L*+4*S*/3 выборок, которые сгенерированы генератором случайного шума. Форма временной огибающей представляется временным доменом, эквивалентным *Line Spectral Frequencies*, которые являются представлением коэффициентов *LPC*. Параметр дополнительного усиления используется, чтобы масштабировать всю огибающую. Так как у интервалов *LSF* имеется перекрытие 25 %, существует потенциальная избыточность в параметрах *LSF* в этой области перекрытия. В случае, когда присутствует эта избыточность, кодируется только один набор *LSFs*, допустимый для двух огибающих в области перекрытия. Об этой ситуации сообщается параметром *n_overlap_lsf*. В случае *refresh_noise* = = %1 первый *LSF* и усиление для этого определенного интервала кодируются в абсолютных значениях посредством параметров *n_lsf* и *n_gain*. Последующие *LSFs* кодируются дифференцированно со ссылкой друг на друга.

В случае *refresh_noise* == %0, параметр усиления для этого интервала кодируется дифференцированно со ссылкой на усиление предыдущего фрейма посредством параметра *n_delta_gain*. Кодирование параметров *LSF* в той ситуации зависит от установки *n_overlap_lsf*. В случае *refresh_noise* == %0 и *n_overlap_lsf* == %0, *LSF* кодируется как в ситуации, где *refresh_noise* == %1. В случае *refresh_noise* == %0 и *n_overlap_lsf*==%1, число коэффициентов *LSF*, *n_nrof_overlap*, вычисляется из предыдущего определения в канале *ch* согласно

n_nrof_overlap = 0;
for (*i* = 0; *i* <*n_nrof_lsf*; *i* ++) {
if (*n_lsf* [*sf*-4] [*ch*] [*i*]> = 192) *n_nrof_overlap* ++
}

Коэффициенты *LSF*, которые перекрываются, копируются из предыдущего определения согласно

for (*i* = 0, *j* = *n_nrof_lsf* – *n_nrof_overlap*; *i* <*n_nrof_overlap*; *i* ++, *j* ++)
{
n_lsf [*sf*] [*ch*] [*i*] = *n_lsf* [*sf*–4] [*ch*] [*j*] – 192
}

6.3.2.1 Декодирование параметров усиления и *LSF*

Усиление масштабирует всю временную огибающую. Существуют две огибающих на фрейм, усиление для первой и второй временной огибающей кодируется в *sf*=0 и *sf*=4 соответственно. Коэффициент усиления *G*, который фактически применяется к временной огибающей, вычисляется как

$$G\begin{cases}0, & \text{если } ngain_{sf} == 0\\ 10^{\frac{ngain_{sf}-21}{20}} & \text{в других случаях}\end{cases}.$$

Декодируемые *LSFs*, $nlsf_q$ преобразовываются в *a*-параметры, используя следующие уравнения. Все *LSFs* преобразовываются к позициям на единичном круге

$z[i] = e^{j\,nlsf_Q[i]}$ i = [1,*n_nr_of_lsf*].

Эти позиции разделяются на два полинома:

$$z_P[i] = z[2i] \qquad i = \left[1, \left\lfloor \frac{n_nr_of_lsf}{2} \right\rfloor\right].$$

$$z_Q[i] = z[2i-1] \qquad i = \left[1, \left\lfloor \frac{n_nr_of_lsf}{2} \right\rfloor\right].$$

Для обоих полиномов комплексно сопряженные элементы связываются

$$z_P[i] = z_P^*\left[i - \left\lfloor \frac{n_nr_of_lsf}{2} \right\rfloor\right] \quad i = \left[1 + \left\lfloor \frac{n-nr_of_lsf}{2} \right\rfloor, 2\left\lfloor \frac{n-nr_of_lsf}{2} \right\rfloor\right],$$

$$z_Q[i] = z_Q^*\left[i - \left\lfloor \frac{n_nr_of_lsf}{2} \right\rfloor\right] \quad i = \left[1 + \left\lfloor \frac{n-nr_of_lsf}{2} \right\rfloor, 2\left\lfloor \frac{n-nr_of_lsf}{2} \right\rfloor\right].$$

Полиномы вычисляются следующим образом

$$P_P = \prod_{i=1}^{2\lfloor n_nr_of_lsf \rfloor} (z - z_p[i]),$$

$$P_Q = \prod_{i=1}^{2\lfloor n_nr_of_lsf \rfloor} (z - z_Q[i]).$$

В случае, если *n_nr_of_lsf* нечетен, полиномы изменяются так

$P_Q = P_Q(z + 1)(z - 1)$.

В случае, если *n_nr_of_lsf* четный

$P_P = P_P(z - 1)$,

$P_Q = P_Q(z - 1)$.

Полином $A(z)$ дается как

$$A_z = \frac{P_P(z) + p_Q(z)}{2}.$$

В конечном счете огибающая $H[n]$, где n является индексом выборки, вычисляется как:

$$H[n] = \frac{G}{\left| A\left(e^{j\frac{n\pi}{2L+\frac{4S}{3}}} \right) \right|}, \quad n = \left\lfloor 0{,}2L + \frac{4}{S} - 1 \right\rfloor.$$

Последовательность шума умножается на временную огибающую. В случае *n_nr_of_lsf* == 0, $H[n]$ определяется как

$$H[n] = G, \quad n = \left\lfloor 0{,}2L + \frac{4S}{3} - 1 \right\rfloor.$$

6.3.3 Шумовая фильтрация

Фильтр синтеза Лагерра применяется после работы по перекрытию — добавлению.

Параметры для фильтрации Лагерра обновляются каждые 2 подфрейма (выборок). Чтобы удостовериться, что в генерации первой выборки уже получается требуемая спектральная плотность, должны быть установлены начальные состояния фильтра. Это реализуется копированием конечных состояний после генерации интервала в начальные состояния для генерации следующего интервала. В случае, если *refresh_noise* == %0, или запуска декодирования, начальные состояния фильтра будут установлены в 0. Так, используя первый набор параметров для следующих 2 подфреймов (с новыми параметрами) в качестве начальных состояний, используются конечные состояния фильтра, полученные в результате его предыдущей работы. Для обновлений параметров *LAR* используется дифференциальное кодирование относительно параметров *LAR* предыдущего интервала.

В потоке битов, для каждого подфрейма, коэффициенты для структуры фильтра Лагерра кодируются в нотации *Log Area Ratio* (*LAR*). Первый шаг реконструкции параметров Лагерра является деквантизацией *LARs*. Коэффициенты *parcor* (*'rfc'*) получаются из деквантованных *LARs*. Последний шаг является преобразованием коэффициентов *FIR* обратно в коэффициенты Лагерра.

6.3.3.1 Квантованные *LARs*

Коэффициенты *LAR* в знаменателе деквантуются, умножая значение, закодированное в потоке битов на постоянное Δ_{LAR}, которое определяется как

$$\Delta_{LAR} = \frac{dynr}{levels - 1},$$

где *dynr*=2*8 является динамическим диапазоном коэффициентов *LAR* (от –8 до +8), и *levels*=2^{bits}–2, с *bits*=9 представляет число уровней представления.

6.3.3.2 Преобразование *LARs* в *parcors*

Следующий алгоритм описывает преобразование *m* коэффициентов *LAR*, $nlar_q$ в m (m=n_nrof_den) parcors 'rfc'.

for (*i*=0; *i* <*m*; *i* ++)

{

rfc [*i*] = (*exp* (*nlar*$_q$ [*i*])–1) / (*exp* (*nlar*$_q$ [*i*]) +1)

}

6.3.3.3 Преобразование parcors в коэффициенты *FIR*

Следующий алгоритм описывает преобразование *m* коэффициентов *parcor, 'rfc'* в *m*+1 *a*-параметров '*p*'.

for (*k*=0; *k* <*m*; *k* ++)

{

d [*k*] = – *rfc* [*k*];

for (*i*=0; *i* <*k*; *i* ++)

{

d [*i*] = *tmp* [*i*] +*rfc* [*k*] **tmp* [*k*–*i*–1];

}

for (*i*=0; *i* <=*k*; *i* ++)

{

tmp [*i*] = *d* [*i*];

}

}

p [*0*] = 1,0;

for (*k* = 0; *k* <*м.*; *k* ++)

{

p [*k+1*] = –*d* [*k*];

}

6.3.3.4 Преобразование коэффициентов *FIR* в коэффициенты Лагерра

a-параметры *p* преобразуются обратно в коэффициенты Лагерра *a*, используя следующий алгоритм

$a'_M = P_M$,

$a'_m = P_m - a'_{m+1}\lambda$,

где *m* = [*n_nrof_den*–1.. 0] *u*

$$a_m = \frac{a'_m}{a'_0}\sqrt{1-\lambda^2}.$$

6.4 Параметрическое стерео

6.4.1 Параметры стерео

В представлении образа стерео используются три различных типа параметров стерео:

1) межканальное различие в интенсивности или *IID*, определенный относительными уровнями ограниченного полосой сигнала;

2) межканальное и полное различие фаз, *IPD* и *OPD*, определяющее поведение фазы ограниченного полосой сигнала;

3) межканальная когерентность *ICC*, определяющая (не)сходство левого и правого ограниченного полосой сигнала.

Ввод в параметрический стереодекодер состоит из монофонического параметрически сгенерированного сигнала, полученного посредством синтеза переходного процесса, синусоид и шума. Вывод состоит из левого и правого представления стерео соответственно.

6.4.2 Блок фильтров анализа *QMF*

Этот блок фильтров идентичен 64 блоку фильтров комплексного анализа *QMF*. Однако в уравнении матрицы *M* (*k, n*) множитель "(2**n*+1)" нужно заменить на "(2**n*–1)". Ввод в блок фильтров является блоками из 64 выборок монофонического синтезируемого сигнала М. Для каждого блока блок фильтров выводит один слот из 64 выборок *QMF*.

6.4.3 Фильтрация низкой частоты

Поддиапазоны *QMF* далее разделяются, чтобы получить более высокую разрешающую способность по частоте, задействующую надлежащий анализ и синтез стерео для более низких частот. В зависимости от числа полос стерео были определены две гибридных конфигурации. Обзор разделений и типа фильтра, который используется, чтобы выполнить разделение, см. в таблице 36.

Таблица 36 — Обзор разделения низкой частоты для доступных конфигураций

Конфигурация, число полос стерео	Поддиапазон *QMF p*	Число полос Q^p	Фильтр
10, 20	0	8	Тип A
	1	2	Тип B
	2	2	
34	0	12	Тип A
	1	8	
	2	4	
	3	4	
	4	4	

Тип A: $G_q^p = g^p[n]\exp\left(j\frac{2\pi}{Q^p}\left(q+\frac{1}{2}\right)(n-6)\right)$.

Тип B: $G_q^p = g^p[n]\cos\left(\frac{2\pi}{Q^p}q(n-6)\right)$,

где g^p представляет фильтры прототипа в поддиапазоне *QMF p*. Q^p представляет число подподдиапазонов в поддиапазоне *QMF p*, *q* индекс подподдиапазона в канале *QMF p* и *n* индекс времени. Все прототипные фильтры длиной 13 имеют задержку 6 выборок *QMF*. Прототипные фильтры перечисляются в таблице 37 и таблице 38 для конфигураций 10, 20 и 34 полосы стерео соответственно.

Таблица 37 — Коэффициенты прототипного фильтра для фильтров, которые разделяют нижние поддиапазоны *QMF* для конфигураций 10 и 20 полос стерео

n	$g^0[n]$, $Q^0 = 8$	$g^{1,2}[n]$, $Q^{1,2} = 2$
0	0,00746082949812	0
1	0,02270420949825	0,01899487526049
2	0,04546865930473	0
3	0,07266113929591	–0,07293139167538
4	0,09885108575264	0
5	0,11793710567217	0,30596630545168
6	0,125	0,5
7	0,11793710567217	0,30596630545168
8	0,09885108575264	0
9	0,07266113929591	–0,07293139167538
10	0,04546865930473	0
11	0,02270420949825	0,01899487526049
12	0,00746082949812	0

Таблица 38 — Коэффициенты прототипного фильтра для фильтров, которые разделяют нижние поддиапазоны *QMF* для конфигурация 34 полосы стерео

n	$g^0[n]$, $Q^0 = 12$	$g^1[n]$, $Q^1 = 8$	$g^{2,3,4}$, $Q^{2,3,4} = 4$
0	0,04081179924692	0,01565675600122	–0,05908211155639
1	0,03812810994926	0,03752716391991	–0,04871498374946
2	0,05144908135699	0,05417891378782	0
3	0,06399831151592	0,08417044116767	0,07778723915851

Окончание таблицы 38

n	$g^0[n]$, $Q^0 = 12$	$g^1[n]$, $Q^1 = 8$	$g^{2,3,4}$, $Q^{2,3,4} = 4$
4	0,07428313801106	0,10307344158036	0,16486303567403
5	0,08100347892914	0,12222452249753	0,23279856662996
6	0,08333333333333	0,12500000000000	0,25000000000000
7	0,08100347892914	0,12222452249753	0,23279856662996
8	0,07428313801106	0,10307344158036	0,16486303567403
9	0,06399831151592	0,08417044116767	0,07778723915851
10	0,05144908135699	0,05417891378782	0
11	0,03812810994926	0,03752716391991	–0,04871498374946
12	0,04081179924692	0,01565675600122	–0,05908211155639

Чтобы выровнять по времени все выборки, приходящие из гибридного блока фильтров, компенсируют задержку остающихся поддиапазонов QMF, которые не фильтровались. Эта задержка составляет 6 выборок поддиапазона QMF. Это означает $G_0^k(z) = z^{-6}$ для k = [3... 63] (10, 20 полос стерео) или k = [5... 63] (34 полосы стерео). Чтобы компенсировать полную задержку гибридного блока фильтров анализа, первые 10 наборов (6 от задержки и 4 от фильтра *QMF*) гибридных поддиапазонов сбрасывается и поэтому не принимается во внимание при обработке.

Результатом этой работы является слот гибридных выборок поддиапазона, состоящих из *LF* (низкая частота) части поддиапазона *sub QMF* и *HF* (высокая частота) части поддиапазона *QMF*.

6.4.4 Кадрирование

Один фрейм параметрического аудио включает два фрейма данных стерео. Параметры стерео в пределах фрейма стерео могут быть присвоены одному или более слотам. Границы фрейма стерео и позиции n_e слотов, которым были присвоены параметры стерео, определяют так называемые области. Параметры стерео определяются для последнего слота.

6.4.5 Декорреляция

Посредством всечастотной фильтрации и задержки, выборки поддиапазона *sub* $s_k(n)$ преобразовываются в декоррелированные выборки поддиапазона *sub* $d_k(n)$, где k представляет частоту в гибридном спектре, а n — индекс времени.

6.4.5.1 Константы

DECAY_SLOPE=0,05 — Наклон затухания всечастотного фильтра

NR_ALLPASS_LINKS=3 — Число связей фильтра для всечастотного фильтра

NR_PAR_BANDS — Число полос частот, которые могут адресоваться индексом параметра, $b(k)$.

$NR_PAR_BANDS \begin{cases} 20 & \text{, 10 или 20 полос стерео} \\ 34 & \text{, 34 полосы стерео} \end{cases}$

NR_BANDS — Число полос частот, которые могут адресоваться индексом поддиапазона *sub*, *k*.

$NR_PAR_BANDS \begin{cases} 71 & \text{, 10 или 20 полос стерео} \\ 91 & \text{, 34 полосы стерео} \end{cases}$

DECAY_CUTOFF — Стартовая полоса частот для наклона затухания всечастотного фильтра

$DECAY_CUTOFF \begin{cases} 10 & \text{, 10 или 20 полос стерео} \\ 32 & \text{, 34 полосы стерео} \end{cases}$

NR_ALLPASS_BANDS — Число полос всечастотного фильтра

$NR_ALLPASS_BANDS \begin{cases} 30 & \text{, 10 или 20 полос стерео} \\ 50 & \text{, 34 полосы стерео} \end{cases}$

SHORT_DELAY_BAND Первая полоса стерео, использующая короткую, в одну выборку, задержку

$$SHORT_DELAY_BAND \begin{cases} 42, & \text{10 или 20 полос стерео} \\ 62, & \text{34 полосы стерео} \end{cases}$$

$a_{Smooth} = 0{,}25$ Коэффициент сглаживания

6.4.5.2 Вычисление декоррелируемого сигнала $d_k(z)$

Процесс декорреляции для первой полосы частот *NR_ALLPASS_BANDS* $s_k(n)$ основан на всечастотном фильтре, описанном в Z-домене. Его функция передачи для каждой полосы k определяется:

$$H_k(z) = z^2 \cdot \varphi_{Fract}(k) \cdot \prod_{m=0}^{NR_ALLPASS_LINKS} \frac{Q_{Fract_allpass}(k,m) z^{-d(m)} - a(m) g_{DecaySlope}(k)}{1 - a(m) g_{DecaySlope}(k) Q_{fract_allpass}(k,m) z^{-d(m)}}$$

для $0 \le k <$ *NR_ALLPASS_BANDS*.

Матрица длины фрагментарной задержки, $Q_{Fract_allpass}(k,m)$ и вектор фрагментарной задержки, $\varphi_{Fract}(k)$ определяются выражением

$$Q_{Fract_allpass}(k,m) = \exp(-i\pi q(m) f_{center}(k)), \begin{cases} 0 \le k < NR_ALLPASS_BANDS \\ 0 \le m < NUM_OF_LINKS \end{cases},$$

и

$\varphi_{Fract}(k) = \exp(-i\pi q_\varphi f_{center}(k))$, $0 \le k <$ *NR_ALLPASS_BANDS*, где $i = \sqrt{-1}$ обозначает мнимую единицу. Постоянная длины фракционной задержки $q_\varphi = 0{,}39$.

Вектора коэффициентов фильтра $a(m)$ и вектора длины задержки $d(m)$ см. в таблице 39.

Таблица 39 — Векторы коэффициентов фильтра, векторы длины задержки $d_{24kHz}(m)$ и $d_{48kHz}(m)$

m	$a(m)$	$d(m)$
0	0,65143905753106	3
1	0,56471812200776	4
2	0,48954165955695	5

Таблица 40 — Вектор длины задержки f_{center_20}

k	$f_{center_20}(k)$	k	$f_{center_20}(k)$
0	–3/8	5	7/8
1	–1/8	6	5/4
2	1/8	7	7/4
3	3/8	8	9/4
4	5/8	9	11/4

Таблица 41 — Вектор длины задержки f_{center_34}

k	$f_{center_34}(k)$	k	$f_{center_34}(k)$
0	1/12	16	9/8
1	3/12	17	11/8
2	5/12	18	13/8
3	7/12	19	15/8
4	9/12	20	9/4
5	11/12	21	11/4

Окончание таблицы 41

k	$f_{center_34}(k)$	k	$f_{center_34}(k)$
6	13/12	22	13/4
7	15/12	23	7/4
8	17/12	24	17/4
9	–5/12	25	11/4
10	–3/12	26	13/4
11	–1/12	27	15/4
12	17/8	28	17/4
13	19/8	29	19/4
14	5/8	30	21/4
15	7/8	31	15/4

Вектор $g_{DecaySlope}$ содержит независимые от времени факторы для того, чтобы сделать всечастотный фильтр вариантом частотнозависимого. Это определяется выражением:

$$g_{DecaySlope}(k)\begin{cases}\max\langle 0{,}1 - DECAY_SLOPE\cdot(k - DECAY_CUTOFF)\rangle, & k - DECAY_CUTOFF\\ 1 & \text{для}\quad 0 \le k < NR_ALLPASS_BANDS.\end{cases}$$

Для верхних полос $NR_ALLPASS_BANDS \le k < NR_BANDS$ функция передачи $H_k(z)$ равняется задержке согласно:

$H_k(z) = z^{-D(k)}$, где $D(k)$ определяется

$$D(k) = \begin{cases}14, & NR_ALLPASS_BANDS \le k < SHORT_DELAY_BAND\\ 1, & SHORT_DELAY_BAND \le k < NR_BANDS\end{cases}$$

6.4.5.3 Обнаружение переходов

Чтобы обработать переходные процессы и другие быстрые огибающие времени, всечастотный фильтр должен быть ослаблен при этих сигналах. Это делается по следующей схеме:

сначала определяется матрица входной мощности $P(i, n)$, которая содержит сумму выборок поддиапазона *sub* в квадрате каждой полосы параметра

$$P(i,n) = \sum_{i=b(k)} |s_k(n)|^2,\ 0 \le I < NR_PAR_BANDS,$$

где $b(k)$ определяется в таблице 48 и таблице 49.

Применяется пиковое затухание к мощности входного сигнала согласно:

$$P_{PeakDecayNrg}(i,n) = \begin{cases}P(i,n), & \alpha P_{PeakDecayNrg}(i,n-1) < P(i,n)\\ \alpha P_{PeakDecayNrg}(i,n-1), & \text{в других случаях}\end{cases}$$

для $0 \le i < NR_PAR_BAND$. α является фактором пикового затухания.

Впоследствии входная мощность и сигналы пикового затухания мощности фильтруются с функцией передачи Z-домена

$H_{Smooth}(z)$:

$P_{SmoothNrg}(i,z) = H_{Smooth}(z)\cdot P(i, z)$,

$P_{SmoothPeakDecayDiffNrg}(i,z) = H_{Smooth}(z)\cdot P_{PeakDecayNrg}(i, z) - P(i, z))$,

для $0 \le I < NR_PAR_BANDS$, где

$$H_{Smooth}(z) = \frac{a_{Smooth}}{1 + (a_{Smooth} - 1)\cdot z^{-1}}.$$

Аттенюатор переходов $G_{TransientRatio}$ вычисляется следующим образом:

$$G_{TransientRatio}(i,n) = \begin{cases} \dfrac{P_{SmoothNrg}(i,n)}{\gamma \cdot P_{SmoothPeakDecayDiffNrg}(i,n)}, & \gamma \cdot P_{SmoothPeakDecayDiffNrg}(i,n) > P_{SmoothNrg}(i,n) \\ 1, & \text{в других случаях} \end{cases}$$

для $0 \le i < NR_PAR_BANDS$, где $\gamma = 1.5$ фактор воздействия перехода.

Наконец аттенюатор переходов $G_{TransientRatio}$ отображается в полосы согласно:

$$G_{TransientRatioMapped}(k,n) = G_{TransientRatio}(b(k),n),\ 0 \le k < NR_PAR_BANDS.$$

6.4.5.4 Примените сокращения для перехода к декоррелируемому сигналу

Пусть $d_k(z)$ будет декоррелируемым сигналом и $s_k(z)$ входным сигналом моно в Z-домене для каждой полосы. Тогда $d_k(z)$ определяется согласно:

$$d_k(z) = G_{TransientRatioMapped}(k,z) \cdot H_k(z) \cdot s_k(z),\ \text{где } 0 \le k < NR_PAR_BANDS.$$

$$f_{center_20}(k) = k + \frac{1}{2} - 7,\ 10 \le k < NR_ALLPASS_BANDS.$$

$$f_{center_34}(k) = k + \frac{1}{2} - 27,\ 32 \le k < NR_ALLPASS_BANDS.$$

$$f_{center} \begin{cases} f_{center_20}, & NR_PAR_BANDS = 20 \\ f_{center_34}, & NR_PAR_BANDS = 34. \end{cases}$$

Таблица 42 — Фракционный вектор длины задержки $q(m)$

(m)	$q(m)$
0	0,43
1	0,75
2	0,347

Таблица 43 — Пиковые факторы затухания α

α	0,76592833836465

6.4.6 Обработка стерео

Наборы выборок поддиапазона *sub* $s_k(n)$ и $d_k(n)$ обрабатываются согласно сигналам стерео. Эти сигналы определяются на полосу стерео. Все гибридные выборки поддиапазона в пределах полосы стерео обрабатываются согласно сигналам в этой соответствующей полосе стерео. Таблица 48 и таблица 49 указывают гибридные выборки поддиапазона, которые попадают в каждый стереодиапазон для конфигурации стереодиапазона (10,20) и 34. k проходит диапазон от [0 … 70] или [0 … 90] для конфигурации стереодиапазона (10,20) или 34, соответственно.

6.4.6.1 Отображение

Число полос стерео, которое фактически используется для обработки сигналов, зависит от числа *IID* и доступных параметров для *ICC* согласно отношению, данному в таблице 44. В случае, если никакие *IID* или параметры *ICC* не были переданы в текущем фрейме (*enable_iid* == %0 или *enable_icc* == %0), число *IID* или параметр *ICC*, соответственно, считается равным 20. В случае, если никакой *IID* и никакие параметры *ICC* не были переданы в текущем фрейме (*enable_iid* == %0 и *enable_icc* == %0), число полос стерео в предыдущем фрейме сохраняется неизменным и используется также для обработки текущего фрейма.

Таблица 44 — Число полос стерео зависит от числа параметров для *IID* и *ICC*

Число параметров *IID*	Число параметров *ICC*	Число полос стерео
10	10	20 (то есть конфигурация 10,20 полос стерео)
10	20	
20	10	
20	20	

Окончание таблицы 44

Число параметров *IID*	Число параметров *ICC*	Число полос стерео
10,20	34	34
34	10, 20	
34	34	

В случае, когда число параметров для *IID* и *ICC* отличается от числа полос стерео, требуется отображение из меньшего числа параметров в большее число параметров. Для отображения от 10 до 20 параметров это реализуется копированием каждого параметра как показано в таблице 45. Для отображения от 20 до 34 параметров это реализуется согласно таблице 45. Для отображения от 10 до 34 параметров сначала 10 параметров отображаются на 20 параметров, а потом на 34 параметра. Таблица 46 дает инверсное отображение от 34 параметров до 20.

Процесс усреднения, обозначенный $(idx_0 + idx_1)/2$ в таблице 45 и $(2*idx_0 + idx_1)/2$ в таблице 46, выполняется для представления целочисленного индекса idx_k параметров *IID* или *ICC* перед деквантизацией, согласно целочисленной арифметике *ANSI-C*.

Таблица 45 — Отображение от 10 до 20 параметров в 34

Сетка параметров			Сетка параметров		
34	20	10	34	20	10
idx_0	idx_0	idx_0	idx_{17}	idx_{11}	idx_5
idx_1	$(idx_0+idx_1)/2$	idx_0	idx_{18}	idx_{12}	idx_6
idx_2	idx_1	idx_0	idx_{19}	idx_{13}	idx_6
idx_3	idx_2	idx_1	idx_{20}	idx_{14}	idx_7
idx_4	$(idx_2+idx_3)/2$	idx_1	idx_{21}	idx_{14}	idx_7
idx_5	idx_3	idx_1	idx_{22}	idx_{15}	idx_7
idx_6	idx_4	idx_2	idx_{23}	idx_{15}	idx_7
idx_7	idx_4	idx_2	idx_{24}	idx_{16}	idx_8
idx_8	idx_5	idx_2	idx_{25}	idx_{16}	idx_8
idx_9	idx_5	idx_2	idx_{26}	idx_{17}	idx_8
idx_{10}	idx_6	idx_3	idx_{27}	idx_{17}	idx_8
idx_{11}	idx_7	idx_3	idx_{28}	idx_{18}	idx_9
idx_{12}	idx_8	idx_4	idx_{29}	idx_{18}	idx_9
idx_{13}	idx_8	idx_4	idx_{30}	idx_{18}	idx_9
idx_{14}	idx_9	idx_4	idx_{31}	idx_{18}	idx_9
idx_{15}	idx_9	idx_4	idx_{32}	idx_{19}	idx_9
idx_{16}	idx_{10}	idx_5	idx_{33}	idx_{19}	idx_9

Таблица 46 — Отображение параметров *IID*, *ICC*, *IPD* и *OPD* из 34 полос стерео в 20 полос стерео. Для параметров *IPD* и *OPD* это отображение применяется до и включая idx_{10} и idx_{16} для 20 и 34 полос стерео соответственно

20 полос стерео	34 полосы стерео
idx_0	$(2*idx_0+idx_1)/3$
idx_1	$(idx_1+2*idx_2)/3$
idx_2	$(2*idx_3+idx_4)/3$
idx_3	$(idx_4+2*idx_5)/3$

Окончание таблицы 46

20 полос стерео	34 полосы стерео
idx_4	$(idx_6+idx_7)/2$
idx_5	$(idx_8+idx_9)/2$
idx_6	idx_{10}
idx_7	idx_{11}
idx_8	$(idx_{12}+idx_{13})/2$
idx_9	$(idx_{14}+idx_{15})/2$
idx_{10}	idx_{16}
idx_{11}	idx_{17}
idx_{12}	idx_{18}
idx_{13}	idx_{19}
idx_{14}	$(idx_{20}+idx_{21})/2$
idx_{15}	$(idx_{22}+idx_{23})/2$
idx_{16}	$(idx_{24}+idx_{25})/2$
idx_{17}	$(idx_{26}+idx_{27})/2$
idx_{18}	$(idx_{28}+idx_{29}+idx_{30}+idx_{31})/4$
idx_{19}	$(idx_{32}+idx_{33})/2$

Параметры *IPD/OPD* следуют за отображением для параметров *IID*, принимая во внимание относительное количество параметров для *IPD/OPD*. Следовательно для *IPD/OPD* применяется то же самое отображение как для *IID*, но только для более низкого числа параметров. Для верхних полос стерео, где никакие данные *IPD/OPD* не передаются, параметры *IPD/OPD* обнуляются.

Если число полос стерео изменяется от 10, 20 в предыдущем фрейме к 34, коэффициенты $h_{11}(b)$, $h_{12}(b)$, $h_{21}(b)$ и $h_{22}(b)$ в конце предыдущего фрейма отображаются от 20 до 34 полос стерео согласно таблице 45 (путем замены idx_b на $h_{ij}(b)$, где ij равно 11, 12, 21 или 22). Коэффициенты $H_{11}(k,n)$, $H_{12}(k,n)$, $H_{21}(k,n)$ и $H_{22}(k,n)$ для конца предыдущего фрейма извлекаются согласно четырем уравнениям, приведенным в 6.4.6.3 перед дальнейшей обработкой, как определено в 6.4.6.4. Параметры состояния сглаживания *IPD/OPD* сбрасываются, то есть $opd(b, n_{e-1})=0$, $ipd(b, n_{e-1})=0$, $opd(b, n_e)=0$ и $ipd(b, n_e)=0$. Разрешающая способность по частоте гибридного блока фильтров анализа *QMF* изменяется на конфигурацию 34 полосы стерео. Параметры состояния процесса декорреляции сбрасываются в нуль.

Если число полос стерео изменяется от 34 в предыдущем фрейме к 10, 20 в текущем фрейме, коэффициенты $h_{11}(b)$, $h_{12}(b)$, $h_{21}(b)$ и $h_{22}(b)$ в конце предыдущего фрейма отображаются от 34 до 20 полос стерео согласно таблице 46 (заменяя idx_b на $hij(b)$, где ij равно 11, 12, 21 или 22). Коэффициенты $H_{11}(k,n)$, $H_{12}(k,n)$, $H_{21}(k,n)$ и $H_{22}(k,n)$ для конца предыдущего фрейма извлекаются согласно четырем уравнениям, приведенным в 6.4.6.3 перед дальнейшей обработкой, как определено в 6.4.6.4. Переменные состояния сглаживания *IPD/OPD* сбрасываются, то есть, $opd(b,n_{e-1})=0$, $ipd(b,n_{e-1})=0$, $opd(b,n_e)=0$ и $ipd(b,n_e)=0$. Разрешающая способность по частоте гибридного блока фильтров анализа *QMF* изменяется на конфигурацию 20 полос стерео. Параметры состояния процесса декорреляции сбрасывается, чтобы обнулиться (таблица 47).

Таблица 47 — Изменение числа полос стерео

Предыдущий фрейм	Текущий фрейм	
	10/20 полос	34 полосы
10/20 полос	—	Отображение $h_{ij}(b)$ согласно таблицы 45, сброс параметров состояния
34 полосы	Отображение $h_{ij}(b)$ согласно таблицы 46, сброс параметров состояния	—

6.4.6.2 Смешивание

Чтобы генерировать сигналы поддиапазона *QMF* для выборок поддиапазона $n = n_e + 1 \ldots n_{e+1}$ требуются параметры в положении n_e и n_{e+1}, а также сигналы как домена поддиапазона $s_k(n)$ и $d_k(n)$ для $n = n_e + 1 \ldots n_{e+1}$. Для *IPD/OPD* дополнительно необходимы параметры в позиции n_{e-1}. n_e представляет стартовую позицию для огибающей e. В случае e=0, n_{-1} представляет позицию последнего параметра предыдущего фрейма стерео. Для первого фрейма стерео параметры *IPD/OPD* в позиции n_{-1} инициализируются в нуль. В случае класса фрейма == %1 (*VAR_BORDERS*), позиции границы n_e получаются с помощью *border_position* [e]. В случае класса фрейма == % 0 (*FIX_BORDERS*), позиции границы n_e получаются посредством формулы

$$n_e \left\lfloor \frac{numQMFSlots * (e \div 1)}{num_env} \right\rfloor - 1, \; e = [0, \ldots, num_env\text{-}1]$$

Сигналы поддиапазона *sub* стерео создаются как:

$l_k(n)=H_{11}(k,n)s_k(n)+H_{21}(k,n)d_k(n)$

$r_k(n)=H_{11}(k,n)s_k(n)+H_{21}(k,n)d_k(n)$

Чтобы получить матрицы $H_{11}(k,n)$, $H_{12}(k,n)$, $H_{21}(k,n)$ и $H_{22}(k,n)$ сначала должны быть вычислены векторы $h_{11}(b)$, $h_{12}(b)$, $h_{21}(b)$, $h_{22}(b)$, где параметр b используется в качестве индекса параметра. Сначала преобразуются к линейному домену различия в интенсивности (*IID*) для позиции параметра n_{e+1}

$$c(b) \div 10^{\frac{iid(b)}{20}},$$

где *iid* (b) представляет декодированное значение *IID* для полосы стерео b в дБ. В зависимости от конфигурации режима *ICC* используются смешивающая процедура R_a или R_b. Для обеих смешивающих процедур используются параметры для позиции параметра n_{e+1}.

6.4.6.2.1 Смешивающая процедура R_a

В случае использования смешивающей процедуры R_a применяется следующий метод.

Из различий в интенсивности вычисляются два вектора масштабного коэффициента c_1 и c_2

$$c_1(b) \div \frac{\sqrt{2}}{\sqrt{1 + c^2(b)}}$$

$$c_2(b) \div \frac{\sqrt{2c(b)}}{\sqrt{1 + c^2(b)}}.$$

Исходя из них и параметра *ICC* $\rho(b)$ вычисляются коэффициенты $h_{xy}(b)$ согласно

$$\alpha(b) \div \frac{1}{2}\arccos(\rho(b))$$

$$\beta(b) = \alpha(b)\frac{c_1(b) - c_2(b)}{\sqrt{2}}$$

$$h_{11}(b) = \cos(\alpha(b) + \beta(b))c_2(b)$$

$$h_{12}(b) = \cos(\beta(b) - \alpha(b))c_1(b)$$

$$h_{21}(b) = \sin(\alpha(b) + \beta(b))c_2(b)$$

$$h_{22}(b) = \sin(\beta(b) - \alpha(b))c_1(b)$$

6.4.6.2.2 Процедура смешивания *Rb*

В случае, когда используется процедура смешивания R_b, применяется следующий метод.

Чтобы предотвратить неустойчивость, в случае значения $\rho(b)$ меньше чем 0,05, $\rho(b)$ устанавливается в 0,05. В случае если $c(b)$ не равно 1

$$\alpha(b) \div \frac{1}{2}\arctan\left(\frac{2c(b)\rho(b)}{c^2(b) - 1}\right),$$

иначе $\alpha(b) = \frac{\pi}{4}$. После исправления по модулю $a(b)$ значения c (b) и ρ (b) снова используются, чтобы получить коэффициенты h_{xy} (b).

$$\alpha(b) = \alpha(b) - \left\lfloor \frac{\alpha(b)}{\frac{1}{2}\pi} \right\rfloor \frac{1}{2}\pi,$$

$$\mu(b) = 1 + \frac{4\rho^2(b) - 4}{(c(b) + c^{-1}(b))^2},$$

$$\gamma(b) = \arctan\left(\sqrt{\frac{1-\sqrt{\mu(b)}}{1+\sqrt{\mu(b)}}} \right),$$

$$h_{11}(b) = \sqrt{2}\cos(\alpha(b))\cos(\gamma(b)),$$

$$h_{12}(b) = \sqrt{2}\sin(\alpha(b))\cos(\gamma(b)),$$

$$h_{21}(b) = -\sqrt{2}\sin(\alpha(b))\sin(\gamma(b)),$$

$$h_{22}(b) = \sqrt{2}\cos(\alpha(b))\sin(\gamma(b)).$$

6.4.6.3 Фазовые параметры

6.4.6.3.1 Фазовые параметры заблокированы

Если *IPD* и *OPD* заблокированы (*enable_ipdopd* == 0), применяется следующая процедура. Чтобы получить $H_{11}(k, n_{e+1})$, $H_{12}(k, n_{e+1})$, $H_{21}(k, n_{e+1})$ и $H_{22}(k, n_{e+1})$, используем следующие уравнения:

$H_{11}(k, n_{e+1}) = h_{11}(b(k))$
$H_{12}(k, n_{e+1}) = h_{12}(b(k))$
$H_{21}(k, n_{e+1}) = h_{21}(b(k))$
$H_{22}(k, n_{e+1}) = h_{22}(b(k))$,

где $b(k)$ определяется в таблице 48 и таблице 49.

Т а б л и ц а 48 — Отображение параметров от 20 полос до 71 поддиапазона

Индекс k поддиапазона	Канал *QMF*	Индекс параметра $b(k)$	Примечание
0	0	1*	*Sub QMF*
1	0	0*	
2	0	0	
3	0	1	
4	0	2	
5	0	3	
6	1	4	
7	1	5	
8	2	6	
9	2	7	
10	3	8	*QMF* (только)
11	4	9	
12	5	10	
13	6	11	
14	7	12	
15	8	13	
16—17	9—10	14	
18—20	11—13	15	

Окончание таблицы 48

Индекс *k* поддиапазона	Канал *QMF*	Индекс параметра *b*(*k*)	Примечание
21—24	14—17	16	
25—29	18—22	17	
30—41	23—34	18	
42—70	35—63	19	

Таблица 49 — Отображение параметров из 34 полос в 91 поддиапазон *sub*

Индекс *k* поддиапазона	Канал *QMF*	Индекс параметра *b* (*k*)	Примечание
0	0	0	*Sub QMF*
1	0	1	
2	0	2	
3	0	3	
4	0	4	
5	0	5	
6—7	0	6	
8	0	7	
9	0	2*	
10	0	1*	
11	0	0*	
12—13	1	10	
14	1	4	
15	1	5	
16	1	6	
17	1	7	
18	1	8	
19	1	9	
20	2	10	
21	2	11	
22	2	12	
23	2	9	
24	3	14	
25	3	11	
26	3	12	
27	3	13	
28	4	14	
29	4	15	
30	4	16	
31	4	13	
32	5	16	*QMF* (только)
33	6	17	
34	7	18	
35	8	19	
36	9	20	

Окончание таблицы 49

Индекс k поддиапазона	Канал *QMF*	Индекс параметра b (k)	Примечание
37	10	21	
38—39	11—12	22	
40—41	13—14	23	
42—43	15—16	24	
44—45	17—18	25	
46—47	19—20	26	
48—50	21—23	27	
51—53	24—26	28	
54—56	27—29	29	
57—59	30—32	30	
60—63	33—36	31	
64—67	37—40	32	
68—90	41—63	33	

6.4.6.3.2 Фазовые параметры задействованы

В случае, если *IPD* и *OPD* задействованы (*enable_ipdopd* == 1), применяется следующая процедура. Сначала значения *IPD* и *OPD* сглаживаются во времени согласно

$$\varphi_{opd}(b) = \left\{ \left[\frac{1}{4}\exp(j \cdot opd(b,e-1)) + \frac{1}{2}\exp(j \cdot opd(b,e)) + \exp(j \cdot opd(b,e+1)) \right] \right\},$$

$$\varphi_{opd}(b) = \left\{ \left[\frac{1}{4}\exp(j \cdot ipd(b,e-1)) + \frac{1}{2}\exp(j \cdot ipd(b,e)) + \exp(j \cdot ipd(b,e+1)) \right] \right\}.$$

В случае, когда число параметров *IPD/OPD* для позиции параметра n_{e-1} и/или n_e отличается от числа параметров *IPD/OPD* для позиции параметра n_{e+1}, они отображаются в число параметров *IPD/OPD* для позиции параметра n_{e+1}, используя таблицу 45 и таблицу 46.

$\varphi_1(b) = \varphi_{opd}(b)$,

$\varphi_2(b) = \varphi_{opd}(b) - \varphi_{ipd}(b)$.

Векторы $h_{11}(b)$, $h_{12}(b)$, $h_{21}(b)$, $h_{22}(b)$ изменяются согласно:

$h_{11}(b) = h_{11}(b) \cdot \exp(j\varphi_1(b))$

$h_{12}(b) = h_{12}(b) \cdot \exp(j\varphi_2(b))$

$h_{21}(b) = h_{21}(b) \cdot \exp(j\varphi_1(b))$

$h_{22}(b) = h_{22}(b) \cdot \exp(j\varphi_2(b))$.

Чтобы получить $H_{11}(k, n_{e+1})$, $H_{12}(k, n_{e+1})$, $H_{21}(k, n_{e+1})$ и $H_{22}(k, n_{e+1})$ используются следующие уравнения

$H_{11}(k, n_{e+1}) = h_{11}(b(k))$

$H_{12}(k, n_{e+1}) = h_{12}(b(k))$

$H_{21}(k, n_{e+1}) = h_{21}(b(k))$

$H_{22}(k, n_{e+1}) = h_{22}(b(k))$,

где $b(k)$ определяется в таблице 48 и таблице 49. Для индексов, обозначенных a^* используются следующие уравнения:

$H_{11}(k, n_{e+1}) = h_{11}{}^*(b(k))$

$H_{12}(k, n_{e+1}) = h_{12}{}^*(b(k))$

$H_{21}(k, n_{e+1}) = h_{21}{}^*(b(k))$

$H_{22}(k, n_{e+1}) = h_{22}{}^*(b(k))$,

где $h_{xx}{}^*$ представляет комплексно сопряженным с h_{xx}.

6.4.6.4 Интерполяция

Промежуточные значения для $H_{11}(k, n)$, $H_{12}(k, n)$, $H_{21}(k, n)$ и $H_{22}(k, n)$ в позициях $n = n_e + 1 \ldots n_{e+1}$ получаются посредством линейной интерполяции, соответствующей

$$H_{11}(k, n) = H_{11}(k, n_e) + (n - n_e)\frac{H_{11}(k,n_{e+1}) - H_{11}(k,n_e)}{n_{e+1} - n_e}$$

$$H_{12}(k, n) = H_{12}(k, n_e) + (n - n_e)\frac{H_{12}(k,n_{e+1}) - H_{12}(k,n_e)}{n_{e+1} - n_e}$$

$$H_{21}(k, n) = H_{21}(k, n_e) + (n - n_e)\frac{H_{21}(k,n_{e+1}) - H_{21}(k,n_e)}{n_{e+1} - n_e}$$

$$H_{22}(k, n) = H_{22}(k, n_e) + (n - n_e)\frac{H_{22}(k,n_{e+1}) - H_{22}(k,n_e)}{n_{e+1} - n_e}.$$

Особые случаи:

а) для первой области фрейма стерео с $n = 0 ... n_0 - 1$ применяется следующее:

$$H_{11}(k, n) = H_{11}(k, n_{-1}) + n\frac{H_{11}(k,n_0) - H_{11}(k,n_{-1})}{n_0}$$

$$H_{12}(k, n) = H_{12}(k, n_{-1}) + n\frac{H_{12}(k,n_0) - H_{12}(k,n_{-1})}{n_0}$$

$$H_{21}(k, n) = H_{21}(k, n_{-1}) + n\frac{H_{21}(k,n_0) - H_{21}(k,n_{-1})}{n_0}$$

$$H_{22}(k, n) = H_{22}(k, n_{-1}) + n\frac{H_{22}(k,n_0) - H_{22}(k,n_{-1})}{n_0},$$

где $H_{xx}(k, n_{-1})$ представляет коэффициенты $H_{xx}(k, n_{num_evn-1})$, полученные из предыдущего фрейма стерео.

Для первого фрейма стерео $H_{xx}(k, n_{-1})$ инициализируется в нуль,

б) для последней области фрейма стерео с $n = (k, n_{num_evn-1}) ...$ *numQMFSlots*–1 применяется следующее:

$H_{11}(k, n) = H_{11}(k, n_{num_env-1})$
$H_{12}(k, n) = H_{12}(k, n_{num_env-1})$
$H_{21}(k, n) = H_{21}(k, n_{num_env-1})$
$H_{22}(k, n) = H_{22}(k, n_{num_env-1})$.

6.4.6.5 Процедура для неполных наборов параметров

В случае, если никакие параметры не были переданы в текущем фрейме ни для *IID*, *ICC*, ни для *IPD/OPD* или их комбинации, значения параметров для текущего фрейма получаются согласно переменной *num_env*, как даётся в таблицах 50, 51 и 52.

Таблица 50 — Источник параметров для *IID* в случае отсутствия передачи параметров

	enable_iid	
	0	1
num_env=0	Набор параметров *IID*, чтобы присвоить значение по умолчанию	Сохраненные параметры *IID*
num_env > 0	Набор параметров *IID*, чтобы присвоить значение по умолчанию	Не определено

Таблица 51 — Источник параметров для *ICC* в случае отсутствия передачи параметров

	enable_icc	
	0	1
num_env=0	Набор параметров *ICC*, чтобы присвоить значение по умолчанию	Сохраненные параметры *ICC*
num_env > 0	Набор параметров *ICC*, чтобы присвоить значение по умолчанию	Не определено

В случае, если параметры должны быть установлены в значение по умолчанию, параметры в позициях, определенных n_e, устанавливаются в их значения по умолчанию (*index*=0).

Таблица 52 — Источник параметров для *IPD/OPD* в случае отсутствия передачи параметров

	enable_ipdopd	
	0	1
num_env=0	Набор параметров *IPD/OPD*, чтобы присвоить значение по умолчанию	Сохраненные параметры *IPD/OPD*
num_env > 0	Набор параметров *IPD/OPD*, чтобы присвоить значение по умолчанию	Не определено

В случае, если параметры должны быть сохранены, различают две ситуации. Если *enable_ipdopd* == == %1, четыре вектора $H_{11}(k, n)$, $H_{12}(k, n)$, $H_{21}(k, n)$ и $H_{22}(k, n)$ для всего n = [0, …, *numQMFSlots*–1] копируются из тех же самых четырех векторов в позиции *n*=*numQMFSlots*–1 в предыдущем элементе *ps_data* (). Если *enable_ipdopd* == %0, четыре вектора $H_{11}(k, n)$, $H_{12}(k, n)$, $H_{21}(k, n)$ и $H_{22}(k, n)$ для всего n = [0, …, *numQMFSlots*-1] устанавливаются в четыре вектора $h_{11}(b(k))$, $h_{12}(b(k))$, $h_{21}(b(k))$ и $h_{22}(b(k))$, соответственно, где *n*=*numQMFSlots*–1 в предыдущем элементе *ps_data* ().

6.4.7 Гибридный блок фильтров синтеза *QMF*

Гибридные сигналы поддиапазона обработки стерео $l_k(n)$ и $r_k(n)$ подаются в гибридные блоки фильтров синтеза, которые реализуются как сумматоры выборок *sub QMF*. Два блока фильтров синтеза идентичны блоку фильтров синтеза 64 комплексных *QMF*. Вводом в блок фильтров являются слоты 64 выборок *QMF*. Для каждого слота блок фильтров выводит один блок из 64 выборок одного канала реконструируемого сигнала стерео.

6.5 Ситуации запуска/остановки для декодирования

Декодирование выборки должно запускаться и заканчиваться определенным образом. Этот подпункт объясняет, как поступать с запуском и остановкой процесса декодирования.

6.5.1 Запуск декодирования

Запуск декодирования происходит для первого фрейма выборки или во время произвольного доступа в выборке.

Для синусоид предыдущий (несуществующий) подфрейм должен быть заполнен сигналом нуля. Тогда методом перекрытия-добавления генерируется естественное постепенное появление синусоидальных компонентов первого подфрейма.

Для шума предыдущий (несуществующий) подфрейм должен быть заполнен сигналом нуля. Метод перекрытия-добавления тогда генерирует естественное постепенное появление шумового компонента первого подфрейма.

Совместимый декодер, который получает данные *PS*, должен вывести моносигнал в два выходных канала, пока не будет принят первый элемент *ps_data* () с *enable_ps_header* == 1, в котором для всех задействованных параметров используется дифференциальное кодирование частоты и *num_env*> 0, гарантируя, что данные *PS* могут правильно декодироваться.

6.5.2 Прекращение декодирования

Прекращение декодирования происходит для последнего фрейма выборки, или во время произвольного доступа в выборке (прекращение процесса декодирования «вручную» (например, остановка, пропуск, пауза)).

Для шага переходного процесса не требуются никакие специальные предосторожности. Для переходного процесса Мейкслера возможно, что хвостовая часть не закончилась в конце выборки. Рекомендуется прекратить генерацию вывода для переходного процесса Мейкслера в конце выборки.

Для синусоид следующий (несуществующий) подфрейм должен быть заполнен сигналом нуля. Метод перекрытия-добавления тогда генерирует естественное постепенное исчезновение для синусоидальных компонентов последнего подфрейма.

Для шума следующий (несуществующий) подфрейм должен быть заполнен сигналом нуля. Метод перекрытия-добавления используется, когда генерирует естественное постепенное исчезновение для шумового компонента последнего подфрейма.

Приложение А
(обязательное)

Комбинация инструмента *SBR* с параметрическим инструментом стерео

А.1 Краткий обзор

Параметрический инструмент кодирования стерео (инструмент *PS*) может использоваться в комбинации с инструментом *SBR*. В этом случае аудиосигнал с первого канала передается в *AAC+SBR* (то есть *HE-AAC*), и инструмент *PS* используется, чтобы восстановить сигнал второго стерео канала от этого монофонического сигнала. Элемент потока битов *ps_data* () передает информацию, необходимую инструменту *PS*, и переносится в *sbr_extension* () контейнер потока битов *SBR*.

Использование этого параметрического расширения стерео *HE AAC* сообщается или неявно присутствием параметрических данных стерео в потоке битов, или явно сигнализируя соответствующий *AudioObjectType* в *audioSpecificConfig*. Следовательно, неявная сигнализация требует, если *sbr_extension* () с *bs_extension_id* == *EXTENSION_ID_PS* находится в части *SBR* потока битов, декодер, поддерживающий комбинацию *SBR* и *PS*, должен управлять инструментом *PS*, чтобы генерировать выходной сигнал стерео. Если никакой элемент *ps_data* () не доступен в части *SBR* монофонического потока битов *HE-AAC*, нормальный монофонический сигнал сгенерирован инструментом *SBR* и отображен на выходной сигнал стерео, в котором левый и правый канал содержат тот же самый монофонический сигнал.

А.2 Синтаксис потока битов и семантика

Элемент потока битов *ps_data* () переносят в *sbr_extension* () контейнер, обеспеченный потоком битов *SBR*.

Таблица А.1 — Синтаксис *sbr_extension* ()

Синтаксис	Количество битов	Мнемосхема

sbr_extension (*bs_extension_id*, *num_bits_left*)
{
switch (*bs_extension_id*) {
case EXTENSION_ID_PS:
num_bits_left - = *ps_data* ();
break;
default:
bs_fill_bits;
num_bits_left = 0;
break;
}
}
	num_bits_left	Примечание
		bslbf

Примечание — *ps_data* () возвращает число чтения битов.		

А.3 Процесс декодирования

Семантика и декодирующий процесс для инструмента *PS* определяется в 5.2 и 6.4. Когда инструмент *PS* объединяется с *SBR*, фрейм стерео идентичен фрейму *SBR* и состоит из 32 сложных выборок на полосу *QMF* для 1024 структурированной *AAC* (30 выборок для 960 структурированной).

Чтобы позволить эффективную реализацию инструмента *PS*, частичный сброс *decorrelator* параметров состояния выполняется для каждого фрейма стерео для всех поддиапазонов *QMF* выше самого высокого поддиапазона *QMF*, сгенерированного инструментом *SBR*

$d_k(n)=0$

$s_k(n)=0$,

где $n<n_e$, $k_{max} \le k < NR_BANDS$, n_e является первой выборкой в текущем фрейме стерео

и

$$k_{max} = k_x + M + \begin{cases} 7, & 10 \text{ или } 20 \text{ полос стерео} \\ 27, & 34 \text{ полосы стерео} \end{cases}$$

Если никакой *ps_data* () элемент не присутствовал в предыдущем фрейме, то полный сброс *decorrelator* параметров состояния выполняется, вызывая состояния:

$d_k(n)$=0

$s_k(n)$=0,

где $n<n_e$, $0 \le k < NR_BANDS$, и n_e является первой выборкой в текущем фрейме стерео.

Инструмент *PS* использует оцененное комплексу представление *QMF*, и поэтому не может использоваться в комбинации с низкой версией питания инструмента *SBR*. Если *DRC* используется в комбинации с *SBR*, то *DRC* применяется в домене *QMF* к выводу инструмента *PS* до синтеза *QMF filterbanks*. Тот же самый *factor*(*k, l*) применяется и к левому и к правому звуковому каналу.

А.4 Базовая версия параметрического инструмента кодирования стерео

Чтобы облегчить реализацию инструмента декодера *PS* на платформах с очень ограниченными вычислительными ресурсами, используется базовая версия инструмента *PS*. Декодер *PS*, реализовывая эту базовую версию, всегда использует гибридную структуру фильтра для 20 полос стерео и не реализует синтез *IPD/OPD* и смешивающийся режим *Rb*. Это приводит к сокращению вычислительной сложности приблизительно на 25 % по сравнению с неограниченным инструментом *PS*. Базовая версия инструмента *PS* поддерживает полный синтаксис потока битов для *ps_data* (). Однако, данные *IPD/OPD* игнорируются и сбрасываются к *IPD=OPD*=0 до синтеза стерео. Если 34 конфигурации полосы стерео используются для *IID* или параметров *ICC* в потоке битов, декодируемые параметры отображаются на 20 полос стерео. Базовый декодер всегда использует смешивающий *Pa* режим независимо от значения *icc_mode*.

Приложение Б
(обязательное)

Нормативные таблицы

Б.1 Таблицы Хаффмана для *SSC*

Функция *ssc_huff_dec* () используется как:
date = ssc_huff_dec (*t_huff, codeword*),
где *t_huff* является выбранной таблицей Хаффмана, и *codeword* является словом, считанным из потока битов.

Таблица Б.1 — *huff_sgrid*

Индекс	*huff_sgrid*	Индекс	*huff_sgrid*
0	100001	11	0101
1	11101	12	0000
2	11110	13	0001
3	1100	14	11100
4	1101	15	01001
5	1010	16	111111
6	0111	17	111110
7	001	18	100000
8	1011	19	010001
9	0110	20	010000
10	1001	21	10001

Таблица Б.2 — *huff_sampba*

Индекс	*huff_sampba*	Индекс	*huff_sampba*	Индекс	*huff_sampba*	Индекс	*huff_sampba*
0	110010010	64	1101	128	0110	192	110010011001
8	0100111	72	001	136	11000	200	110010011000100
16	1100101	80	000	144	01000	208	110010011000101
24	110011	88	1111	152	010010	216	110010011000110
32	01110	96	1110	160	0100110	224	110010011000111
40	01111	104	1011	168	11001000	232	11001001100000
48	0101	112	1010	176	1100100111	240	11001001100001
56	1001	120	1000	184	11001001101		

Таблица Б.3 — *huff_sampbr*

Индекс	*huff_sampbr*	Индекс	*huff_sampbr*	Индекс	*huff_sampbr*
–240	111111110110001000010	–184	111111110110001001001	–128	111111110110001010000
–232	111111110110001000011	–176	111111110110001001010	–120	111111110110100
–224	111111110110001000100	–168	111111110110001001011	–112	11111111011001
–216	111111110110001000101	–160	111111110110001001100	–104	1111111111100
–208	111111110110001000110	–152	111111110110001001101	–96	111111110111
–200	111111110110001000111	–144	111111110110001001110	–88	11111111110
–192	111111110110001001000	–136	111111110110001001111	–80	1111111100

Окончание таблицы Б.3

Индекс	*huff_sampbr*	Индекс	*huff_sampbr*	Индекс	*huff_sampbr*
–72	111101101	40	11111110	152	11111111010001010101
–64	11110111	48	111101100	160	11111111010001010110
–56	1111010	56	1111111110	168	11111111010001010111
–48	111100	64	11111111010	176	11111111010001011000
–40	111110	72	111111111111	184	11111111010001011001
–32	11101	80	1111111111101	192	11111111010001011010
–24	0111	88	11111111011011	200	11111111010001011011
–16	010	96	1111111101100011	208	11111111010001011100
–8	00	104	111111110110101	216	11111111010001011101
0	10	112	111111110110000	224	11111111010001011110
8	110	120	11111111010001010001	232	11111111010001011111
16	0110	128	11111111010001010010	240	11111111011000100000
24	11100	136	11111111010001010011		
32	1111110	144	11111111010001010100		

Таблица Б.4 — *huff_sampca*

Индекс	*huff_sampca*	Индекс	*huff_sampca*	Индекс	*huff_sampca*
0	01101101011	88	000	176	0011001
8	01101100	96	1111	184	011011011
16	0011000	104	1110	192	0110110100
24	0110111	112	1100	200	011011010101
32	011010	120	1011	208	0110110101001
40	00111	128	1000	216	01101101010001
48	10011	136	0101	224	0110110101000010
56	0100	144	0010	232	0110110101000011
64	0111	152	10010	240	011011010100000
72	1010	160	01100		
80	1101	168	001101		

Таблица Б.5 — *huff_sampcr* [0]

Индекс	*huff_sampcr* [0]	Индекс	*huff_sampcr* [0]	Индекс	*huff_sampcr* [0]
–26	111001010111000	–16	0110101110	–6	111000
–25	01101101110	–15	1110010100	–5	00101
–24	1110010101111	–14	000111000	–4	11101
–23	000111001011	–13	011010110	–3	0011
–22	011010111110	–12	00011011	–2	1111
–21	111001010100	–11	01101010	–1	110
–20	00011100100	–10	11100100	0	10
–19	01101011110	–9	0001111	1	010
–18	11100101100	–8	0110111	2	0111
–17	0001110110	–7	001000	3	0000

Окончание таблицы Б.5

Индекс	*huff_sampcr* [0]	Индекс	*huff_sampcr* [0]	Индекс	*huff_sampcr* [0]
4	01100	12	011011010	20	111001010110
5	00010	13	000111010	21	111001010101
6	001001	14	1110010111	22	011010111111
7	1110011	15	0110110110	23	000111001010
8	0110100	16	0001110011	24	11100101011101
9	0001100	17	11100101101	25	00011101111
10	01101100	18	01101101111	26	111001010111001
11	00011010	19	00011101110		

Таблица Б.6 — *huff_sampcr* [1]

Индекс	*huff_sampcr* [1]	Индекс	*huff_sampcr* [1]	Индекс	*huff_sampcr* [1]
–26	100111001011	–8	110110	10	11011111
–24	11011100010	–6	11010	12	10011101
–22	110111101110	–4	1100	14	110111001
–20	11011110110	–2	111	16	1101111010
–18	1101110000	0	0	18	1001110011
–16	100111000	2	101	20	11011100011
–14	110111100	4	1000	22	10011100100
–12	11011101	6	10010	24	110111101111
–10	1001111	8	100110	26	100111001010

Таблица Б.7 — *huff_sampcr* [2]

Индекс	*huff_sampcr* [2]	Индекс	*huff_sampcr* [2]	Индекс	*huff_sampcr* [2]
–28	01011000101	–8	0100	12	0101110
–24	010110000	–4	00	16	01011010
–20	010110011	0	1	20	010110010
–16	01011011	4	011	24	0101100011
–12	0101111	8	01010	28	01011000100

Таблица Б.8 — *huff_sampcr* [3]

Индекс	*huff_sampcr* [3]	Индекс	*huff_sampcr* [3]	Индекс	*huff_sampcr* [3]
–32	00010101	–8	01	16	00011
–24	000100	0	1	24	0001011
–16	0000	8	001	32	00010100

Таблица Б.9 — *huff_sfreqba*

Индекс	*huff_sfreqba*	Индекс	*huff_sfreqba*
0	1011111100100101100100	24	101111110100101100111
8	101111110100101100101	32	101111110100101101000
16	1011111110100101100110	40	101111110100101101001

Продолжение таблицы Б.9

Индекс	huff_sfreqba	Индекс	huff_sfreqba
48	101111110100101101010	384	011101010
56	101111110100101101011	392	011101011
64	101111110100101101100	400	00010100
72	101111110100101101101	408	010111100
80	101111110100101101110	416	111100011
88	101111110100101101111	424	110000001
96	101111110100101110000	432	010111101
104	101111110100101110001	440	011101100
112	101111110100101110010	448	011101101
120	101111110100101110011	456	1101001100
128	101111110100101110100	464	1010101001
136	10111111010011	472	100111100
144	101111110100101110101	480	00011111
152	101111110101	488	010111110
160	10111111011	496	1111101101
168	100010000	504	111010000
176	001010010	512	110000010
184	01000010	520	00000100
192	101111111	528	00100000
200	111001101	536	111010001
208	00010001	544	111100100
216	111001110	552	111010010
224	111100001	560	110000011
232	101010101	568	01000100
240	010000011	576	10010101
248	1111101001	584	01010001
256	111100010	592	10101001
264	111001111	600	101010110
272	100111011	608	111100101
280	010111011	616	011101110
288	1111101010	624	00000101
296	100010001	632	10001001
304	0111000011	640	01100000
312	001010011	648	01100001
320	1111101011	656	101010111
328	000100100	664	100111101
336	000100101	672	111010011
344	1111101100	680	101011000
352	000100110	688	10101101
360	110000000	696	10000010
368	00110110	704	110100111
376	010000110	712	00000110

Продолжение таблицы Б.9

Индекс	*huff_sfreqba*	Индекс	*huff_sfreqba*
720	100010100	1056	10111000
728	001010100	1064	10111001
736	110101000	1072	11010110
744	010111111	1080	11011101
752	01010010	1088	111010101
760	011000100	1096	100111111
768	011101111	1104	101000000
776	111100110	1112	101011111
784	110101001	1120	110000110
792	111110111	1128	110111110
800	110000100	1136	00000111
808	111010100	1144	101000001
816	011000101	1152	010000111
824	1010110010	1160	101100000
832	111100111	1168	001010101
840	01110001	1176	110111111
848	11001110	1184	00111001
856	01110010	1192	111111001
864	011110000	1200	110000111
872	110111001	1208	101000010
880	110101010	1216	111000000
888	101011100	1224	101100001
896	00101011	1232	101100010
904	01000101	1240	111000001
912	11001111	1248	111000010
920	01100011	1256	01010011
928	110111100	1264	00111010
936	011001000	1272	111000011
944	011001001	1280	1101001101
952	011001010	1288	111101000
960	110111101	1296	110101011
968	101011101	1304	111000100
976	000100111	1312	01010100
984	00110111	1320	111111010
992	01100110	1328	11010000
1000	10000011	1336	01100111
1008	00111000	1344	01101000
1016	011110001	1352	101000011
1024	110000101	1360	00001000
1032	101011110	1368	11000100
1040	1111110000	1376	10001011
1048	100111110	1384	011110010

Продолжение таблицы Б.9

Индекс	huff_sfreqba	Индекс	huff_sfreqba
1392	111101001	1728	00100001
1400	10001100	1736	00001011
1408	111101010	1744	00100010
1416	01010101	1752	01000111
1424	111000101	1760	00001100
1432	00001001	1768	111011000
1440	111010110	1776	11000110
1448	111111011	1784	11010010
1456	10100010	1792	00010110
1464	11111000	1800	01001000
1472	01101001	1808	101001000
1480	01010110	1816	110001011
1488	01101010	1824	01001001
1496	10001101	1832	10111010
1504	01101011	1840	01111100
1512	110101110	1848	000101110
1520	10010110	1856	00001101
1528	10010111	1864	10000100
1536	01010111	1872	01011010
1544	01000110	1880	00101101
1552	111000110	1888	01001010
1560	01101100	1896	111011001
1568	00001010	1904	111111101
1576	01111010	1912	10001111
1584	00101100	1920	10011001
1592	111101011	1928	10100101
1600	00111011	1936	101100011
1608	01110011	1944	110001110
1616	01111011	1952	111111110
1624	11010001	1960	10000101
1632	111111100	1968	00101110
1640	100010101	1976	10000110
1648	01011000	1984	00100011
1656	10011000	1992	11100100
1664	100011100	2000	10011010
1672	10100011	2008	00101111
1680	110001010	2016	111101100
1688	0001110	2024	01111101
1696	00010101	2032	01110100
1704	10110010	2040	111011010
1712	111010111	2048	10111011
1720	01011001	2056	10011011

Продолжение таблицы Б.9

Индекс	*huff_sfreqba*	Индекс	*huff_sfreqba*
2064	10010000	2400	110010010
2072	00110000	2408	00110011
2080	10111100	2416	10000111
2088	01011011	2424	011001011
2096	111011011	2432	01111111
2104	01001011	2440	00000010
2112	10111101	2448	101101000
2120	10011100	2456	01001101
2128	00110001	2464	111101101
2136	110001111	2472	00100111
2144	01111110	2480	101101001
2152	10010001	2488	110010011
2160	10110011	2496	111101110
2168	111011100	2504	111000111
2176	00001110	2512	101101010
2184	10010010	2520	00111111
2192	10111110	2528	100100110
2200	10100110	2536	111011101
2208	00110010	2544	010011100
2216	00001111	2552	110110010
2224	00011000	2560	101101011
2232	110101111	2568	101001001
2240	110110000	2576	01000000
2248	111111111	2584	101101100
2256	11111001	2592	011110011
2264	00111100	2600	100000000
2272	00010000	2608	110010100
2280	10100111	2616	00011010
2288	01101101	2624	01001111
2296	01011100	2632	100100111
2304	00000000	2640	111001010
2312	00111101	2648	100000001
2320	01001100	2656	100000010
2328	00011001	2664	101101101
2336	00100100	2672	110010101
2344	110110001	2680	000101111
2352	00111110	2688	001101000
2360	00100101	2696	111001011
2368	11001000	2704	1110011001
2376	00100110	2712	1101100110
2384	00000001	2720	011011100
2392	100011101	2728	000110110

Продолжение таблицы Б.9

Индекс	huff_sfreqba	Индекс	huff_sfreqba
2736	111011110	3072	01110000101
2744	110010110	3080	0110111110
2752	111101111	3088	110011011
2760	010011101	3096	00110101101
2768	0001111011	3104	11011011010
2776	110010111	3112	1101101110
2784	010100000	3120	1011111100
2792	101101110	3128	0000001101
2800	101010000	3136	0110111111
2808	1110111110	3144	000111100
2816	101010001	3152	10011101010
2824	100000011	3160	11011011011
2832	1110111111	3168	11011011110
2840	010100001	3176	0011010111
2848	100101000	3184	0000001110
2856	1111000000	3192	0111000000
2864	110011000	3200	11011011111
2872	110110100	3208	01000001000
2880	1010110011	3216	101111110100101110110
2888	1101100111	3224	110111000000
2896	1111110001	3232	1101110001
2904	001101001	3240	10011101011
2912	1101101010	3248	110111000001
2920	11011010110	3256	1110011000
2928	110011001	3264	0000001111
2936	00101000	3272	0111000001
2944	11011010111	3280	1011111101001011110111
2952	011011101	3288	110111000010
2960	001101010	3296	010000010010
2968	1001010010	3304	01000001010
2976	1101101100	3312	010000010011
2984	011011110	3320	10101010000
2992	00110101100	3328	0100000101100
3000	000110111	3336	0001111010
3008	101101111	3344	010000010111
3016	1001010011	3352	1011111101001011111000
3024	0101110101	3360	0100000101101
3032	110011010	3368	110111000011
3040	1111000001	3376	10101010001
3048	0000001100	3384	01011101000
3056	1111101000	3392	010111010010
3064	1001110100	3400	010111010011

Окончание таблицы Б.9

Индекс	*huff_sfreqba*	Индекс	*huff_sfreqba*
3408	1011111101000	3648	10111111010010010100
3416	101111110100101111001	3656	10111111010010010101
3424	011100001000	3664	10111111010010010110
3432	101111110100101111010	3672	10111111010010010111
3440	101111110100101111011	3680	10111111010010011000
3448	101111110100101111100	3688	10111111010010011001
3456	101111110100101111101	3696	10111111010010011010
3464	101111110100101111110	3704	10111111010010011011
3472	101111110100101111111	3712	10111111010010011100
3480	011100001001	3720	10111111010010011101
3488	101111110100100000000	3728	10111111010010011110
3496	101111110100100000001	3736	10111111010010011111
3504	101111110100100000010	3744	10111111010010100000
3512	101111110100100000011	3752	10111111010010100001
3520	101111110100100000100	3760	10111111010010100010
3528	101111110100100000101	3768	10111111010010100011
3536	101111110100100000110	3776	10111111010010100100
3544	101111110100100000111	3784	10111111010010100101
3552	101111110100100001000	3792	10111111010010100110
3560	101111110100100001001	3800	10111111010010100111
3568	101111110100100001010	3808	10111111010010101000
3576	101111110100100001011	3816	10111111010010101001
3584	101111110100100001100	3824	10111111010010101010
3592	101111110100100001101	3832	10111111010010101011
3600	101111110100100001110	3840	10111111010010101100
3608	101111110100100001111	3848	10111111010010101101
3616	101111110100100010000	3856	10111111010010101110
3624	101111110100100010001	3864	10111111010010101111
3632	101111110100100010010	3872	10111111010010110000
3640	101111110100100010011	3880	10111111010010110001

Таблица Б.10 — *huff_sfreqbr*

Индекс	*huff_sfreqbr*	Индекс	*huff_sfreqbr*
0	0000001011	72	01011
8	101111	80	00111
16	10101	88	111101
24	11011	96	111110
32	11001	104	111010
40	10100	112	111001
48	10001	120	100111
56	01100	128	101100
64	01101	136	011111

Продолжение таблицы Б.10

Индекс	*huff_sfreqbr*	Индекс	*huff_sfreqbr*
144	100100	480	11101100
152	010101	488	10110110
160	010000	496	10111011
168	001100	504	10110111
176	001001	512	01001001
184	000010	520	00010011
192	000001	528	11010010
200	000111	536	01010011
208	1111110	544	10000110
216	1111000	552	01110110
224	1100011	560	00101110
232	1110111	568	01001110
240	1101000	576	01010000
248	1001100	584	01001111
256	1011010	592	00101101
264	1100001	600	00000011
272	1011100	608	00000001
280	1000000	616	111100111
288	1000010	624	111100110
296	1001101	632	110100110
304	0111101	640	111100100
312	1001010	648	00001111
320	0100010	656	00101111
328	0111010	664	111111110
336	0100011	672	101110101
344	0001011	680	110101110
352	0100101	688	100101100
360	0100110	696	00010100
368	0001100	704	110101111
376	0010100	712	111000011
384	0011010	720	011100011
392	0010000	728	110100111
400	0011011	736	100000100
408	0010101	744	100000101
416	11111110	752	011101111
424	0001000	760	011100001
432	11101101	768	100001110
440	11100000	776	101110100
448	11010101	784	000110110
456	11100010	792	011100100
464	11010110	800	000100101
472	11000001	808	001011001

Продолжение таблицы Б.10

Индекс	*huff_sfreqbr*	Индекс	*huff_sfreqbr*
816	1111111110	1152	0010001100
824	1111001011	1160	110000001100
832	100101101	1168	0111001111
840	010010001	1176	11111111111
848	001000101	1184	0111011100
856	010100010	1192	0010001101
864	1100000001	1200	1000001100
872	011110010	1208	0010001110
880	1001011101	1216	0010110001
888	000110111	1224	0111011101
896	000101010	1232	10010111110
904	1110001100	1240	0100100000
912	000000100	1248	01111001101
920	000011000	1256	0000000000
928	1101010011	1264	1001011100
936	1110000100	1272	0001101001
944	000100100	1280	0000000001
952	011100010	1288	0000000010
960	1110001101	1296	01001000011
968	0111100000	1304	11100001010
976	000101011	1312	110000001101
984	0111100001	1320	0001101010
992	1101010000	1328	0000000011
1000	0111000001	1336	0001101011
1008	1101010001	1344	01110011011
1016	0111001010	1352	10010111111
1024	0111001011	1360	01010001100
1032	1000011111	1368	11100011101
1040	1100000010	1376	01010001101
1048	0010001001	1384	110000001110
1056	1101010010	1392	1000011110111
1064	1100000000	1400	11100001011
1072	0111001110	1408	11000100000
1080	1110001111	1416	01111001100
1088	11111111110	1424	00100011110
1096	0000111011	1432	0010001000
1104	0111100010	1440	11000100001
1112	0111100011	1448	11000100010
1120	0001101000	1456	01010001110
1128	0111001100	1464	01010001111
1136	0111100111	1472	0000110010
1144	1001011110	1480	111000111001

Продолжение таблицы Б.10

Индекс	*huff_sfreqbr*	Индекс	*huff_sfreqbr*
1488	00100011111	1824	100001111000
1496	10000011010	1832	0101001001001
1504	11000100011	1840	010100101000
1512	00000010101	1848	111100101010
1520	10000011011	1856	000011011100
1528	111100101000	1864	010100101001
1536	10000011100	1872	0101001010100
1544	01010010000	1880	100001111001
1552	010010000101	1888	1100010100011
1560	01010010001	1896	010100101011
1568	110000001111	1904	000011011101
1576	00001100110	1912	1100010100100
1584	0100100001001	1920	010100101100
1592	0101001001000	1928	0101001010101
1600	110001001000	1936	000011011110
1608	00001100111	1944	11000101001010
1616	100000111010	1952	000011011111
1624	1100010010010	1960	000011100000
1632	110001001010	1968	010100101101
1640	00101100000	1976	111100101011
1648	010100100101	1984	110001010011
1656	00001101000	1992	010100101110
1664	100000111011	2000	000011100001
1672	01010010011	2008	000011100010
1680	10000011110	2016	010100101111
1688	00101100001	2024	000011100011
1696	00001101001	2032	1100010101000
1704	110001001011	2040	0111000000000
1712	11000100110	2048	0111000000001
1720	110001001110	2056	000011100100
1728	110001001111	2064	010010000100011
1736	111100101001	2072	000011100101
1744	100000111110	2080	011100000001
1752	00001101010	2088	100001111010
1760	1100010010011	2096	0111000000100
1768	00001101011	2104	1100010101001
1776	00001101100	2112	000011100110
1784	00001101101	2120	1100010101010
1792	110001010000	2128	000011100111
1800	100000111111	2136	0111000000101
1808	000000101001	2144	000011101000
1816	1100010100010	2152	110001010010110

Продолжение таблицы Б.10

Индекс	*huff_sfreqbr*	Индекс	*huff_sfreqbr*
2160	000011101001	2496	11000101110000
2168	0100100001000101110010	2504	0100100001000101110111
2176	110001010010111	2512	110001011011111
2184	000011101010	2520	0100100001000101110100
2192	11000101010110	2528	11000101110001
2200	11000101010111	2536	0100100001000101110100
2208	0111000000110	2544	0100100001000101110101
2216	0111000000111	2552	110001011100100
2224	1100010101100	2560	0100100001000101110101
2232	0111001101000	2568	110001011100101
2240	1100010101101	2576	0100100001000101110110
2248	11000101011100	2584	0100100001000101110110
2256	0111001101001	2592	110001011100110
2264	0111001101010	2600	110001011100111
2272	000011101011	2608	1100010111010
2280	11000101011101	2616	0100100001000101110111
2288	11000101011110	2624	110001011101100
2296	0100100001000101110011	2632	0100100001000101110111
2304	11000101011111	2640	110001011101101
2312	0111001101011	2648	0100100001000101111000
2320	110001011000000	2656	0100100001000101111001
2328	110001011000001	2664	0100100001000101111010
2336	1100010110001	2672	110001011101110
2344	1100010110010	2680	110001011101111
2352	110001011000010	2688	110001011110000
2360	0100100001000101110100	2696	110001011110001
2368	110001011000011	2704	110001011110010
2376	11000101100110	2712	11000101111010
2384	110001011001110	2720	11000101111011
2392	0100100001000101110101	2728	110001011110011
2400	0100100001000101110110	2736	110001011111000
2408	11000101101000	2744	0100100001000101111001
2416	1000011110110	2752	0100100001000101111010
2424	11000101101001	2760	0100100001000101111010
2432	110001011001111	2768	11000101111101
2440	11000101101010	2776	0100100001000101111011
2448	110001011010110	2784	110001011111001
2456	1100010110110	2792	0100100001000101111011
2464	110001011010111	2800	0100100001000101111100
2472	110001011011100	2808	110001011111100
2480	110001011011101	2816	0100100001000101111001
2488	110001011011110	2824	110001011111101

Продолжение таблицы Б.10

Индекс	*huff_sfreqbr*	Индекс	*huff_sfreqbr*
2832	0100100001000101111110	3168	010010000100010010111
2840	11000101111111	3176	010010000100010011000
2848	0100100001000101111011	3184	010010000100010011001
2856	0100100001000101111100	3192	010010000100010011010
2864	111000111000000	3200	010010000100010011011
2872	0100100001000101111101	3208	010010000100010011100
2880	111000111000001	3216	010010000100010011101
2888	111000111000010	3224	010010000100010011110
2896	0100100001000101111110	3232	010010000100010011111
2904	0100100001000101111111	3240	010010000100010100000
2912	0100100001000100000000	3248	010010000100010100001
2920	11100011100010	3256	010010000100010100010
2928	0100100001000100000001	3264	010010000100010100011
2936	111000111000011	3272	010010000100010100100
2944	111000111000110	3280	010010000100010100101
2952	0100100001000100000010	3288	010010000100010100110
2960	111000111000111	3296	010010000100010100111
2968	00000010100000	3304	010010000100010101000
2976	0100100001000100000011	3312	010010000100010101001
2984	0100100001000100000100	3320	010010000100010101010
2992	0100100001000100000101	3328	010010000100010101011
3000	0100100001000100000110	3336	010010000100010101100
3008	0100100001000100000111	3344	010010000100010101101
3016	0100100001000100001000	3352	010010000100010101110
3024	00000010100001	3360	010010000100010101111
3032	0100100001000100001001	3368	010010000100010110000
3040	0100100001000100001010	3376	010010000100010110001
3048	0100100001000100001011	3384	010010000100010110010
3056	0100100001000100001100	3392	010010000100010110011
3064	0100100001000100001101	3400	010010000100010110100
3072	00000010100010	3408	010010000100010110101
3080	0100100001000100001110	3416	010010000100010110110
3088	0100100001000100001111	3424	010010000100010110111
3096	0100100001000100010000	3432	010010000100010111000
3104	00000010100011	3440	010010000100010111001
3112	0100100001000100010001	3448	010010000100010111010
3120	01001000010000	3456	010010000100010111011
3128	0100100001000100010010	3464	010010000100010111100
3136	0100100001000100010011	3472	010010000100010111101
3144	0100100001000100010100	3480	010010000100010111110
3152	0100100001000100010101	3488	010010000100010111111
3160	0100100001000100010110	3496	010010000100010100000

Окончание таблицы Б.10

Индекс	*huff_sfreqbr*	Индекс	*huff_sfreqbr*
3504	010010000100010100001	3696	010010000100010101001
3512	010010000100010100010	3704	010010000100010101010
3520	010010000100010100011	3712	010010000100010101011
3528	010010000100010100100	3720	010010000100010101100
3536	010010000100010100101	3728	010010000100010101101
3544	010010000100010100110	3736	010010000100010101110
3552	010010000100010100111	3744	010010000100010101111
3560	010010000100010101000	3752	010010000100010110000
3568	010010000100010101001	3760	010010000100010110001
3576	010010000100010101010	3768	010010000100010110010
3584	010010000100010101011	3776	010010000100010110011
3592	010010000100010101100	3784	010010000100010110100
3600	010010000100010101101	3792	010010000100010110101
3608	010010000100010101110	3800	010010000100010110110
3616	010010000100010101111	3808	010010000100010110111
3624	010010000100010110000	3816	010010000100010111000
3632	010010000100010110001	3824	010010000100010111001
3640	010010000100010110010	3832	010010000100010111010
3648	010010000100010110011	3840	010010000100010111011
3656	010010000100010110100	3848	010010000100010111100
3664	010010000100010110101	3856	010010000100010111101
3672	010010000100010110110	3864	010010000100010111110
3680	010010000100010110111	3872	010010000100010111111
3688	010010000100010111000	3880	010010000100011110000

Таблица Б.11 — *huff_sfreqc*

Индекс	*huff_sfreqc*	Индекс	*huff_sfreqc*
0	00100011011100111010010000	120	100110101011010
8	00100011011100111010010001	128	100110101011000
16	00100011011100111010010010	136	100110101011001
24	00100011011100111010010011	144	0010001101100
32	00100011011100111010010100	152	0010001101111
40	00100011011100111010010101	160	011001001001
48	00100011011100111010010110	168	111101000111
56	00100011011100111010010111	176	11000000000
64	00100011011100111010011000	184	0111000010
72	00100011011100111010011001	192	0010011010
80	0010001101110011011	200	11100010010
88	00100011011100111010011010	208	0000101111
96	001000110111001100	216	0000001100
104	0010001101110010	224	10110110111
112	00100011011100111	232	11101011000

Продолжение таблицы Б.11

Индекс	*huff_sfreqc*	Индекс	*huff_sfreqc*
240	11011100010	576	011100011
248	10111101000	584	001000111
256	10101001000	592	110000011
264	01101011101	600	111111001
272	10001011101	608	10110110110
280	11001101101	616	11110100010
288	0101100011	624	1000001100
296	001011100	632	011010010
304	0000001000	640	100001010
312	11000000001	648	100100110
320	01100100101	656	1010001011
328	10011010001	664	0000100011
336	0001100010	672	11001111111
344	001010000	680	1100000001
352	110001001	688	01011100
360	01101011100	696	00000111
368	10101110100	704	110001101
376	0000000001	712	0100100111
384	1000001101	720	1010100101
392	0111000011	728	001100011
400	0011101010	736	0100100110
408	000110010	744	0101000010
416	1111011110	752	010100000
424	11101101001	760	000110000
432	01101000000	768	011111100
440	10001010001	776	110000001
448	111101000110	784	00110111
456	00100011010	792	00000100
464	11011100000	800	11001111110
472	0000001101	808	10001010000
480	1001001011	816	11011100001
488	11100111000	824	11100010011
496	01101000001	832	111100011
504	11101011001	840	100001011
512	1011001011	848	111000110
520	1101010011	856	1011101111
528	1100010000	864	11010100100
536	0000100010	872	0110101100
544	0000101110	880	0110010011
552	0110101101	888	1001001010
560	1001110111	896	1111011111
568	001110100	904	100100000

Продолжение таблицы Б.11

Индекс	*huff_sfreqc*	Индекс	*huff_sfreqc*
912	00111100	1248	101001110
920	1100111110	1256	100111010
928	1000101111	1264	111100111
936	1011001010	1272	001011101
944	00010100	1280	1100010001
952	1111011010	1288	1111011011
960	10100010100	1296	1010111001
968	10101110101	1304	0011001010
976	11001101000	1312	0111000101
984	0001100011	1320	001001100
992	000110011	1328	011011110
1000	010110000	1336	11010010
1008	100000111	1344	00001111
1016	010101000	1352	1010111000
1024	0110101111	1360	000110100
1032	10111101001	1368	010101001
1040	11101101000	1376	110111111
1048	11001101100	1384	01000110
1056	0111000100	1392	1001101011
1064	011101110	1400	1110011101
1072	00110011	1408	010010010
1080	01111000	1416	011010001
1088	1001110110	1424	101000100
1096	11011100011	1432	001010001
1104	0101000011	1440	011000101
1112	1011110101	1448	101110110
1120	1111110000	1456	10011110
1128	1101100111	1464	10010001
1136	1110001000	1472	000010000
1144	011100100	1480	1010111011
1152	01001101	1488	1100110101
1160	011001000	1496	1111110001
1168	0111001010	1504	010111111
1176	1000101001	1512	1111010000
1184	110110010	1520	111100101
1192	10110001	1528	111100100
1200	0111001011	1536	0001011
1208	10011010000	1544	01001100
1216	00111010111	1552	000110101
1224	10100010101	1560	1110110101
1232	11100111001	1568	010111110
1240	1100100011	1576	101100100

Продолжение таблицы Б.11

Индекс	huff_sfreqc	Индекс	huff_sfreqc
1584	110010110	1920	10011011
1592	100100111	1928	11011011
1600	111011011	1936	110111110
1608	00001001	1944	011101111
1616	01101010	1952	11101110
1624	11111101	1960	10011100
1632	111001111	1968	11010011
1640	000000001	1976	00011011
1648	00111001	1984	110010111
1656	00100101	1992	111110001
1664	11000101	2000	11001110
1672	10001110	2008	10011001
1680	001100100	2016	111111100
1688	110110100	2024	00001100
1696	01100001	2032	111010100
1704	111011110	2040	01000101
1712	100100100	2048	11110000
1720	101101010	2056	0100111
1728	111000101	2064	00001110
1736	00011101	2072	01000010
1744	01111100	2080	111111101
1752	10000100	2088	111101100
1760	00000001	2096	00011100
1768	00000101	2104	01110101
1776	111011111	2112	01100000
1784	00111011	2120	10001111
1792	110000010	2128	10111110
1800	110001100	2136	10010101
1808	100010101	2144	1010010
1816	00010000	2152	11100100
1824	00010001	2160	01010111
1832	0110110	2168	00101010
1840	10000110	2176	01110011
1848	100100001	2184	10101011
1856	011010011	2192	11011000
1864	01000111	2200	01111101
1872	00110000	2208	00101011
1880	10000010	2216	00110101
1888	00110100	2224	11000011
1896	110111101	2232	10001001
1904	110111001	2240	111100110
1912	01010010	2248	111100010

Продолжение таблицы Б.11

Индекс	*huff_sfreqc*	Индекс	*huff_sfreqc*
2256	10010100	2592	10101111
2264	11010111	2600	0010110
2272	11011101	2608	11101001
2280	11111111	2616	10011000
2288	10111000	2624	11001001
2296	10110111	2632	10110100
2304	11000111	2640	01010001
2312	01100110	2648	10100110
2320	11010101	2656	10100011
2328	10110011	2664	00110110
2336	111101010	2672	10001100
2344	110111100	2680	01111111
2352	01001010	2688	01001011
2360	10000000	2696	0111101
2368	1010110	2704	11010000
2376	0001001	2712	101111011
2384	01010101	2720	110110101
2392	01010011	2728	01011010
2400	01111001	2736	10001000
2408	10010110	2744	11000010
2416	10101000	2752	11100001
2424	01011101	2760	10100000
2432	11010001	2768	01011001
2440	11001100	2776	111110011
2448	00100000	2784	10010111
2456	10111010	2792	00111000
2464	0001111	2800	01011110
2472	11101000	2808	000010110
2480	00101111	2816	110010100
2488	01110110	2824	11100110
2496	11100101	2832	01100111
2504	10101010	2840	111010111
2512	10011111	2848	00001101
2520	11111011	2856	111110100
2528	10001101	2864	110101000
2536	10000111	2872	110101101
2544	11101100	2880	11100000
2552	0100000	2888	111101110
2560	01000011	2896	00010101
2568	111101011	2904	00101001
2576	111101001	2912	011011111
2584	00111101	2920	011100000

Продолжение таблицы Б.11

Индекс	*huff_sfreqc*	Индекс	*huff_sfreqc*
2928	10000001	3264	0010001100
2936	10111111	3272	1100100010
2944	000000101	3280	0000001001
2952	101111000	3288	000000111
2960	01100011	3296	10101001001
2968	111110010	3304	10011010100
2976	111010101	3312	0110100001
2984	10100001	3320	00000110
2992	101101100	3328	0000101011
3000	01000100	3336	1011101110
3008	110010000	3344	0000000000
3016	100010110	3352	0011001011
3024	00100001	3360	11010100101
3032	0011111	3368	10001011100
3040	01011011	3376	1100110111
3048	011111101	3384	1011011010
3056	01101110	3392	11001101001
3064	01010110	3400	00001010101
3072	10111001	3408	00001010100
3080	111000111	3416	0010001101101
3088	10110000	3424	0101100010
3096	110011110	3432	0010011011
3104	111110101	3440	011001001000
3112	111110000	3448	001000110111000
3120	00100111	3456	10011010101110
3128	01001000	3464	10011010101111
3136	011000100	3472	1001101010101
3144	101111001	3480	00111010110
3152	00100010	3488	001000110111010
3160	01110100	3496	1001101010100
3168	110010101	3504	00100011011100110100110111
3176	101001111	3512	001000110111001101001100
3184	00100100	3520	1001101010110111
3192	01100101	3528	001000110111011
3200	101101011	3536	00100011011100110100111101
3208	101010011	3544	1001101010110110
3216	110101100	3552	00100011011100110100111110
3224	001100010	3560	00100011011100110100111111
3232	000010100	3568	00100011011100110101000000
3240	1110101101	3576	00100011011100110101000001
3248	1001101001	3584	00100011011100110101000010
3256	1101100110	3592	00100011011100110101000011

Окончание таблицы Б.11

Индекс	*huff_sfreqc*	Индекс	*huff_sfreqc*
3600	00100011011100011010100100	3744	00100011011100011010110110
3608	00100011011100011010100101	3752	00100011011100011010110111
3616	00100011011100011010100110	3760	00100011011100011010111000
3624	00100011011100011010100111	3768	00100011011100011010111001
3632	00100011011100011010101000	3776	00100011011100011010111010
3640	00100011011100011010101001	3784	00100011011100011010111011
3648	00100011011100011010101010	3792	00100011011100011010111100
3656	00100011011100011010101011	3800	00100011011100011010111101
3664	00100011011100011010101100	3808	00100011011100011010111110
3672	00100011011100011010101101	3816	00100011011100011010111111
3680	00100011011100011010101110	3824	00100011011100011101000000
3688	00100011011100011010101111	3832	00100011011100011101000001
3696	00100011011100011010110000	3840	00100011011100011101000010
3704	00100011011100011010110001	3848	00100011011100011101000011
3712	00100011011100011010110010	3856	00100011011100011101000100
3720	00100011011100011010110011	3864	00100011011100011101000101
3728	00100011011100011010110100	3872	00100011011100011101000110
3736	00100011011100011010110101	3880	00100011011100011101000111

Таблица Б.12 — *huff_nlag*

Индекс	*huff_nlag*	Индекс	*huff_nlag*
–512	100010101101101100	–428	100010111010010001
–508	100010101101101101	–424	100010111010010010
–504	100010101101101110	–420	100010111010010011
–500	100010101101101111	–416	100010111010010100
–496	100010111010000000	–412	100010111010010101
–492	100010111010000001	–408	100010111010010110
–488	100010111010000010	–404	100010111010010111
–484	100010111010000011	–400	100010111010011000
–480	100010111010000100	–396	100010111010011001
–476	100010111010000101	–392	100010111010011010
–472	100010111010000110	–388	100010111010011011
–468	100010111010000111	–384	100010111010011100
–464	100010111010001000	–380	100010111010011101
–460	100010111010001001	–376	100010111010011110
–456	100010111010001010	–372	100010111010011111
–452	100010111010001011	–368	100010111010100000
–448	100010111010001100	–364	100010111010100001
–444	100010111010001101	–360	100010111010100010
–440	100010111010001110	–356	100010111010100011
–436	100010111010001111	–352	100010111010100100
–432	100010111010010000	–348	100010111010100101

Продолжение таблицы Б.12

Индекс	*huff_nlag*	Индекс	*huff_nlag*
–344	100010111010100110	–176	100010111011010001
–340	100010111010100111	–172	100010111011010010
–336	100010111010101000	–168	100010111011010011
–332	100010111010101001	–164	100010111011010100
–328	100010111010101010	–160	10001011101101011
–324	100010111010101011	–156	1000101001011001
–320	100010111010101100	–152	100010111011010101
–316	100010111010101101	–148	100010111011011000
–312	100010111010101110	–144	100010111011011001
–308	100010111010101111	–140	100010111011011010
–304	100010111010110000	–136	100010111011011011
–300	100010111010110001	–132	10001011101101110
–296	100010111010110010	–128	10001011101101111
–292	100010111010110011	–124	100010111011100000
–288	100010111010110100	–120	100010111011100001
–284	100010111010110101	–116	100010111011100010
–280	100010111010110110	–112	100010111011100011
–276	100010111010110111	–108	100010111011100100
–272	100010111010111000	–104	100010111011100101
–268	100010111010111001	–100	100010111011100110
–264	100010111010111010	–96	1000101001011010
–260	100010111010111011	–92	100010100100111
–256	100010111010111100	–88	1000101001011011
–252	100010111010111101	–84	1000101110111010
–248	100010111010111110	–80	10001010010100
–244	100010111010111111	–76	10001010010101
–240	100010111011000000	–72	10001010110010
–236	100010111011000001	–68	10000001101100
–232	100010111011000010	–64	10000001101101
–228	100010111011000011	–60	10001010111111
–224	100010111011000100	–56	1000000110111
–220	100010111011000101	–52	1000101011000
–216	100010111011000110	–48	100010101010
–212	100010111011000111	–44	100010101011
–208	100010111011001000	–40	100010111000
–204	100010111011001001	–36	10001010100
–200	1000101101100101	–32	1000000111
–196	100010111011001100	–28	1000101111
–192	100010111011001101	–24	100010110
–188	100010111011001110	–20	1000001
–184	100010111011001111	–16	100011
–180	100010111011010000	–12	10011

Продолжение таблицы Б.12

Индекс	*huff_nlag*	Индекс	*huff_nlag*
–8	1110	160	10000001100001001
–4	101	164	1000000110000101
0	0	168	10000001100001100
4	110	172	10000001100001101
8	1111	176	1000000110000111
12	10010	180	10000001100010000
16	100001	184	10000001100010001
20	1000100	188	10000001100010010
24	10000000	192	1000000110001010
28	100000010	196	10000001100010011
32	1000101000	200	10000001100010110
36	10001010011	204	10000001100010111
40	100010111001	208	10000001100011000
44	100010101110	212	10000001100011001
48	1000101011010	216	10000001100011010
52	1000101011110	220	1000000110001110
56	10001010110111	224	10000001100011011
60	1000101001000	228	10000001100011110
64	10001010110011	232	10000001100011111
68	100010101111100	236	10000001100100000
72	10001010010010	240	10000001100100001
76	100010111011110	244	10000001100100010
80	100010100101110	248	10000001100100011
84	10001011101110110	252	10000001100100100
88	100010101111101	256	10000001100100101
92	100010100101111	260	10000001100100110
96	1000101110111110	264	10000001100100111
100	10001011101110111	268	10000001100101000
104	1000101110111111	272	10000001100101001
108	1000101011011000	276	10000001100101010
112	100010111011100111	280	10000001100101011
116	1000101011011001	284	10000001100101100
120	10000001100000000	288	10000001100101101
124	1000101011011010	292	10000001100101110
128	10000001100000001	296	10000001100101111
132	10000001100000010	300	10000001100110000
136	10000001100000011	304	10000001100110001
140	10000001100000100	308	10000001100110010
144	10000001100000101	312	10000001100110011
148	10000001100000110	316	10000001100110100
152	10000001100000111	320	10000001100110101
156	10000001100001000	324	10000001100110110

Окончание таблицы Б.12

Индекс	*huff_nlag*	Индекс	*huff_nlag*
328	10000001100110111	424	10000001101001111
332	10000001100111000	428	10000001101010000
336	10000001100111001	432	10000001101010001
340	10000001100111010	436	10000001101010010
344	10000001100111011	440	10000001101010011
348	10000001100111100	444	10000001101010100
352	10000001100111101	448	10000001101010101
356	10000001100111110	452	10000001101010110
360	10000001100111111	456	10000001101010111
364	10000001101000000	460	10000001101011000
368	10000001101000001	464	10000001101011001
372	10000001101000010	468	10000001101011010
376	10000001101000011	472	10000001101011011
380	10000001101000100	476	10000001101011100
384	10000001101000101	480	10000001101011101
388	10000001101000110	484	10000001101011110
392	10000001101000111	488	10000001101011111
396	10000001101001000	492	10001010010011000
400	10000001101001001	496	10001010010011001
404	10000001101001010	500	10001010010011010
408	10000001101001011	504	10001010010011011
412	10000001101001100	508	10001010010110000
416	10000001101001101	512	10001010010110001
420	10000001101001110		

Таблица Б.13 — *huff_nlsf*

Индекс	*huff_nlsf*	Индекс	*huff_nlsf*	Индекс	*huff_nlsf*
7	10110011	14	1000	21	1011010
8	101101110	15	00	22	10010011
9	101101111	16	11	23	100100101
10	10110010	17	01	24	100100100
11	1001000	18	1010	25	10110110
12	1011000	19	10111		
13	100101	20	10011		

Таблица Б.14 — *huff_ngain*

Индекс	*huff_ngain*	Индекс	*huff_ngain*	Индекс	*huff_ngain*
–13	010011100	–9	01001110100	–5	01000110
–12	010011011001	–8	01001110110	–4	0100010
–11	010011101110	–7	0100110101	–3	010010
–10	010011101011	–6	010011000	–2	01011

Окончание таблицы Б.14

Индекс	*huff_ngain*	Индекс	*huff_ngain*	Индекс	*huff_ngain*
–1	00	4	01001111	9	01001101101
0	1	5	01000111	10	010011101111
1	011	6	010011001	11	010011101010
2	01010	7	0100110111	12	010011011000
3	010000	8	0100110100		

Таблица Б.15 — *huff_scont*

Индекс	*huff_scont*	Индекс	*huff_scont*	Индекс	*huff_scont*
0	1000	4	001	8	0000
1	0100	5	1001	9	11
2	101	6	0101		
3	011	7	0001		

Таблица Б.16 — *huff_nrofbirths*

Индекс	*huff_nrofbirths*	Индекс	*huff_nrofbirths*	Индекс	*huff_nrofbirths*
0	010	21	000001001	42	000000100001110
1	1010	22	00000101	43	000000100001111
2	1110	23	000001100	44	000000100010000
3	011	24	00000001	45	000000100010001
4	110	25	0000010001	46	000000100010010
5	100	26	0000011010	47	000000100010011
6	001	27	0000011011	48	000000100010100
7	1111	28	000000100000100	49	000000100010101
8	0001	29	0000011100	50	000000100010110
9	10111	30	000000100000101	51	000000100010111
10	10110	31	000000100000110	52	000000100011000
11	0000111	32	000000100000111	53	000000100011001
12	000010	33	0000011101	54	000000100011010
13	0000110	34	000000100001000	55	000000100011011
14	00000011	35	0000011110	56	000000100011100
15	00000000	36	000000100001001	57	000000100011101
16	000000101	37	000000100001010	58	000000100011110
17	0000001001	38	0000011111	59	000000100011111
18	0000010000	39	000000100001011	60	00000010000000
19	000000100000010	40	000000100001100		
20	000000100000011	41	000000100001101		

Таблица Б.17 — *huff_ipd_df* и *huff_ipd_dt*

Индекс	*huff_ipd_df*	*huff_ipd_dt*
0	1	1
1	000	010

Окончание таблицы Б.17

Индекс	*huff_ipd_df*	*huff_ipd_dt*
2	0110	0010
3	0100	00011
4	0010	00010
5	0011	0000
6	0101	0011
7	0111	011

Таблица Б.18 — *huff_opd_df* и *huff_opd_dt*

Индекс	*huff_opd_df*	*huff_opd_dt*
0	1	1
1	001	010
2	0110	0001
3	0100	00111
4	01111	00110
5	01110	0000
6	0101	0010
7	000	011

Таблица Б.19 — *huff_iid_df* [1] и *huff_iid_dt* [1]

Индекс	*huff_iid_df* [1]	*huff_iid_dt* [1]	Индекс	*huff_iid_df* [1]	*huff_iid_dt* [1]
–30	011111111010110100	0100111011010100	–9	01111111011	010011110111
–29	011111111010110101	0100111011010101	–8	0111111011	01001111000
–28	011111110101110110	0100111011001110	–7	0111111111	0100111001
–27	011111110101110111	0100111011001111	–6	01111100	010011010
–26	011111110101110100	0100111011001100	–5	0111100	010011111
–25	011111110101110101	0100111011010110	–4	011100	0100000
–24	011111111010001010	0100111011011000	–3	01100	010001
–23	011111111010001011	0100111101000110	–2	0000	01010
–22	011111111010001000	0100111101100000	–1	001	011
–21	011111110100000000	010011100011000	0	1	1
–20	011111111010110110	010011100011001	1	010	00
–19	011111110100000010	010011101100100	2	0001	01011
–18	011111110101110000	010011101100101	3	01101	010010
–17	0111111101000010	010011101101101	4	011101	0100001
–16	0111111110101110	010011110110001	5	0111101	01001100
–15	011111110101111	01001110110111	6	01111101	010011011
–14	01111111010001	01001111010110	7	011111100	0100111010
–13	01111111101001	0100111000111	8	0111111100	01001111001
–12	0111111101001	0100111101001	9	01111111100	01001110000
–11	011111101010	0100111101101	10	01111110100	010011101111
–10	011111111011	010011101110	11	011111101011	010011100010

Окончание таблицы Б.19

Индекс	*huff_iid_df* [1]	*huff_iid_dt* [1]	Индекс	*huff_iid_df* [1]	*huff_iid_dt* [1]
12	0111111101010	0100111101010	22	011111111010001001	0100111101100001
13	01111111101010	0100111011000	23	011111111010001110	0100111101000111
14	01111111010110	01001111010111	24	011111111010001111	0100111011011001
15	011111111010000	01001111010000	25	011111111010001100	0100111011010111
16	0111111110101111	010011110110010	26	011111111010001101	0100111011001101
17	0111111101000011	010011110100010	27	011111111010110010	0100111011010010
18	01111111010111001	010011100011010	28	011111111010110011	0100111011010011
19	01111111010000011	010011100011011	29	011111111010110000	0100111011010000
20	011111111010110111	0100111101100110	30	011111111010110001	0100111011010001
21	01111111010000001	0100111101100111			

Таблица Б.20 — *huff_icc_dt* и *huff_icc_df*

Индекс	*huff_icc_df*	*huff_icc_dt*
–7	11111111111111	11111111111110
–6	11111111111110	1111111111110
–5	111111111110	11111111110
–4	1111111110	111111110
–3	1111110	1111110
–2	11110	11110
–1	110	110
0	0	0
1	10	10
2	1110	1110
3	111110	111110
4	11111110	11111110
5	111111110	1111111110
6	11111111110	111111111110
7	1111111111110	11111111111111

Таблица Б.21 — *huff_iid_df* [0] и *huff_iid_dt* [0]

Индекс	*huff_iid_df* [0]	*huff_iid_dt* [0]
–14	11111111111111011	111111111111111001
–13	11111111111111100	111111111111111010
–12	11111111111111101	111111111111111011
–11	11111111111111010	1111111111111111000
–10	1111111111111100	1111111111111111001
–9	111111111111100	1111111111111111010
–8	1111111111101	1111111111111101
–7	1111111110	111111111111110
–6	111111110	111111111110
–5	1111110	1111111110

Окончание таблицы Б.21

Индекс	*huff_iid_df* [0]	*huff_iid_dt* [0]
–4	111100	11111110
–3	11101	111110
–2	1101	1110
–1	101	10
0	0	0
1	100	110
2	1100	11110
3	11100	1111110
4	111101	111111110
5	111110	11111111110
6	11111110	1111111111110
7	11111111110	11111111111110
8	1111111111100	1111111111111100
9	11111111111100	11111111111111000
10	11111111111101	11111111111111011
11	11111111111101	11111111111111100
12	1111111111111110	11111111111111101
13	11111111111111110	11111111111111110
14	11111111111111111	11111111111111111

Приложение В
(справочное)

Темп и подача, масштабируемые в декодере

Темп, масштабирующий аудиосигнал, необходим, чтобы изменить (мгновенную) продолжительность сигнала, сохраняя воспринятую подачу. Как правило, в темпе декодера масштабирование обрабатывается для каждого объекта отдельно:

Переходные процессы; масштабирование темпа обычно не применяется к переходному объекту. Перцепционно переходные процессы характеризуются временами атаки и временами затухания. Намного более естественный результат получается, когда продолжительность переходного процесса сохраняется.

Синусоиды; для синусоидального объектного темпа масштабирование может быть получено, применяя два изменения в декодере. Прежде всего длина окна синтеза должна быть изменена согласно масштабному коэффициенту времени, гарантирующему, что унитарное перекрытие добавляется. Во-вторых, новое фазовое значение получается для продолжений, применяя непрерывную фазовую функцию.

Шум; в пределах объекта шума темп может масштабироваться, масштабируя оба окна синтеза.

Стерео; масштабирование темпа делается, изменяя позицию параметра согласно масштабному коэффициенту времени.

Поскольку подача главным образом определяется синусоидальными частотами, один подход должен масштабировать только частоты синусоид $f_{new} = p \cdot f_{old}$ с масштабным коэффициентом подачи p.

В.1 Непрерывная фаза

Так как фазовые значения зависят от размера подкадра S, в случае темпа и масштабирования подачи декодер переключается на непрерывную фазу. Темп и масштабирующая подачу процедура для синусоидальных компонентов делаются следующим образом. Во-первых, исходная частота и фаза получаются процессом декодирования. В случае *phase_jitter_present* устанавливается в %1, полученная частота повторно квантуется, приводя к уровню представления sf_{rl}. Уровень представления частоты sf_{rl} становится $sf_{rl} + sf_{jitter}$ и деквантуется. Эта частота умножается на масштабный коэффициент подачи и используется в непрерывном фазовом вычислении.

Непрерывная фаза φ_2 вычисляется так, что фаза двух последовательных сегментов перекрывается в середине области.

Фазовая информация φ_1 и φ_2 определяется в течение середины сегмента (= $(L–1)/2$). Где f_1 и f_2 представляют частоту в радианах.

Фаза продолжающегося синусоидального компонента в первом подкадре фрейма с *refresh_sinusoids* = = %1 не устанавливается в значение, считанное из потока (*s_phi*), а вычисляется из выражения для φ_2. Если процесс декодирования запускается в этом подкадре, наугад от отмеченного подкадра, фаза продолжающегося синусоидального компонента должна быть установлена в значение, считанное из потока.

В.2 Темп, масштабирующийся для стерео

Темп, масштабирующийся для параметрического инструмента стерео, может быть достигнут, адаптируя позицию параметров. μ — масштабный коэффициент темпа. Считывается предыдущее и текущее масштабируемыми позициями параметры $\hat{n}_{prev} = n_{e-1} \cdot \mu$ и $\hat{n}_{curr} = n_e \cdot \mu$ соответственно. Перемещая эти позиции в целочисленные позиции, декодирование стерео может быть выполнено как обычно. Метод для отображения позиции параметра нецелого числа $\hat{n}_{prev}$ и $\hat{n}_{curr}$ к целочисленным позициям $\hat{n}_{prev}$ и $\hat{n}_{curr}$ дается, используя следующую рекурсию

$$m = \mathrm{mod}(\hat{n}_{curr} - n_{prev}, 1),$$

где n_{prev} является предыдущей целочисленной позицией.

Затем текущая целочисленная позиция параметра вычисляется следующим образом:

$$n_{curr} = \hat{n}_{curr} + 1 - m.$$

Чтобы инициировать рекурсию $n_{prev} = 0$.

Библиография

[1] ИСО/МЭК 14496-3:2009* Информационные технологии. Кодирование аудиовизуальных объектов. Часть 3. Аудио (ISO/IEC14496-3:2009 *Information technology — Coding of audio-visual objects — Part 3: Audio*)

* Заменен на ISO/IEC 14496-3:2019.

УДК 621.396:006.354 ОКС 33.170

Ключевые слова: звуковое вещание, электрические параметры, каналы и тракты, технологии MPEG-кодирования, синтетический звук, масштабирование, защита от ошибок, поток битов расширения, психоакустическая модель

Редактор переиздания *Е.Ю. Яковлева*
Технический редактор *В.Н. Прусакова*
Корректор *М.В. Бучная*
Компьютерная верстка *Е.О. Асташина*

Сдано в набор 24.08.2020. Подписано в печать 21.12.2020. Формат 60×84 1/8. Гарнитура Ариал.
Усл. печ. л. 9,30. Уч.-изд. л. 9,20.
Подготовлено на основе электронной версии, предоставленной разработчиком стандарта

Создано в единичном исполнении во ФГУП «СТАНДАРТИНФОРМ» для комплектования Федерального информационного фонда стандартов, 117418 Москва, Нахимовский пр-т, д. 31, к. 2.

gost.e-ecolog.ru gost.e-ecolog.ru

gost.e-ecolog.ru