ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р 54713— 2011

Звуковое вещание цифровое

КОДИРОВАНИЕ СИГНАЛОВ ЗВУКОВОГО ВЕЩАНИЯ С СОКРАЩЕНИЕМ ИЗБЫТОЧНОСТИ ДЛЯ ПЕРЕДАЧИ ПО ЦИФРОВЫМ КАНАЛАМ СВЯЗИ. MPEG-2, ЧАСТЬ VII: УСОВЕРШЕНСТВОВАННОЕ КОДИРОВАНИЕ ЗВУКА (MPEG-2 AAC)

Издание официальное

Москва
Стандартинформ
2013

gost.e-ecolog.ru

Предисловие

Цели и принципы стандартизации в Российской Федерации установлены Федеральным законом от 27 декабря 2002 г. № 184-ФЗ «О техническом регулировании», а правила применения национальных стандартов Российской Федерации — ГОСТ Р 1.0—2004 «Стандартизация в Российской Федерации. Основные положения»

Сведения о стандарте

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 13 декабря 2011 г. № 874-ст

4 Настоящий стандарт разработан с учетом основных нормативных положений международного стандарта ИСО/МЭК 13818-7:2006 Информационные технологии — Универсальное кодирование движущихся изображений и сопутствующего звука — Часть 7: Усовершенствованное кодирование звука (AAC) (ISO/IEC 13818-7:2006 Information technology — Generic coding of moving pictures and associated audio information — Part 7: Advanced Audio Coding (AAC) [1]

5 ВВЕДЕН ВПЕРВЫЕ

Информация об изменениях к настоящему стандарту публикуется в ежегодно издаваемом информационном указателе «Национальные стандарты», а текст изменений и поправок — в ежемесячно издаваемых информационных указателях «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ежемесячно издаваемом информационном указателе «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

© Стандартинформ, 2013

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
1.1 Цель разработки стандарта . . . 1
1.2 Условия применения стандарта . . . 1
2 Нормативные ссылки . . . 7
3 Термины, определения, символы и сокращения . . . 7
3.1 Термины и определения . . . 7
3.2 Символы и сокращения . . . 7
3.3 Метод описания синтаксиса потока битов . . . 9
4 Синтаксис . . . 10
4.1 Формат обмена аудиоданными, ADIF . . . 10
4.2 Транспортный поток аудиоданных, ADTS . . . 11
4.3 Необработанные данные . . . 13
5 Профили и функциональная совместимость профилей . . . 22
5.1 Состав профилей . . . 22
5.2 Функциональная совместимость профилей . . . 24
6 Общая структура данных . . . 24
6.1 Форматы обмена AAC . . . 24
6.2 Необработанные данные . . . 28
6.3 Элемент одиночного канала (SCE), элемент парного канала (CPE) и поток индивидуального канала (ICS) . . . 31
6.4 Канал низкой частоты (LFE) . . . 37
6.5 Элемент конфигурации программы (PCE) . . . 37
6.6 Элемент потока данных (DSE) . . . 40
6.7 Элемент заполнения (FIL) . . . 40
6.8 Расширение . . . 41
6.9 Таблицы . . . 44
6.10 Рисунки . . . 53
7 Прозрачное кодирование . . . 54
7.1 Описание инструмента . . . 54
7.2 Определения . . . 54
7.3 Процесс декодирования . . . 56
7.4 Таблицы . . . 58
8 Квантование . . . 59
8.1 Описание инструмента . . . 59
8.2 Элементы справки . . . 59
8.3 Процесс декодирования . . . 59
9 Масштабные коэффициенты . . . 60
9.1 Описание инструмента . . . 60
9.2 Определения . . . 60
9.3 Процесс декодирования . . . 60
10 Совместное кодирование . . . 62
10.1 M/S stereo . . . 62
10.2 Intensity stereo . . . 63
10.3 Спаренный канал . . . 64
11 Предсказание . . . 67
11.1 Описание инструмента . . . 67
11.2 Элементы данных . . . 68
11.3 Процесс декодирования . . . 68
11.4 Схемы . . . 73
12 Временное формирование шума (TNS) . . . 74
12.1 Описание инструмента . . . 74
12.2 Элементы данных . . . 74
12.3 Процесс декодирования . . . 74

13 Банк фильтров и переключение окон . . . 76
13.1 Описание инструмента . . . 76
13.2 Определения . . . 76
13.3 Процесс декодирования . . . 76
14 Управление усилением . . . 80
14.1 Описание инструмента . . . 80
14.2 Определения . . . 81
14.3 Процесс декодирования . . . 81
14.4 Схемы . . . 85
14.5 Таблицы . . . 86
Приложение А (обязательное) Таблицы кодов Хаффмана . . . 87
Приложение Б (обязательное) Информация о неиспользованных кодовых книгах . . . 102
Приложение В (обязательное) Кодер . . . 103
Библиография . . . 144

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Звуковое вещание цифровое

КОДИРОВАНИЕ СИГНАЛОВ ЗВУКОВОГО ВЕЩАНИЯ С СОКРАЩЕНИЕМ ИЗБЫТОЧНОСТИ ДЛЯ ПЕРЕДАЧИ ПО ЦИФРОВЫМ КАНАЛАМ СВЯЗИ. MPEG-2, ЧАСТЬ VII: УСОВЕРШЕНСТВОВАННОЕ КОДИРОВАНИЕ ЗВУКА (MPEG-2 AAC)

Sound broadcasting digital. Coding of signals of a sound broadcasting with reduction of redundancy for transfer on digital communication channels. MPEG-2, part VII: Advanced Audio Coding (MPEG-2 AAC)

Дата введения — 2012—12—01

1 Область применения

1.1 Цель разработки стандарта

Настоящий стандарт позволяет получить более высокое качество многоканального звучания. С его помощью достигается «неразличимое» с оригиналом качество по шкале ITU-R согласно [2] на скоростях передачи данных 320 Кбит/с для пяти звуковых сигналов с полной пропускной способностью.

Действие настоящего стандарта распространяется на услуги местной, внутризоновой, междугородной и международной сети звукового вещания, независимо от используемой сетевой технологии, что способствует обеспечению целостности сетей звукового вещания, устойчивости работы сети, выполнению норм на основные электрические параметры при разработке и проектированию каналообразующей аппаратуры звукового вещания (ГОСТ Р 52742 и ГОСТ Р 53537).

Показатели, определенные настоящим стандартом, являются базовыми для профессиональной и бытовой аппаратуры — проигрывателей компакт-дисков, усилителей сигналов звуковой частоты и другого оборудования класса Hi-Fi.

Универсальная и совместимая многоканальная аудиосистема применима для спутникового и наземного телевизионного вещания, цифрового звукового вещания (наземного и спутникового), так же как и для других носителей, например:

CATV — кабельное телевидение;

CDAD — кабельное цифровое звуковое вещание;

DAB — широковещательная передача цифрового звукового сигнала;

DVD — цифровой универсальный диск;

ENG — электронные новости (включая новости по спутнику);

HDTV — телевидение высокой четкости;

IPC — межличностное общение (видеоконференция, видеотелефон и т. д.);

ISM — интерактивные носители (оптические диски и т. д.).

Вход кодера и выход декодера совместимы со стандартами импульснокодовой модуляции (ИКМ), такими как ГОСТ 28376, ГОСТ 27667 и др.

1.2 Условия применения стандарта

1.2.1 Спецификация инструментов MPEG-2 AAC

В процессе декодирования AAC используется ряд необходимых и опциональных инструментов. В таблице 1 перечислены инструменты и их статус (необходимые или опциональные). Необходимые

Издание официальное

инструменты обязательны в любом возможном профиле. Опциональные инструменты могут не использоваться в некоторых профилях.

Таблица 1 — Перечень инструментов декодера AAC

Предназначение	Статус
Средство форматирования потока битов	Необходимый
Прозрачное декодирование	Необходимый
Обратное квантование	Необходимый
Перемасштабирование	Необходимый
M/S	Опциональный
Предсказание	Опциональный
Интенсивность	Опциональный
Зависимое спаривание каналов	Опциональный
TNS	Опциональный
Банк фильтров/переключение окон	Необходимый
Регулирование усиления	Опциональный
Независимое спаривание каналов	Опциональный

1.2.2 Назначение инструментов декодирования

Общая структура системы MPEG-2 AAC приведена на рисунках 1 и 2. В соответствии с таблицей 1 структура декодера состоит из необходимых и опциональных инструментов. Направление потока данных в этой схеме слева направо, сверху вниз. В задачи декодера входят обнаружение описания квантованных спектральных значений в потоке битов, декодирование квантованных значений и другой информации для восстановления, восстановление квантованных спектральных значений, обработка восстановленных спектральных значений соответствующими инструментами, активными для данного потока битов, с целью достигнуть исходного спектра входного звукового сигнала, и, наконец, преобразование спектральных значений во временные отсчеты, с (или без) дополнительного инструмента регулирования усиления. После начального восстановления и масштабирования восстановленных спектральных значений может применяться множество дополнительных инструментов, используемых для обеспечения более эффективного кодирования. Для каждого из дополнительных инструментов, которые работают в спектральном пространстве, предусмотрена опция отключения, и во всех случаях, когда обработка в спектральном пространстве не используется, входные спектральные значения поступают непосредственно на выход инструмента без изменений.

1.2.3 Вход и выход инструментов демультиплексирования

На вход инструмента демультиплексирования поступает поток битов MPEG-2 AAC. Демультиплексор разделяет поток данных MPEG-AAC на части, предзначенные для каждого инструмента, и предоставляет для каждого из инструментов информацию о потоке битов, относящуюся к этому инструменту.

На выходе инструмента демультиплексирования потока битов содержится:

- информация о разделении для прозрачного кодирования;
- прозрачно-кодированные спектральные значения;
- информация о *M/S* (опционально);
- информация о состоянии предсказывающего устройства (опционально);
- информация для управления intensity stereo и информация для управления спаренным каналом (опциональные);
- информация о временном формировании шума (*TNS*) (опционально);
- информация для управления банком фильтров;
- информация о регулировании усиления (опционально).

1.2.4 Инструмент прозрачного декодирования

Информация потока битов с демультиплексора поступает на инструмент прозрачного декодирования, который анализирует ее, декодирует коды Хаффмана и восстанавливает квантованные спектральные значения, а также кодированные с помощью кодов Хаффмана и ДИКМ масштабные коэффициенты.

На вход инструмента прозрачного декодирования поступают информация о разделении для прозрачного кодирования и прозрачно-кодированные спектральные значения.

Выход инструмента прозрачного декодирования содержит декодированное целочисленное представление масштабных коэффициентов и квантованные спектральные значения.

1.2.5 Инструмент деквантования

Квантованные спектральные сигналы поступают на вход инструмента деквантования, который преобразует целочисленные значения в восстановленные деквантованные спектральные значения. Этот деквантователь является неоднородным.

1.2.6 Инструмент перемасштабирования

Инструмент перемасштабирования преобразует целочисленное представление масштабных коэффициентов в их фактические значения и умножает восстановленные, деквантованные спектральные значения на соответствующие масштабные коэффициенты.

На вход инструмента перемасштабирования поступает декодированное целочисленное представление масштабных коэффициентов и восстановленные, деквантованные спектральные значения.

На выходе инструмента перемасштабирования содержатся масштабированные, деквантованные спектральные значения.

1.2.7 Инструмент *M/S*

На вход инструмента поступает информация о *M/S* (середина/сторона) и масштабированные, деквантованные спектральные значения, относящиеся к парам каналов. Инструмент *M/S* преобразует пары спектральных значений из *M/S* в *L/R* под управлением информации о *M/S* с целью улучшения кодирования.

На выходе инструмента *M/S* присутствуют масштабированные, деквантованные спектральные значения сигналов, относящиеся к парам каналов после декодирования *M/S*.

Следует учитывать, что масштабированные, деквантованные спектральные значения индивидуально кодированных каналов не обрабатываются блоком *M/S* и передаются непосредственно на выход инструмента *M/S* без изменений. Если инструмент *M/S* не является активным, все спектральные значения проходят через этот блок без изменений.

1.2.8 Инструмент предсказания

Этот инструмент обращает процесс предсказания, выполненный в кодере. Обратный процесс предсказания добавляет избыточность, которая была устранена инструментом предсказания в кодере, под управлением информации о состоянии предсказывающего устройства. Данный инструмент представляет собой обратное адаптивное предсказывающее устройство второго порядка. На вход инструмента предсказания поступают информация о состоянии предсказывающего устройства и масштабированные, деквантованные спектральные значения. На выходе инструмента предсказания — масштабированные, деквантованные спектральные значения после предсказания.

Если предсказание не используется, масштабированные, деквантованные спектральные значения поступают непосредственно на выход блока без изменений.

1.2.9 Инструмент *intensity stereo*

Данный инструмент реализует декодирование *intensity stereo* спектральных пар. На вход инструмента *intensity stereo* поступают деквантованные спектральные значения и управляющая информация *intensity stereo*.

На выходе инструмента *intensity stereo* — деквантованные спектральные значения после декодирования канала интенсивности.

Масштабированные, деквантованные спектральные значения индивидуально кодированных каналов поступают непосредственно на выход этого инструмента без изменений, если *intensity stereo* не используется. Инструмент *intensity stereo* и инструмент *M/S* располагаются так, чтобы работа *M/S* и *intensity stereo* была взаимоисключающей для любой полосы масштабных коэффициентов и группы одной пары спектральных значений.

1.2.10 Инструмент спаривания для зависимо коммутируемых каналов

Инструмент добавляет соответствующие данные от зависимо коммутируемых каналов к спектральным значениям в соответствии с информацией для управления спариванием каналов. На входе инструмента спаривания — деквантованные спектральные значения и информация для управления спариванием каналов. На выходе инструмента — деквантованные спектральные значения вместе с зависимо коммутируемыми каналами.

Масштабированные, деквантованные спектральные значения поступают непосредственно на выход этого инструмента без изменений, если спаривание каналов не применяется. В зависимости от информации для управления спариванием каналов, зависимо коммутируемые каналы могут быть спарены до или после обработки *TNS*.

1.2.11 Инструмент спаривания для независимо коммутируемых каналов

Инструмент добавляет соответствующие данные от независимо коммутируемых каналов к временному сигналу в соответствии с информацией для управления спариванием каналов. На вход инструмента спаривания поступает временной сигнал, аналогичный сигналу на выходе набора фильтров и информация для управления спариванием каналов.

На выходе инструмента — временной сигнал вместе с независимо коммутируемыми каналами.

Временной сигнал поступает непосредственно на выход этого инструмента без изменений, если спаривание каналов не используется.

1.2.12 Инструмент временного формирования шума (*TNS*)

С помощью данного инструмента реализуется управление точной временной структурой шума кодирования. В кодере в результате процесса *TNS* сглаживается временная огибающая сигнала, к которому это было применено. В декодере происходит обратный процесс для восстановления фактической временной огибающей (их) под управлением информации о *TNS*. Это выполняется фильтрацией частей спектральных данных. На входе инструмента *TNS* присутствуют деквантованные спектральные значения и информация о *TNS*, на выходе — деквантованные спектральные значения.

Если этот блок отключен, деквантованные спектральные значения поступают на его выход без изменений.

1.2.13 Банк фильтров/инструмент переключения окон

Банк фильтров реализует обратное спектральное преобразование. Обратное дискретное косинусное преобразование (ОДКП) используется в качестве банка фильтров. ОДКП может поддерживать либо один набор из 128 или 1024, или четыре набора из 32 или 256 спектральных коэффициентов.

На вход банка фильтров поступают деквантованные спектральные значения и информация для управления банком фильтров, на выход(ы) банка фильтров — восстановленные временные отсчеты звукового сигнала(ов).

1.2.14 Инструмент регулирования усиления

При использовании данного инструмента осуществляется отдельное регулирование усиления временных отсчетов в каждой из четырех частотных полос, которые были получены регулированием усиления *PQF*-банка фильтров кодера. Далее инструмент собирает четыре частотных полосы и восстанавливает форму временного сигнала с помощью банка фильтров инструмента регулирования усиления. При этом на вход инструмента регулирования усиления поступают восстановленные временные отсчеты звукового сигнала(ов) и информация о регулировании усиления, а на выход(ы) инструмента регулирования усиления — восстановленные временные остчеты звукового сигнала(ов).

Если инструмент регулирования усиления не используется, восстановленные временные отсчеты звукового сигнала(ов) поступают непосредственно из банка фильтров на выходе декодера. Этот инструмент используется в профиле масштабируемой частоты дискретизации (*SSR*).

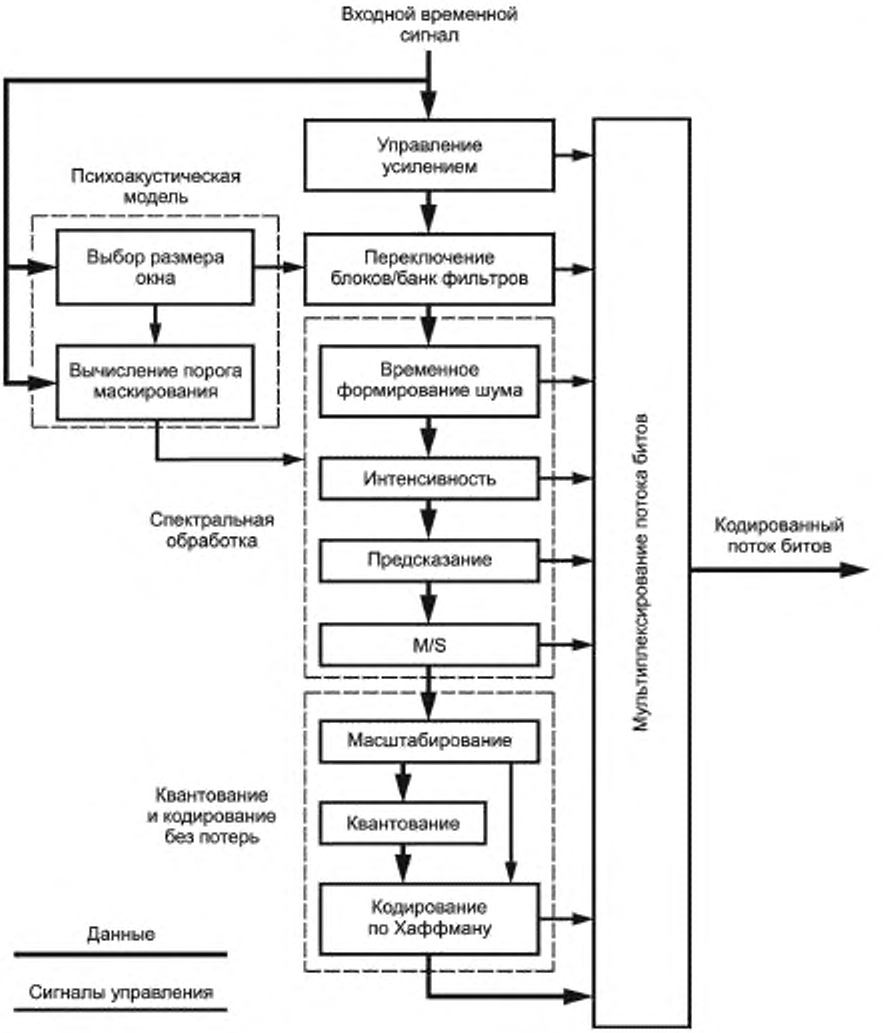


Рисунок 1 — Блок-схема кодера MPEG-2 AAC

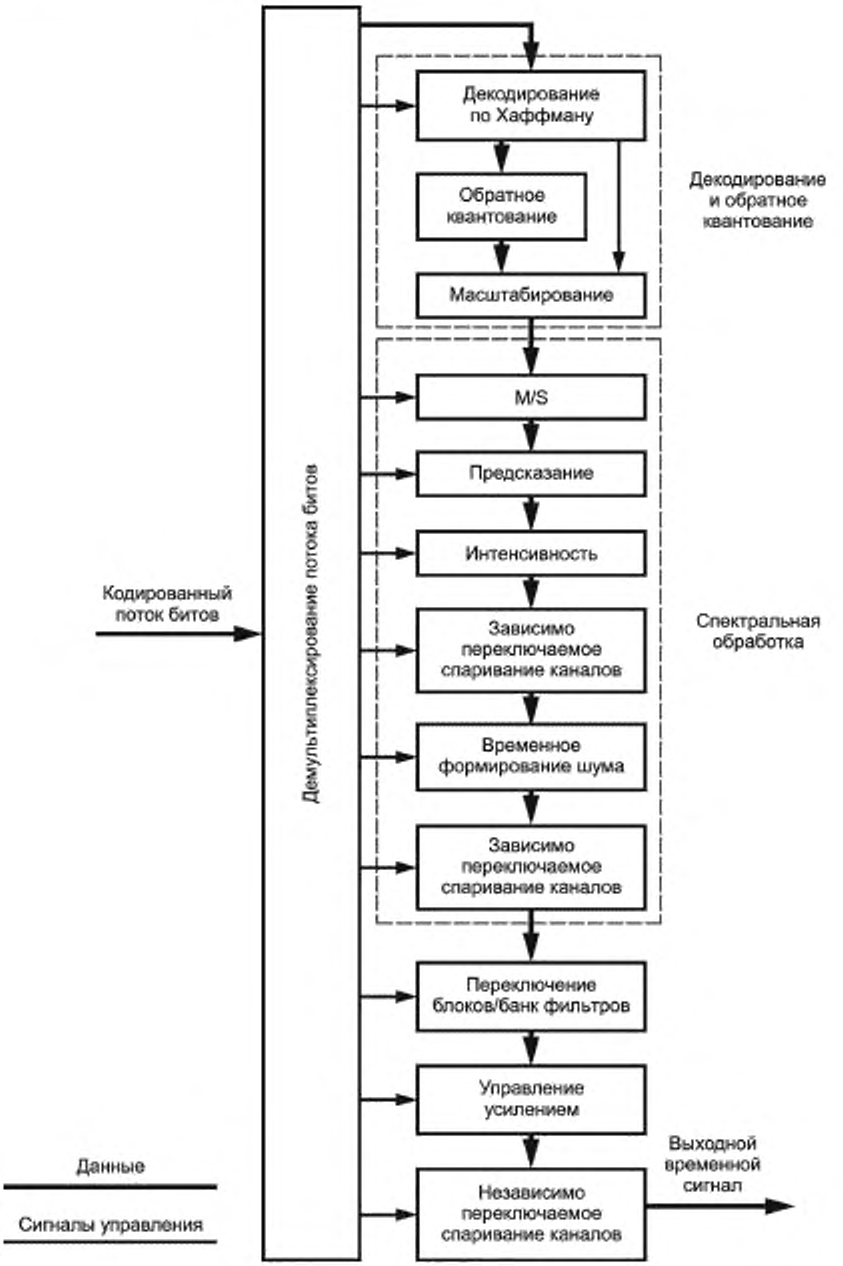


Рисунок 2 — Блок-схема декодера MPEG-2 AAC

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты:

ГОСТ Р 52742—2007 Каналы и тракты звукового вещания. Типовые структуры. Основные параметры качества. Методы измерений

ГОСТ Р 53537—2009 Звуковое вещание. Основные электрические параметры каналов и трактов студийного качества (с полосой частот 20…20000 Гц)

ГОСТ 27667—88 Система цифровая звуковая «Компакт-диск». Параметры

ГОСТ 28376—89 Компакт-диск. Параметры и размеры.

Примечание — При пользовании настоящим стандартом целесообразно проверить действие ссылочных стандартов в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет или по ежегодно издаваемому информационному указателю «Национальные стандарты», который опубликован по состоянию на 1 января текущего года, и по соответствующим ежемесячно издаваемым информационным указателям, опубликованным в текущем году. Если ссылочный стандарт заменен (изменен), то при пользовании настоящим стандартом следует руководствоваться заменяющим (измененным) стандартом. Если ссылочный стандарт отменен без замены, то положение, в котором дана ссылка на него, применяется в части, не затрагивающей эту ссылку.

3 Термины, определения, символы и сокращения

3.1 Термины и определения

основные звуковые каналы: Все каналы, представленные либо *single_channel_element()* (см. 6.2.1), либо *channel_pair_element()* (см. 6.2.1).

программа: Набор основных звуковых каналов, *coupling_channel_element()* (см. 6.2.1), *lfe_channel_element()* (см. 6.2.1) и сопутствующих потоков данных, требующих одновременного декодирования и воспроизведения.

Примечание — Программа может быть определена по умолчанию (см. 6.5.3.1 и 6.5.3.3) или указана *program_config_element*() (см. 6.5.3.2). Данные *single_channel_element*() (см. 6.2.1), *channel_pair_element*() (см. 6.2.1), *coupling_channel_element*(), *lfe_channel_element*() или канал передачи данных могут сопровождать одну или несколько программ в любом потоке битов.

синтаксический анализатор: Функциональный элемент декодера, который извлекает из кодированного потока битов серию битов, представляющих кодированные элементы.

спектральные коэффициенты: Дискретные значения в частотном пространстве на выходе набора фильтров анализа.

стерео избыточность: Часть стереофонического звукового сигнала обладает стерео избыточностью, если она не способствует пространственному восприятию сигнала.

число учитываемых каналов; *NCC:* Число каналов, представленных *SCE* элементами, независимо переключаемых *CCE* и *CPE*, т. е. единожды взятое количество *SCE* плюс единожды взятое количество независимо коммутируемых *CCE* плюс дважды взятое число *CPE*, в соответствии с соглашением о присвоении имен в декодерах и потоках битов MPEG-AAC, $NCC = A + I$.

Примечание — Число учитываемых каналов используется для получения требуемого размера входного буфера декодера (см. 6.2.2).

3.2 Символы и сокращения

Математические операторы, используемые в этом стандарте, аналогичны используемым в языке программирования С. Однако целочисленное деление с усечением и округление определены особым образом. Побитные операторы определяются с учетом представления чисел в дополнительном коде. Нумерация и счетчики циклов обычно начинаются с нуля.

3.2.1 Арифметические операторы

\+ — Сложение.

– — Вычитание (как бинарный оператор) или отрицание (как унарный оператор).

++ — Инкремент.

– — Декремент.

* — Умножение.

^ — Возведение в степень.

/ — Целочисленное деление с округлением к меньшему по модулю целому. Например, 7/4 и –7/4 округляются до 1, а –7/4 и 7/–4 округляются до –1.

// — Целочисленное деление с округлением к ближайшему целому числу. Полуцелые числа округляются в сторону ближайшего большего по модулю числа, если не указано другое. Например, 3//2 округляется до 2, а –3//2 округляется до – 2.

DIV — Целочисленное разделение с округлением результата в сторону – ∞.

| | — Абсолютное значение. | x | = x, когда x> 0
| x | = 0, когда x == 0
| x | = –x, когда x <0

% — Деление с остатком. Операция определена только для положительных чисел.

Sign ( ) – Sign (x) = 1, когда x> 0
Sign (x) = 0, когда x == 0
Sign (x) = –1, когда x <0

NINT ( ) — Округление до ближайшего целого. Возвращает самое близкое к вещественному аргументу целочисленное значение. Полуцелые числа округляются в сторону от нуля.

sin — Синус.

cos — Косинус.

exp — Экспонента.

√ — Квадратный корень.

log10 — Логарифм по основанию 10.

$\log_e$ — Натуральный логарифм.

$\log_2$ — Логарифм по основанию 2.

3.2.2 Логические операторы

|| — Логическое ИЛИ.

&& — Логическое И.

! — Логическое НЕТ.

3.2.3 Операторы сравнения

> — Больше

> = — Больше или равно

< — Меньше

<= — Меньше или равно

== — Равно

! = — Не равно

max [,...,] — максимальное значение.

min [,...,] — минимальное значение.

3.2.4 Побитные операторы

Использование побитных операций подразумевает представление чисел в дополнительном коде.

& — Побитное И.

| — Побитное ИЛИ.

>> — Сдвиг вправо со знаком.

<< — Сдвиг влево с нулевым заполнением.

3.2.5 Присвоение

= — Оператор присвоения.

3.2.6 Мнемоники

Следующие мнемоники подлежат определению для описания различные типов данных, используемых в кодированном потоке битов.

bslbf — Битовая строка, младший бит слева. Битовые строки пишутся как строка единиц и нулей внутри одинарных кавычек, например '1000 0001'. Пробелы внутри битовой строки вводятся для удобства чтения и не имеют никакого значения.

L, C, R, LS, RS — Аудиосигналы: левый, центральный, правый, левый окружения, правый окружения.
M/S — Середина/сторона *M* = (*L*+*R*)/2 и *S* = (*L*–*R*)/2.
rpchof — Коэффициенты остатка от деления на порождающий полином, сначала следует коэффициент высшего порядка. (Аудио).
uimsbf — Целое число без знака, старший бит первый.
vlclbf — Код с переменной длиной слова, левый бит первый, где левый относится к порядку, в котором пишутся коды с переменной длиной.
window — Номер фактического временного слота в случае *block_type* == 2, 0 <= *window* <= 2. (Аудио).

В многобайтовых словах старший байт является первым.

3.2.7 Константы

π — 3,14159265358...
e — 2,71828182845...

3.3 Метод описания синтаксиса потока битов

Поток битов на входе декодера описывается в разделе 4. Каждый элемент данных выделен жирным. При описании элемента указываются
- его имя;
- его длина в битах, где *X*.. *Y* указывает, что количество битов принадлежит диапазону от *X* до *Y*, включая *X* и *Y*. {*X*; *Y*} означает, что количество битов равно *X* или *Y*, в зависимости от значения других элементов данных в потоке битов;
- мнемоника для его типа и порядок передачи.

Действие, вызванное декодируемым элементом данных в потоке битов, зависит от значения того элемента данных и на элементах данных, ранее декодируемых. Декодирование элементов данных и определение параметров состояния, используемых в их декодировании, описываются в пунктах, следующих за описанием синтаксиса. Следующие конструкции используются, чтобы выразить условия, когда элементы данных присутствуют, и указаны обычным шрифтом.

Следует обратить внимание, что в этом синтаксисе используется принятое в языке С соглашение о том, что переменная или выражение, возвращающие ненулевое значение, эквивалентны результату «истина»:

while (condition) { *data_element* . . . *}*	Если «истина», то группа элементов данных появляется в потоке данных. Это повторяется, пока условие не «ложь».
do { *data_element* . . . *} while (condition)*	Элемент данных всегда появляется, по крайней мере, однажды. Элемент данных повторяется, пока условие не «ложь».
if (condition) { *data_element* . . . *}*	Если условие является «истина», то первая группа элементов данных появляется в потоке данных.
else { *data_element* . . . *}*	Если условие не является «истиной», то вторая группа элементов данных появляется в потоке данных.
for (expr1; expr2; expr3) { *data_element* . . . *}*	Expr1 является инициализирующим выражением цикла. Обычно оно определяет начальное состояние счетчика. Expr2 является условием, определяющим проверку перед каждой итерацией цикла. Цикл завершается, когда условие не является «истиной». Expr3 является выражением, которое выполняется в конце каждой итерации цикла, обычно оно инкрементирует счетчик.

Следует обратить внимание на следующие наиболее распространенные варианты использования этой конструкции:

for (i = 0; i < n; i++) {
data_element
. . .
}

Группа элементов данных появляется *n* раз. Условия в пределах группы элементов данных могут зависеть от значения переменной управления циклом *i*, которая обнуляется при первом появлении, увеличивается на 1 при втором появлении и т. д.

Как отмечено, группа элементов данных может содержать вложенные условные конструкции. Для компактности {} может быть опущен, когда следует только один элемент данных :

data_element [] — *data_element* [] является массивом данных. Количество элементов массива зависит от контекста.

data_element [n] — *data_element [n]* является (*n*+1)-ым элементом массива данных.

data_element [m] [n] — *data_element [m] [n]* является элементом (*m*+1)-ой строки (*n*+1)-го столбца двухмерного массива данных.

data_element [l] [m] [n] — *data_element [l] [m] [n]* является (*l*+1), (*m*+1), (*n*+1)-ым элементом трехмерного массива данных.

data_element [m...n] — *data_element [m...n]* содержит биты массива *data_element* с *m* по *n* включительно.

Знание самого синтаксиса потока битов в разделе 4 не следует считать достаточным для декодирования. В частности, это лишь определяет корректный и свободный от ошибок входной поток битов. Реальные декодеры для того, чтобы правильно начать декодирование, должны иметь средства обнаружения стартовых последовательностей.

Определение функции *nextbits*

Функция *nextbits*() реализует сравнение строки битов со строкой битов на входе декодера.

4 Синтаксис

Описание синтаксиса потока битов представлено в таблицах 2—30.

4.1 Формат обмена аудиоданными, ADIF

Таблица 2 — Синтаксис *adif_sequence* ()

Синтаксис	Количество битов	Мнемоника
adif_sequence() *{* *adif_header();* *byte_alignment();* *raw_data_stream();* *}*		

Таблица 3 — Синтаксис *adif_header* ()

Синтаксис	Количество битов	Мнемоника
adif_header() *{*		
adif_id;	32	*bslbf*
copyright_id_present;	1	*bslbf*
if (copyright_id_present) {		
copyright_id; *}*	72	*bslbf*
original_copy;	1	*bslbf*

Окончание таблицы 3

Синтаксис	Количество битов	Мнемоника
home;	1	*bslbf*
bitstream_type;	1	*bslbf*
bitrate;	23	*uimsbf*
num_program_config_elements;	4	*bslbf*
if (bitstream_type == '0') {		
adif_buffer_fullness;	20	*uimsbf*
} *for (i = 0; i < num_program_config_elements + 1; i++) {*		
program_config_element(); *}* *}*		

4.2 Транспортный поток аудиоданных, ADTS

Таблица 4 — Синтаксис *adts_sequence* ()

Синтаксис	Количество битов	Мнемоника
adts_sequence() *{* *while (nextbits() == syncword) {* *adts_frame();* *}* *}*		

Таблица 5 — Синтаксис *adts_frame* ()

Синтаксис	Количество битов	Мнемоника
adts_frame() *{* *adts_fixed_header();* *adts_variable_header();* *if (number_of_raw_data_blocks_in_frame == 0) {* *adts_error_check();* *raw_data_block();* *}* *else{* *adts_header_error_check();* *for (i = 0; i <= number_of_raw_data_blocks_in_frame;* *i++) {* *raw_data_block();* *adts_raw_data_block_error_check();* *}* *}* *}*		

Таблица 6 — Синтаксис *adts_header_error_check* ()

Синтаксис	Количество битов	Мнемоника
adts header error check () *{* *if (protection_absent == '0') {* *for (i = 1; i <= number of raw data blocks in frame; i++) {* *raw data block position[i];* *}* *crc check;* *}* *}*	 16 16	 *Uimsfb* *rpchof*

Таблица 7 — Синтаксис *of adts_raw_data_block_error_check*()

Синтаксис	Количество битов	Мнемоника
adts raw data block error check() *{* *if (protection absent == '0')* *crc check;* *}*	 16	 *rpchof*

4.2.1 Фиксированный заголовок ADTS

Таблица 8 — Синтаксис *adts_fixed_header*()

Синтаксис	Количество битов	Мнемоника
adts_fixed_header()		
{ *syncword;*	12	*bslbf*
ID;	1	*bslbf*
layer;	2	*uimsbf*
protection_absent;	1	*bslbf*
profile;	2	*uimsbf*
sampling_frequency_index;	4	*uimsbf*
private_bit;	1	*bslbf*
channel_configuration;	3	*uimsbf*
original_copy;	1	*bslbf*
home; *}*	1	*bslbf*

4.2.2 Переменный заголовок ADTS

Таблица 9 — Синтаксис *adts_variable_header* ()

Синтаксис	Количество битов	Мнемоника
adts_variable_header() *{* *copyright_identification_bit;* *copyright_identification_start;* *aac_frame_length;* *adts_buffer_fullness;* *number_of_raw_data_blocks_in_frame;* *}*	 1 1 13 11 2	 *bslbf* *bslbf* *bslbf* *bslbf* *uimsfb*

4.2.3 Обнаружение ошибок

Таблица 10 — Синтаксис *adts_error_check* ()

Синтаксис	Количество битов	Мнемоника
adts_error_check() *{* *if (protection absent == '0')* *crc check;* *}*	16	*rpchof*

4.3 Необработанные данные

Таблица 11 — Синтаксис *raw_data_stream* ()

Синтаксис	Количество битов	Мнемоника
raw_data_stream () *{* *while (data available()) {* *raw data block();* *}* *}*		

Таблица 12 — Синтаксис *raw_data_block*()

Синтаксис	Количество битов	Мнемоника
raw_data_block() *{*		
while ((id = id_syn_ele) != ID_END) {	3	*uimsbf*
switch (id) {		
case ID_SCE: single_channel_element ();		
break;		
case ID_CPE: channel_pair_element();		
break;		
case ID_CCE: coupling_channel_element ();		
break;		
case ID_LFE: lfe_channel_element();		
break;		
case ID_DSE: data_stream_element();		
break;		
case ID_PCE: program_config_element ();		
break;		
case ID FIL: } fill_element();		
} byte _alignment(); *}*		

Таблица 13 — Синтаксис *single_channel_element*()

Синтаксис	Количество битов	Мнемоника
single_channel_element() *{* *element_instance_tag;* *individual_channel_stream(0);* *}*	4	*uimsbf*

Таблица 14 — Синтаксис *channel_pair_element* ()

Синтаксис	Количество битов	Мнемоника
channel_pair_element() *{* *element_instance_tag;* *common_window;* *if (common_window) {* *ics_info();* *ms_mask_present;* *if (ms_mask_present == 1) {* *for (g = 0; g < num_window_groups; g++) {* *for (sfb = 0; sfb < max_sfb; sfb++) {* *ms_used[g][sfb];* *}* *}* *}* *}* *individual_channel_stream(common_window);* *individual channel stream(common window);* *}*	 4 1 2 1	 *uimsbf* *uimsbf* *uimsbf* *uimsbf*

Таблица 15 — Синтаксис *ics_info* ()

Синтаксис	Количество битов	Мнемоника
ics_info() *{* *ics_reserved_bit;* *window_sequence;* *window shape;* *if (window sequence == EIGHT SHORT SEQUENCE) {* *max sfb;* *scale factor grouping;* *}* *else {* *max sfb;* *predictor data present;* *if (predictor data present) {* *predictor reset;* *if (predictor reset) {* *predictor reset group number;* *}* *for (sfb = 0; sfb< min(max sfb,* *PRED SFB MAX); sfb++) {* *prediction used[sfb];* *}* *}* *}* *}*	 1 2 1 4 7 6 1 1 5 1	 *bslbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf*

Таблица 16 — Синтаксис *individual_channel_stream* ()

Синтаксис	Количество битов	Мнемоника

individual_channel_stream(common_window)
{
global_gain;
if (!common_window)
ics_info();
section_data();
scale_factor_data();
pulse_data_present;
if (pulse_data_present) {
pulse data();
}
tns_data_present;
if (tns_data_present) {
tns data();
}
gain_control_data_present;
if (gain_control_data_present) {
gain control data();
}
spectral data();
}
	
	8
	
	1
	
	1
	
	1
		
		uimsbf
		
		uismbf
		
		uimsbf
		
		uimsbf

Таблица 17 — Синтаксис *section_data*()

Синтаксис	Количество битов	Мнемоника
section_data() *{* *if (window_sequence == EIGHT_SHORT_SEQUENCE)* *sect_esc_val = (1<<3) –1;* *else sect_esc_val = (1<<5) –1;* *for (g = 0; g < num window groups; g++) {* *k = 0;* *i = 0;* *while (k < max_sfb) {* *sect_cb[g][i];* *sect_len = 0;* *while (sect_len _incr == sect_esc_val) {* *sect_len += sect_esc_val;* *}* *sect_len += sect_len_incr;* *sect_start[g][i] = k;* *sect_end[g][i] = k+sect_len;* *for (sfb = k; sfb < k+sect_len; sfb++)* *sfb_cb[g][sfb] = sect_cb[g][i];* *к += sect_len;* *I++;* *}* *num_sec[g] = i;* *}* *}*	 4 {3; 5} 	 *uimsbf* *uismbf*

Таблица 18 — Синтаксис *scale_factor_data* ()

Синтаксис	Количество битов	Мнемоника
scale_factor_data() *{* *for (g = 0; g < num_window_groups; g++) {* *for (sfb = 0; sfb < max sfb; sfb++) {* *if (sfb_cb[g][sfb] != ZERO_HCB) {* *if (is_intensity(g,sfb))* *hcod_sf[dpcm_is_position[g][sfb]];* *else* *hcod_sf [dpcm_sf [g] [sfb]];* *}* *}* *}* *}*	 1..19 1..19	 vlclbf vlclbf

Таблица 19 — Синтаксис *tns_data* ()

Синтаксис	Количество битов	Мнемоника

tns_data()
{
for (w = 0; w < num windows; w++) {
n filt[w];
if (n_filt[w])
coef res[w];
for (filt = 0; filt < n filt[w]; filt++) {
length[w][filt];
order[w][filt];
if (order[w][filt]) {
direction[w][filt];
coef_compress[w][filt];
for (i = 0; i < order[w][filt]; i++)
coef[w][filt][i];
}
}
}
}
	
	1..2
	
	1
	
	{4;6}
	{3;5}
	
	1
	1
	2..4
		
		uimsbf
		
		uimsbf
		
		uimsbf
		uimsbf
		
		uimsbf
		uimsbf
		uimsbf

Таблица 20 — Синтаксис *spectral_data* ()

Синтаксис	Количество битов	Мнемоника
spectral_data() *{* *for (g = 0; g < num window groups; g++) {* *for (i = 0; i < num sec[g]; i++) {* *if (sect cb[g][i] != ZERO HCB &&* *sect_cb[g][i] <= ESC_HCB) {* *for (k = sect_sfb_offset[g][sect_start[g][i]];* *к < sect sfb offset[g][sect end[g][i]];) {* *if (sect cb[g][i]<FIRST PAIR HCB){* *hcod[sect_cb[g][i]][w][x][y][z];* *if (unsigned_cb[sect_cb[g][i]])* *quad_sign_bits;* *к += QUAD_LEN;* *} else {* *hcod[sect_cb[g][i]][y][z];*	 1..16 0..4	 *vlclbf* *bslbf*

Окончание таблицы 20

Синтаксис	Количество битов	Мнемоника
if (unsigned_cb[sect_cb[g][i]) *pair_sign_bits;* *к += PAIR_LEN;* *if (sect cb[g][i] == ESC_HCB) {* *if (y == ESC_FLAG)* *hcod_esc_y;* *if (z == ESC_FLAG)* *hcod_esc_z;* *}* *}* *}* *}* *}* *}*	1..15 0..2 5..21 5..21	*vlclbf* *bslbf* *vlclbf* *vlclbf*

Таблица 21 — Синтаксис *pulse_data* ()

Синтаксис	Количество битов	Мнемоника
pulse_data() *{* *number_pulse;* *pulse_start_sfb;* *for (i = 0; i < number_pulse+1; i++) {* *pulse_offset[i];* *pulse_amp[i];* *}* *}*	 2 6 5 4	 *uimsbf* *uimsbf* *uimsbf* *uimsbf*

Таблица 22 — Синтаксис *coupling_channel_element*()

Синтаксис	Количество битов	Мнемоника
coupling_channel_element() *{* *element_instance_tag;* *ind_sw_cce_f l ag;* *num_coupled_elements;* *num_gain_element_lists = 0;* *for (c = 0; c < num_coupled_elements+1; c++) {* *num_gain_element_lists++;* *cc_target_is_cpe[c];* *cc_target_tag_select[c];* *if (cc target is cpe[c]) {* *cc_l[c];* *cc_r[c];* *if (cc_l[c] && cc_r[c])* *num gain element lists++;* *}* *}* *cc_domain;* *gain_element_sign;* *gain_element_scale;* *individual_channel_stream(0);*	 4 1 3 1 4 1 1 1 1 2 1	 *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf*

Окончание таблицы 22

Синтаксис	Количество битов	Мнемоника
for (c = 1; c < num_gain_element_lists; c++) { *if (ind_sw_cce_flag) {* *cge = 1;* *} else {* *common_gain_element_present[c];* *cge = common_gain_element_present[c];* *}* *if (cge)* *hcod_sf[common_gain_element[c]];* *else{* *for (g = 0; g < num_window_groups; g++) {* *for (sfb = 0; sfb < max_sfb; sfb++) {* *if (sfb_cb[g][sfb] != ZERO_HCB);* *hcod sf[dpcm_gain_element[c][g][sfb]];* *}* *}* *}* *}* *}*	 1..19 1..19 	 *vlclbf* *vlclbf*

Таблица 23 — Синтаксис *lfe_channel_element*()

Синтаксис	Количество битов	Мнемоника
lfe_channel_element() *{* *element_instance_tag;* *individual_channel_stream(0);* *}*	 4 	 *uimsbf*

Таблица 24 — Синтаксис *data_stream_element* ()

Синтаксис	Количество битов	Мнемоника
data_stream_element() *{* *element_instance_tag;* *data_byte_align_flag;* *cnt = count;* *if (cnt == 255) {* *cnt += esc_count;* *if (data_byte_align_flag) {* *byte_alignment();* *for (i = 0; i < cnt; i++) {* *data_stream_byte[element_instance_tag][i];* *}* *}*	 4 1 8 8 8 	 *uimsbf* *uimsbf* *uimsbf* *uimsbf* *uimsbf*

Таблица 25 — Синтаксис *program_config_element* ()

Синтаксис	Количество битов	Мнемоника
program_config_element() *{* *element_instance_tag;* *profile;* *sampling_frequency_index;*	 4 2 4	 *uimsbf* *uimsbf* *uimsbf*

Окончание таблицы 25

Синтаксис	Количество битов	Мнемоника
num_front_channel_elements;	4	*uimsbf*
num_side_channel_elements;	4	*uimsbf*
num_back_channel_elements;	4	*uimsbf*
num_lfe_channel_elements;	2	*uimsbf*
num_assoc_data_elements;	3	*uimsbf*
num_valid_cc_elements;	4	*uimsbf*
mono_mixdown_present;	1	*uimsbf*
if (mono_mixdown_present ==1)		
mono_mixdown_element_number;	4	*uimsbf*
stereo_mixdown_present;	1	*uimsbf*
if (stereo_mixdown_present ==1)		
stereo_mixdown_element_number;	4	*uimsbf*
matrix_mixdown_idx_present;	1	*uimsbf*
if (matrix_mixdown_idx_present == 1) {		
matrix_mixdown_idx;	2	*uimsbf*
pseudo_surround_enable;	1	*uimsbf*
}		
for (i = 0; i < num_front_channel_elements; i++) {	1	*bslbf*
front_element_is_cpe[i];	4	*uimsbf*
front_element_tag_select[i];		
}	1	*bslbf*
for (i = 0; i < num_side_channel_elements; i++) {	4	*uimsbf*
side_element_is_cpe[i];		
side_element_tag_select[i];	1	*bslbf*
}	4	*uimsbf*
for (i = 0; i < num_back_channel_elements; i++) {		
back_element_is_cpe[i];	4	*uimsbf*
back_element_tag_select[i];		
}	4	*uimsbf*
for (i = 0; i < num_lfe_channel_elements; i++)		
lf e_element_tag_select[i];	1	*uimsbf*
for (i = 0; i < num_assoc_data_elements; i++)		
assoc_data_element_tag_select[i];	4	*uimsbf*
for (i = 0; i < num_valid_cc_elements; i++) {	8	*uimsbf*
cc_element_is_ind_sw[i];		
valid_cc_element_tag_select[i];	8	*uimsbf*
}		
byte_alignment();		
comment_field_bytes;		
for (i = 0; i < comment_field_bytes; i++)		
comment field data[i];		
}		

Таблица 26 — Синтаксис *fill_element*()

Синтаксис	Количество битов	Мнемоника
fill_element()		
{		
cnt = count;	4	*uimsbf*
if (cnt== 15)		
cnt += esc count –1;	8	*uimsbf*
while (cnt > 0) {		
cnt –= extension_payload(cnt);		
}		
}		

Таблица 27 — Синтаксис *gain_control_data*()

Синтаксис	Количество битов	Мнемоника
gain_control_data()		
{		
max_band;	2	*uimsbf*
if (window_sequence == ONLY_LONG_SEQUENCE) {		
for (bd = 1; bd <= max_band; bd++) {		
for (wd = 0; wd < 1; wd++) {		
adjust_num[bd][wd];	3	*uimsbf*
for (ad = 0; ad < adjust_num[bd][wd]; ad++) {		
alevcode[bd][wd][ad];	4	*uimsbf*
aloccode[bd][wd][ad];	5	*uimsbf*
}		
}		
}		
}	3	*uimsbf*
else if (window_sequence == LONG_START_SEQUENCE)		
{	4	*uimsbf*
for (bd = 1; bd <= max_band; bd++) {		
for (wd = 0; wd < 2; wd++) {	4	*uimsbf*
adjust_num[bd][wd];		
for (ad = 0; ad < adjust_num[bd][wd]; ad++) {	2	*uimsbf*
alevcode[bd] [wd] [ad];		
if (wd == 0)		
aloccode[bd] [wd] [ad];		
else		
aloccode[bd][wd][ad];		
}		
}		
}		
}		
else if (window sequence ==		
EIGHT_SHORT_SEQUENCE) {		
for (bd = 1; bd <= max_band; bd++) {		
for (wd = 0; wd < 8; wd++) {		
adjust_num[bd][wd];	3	*uimsbf*
for (ad = 0; ad < adjust_num[bd][wd]; ad++) {		
alevcode[bd] [wd] [ad];	4	*uimsbf*
aloccode[bd] [wd] [ad];	2	*uimsbf*
}		
}		
}		
}		
else if (window_sequence == LONG_STOP_SEQUENCE) {		
for (bd = 1; bd <= max_band; bd++) {		
for (wd = 0; wd < 2; wd++) {		
adjust_num[bd][wd];	3	*uimsbf*
for (ad = 0; ad < adjust_num[bd][wd]; ad++) {		
alevcode[bd][wd][ad];	4	*uimsbf*
if (wd == 0)		
aloccode[bd][wd][ad];	4	*uimsbf*
else	5	*uimsbf*
aloccode[bd][wd][ad];		
}		
}		
}		
}		
}		

Таблица 28 — Синтаксис *extension_payload* ()

Синтаксис	Количество битов	Мнемоника
extension_payload(cnt)		**uimsbf**
{		
extension_type;	**4**	
switch (extension type) {		
case EXT_DYNAMIC_RANGE:		
n = dynamic_range_info();		
return n;		Note1
case EXT_SBR_DATA:		
return_sbr_extension_data(id aac, 0);		Note1
case EXT_SBR_DATA_CRC:		
return_sbr_extension_data(id aac, 1);		**Uimsbf**
case EXT FILL DATA:		
fill_nibble; /* равен "0000" */	**4**	**Uimsbf**
for (i = 0; i < cnt–1; i++)		
fill_byte[i]; /* равен "10100101" */	**8**	
return cnt;		
case default:		**uimsbf**
for(i = 0; i<8*(cnt–1)+4;i++)		
other_bits[i];	**1**	
return cnt;		
}		
}		

Примечание — 1. *id_aac* является *id_syn_ele* соответствующего элемента *AAC* (*ID_SCE* или *ID_CPE*) или *SCE ID* в случае *CCE*.

Таблица 29 — Синтаксис *dynamic_range_info* ()

Синтаксис	Количество битов	Мнемоника
dynamic range info()		
{		
n = 1; drc_num_bands = 1;		
pce_tag_present; if	1	*uimsbf*
(pce_tag_present == 1) {		
pce_ instance_tag;	4	*uimsbf*
drc_tag_reserved_bits;	4	*uimsbf*
excluded_chns_present;	1	*Uimsbf*
if (excluded_chns_present == 1) {		
n += excluded_channels();		
drc_bands_present; if	1	*Uimsbf*
(drc_bands_present == 1) {		
drc_band_incr;	4	*uimsbf*
drc_bands_reserved_bits;	4	*uimsbf*
drc_num_bands = drc_num_bands + drc_band_incr;		
for (i = 0; i < drc_num_bands; i++) {		
drc_band_top[i];	8	*uimsbf*
prog_ref_level_present; if	1..7	*uimsbf*
(prog_ref_level_present == 1) {		
prog_ref_level;	1	*uimsbf*
prog_ref_level_reserved_bits;	1	*uimsbf*
for (i = 0; i < drc_num_bands; i++)		*uimsbf*
{ dyn_rng_sgn[i]; dyn_rng_ctl[i];	7	*uimsbf*
return n;		

Таблица 30 — Синтаксис *excluded_channels* ()

Синтаксис	Количество битов	Мнемоника
excluded_channels() *{* *n = 0;* *num_excl_chan = 70;* *for (i = 0; i < 7; i++)* *exclude_mask[i];* *while (additional_excluded_chns[n–1] == 1) {* *for (i = num_excl_chan; i < num_excl_chan+7; i++)* *exclude_mask[i];* *num_excl_chan += 7;* *return n;*	 1 1 1	 *uimsbf* *uimsbf* *uimsbf*

5 Профили и функциональная совместимость профилей

5.1 Состав профилей

Стандартом устанавливаются три профиля (таблица 31):

Основной профиль

Профиль пониженной сложности

Профиль с масштабируемой частотой дискретизации

В *program_config_element()* и *adts_fixed_header()* поле из двух битов указывает на используемый профиль:

Таблица 31 — Профили

Индекс	Профиль
0	Основной профиль
1	Профиль пониженной сложности (LC)
2	Профиль с масштабируемой частотой дискретизации (SSR)
3	(Зарезервированный)

5.1.1 Основной профиль

Основной профиль используется тогда, когда затраты в памяти не имеют особого значения и доступны существенные вычислительные ресурсы. Для обеспечения лучшего возможного сжатия могут использоваться все инструменты, за исключением инструмента регулирования усиления. В потоке битов основного профиля должна содержаться только одна программа (*program_config_element()*). Программа в потоке битов основого профиля не должна содержать никаких элементов моно или стерео сведения.

5.1.2 Профиль с пониженной сложностью

Профиль с пониженной сложностью используется тогда, когда ресурс *RAM*, вычислительных мощностей и требования сжатия ограничены определенными рамками. В профиле с пониженной сложностью не используются предсказание и инструмент регулирования усиления, кроме того, порядок *TNS* ограничен. В потоке битов профиля с пониженной сложностью должна содержаться только одна программа (*program_config_element ()*). Программа в потоке битов профиля с пониженной сложностью не должна содержать никаких элементов моно или стерео сведения.

5.1.3 Профиль с масштабируемой частотой дискретизации

В профиле с масштабируемой частотой дискретизации требуется инструмент регулирования усиления. Предсказание и спаривание каналов не разрешены, порядок *TNS* и ширина канала ограничены. Для самого низкого из 4 *PQF* диапазонов регулирование усиления не используется. В случае сокращения полосы частот звукового сигнала профиль *SSR* масштабируется соответственно сложности. В потоке битов

профиля с масштабируемой частотой дискретизации должна содержаться только одна программа (*program_config_element ()*). Программа в потоке битов профиля с масштабируемой частотой дискретизации не должна содержать никаких элементов моно или стерео сведения.

5.1.4 Соглашение о присвоении имен для декодеров и потоков битов MPEG-2 AAC

Декодер или поток битов могут быть определены как *A.L.I.D* <Имя профиля> MPEG-2 AAC декодер или поток битов, где *A* — число основных звуковых каналов, *L* — число каналов *LFE*, *I* — число независимо переключаемых каналов, *D* — число зависимо переключаемых каналов и <Имя профиля> — фактическое имя профиля. Пример: название «декодер 5.1.1.1 основого профиля MPEG-2 AAC» соответствует декодеру, способному декодировать 5 основных звуковых каналов, один канал *LFE* и по одному независимо и зависимо переключаемых CCE, используя с каждым из каналов определенный профиль. Это может быть сокращено как *M*.5.1.1.1, где *M* указывает на основной профиль декодера. Точно так же, декодер с профилем пониженного уровня сложности может быть определен как *L*, а профиль *SSR* — *S*.

5.1.5 Соглашение о присвоении имен для декодеров и потоков битов MPEG-2 AAC + MPEG 4 SBR

Декодер или поток битов, дополнительно соответствующий MPEG–4 *AOT SBR* на определенном уровне может быть обозначен аналогично, путем добавления "+ *SBR* / *X* [*HQ*/*LP*]" к имени декодера, где *X* — уровень декодера/потока битов профиля HE-AAC с характеристиками, определенными в ИСО/МЭК 14496-3 [3]. Пример. декодер 5.1.1.1 основного профиля MPEG-2 AAC + SBR / 5 HQ.

5.1.6 Минимальное количество основных звуковых каналов и профилей, поддерживаемых декодером

Чтобы обеспечить определенный уровень функциональной совместимости, определяются следующие минимальные возможности декодеров для данного профиля и числа основных звуковых каналов (таблица 32).

Таблица 32 — Минимальные возможности декодера с точки зрения конфигурации каналов

Число основных звуковых каналов	Основной профиль	Профиль с пониженной сложностью	Профиль *SSR*
1	1.0.0.0	1.0.0.0	1.0.0.0
2	2.0.0.0	2.0.0.0	2.0.0.0
3	3.0.1.0	3.0.0.1	3.0.0.0
4	4.0.1.0	4.0.0.1	4.0.0.0
5	5.1.1.1	5.1.0.1	5.1.0.0
7	7.1.1.2	7.1.0.2	7.1.0.0

5.1.7 Параметры инструментов, зависящие от профиля

В соответствии с используемым профилем значение константы *TNS_MAX_ORDER* устанавливается для длинных окон следующим образом: *TNS_MAX_ORDER*==20 для основного профиля, *TNS_MAX_ORDER*==12 для профиля с пониженной сложностью и профиля с масштабируемой частотой дискретизации. Для коротких окон константа *TNS_MAX_ORDER*==7 для всех профилей.

Согласно частоте дискретизации и используемому профилю значение константы *TNS_MAX_BANDS* устанавливается в соответствии с таблицей 33.

Таблица 33 — Значения *TNS_MAX_BANDS* в зависимости от профиля и частоты дискретизации

Частота дискретизации, Гц	Низкая сложность/основной профиль (длинные окна)	Низкая сложность/основной профиль (короткие окна)	Профиль с масштабируемой частотой дискретизации (длинные окна)	Профиль с масштабируемой частотой дискретизации (длинные окна)
96000	31	9	28	7
88200	31	9	28	7
64000	34	10	27	7
48000	40	14	26	6
44100	42	14	26	6

Окончание таблицы 33

Частота дискретизации, Гц	Низкая сложность/основной профиль (длинные окна)	Низкая сложность/основной профиль (короткие окна)	Профиль с масштабируемой частотой дискретизации (длинные окна)	Профиль с масштабируемой частотой дискретизации (длинные окна)
32000	51	14	26	6
24000	46	14	29	7
22050	46	14	29	7
16000	42	14	23	8
12000	42	14	23	8
11025	42	14	23	8
8000	39	14	19	7

5.2 Функциональная совместимость профилей

Любой поток битов данного профиля (таблица 34), число основных звуковых каналов, каналов *LFE*, независимых и зависимых каналов которого меньше или равно соответствующему числу каналов, поддерживаемых декодером того же самого профиля, может быть декодирован этим декодером.

Таблица 34 и рисунок 3 описывают функциональную совместимость трех профилей.

Таблица 34 — Функциональная совместимость профилей

Профиль декодера	Профиль кодера		
	Основной профиль	Профиль LC	Профиль SSR
Основной профиль	да	да	нет*
Профиль *LC*	нет	да	нет*
Профиль *SSR*	нет	нет**	да

* Эти потоки могут быть декодированы, если декодер основного профиля или LC профиля будет в состоянии обнаружить, но не декодировать информацию о регулировании усиления, однако у восстановленного звука будет ограниченный частотный диапазон.

** Эти потоки могут быть декодированы, однако частотный диапазон декодируемого сигнала будет ограничен приблизительно до 5 кГц, что соответствует неискаженной части первой полосы фильтра *PQMF*.

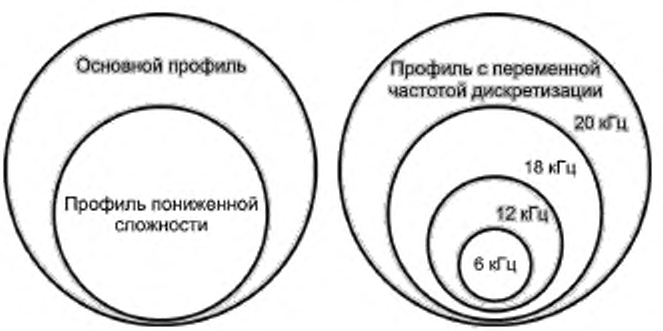


Рисунок 3 — Функциональная совместимость профилей

6 Общая структура данных

6.1 Форматы обмена AAC

6.1.1 Краткий обзор

Блок *raw_data_block*() содержит все данные, относящиеся к аудио (включая вспомогательные данные). Кроме того, дополнительная информация, такая как *sampling_frequency*, необходима, чтобы

полностью описать аудио последовательность. Формат обмена аудиоданными (ADIF) содержит все элементы, которые необходимы для описания потока битов согласно этому стандарту.

В определенных приложениях некоторые или все элементы синтаксиса, подобно тем, которые определены в заголовке ADIF, например, *sampling_rate*, могут быть известны декодеру за счет других средств и, следовательно, не появляются в потоке битов.

Кроме того, может требоваться дополнительная информация, которая изменяется от блока до блока (например, для улучшения считывания или устойчивости к ошибкам). Поэтому транспортные потоки могут быть разработаны для определенного приложения и не определяются в этом стандарте. Однако здесь описывается один нестандартный транспортный поток под названием Транспортный поток аудиоданных (*ADTS*). Он может использоваться для приложений, в которых декодер может считать этот поток.

6.1.2 Формат обмена аудиоданными (*ADIF*)

6.1.2.1 Краткий обзор

Формат обмена аудиоданными (*ADIF*) содержит один заголовок в начале последовательности, сопровождаемый *raw_data_stream()*. *raw_data_stream*() может не содержать дальнейших *program_config_element* () элементов.

ADIF применим только для систем с определенной точкой начала декодирования, не подразумевающих начало процесса декодирования с заранее неизвестной позиции потока аудиоданных. Он может использоваться в качестве формата обмена, в котором содержится вся информация, необходимая для декодирования и воспроизведения аудиоданных.

6.1.2.2 Определения

6.1.2.2.1 Функции данных

adif_sequence ()	Последовательность согласно формату обмена аудиоданных (таблица 2).
adif_header ()	Заголок формата обмена аудиоданными, располагается в начале adif_sequence (таблица 3).
byte_alignment ()	Выравнивание относительно первого бита заголовка.
raw_data_stream ()	См. 6.2.1 и таблицу 11.
program_config_element ()	Содержит информацию о конфигурации для одной программы (таблица 3). См. 6.5.

6.1.2.2.2 Элементы данных

adif_id	Идентификатор формата обмена аудиоданными. Его значение — 0x41444946 (старший бит первый), ASCII код строки «ADIF» (таблица 3).
copyright_id_present	Указывает, присутствует ли copyright_id (таблица 3).
copyright_id	Поле состоит из 8-битового *copyright_identifier*, сопровождаемого 64-битовым *copyright_number* (таблица 3). *copyright_number* — значение, которое идентифицирует защищенный авторским правом материал.
original_copy	Определение элемента данных *copyright*.
home	Определение элемента данных *original/copy*.
bitstream_type	флаг, указывающий на тип потока битов (таблица 3): '0' — постоянный поток битов. Этот поток битов может быть передан по каналу с постоянной скоростью; '1' — поток битов с переменной скоростью. Этот поток битов не предназначен для передачи по каналам с постоянной скоростью.
bitrate	беззнаковое целое число размером 23 бита, указывающее на скорость передачи потока битов в бит/с в случае постоянного потока битов или на максимальную пиковую скорость передачи (измеренную за один фрейм) в случае потоков битов с переменной скоростью. Значение 0 указывает, что скорость передачи не известна (таблица 3).
num_program_config_element	Количество элементов *program_config_element*(), определенных для текущей *adif_sequence* (); равно *num_program_config_element*+1 (таблица 3). Минимальное значение 0 указывает на один элемент *program_config_element()*.
adif_buffer_fullness	состояние резервуара битов после кодирования первого raw_*data_block()* в *adif_sequence()*. Передается как количество доступных битов в резервуаре битов (таблица 3).

6.1.2.2.3 Элементы справки

data_available () Функция, которая возвращает '1', если данные доступны, иначе '0'.

6.1.3 Транспортный поток аудиоданных (ADTS)

6.1.3.1 Краткий обзор

Транспортный поток аудиоданных (ADTS) будет распознан декодерами как «Уровень 4» потока битов.

Фиксированный заголовок ADTS содержит синхрослово плюс все части заголовка, которые необходимы для декодирования и которые не изменяются от фрейма к фрейму. Переменный заголовок ADTS содержит данные заголовка, которые изменяются от фрейма к фрейму.

6.1.3.2 Определения

6.1.3.2.1 Функции данных

adts_sequence ()	Последовательность согласно транспортному потоку аудиоданных ADTS (таблица 4).
adts_frame ()	Фрейм ADTS, состоящий из фиксированного заголовка, переменного заголовка, опциональной проверки на наличие ошибок и конкретного количества блоков *raw_data_block()* (таблица 5).
adts_fixed_header ()	Фиксированный заголовок ADTS. Информация в этом заголовке не изменяется от фрейма к фрейму. Он повторяется в каждом фрейме, чтобы обеспечить произвольный доступ к потоку битов (таблица 8).
adts_variable_header ()	Переменный заголовок ADTS. Этот заголовок передается в каждом фрейме, так же как и фиксированный заголовок, однако содержит данные, которые изменяются от фрейма к фрейму (таблица 9).
adts_error_check ()	Следующие биты защищены и поступают в алгоритм CRC в порядке появления: все биты *adts_fixed_header()* все биты adts_variable_header() первые 192 бита любого: *single_channel_element()* *channel_pair_element()* *coupling_channel_element()* *lfe_channel_element()* Первые 128 битов второго *individual_channel_stream* () в *channel_pair_element* () должны быть защищены. Вся информация в любом *program_config_element ()* или *data_stream_element* () должна быть защищена. Любой элемент с фактической длиной меньше указанной продолжительности поля защиты 128 или 192 битов дополняется нулями до указанной продолжительности поля защиты для CRC расчета. Биты *id_syn_ele* должны быть исключены из CRC защиты. Если продолжительность CPE короче 192 битов, нулевые биты добавляются, чтобы достигнуть продолжительности 192 битов. Кроме того, если первый ICS из CPE заканчивается на N-м бите (N <192), первые (192 – N) битов второго ICS защищаются дважды, каждый раз в порядке появления. Например, если второй ICS начинается со 190 бита CPE, первые 3 бита второго ICS защищаются дважды. Наконец, если продолжительность второго ICS короче128 битов, нулевые биты добавляются, чтобы достигнуть продолжительности 128 битов.
adts_header_error_check ()	Следующие биты защищены и поступают в алгоритм CRC в порядке появления: все биты adts_fixed_header () все биты *adts_variable_header*() все биты каждого *raw_data_block_position[i]*.
adts_raw_data_block_error_check ()	Относительно *i*-ой проверки *adts_raw_data_block_error_check()*, биты *i*–го блока *raw_data_block* () защищены и поступают в алгоритм CRC в порядке появления, согласно тому, что определено для *adts_error_check* () за исключением того, что никакие биты заголовка не рассматриваются.
raw_data_block ()	см. 6.2.1 и таблицу 12.

6.1.3.2.2 Элементы данных

raw_data_block_position[i] Начальная позиция *i*–го блока *raw_data_block*() в *adts_frame*(), определяемая как смещение в байтах от начальной позиции первого блока *raw_data_block*() в *adts_frame*().

crc_check Данные об обнаружении ошибок CRC, произведенные согласно таблицам 6, 7 и 10.

syncword Строка битов '1111 1111 1111'. См. таблицу 8

ID Идентификатор MPEG, установлен в '1'. См. таблицу 8.

layer Указывает, какой уровень используется. Установлен в '00'. См. таблицу 8.

protection_absent Указывает, присутствуют ли данные error_check().

profile Используемый профиль. См. таблицу 8.

sampling_frequency_index Указывает на частоту дискретизации согласно таблицам 8 и 35:

Таблица 35 — Частота дискретизации в зависимости от *sampling_frequency_index*

sampling_frequency_index	Частота дискретизации, Гц
0x0	96000
0x1	88200
0x2	64000
0x3	48000
0x4	44100
0x5	32000
0x6	24000
0x7	22050
0x8	16000
0x9	12000
0xa	11025
0xb	8000
0xc	резерв
0xd	резерв
0xe	резерв
0xf	резерв

private_bit См. таблицу 8.

channel_configuration Указывает на используемую конфигурацию канала. Если channel_configuration > 0, конфигурация канала дается в таблице 42 и 6.5.3.1. Если channel_configuration==0, конфигурация канала не определяется в заголовке и должна быть задана элементом *program_config_element()*, первым элементом синтаксиса первого *raw_data_block()* после заголовка (см. 6.5.3.2) или является неявной конфигурацией (см. 6.5.3.3), или должна быть известна из приложения (таблица 8).

original_copy См. определение в 6.1.2.2.2.

home См. определение в 6.1.2.2.2.

copyright_identification_bit Один бит 72-битного поля идентификации авторского права (см. *copyright_id* выше). Биты этого поля передаются фрейм за фреймом; первый бит *copyright_identification_start* равен '1'. Поле состоит из 8-битного *copyright_identifier*, сопровождаемого 64-битным *copyright_number*. *copyright_number* — значение, которое идентифицирует защищенный авторским правом материал. См. таблицу 9.

copyright_identification_start Один бит, указывающий, что *copyright_identification_bit* в этом аудио фрейме — первый бит 72-битной идентификации авторского права. Если идентификация авторского права не передается, этот бит должен быть равен '0'. '0' идентификация авторского права в этом аудио фрейме не используется '1' идентификация авторского права в этом аудио фрейме используется, см. таблицу 9.

aac_frame_length — Размер фрейма, включая заголовки *error_check* в байтах (таблица 9).

adts_buffer_fullness — Состояние резервуара битов в ходе кодирования фрейма ADTS, до первого блока *raw_data_block*() включительно и, опционально, после *adts_raw_data_block_error_check*(). Передается как количество доступных битов в резервуаре битов, деленное на NCC, деленное на 32 и округленное до целого значения (таблица 9). Шестнадцатеричное значение 7FF сигнализирует, что поток битов является потоком с переменной скоростью. В этом случае заполнение буфера не применимо.

number_of_raw_data_blocks_in_frame — Количество блоков *raw_data_block* (), которые мультиплексируютс в *adts_frame*(), равно *number_of_raw_data_blocks_in_frame* + 1. Минимальное значение 0 указывает на один *raw_data_block*() (таблица 9).

6.2 Необработанные данные

6.2.1. Определения

6.2.1.1 Функции данных

raw_data_stream () — Последовательность блоков *raw_data_block* ().

raw_data_block () — Блок необработанных данных, который содержит аудиоданные за период времени, соответствующий 1024 отсчетам, относящейся к ним информации и других данных. Имеется семь синтаксических элементов, идентифицируемых по элементу данных *id_syn_ele*. Элементы audio_channel_element () в одном *raw_data_stream* () и одном *raw_data_block*() должны иметь одну и только одну частоту дискретизации. В *raw_data_block* () возможно появление того же самого синтаксического элемента несколько раз, однако *element_instance_tag* должны иметь различные 4 бита, за исключением *data_stream_element*() и *fill_element*(). Поэтому в одном *raw_data_block* () может быть от 0 до максимум 16 случаев появления любого синтаксического элемента, за исключением *data_stream_element* () и fill_element (), где это ограничение не применяется. Если встречаются несколько *data_stream_element*() с одинаковым *element_instance_tag*, они являются частью одного потока данных. У *fill_element* () нет *element_instance_tag* (так как контент не требует последующей ссылки), и может встретиться любое число раз. Конец блока *raw_data_block()* обозначается специальным *id_syn_ele* (TERM), который встречается в *raw_data_block* () лишь единожды (таблица 12).

single_channel_element () — Сокращенно SCE. Синтаксический элемент потока битов, содержащий кодированные данные для единственного звукового канала. *single_channel_element*() в основном состоит из *individual_channel_stream*(). В блоке необработанных данных может присутствовать до 16 таких элементов, у каждого из которых должен быть уникальный *element_instance_tag* (таблица 13).

channel_pair_element () — Сокращенно CPE. Синтаксический элемент потока битов, содержащий данные для пары каналов. *channel_pair_element* () состоит из двух *individual_channel_stream* () и дополнительной информации об объединенном кодировании. Эти два канала могут иметь общую служебную информацию. Элемент *channel_pair_element* () имеет те же самые ограничения, что и single_channel_element, *element_instance_tag*, и число появлений (таблица 14).

coupling_channel_element () — Сокращенно CCE. Синтаксический элемент, который содержит аудиоданные для спаренного канала. В спаренном канале представлена информация об одном блоке многоканальной интенсивности или диалогах многоязычных программ. Правила для количества *coupling_channel_element* () и instance_tag аналогичны правилам для *single_channel_element* () (таблица 22). См. 10.3.

lfe_channel_element () — Сокращенно LFE. Синтаксический элемент, который содержит канал с низкой частотой дискретизации. Правила для числа *lfe_channel_element*() и instance_tag аналогичны правилам для *single_channel_element* () 's (таблица 23). См. 6.4.

audio_channel_element () — Общее обозначение для single_channel_element (), *channel_pair_element* (), coupling_ channel_element () и *lfe_channel_element* ().

program_config_element () — Сокращенно PCE. Синтаксический элемент, который содержит данные конфигурации программы. Правила для числа *program_config_element* () и *element_instance_tag* аналогичны правилам для single_channel_element () (таблица 25). PCE должны следовать перед всеми другими синтаксическими элементами в *raw_data_block*(). См. 6.5.

fill_element () — Сокращенно FIL. Синтаксический элемент, который содержит данные заполнения. Может присутствовать любое число элементов, которые могут встретиться в любом порядке в *raw_data_block* (таблица 26). См. 6.7.

data_stream_element () — Сокращенно DSE. Синтаксический элемент, который содержит данные. Снова, есть 16 *element_instance_tags*. Однако нет ограничения на количество *data_stream_element()* ни с каким *instance_tag*, в то время как единственный поток данных может продолжаться через многократный *data_stream_element* () с тем же самым *instance_tag* (таблица 24). См. 6.5.3.

byte_alignment () — Выравнивание относительно первого бита *raw_data_block*().

6.2.1.2 Элементы данных

id_syn_ele — Элемент данных, который идентифицирует синтаксический элемент или конец *raw_data_block*() (таблица 12):

Таблица 36 — Идентификация синтаксических элементов

ID	*Код*	*Сокращение*	*Синтаксический элемент*
ID_SCE	0x0	*SCE*	*single_channel_element()*
ID_CPE	0x1	*CPE*	*channel_pair_element()*
ID_CCE	0x2	*CCE*	*coupling_channel_element()*
ID_LFE	0x3	*LFE*	*lfe_channel_element()*
ID_DSE	0x4	*DSE*	*data_stream_element()*
ID_PCE	0x5	*PCE*	*program_config_element()*
ID_FIL	0x6	*FIL*	*fill_element()*
ID_END	0x7	*TERM*	

element_instance_tag — Уникальный тег для всех синтаксических элементов, кроме *fill_element*(). Все синтаксические элементы, содержащие теги, могут встречаться более одного раза, но, за исключением *data_stream_element*(), должны иметь уникальный *element_instance_tag* в каждом *raw_data_block*(). Этот тег также используется для указания на синтаксические элементы аудио в *single_channel_element()*, *channel_pair_element()*, *lfe_channel_element()*, *data_channel_element*() и *coupling_channel_element*() в *program_config_element*() и обеспечивает возможность использования до 16 независимых *program_config_element*() (таблицы 13, 14, 22, 23, 24, 25, 26).

6.2.2 Требования к буферу

6.2.2.1 Минимальный входной буфер декодера

Следующие правила используются для вычисления максимального количества битов во входном буфере как для потока битов в целом, для любой данной программы, так и для любого данного *SCE/CPE/CCE*:

Входной размер буфера составляет 6144 бита на *SCE* или независимо переключаемого *CCE*, плюс 12288 битов на *CPE* (6144**NCC*). Размеры общего буфера и отдельного буфера ограничиваются так, чтобы предел буферизации мог быть вычислен для всего потока битов, всей программы или для отдельного audio_channel_element(), позволяя декодеру разбивать многоканальный поток битов на отдельные моно и стерео потоки битов, которые декодируются отдельными моно и стерео декодерами соответственно. Все биты для *LFE* или зависимого *CCE* должны выделяться исходя из общих требований к буферу, основанных на независимых *CCE*, *SCE* и *CPE*. Кроме того, все биты, требуемые для любых *DSE*, *PCE*, *FIL*, фиксированных заголовков, переменных заголовков, *byte_alignment* и *CRC*, должны быть также выделены исходя из тех же самых общих требований к буферу.

6.2.2.2 Резервуар битов

Управление резервуаром битов осуществляется в кодере. Максимальный резервуар битов в кодере зависит от *NCC* и средней скорости передачи. Максимальный размер резервуара битов для каналов с постоянной скоростью передачи может быть вычислен путем вычитания среднего числа битов на блок из минимального входного размера буфера декодера. Например, при 96 кбит/с для сигнала стерео на частоте дискретизации 44,1 кГц среднее число битов на блок (mean_framelength) (96000 *bit/s*/44100 *1/s** 1024) = = 2229,1156 .... Это приводит к максимальному размеру резервуара битов (*max_bit_reservoir*) *INT* (12288 битов – 2229,1156 ...) = 10058. Для каналов с переменной скоростью передачи кодер должен работать так, чтобы требования к входному буферу не превышали минимальный входной буфер декодера.

Состояние резервуара битов (*bit_reservoir_state*) передается в поле *buffer_fullness* либо как состояние резервуара битов, округленного до целого значения (*adif_buffer_fullness*), либо как состояние резервуара битов, разделенного на NCC, разделенного на 32 и округленного до целого значения (*adts_buffer_fullness*).

bit_reservoir_state последующих фреймов может быть получен следующим образом:

bit reservoir_state[frame] = bit_reservoir_state[frame – 1] + mean_framelength – framelength[frame].

Framelengths должны быть выбраны так, чтобы следующее условие выполнялось:

$0 \le bit\_reservoir\_state[frame] \le max\_bit\_reservoir$.

6.2.2.3 Максимальная скорость передачи

Максимальная скорость передачи зависит от частоты дискретизации звука. Она может быть вычислена по минимальному размеру входного буфера согласно формуле

$$\frac{6144\frac{bit}{block}}{1024\frac{samples}{block}} gsampling\_frequencygNSS.$$

В таблице 37 содержится несколько примеров максимальных скоростей передачи на канал в зависимости от используемой частоты дискретизации.

Таблица 37 — Максимальная скорость передачи в зависимости от частоты дискретизации

sampling_frequency	максимальная скорость передачи / *NCC*
48 кГц	288 кбит/с
44,1 кГц	264,6 кбит/с
32 кГц	192 кбит/с

6.2.3 Процесс декодирования

Предполагая, что начало *raw_data_block()* известно, он может быть декодирован без использования дополнительной информации транспортного уровня и содержит 1024 отсчета звукового сигнала на один выходной канал. Частота дискретизации звукового сигнала, *sampling_frequency_index*, может быть указана в *program_config_element()* или специальным образом в зависимости от приложения. В последнем случае *sampling_frequency_index* должен быть обнаружен в потоке битов.

Так как данная частота дискретизации связана только с одной таблицей частот дискретизации, и так как максимальная гибкость требуется в диапазоне возможных частот дискретизации, следующая таблица должна использоваться, чтобы связать требуемую частоту дискретизации с желаемой.

Таблица 38 — Соответствие частот дискретизации

Диапазон частот, Гц	Частота дискретизации, Гц
f> = 92017	96000
92017> f> = 75132	88200
75132> f> = 55426	64000
55426> f> = 46009	48000
46009> f> = 37566	44100
37566> f> = 27713	32000

Окончание таблицы 38

Диапазон частот, Гц	Частота дискретизации, Гц
27713> f> = 23004	24000
23004> f> = 18783	22050
18783> f> = 13856	16000
13856> f> = 11502	12000
11502> f> = 9391	11025
9391> f	8000

raw_data_stream поддерживает кодирование каналов как с постоянной скоростью, так и с переменной. В каждом случае структура потока битов и работа декодера идентичны, за исключением некоторых незначительных отличий.

Для каналов с постоянной скоростью передачи кодер может добавлять элемент FIL, чтобы увеличить скорость до требуемого уровня. Декодер, считывающий данные канала с постоянной скоростью передачи, должен накопить минимальное число битов в его входном буфере до начала декодирования так, чтобы не произошло переполнение выходного буфера. В случае переменной скорости каждый raw_data_block() может обладать минимальной длиной (скоростью) так, чтобы достигалось требуемое качество звучания, и в декодере отсутствовали минимальные входные требования к данным до начала декодирования.

Примеры самых простых потоков битов:

Сегмент потока битов	Выходной сигнал
<SCE> <TERM> <SCE> <TERM> ...	Моно сигнал
<CPE> <TERM> <CPE> <TERM>...	Стерео сигнал
<SCE> <CPE> <CPE> <LFE> <TERM> <SCE> <CPE> <CPE> <LFE> <TERM> ...	Сигнал 5.1,

где угловые скобки (<>) используются, чтобы разграничить синтаксические элементы. Для моно сигнала у каждого SCE должно быть то же самое значение в его *element_instance_tag*, и точно так же для сигнала стерео у каждого CPE должно быть то же самое значение в его *element_instance_tag*. Для сигналов 5.1 у каждого SCE должно быть то же самое значение в его *element_instance_tag*, каждый CPE, связанный с фронтальной парой каналов, должен иметь то же самое значение в его *element_instance_tag*, и каждый CPE, связанный с последней парой каналов, должен иметь то же самое значение в его *element_instance_tag*.

Если эти потоки битов должны быть переданы по каналу с постоянной скоростью передачи, то они могут включать в себя элемент *fill_element*() для подстройки мгновенной скорости передачи. В этом случае пример кодированного сигнала стерео

<CPE> <FIL> <TERM> <CPE> <FIL> <TERM> ...

Если потоки битов должны переносить вспомогательные данные при передаче по каналу с постояной скоростью, то пример кодированного сигнала стерео

<CPE><DSE><FIL><TERM><CPE><DSE><FIL><TERM>....

Все элементы data_stream_element() имеют одинаковый *element_instance_tag*, если они — часть одного потока данных.

6.3 Элемент одиночного канала (*SCE*), элемент парного канала (*CPE*) и поток индивидуального канала (*ICS*)

6.3.1 Определения

6.3.1.1 Элементы данных

common_window Флаг, указывающий на совместное использование общего *ics_info*() двумя *individual_channel_stream*(). В случае совместного использования *ics_info*() является частью *channel_pair_element*() и должен использоваться для обоих каналов. Иначе *ics_info*() является частью каждого *individual_channel_stream*() (таблица 14).

ics_reserved_bit Флаг, зарезервированный для будущего использования. Должен быть равен '0'.

window_sequence Указывает на последовательность окон, как определено в таблице 44 (таблица 15).

window_shape 1-битное поле, определяющее тип окна для переключения окон (таблица 15).

max_sfb Количество полос масштабных коэффициентов на группу (таблица 15).

scale_factor_grouping Битовое поле, которое содержит информацию о группировке коротких спектральных данных (таблица 15).

6.3.1.2 Функции данных

individual_channel_stream () — Содержит данные, необходимые для декодирования одного канала (таблица 16).

ics_info () — Содержит служебную информацию, необходимую для декодирования *individual_channel_stream*(). Потоки *individual_channel_stream ()* элемента *channel_pair_element* () могут совместно использовать один общий *ics_info* () (таблица 15).

6.3.1.3 Элементы справки

scalefactor window band — Термин для полос масштабных коэффициентов в пределах окна, см. таблицы 45—57.

scalefactor band — Термин для полос масштабных коэффициентов в пределах группы. В случае *EIGHT_SHORT_SEQUENCE* и группировки, полоса масштабных коэффициентов может содержать несколько масштабных коэффициентов окна соответствующей частоты. Для всех остальных window_sequences полосы масштабных коэффициентов и полосы масштабных коэффициентов в окне идентичны.

g — Индекс группы.

win — Индекс окна в пределах группы.

sfb — Индекс полосы масштабных коэффициентов в пределах группы.

swb — Индекс окна масштабных коэффициентов в пределах окна.

bin — Индекс коэффициента.

num_ window_groups — Количество групп окон, которые совместно используют один набор масштабных коэффициентов.

window_group_length[g] — Число окон в каждой группе.

bit_set (bit_field, bit_num) — Функция, которая возвращает значение *bit_num* поля *bit_field* (самый правый бит — бит 0).

num_windows — Количество окон в фактической последовательности окон.

num_swb_long_window — Количество полос масштабных коэффициентов для длинных окон. Это число должно быть выбрано в зависимости от частоты дискретизации. См. 6.9.

num_swb_short_window — Количество оконных масштабных коэффициентов для коротких окон. Это число должно быть выбрано в зависимости от частоты дискретизации. См. 6.9.

num_swb — Количество оконных полос масштабных коэффициентов для коротких окон в случае EIGHT_SHORT_SEQUENCE, в противном случае — количество оконных полос масштабных коэффициентов для длинных окон.

swb_offset_long_window [swb] — Таблица, содержащая индекс самого низкого спектрального коэффициента полосы масштабных коэффициентов sfb для длинных окон. Эта таблица должна быть выбрана в зависимости от частоты дискретизации. См. 6.9.

swb_offset_short_window [swb] — Таблица, содержащая индекс самого низкого спектрального коэффициента полосы масштабных коэффициентов sfb для коротких окон. Эта таблица должна быть выбрана в зависимости от частоты дискретизации. См. 6.9

swb_offset [swb] — Таблица, содержащая индекс самого низкого спектрального коэффициента полосы масштабных коэффициентов sfb для коротких окон в случае *EIGHT_SHORT_SEQUENCE*, иначе для длинных окон.

sect_sfb_offset [g] [section] — Таблица, которая содержит номер первого коэффициента для *section_data* () в пределах группы. Это смещение зависит от *window_sequence* и *scale_factor_grouping.*

sampling_frequency_index — Указывает на частоту дискретизации согласно таблицам 8 и 35.

6.3.2 Процесс декодирования

6.3.2.1 Декодирование *single_channel_element* () и *channel_pair_element* ()

Элемент *single_channel_element* () составляется из *element_instance_tag* и *individual_channel_stream*. В этом случае *ics_info* () всегда располагается в *individual_channel_stream*.

Элемент *channel_pair_element* () начинается с *element_instance_tag* и флага *common_window*. Если *common_window* равен '1', то *ics_info*() используется совместно для двух *individual_channel_stream* элементов, и информация о *MS* передается. Если *common_window* равен '0', то *ics_info*() существует в каждом *individual_channel_stream* и информация *MS* не представлена.

6.3.2.2 Декодирование *individual_channel_stream* ()

В *individual_channel_stream* используется следующий порядок декодирования:

- получить *global_gain*;
- получить *ics_info* () (искать в потоке битов, если общая информация не присутствует);
- получить *section_data* ();
- получить *scalefactor_data* (), если есть;
- получить *pulse_data* (), если есть;
- получить *tns_data* (), если есть;
- получить *gain_control_data* (), если есть;
- получить *spectral_data* (), если есть.

Процесс восстановления *pulse_data* описан в разделе 7, *tns_data* — в разделе 12, *gain_control* данных — в разделе 14. Краткий обзор декодирования *ics_info()* (см. 6.3), данных раздела 7, масштабных коэффициентов (разделы 7 и 9) и спектральных данных (раздел 7) дается ниже.

6.3.2.3 Восстановление *ics_info* ()

Для элементов *single_channel_element* () *ics_info* () всегда располагается непосредственно после *global_gain* в *inidividual_channel_stream* (). Для *channel_pair_element* () есть два возможных расположения *ics_info* (). Если оба отдельных канала в парном окне переключаются вместе, то *ics_info* () располагается непосредственно после *common_window* в *channel_pair_element* () и *common_window* устанавливается в 1. Иначе *ics_info* () присутствует сразу после *global_gain* в каждом из двух *individual_channel_stream* () в *channel_pair_element* (), и *common_window* устанавливается в 0.

ics_info () содержит информацию об окне, связанную с *ICS*, и таким образом позволяет каналам в *channel_pair* переключаться при необходимости независимо. Кроме того, *ics_info* () содержит *max_sfb*, который определяет верхнюю границу количества *ms_used* [] и *predictor_used* [] бит, которые должны быть переданы. Если *window_sequence* является *EIGHT_SHORT_SEQUENCE*, то *scale_factor_grouping* передается. Если ряд коротких окон образует группу, то они совместно используют масштабные коэффициенты, так же как и позиции *intensity stereo* и их спектральные коэффициенты чередуются. Первое короткое окно всегда является новой группой, таким образом, группировка битов не передается. Последующие короткие окна находятся в той же самой группе, если соответствующий бит группировки равен 1. Новая группа начинается, если соответствующий бит группировки равен 0. Предполагается, что сгруппированные короткие окна обладают подобной сигнальной статистикой. Следовательно, их спектральные значения чередуются для размещения коррелированных коэффициентов друг с другом. Способ чередования показан на рисунке 6. *ics_info* () также содержит данные предсказания для отдельного канала или пары каналов (см. раздел 11).

6.3.2.4 Восстановление данных разделения

В ICS восстанавливается информация об одном длинном окне или восьми коротких окнах. Данные разделения являются первым полем, которое должно быть декодировано и описывают коды Хаффмана, которые применяются к полосам масштабных коэффициентов в ICS (см. разделы 7 и 9). Форма данных разделения: *sect_cb* кодовая книга для раздела и *sect_len* длина раздела.

Эта длина восстанавливается путем последовательного считывания разделов из потока битов, с добавлением символа escape к разделу, до тех пор, пока значение, отличное от escape не будет найдено и добавлено к разделу. Этот процесс ясно объясняется при использовании С-подобного описания синтаксиса. Следует учесть, что в пределах каждой группы разделы должны формировать полосы масштабных коэффициентов от нуля до *max_sfb* так, чтобы первый раздел в пределах каждой группы начинался с нулевой полосы, а последний раздел заканчивался на полосе *max_sfb*.

Данные разделения описывают кодовую книгу и затем длину раздела, использующего эту кодовую книгу, начинающегося с первой полосы масштабных коэффициентов и продолжающегося, пока не будет достигнуто полное количество полос масштабных коэффициентов.

После того, как это описание предоставлено, все масштабные коэффициенты и спектральные данные, соответствующие нулевой кодовой книге, обнуляются, и значения, соответствующие этим масштабным коэффициентам или спектральным данным, не будут переданы. Масштабные коэффициенты любых полос масштабных коэффициентов, кодовая книга Хаффмана которых является нулем, будут отброшены. Точно так же, все спектральные данные, относящиеся к нулевой кодовой книге Хаффмана, исключаются (см. разделы 7 и 9).

Кроме того, спектральные данные, относящиеся к полосам масштабных коэффициентов, у которых есть сборник интенсивностных кодов, не будут переданы, однако управляющие интенсивностные коэффициенты будут переданы вместо масштабных коэффициентов, как описано в 10.2.

6.3.2.5 Поиск данных о масштабных коэффициентах и декодирование

Для каждой полосы масштабных коэффициентов, которая не находится в разделе, кодированном с помощью нулевой кодовой книги (*ZERO_HCB*), передается масштабный коэффициент. Такие полосы будут обозначены как «активные» полосы масштабных коэффициентов, а соответствующие масштабные коэффициенты — как «активные» масштабные коэффициенты. Глобальное усиление, первый элемент данных в *ICS*, обычно является значением первого активного масштабного коэффициента. Все масштабные коэффициенты (и управляющие коэффициенты) передаются с помощью кодированных Хаффманом ДИКМ значений относительно предыдущего «активного» масштабного коэффициента (см. разделы 9 и 11). Первый активный масштабный коэффициент дифференциально кодируется относительно значения глобального усиления. Следует учесть, что неэффективно использовать global_gain, отличный от первого активного масштабного коэффициента и затем ненулевое значение ДИКМ для первого ДИКМ значения масштабного коэффициента. Если какие-либо управляющие интенсивностные коэффициенты получены вперемешку с ДИКМ масштабными коэффициентами, они отправляются в модуль *intensity stereo*, и не включаются в ДИКМ-кодирование значений масштабных коэффициентов (см. 10.2). Значение первого активного масштабного коэффициента обычно передается как global_gain с первым ДИКМ масштабным коэффициентом, имеющим нулевое значение. Как только масштабные коэффициенты декодируются и их целые значения получены, фактические значения находятся через степенную функцию (см. раздел 9).

6.3.2.6 Поиск спектральных данных и декодирование

Спектральные данные восстанавливаются как последняя часть анализа *ICS*. Они состоят из всех ненулевых коэффициентов, оставшихся в спектре или спектрах, упорядоченных, как описано в *ICS_info*. Для каждой ненулевой, неинтенсивностной кодовой книги данные восстанавливаются с помощью Хаффмана, декодированием четверок или пар, как показано в инструменте прозрачного кодирования (см. раздел 7). Если спектральные данные связываются с беззнаковой кодовой книгой Хаффмана, необходимые знаковые биты следуют за кодовым словом Хаффмана (см. 7.3). В случае сборника кодов *ESCAPE*, при получении значения escape, соответствующая escape-последовательность появится после указанного кода Хаффмана. Может встречаться от нуля до двух escape-последовательностей для каждой кодовой комбинации в сборнике кодов *ESCAPE*, в соответствии с наличием escape-значений в декодируемой кодовой комбинации. Для каждого раздела декодирование Хаффмана продолжается до тех пор, пока все спектральные значения в этом разделе не будут декодированы. Как только все разделы декодированы, данные умножаются на декодированные масштабные коэффициенты, и, в случае необходимости, устраняется чередование.

6.3.3 Окна и последовательности окон

Квантование и кодирование выполняются в частотном пространстве. С этой целью временной сигнал в кодере трансформируется в частотное пространство. Декодер выполняет обратное преобразование (см. раздел 13). В зависимости от сигнала кодер может варьировать частотно-временное разрешение путем использования двух различных окон: *LONG_WINDOW* и *SHORT_WINDOW*. Для переключения между окнами используются промежуточные окна перехода, *LONG_START_WINDOW* и *LONG_STOP_WINDOW*. В таблице 43 даны окна, соответствующая длина преобразования и форма окон. Используются 2 размера преобразования: 1024 (называемое длинным преобразованием) и 128 коэффициентов (называемое коротким преобразованием).

Последовательности окон образуются из окон исходя из того, что raw_data_block () всегда содержит данные, соответствующие 1024 выходным отсчетам. Элемент данных **window_sequence** определяет используемую последовательность окон. В таблице 44 показано, как последовательности окон образуются из отдельных окон.

6.3.4 Полосы масштабных коэффициентов и группировка

Множество инструментов декодера выполняют операции над группами последовательных спектральных значений, называемыми полосами масштабных коэффициентов (сокращенно '*sfb*'). Ширина полос масштабных коэффициентов выбирается так, чтобы имитировать критические полосы слуховой системы человека. По этой причине число полос масштабных коэффициентов и их ширина зависят от длины преобразования и частоты дискретизации. В таблицах 45 — 57 приведены смещения начала каждой полосы масштабных коэффициентов для длин преобразования 1024 и 128 и различных частот дискретизации, соответственно.

Чтобы уменьшить количество служебной информации в случае использования последовательностей, которые содержат *SHORT_WINDOWS*, соседние *SHORT_WINDOWS* могут быть сгруппированы (см. рису-

нок 4). Информация о группировке содержится в элементе данных *scale_factor_grouping*. Группировка означает, что передается только один набор масштабных коэффициентов для всех сгруппированных окон, как будто это одно окно. Масштабные коэффициенты в этом случае применяются к соответствующим спектральным данным всех сгруппированных окон. Чтобы увеличить эффективность прозрачного кодирования (см. раздел 7), спектральные данные группы передаются в порядке чередования (см. 6.3.5). Чередование выполняется по полосам масштабных коэффициентов так, чтобы спектральные данные могли быть сгруппированы, чтобы сформировать виртуальную полосу масштабных коэффициентов, к которой может быть применен общий масштабный коэффициент. В пределах этого документа выражение «полоса масштабных коэффициентов» (сокращенно '*sfb*') означает эти виртуальные полосы масштабных коэффициентов. Если речь идет о полосах масштабных коэффициентов отдельных окон, используется выражение «оконная полоса масштабных коэффициентов» (сокращенно '*sfb*'). Из-за влияния на полосы масштабных коэффициентов группировка сказывается на *section_data*, порядке следования спектральных данных (см. 6.3.5) и общем количестве полос масштабных коэффициентов. Для *LONG_WINDOW* полосы масштабных коэффициентов и оконные полосы масштабных коэффициентов окна идентичны, так как на одно окно приходится только одна группа.

Чтобы уменьшить количество информации, необходимой для передачи служебных данных, относящихся к каждой полосе масштабных коэффициентов, передается элемент данных *max_sfb*. Его значение превышает на единицу самую высокую активную полосу масштабных коэффициентов во всех группах. *max_sfb* влияет на интерпретацию данных раздела (см. раздел 7), передачу масштабных коэффициентов (см. разделы 7 и 9), передачу данных предсказывающего устройства (см. раздел 11) и передачу ms_mask (см. 10.1).

Так как полосы масштабных коэффициентов являются основным элементом алгоритма кодирования, необходимы некоторые справочные переменные и массивы для описания процесса декодирования во всех инструментах, с использованием полос масштабных коэффициентов. Эти справочные переменные зависят от *sampling_frequency*, *window_sequence*, *scalefactor_grouping* и *max_sfb* и должны быть созданы для каждого *raw_data_block* (). Псевдокод, показанный ниже, описывает:

- как определить число окон в *window_sequence num_windows;*
- как определить число *window_groups num_window_groups;*
- как определить число окон в каждой группе *window_group_length [g];*
- как определить общее количество оконных полос масштабных коэффициентов *num_swb* для фактического типа окна;
- как определить *swb_offset [swb]*, смещение первого коэффициента в оконной полосе масштабных коэффициентов *swb* используемого окна;
- как определить *sect_sfb_offset [g][section]*, смещение первого коэффициента в разделе *section*.

Это смещение зависит от *window_sequence u scale_factor_grouping* и требуется, чтобы декодировать *spectral_data* ().

Длинное окно преобразования всегда описывается как *window_group*, содержащая единственное окно. Так как число полос масштабных коэффициентов и их ширина зависят от частоты дискретизации, зависимые переменные индексируются с *sampling_frequency_index* для выбора соответствующей таблицы.

```
    fs_index =
sampling_frequency_index;
      switch  (window_sequence)  {
      case
ONLY_LONG_SEQUENCE:
      case
LONG_START_SEQUENCE:
      case LONG_STOP_SEQUENCE:
        num_windows  =  1;
        num_window_groups  =  1;
      window_group_length [num_window_groups–1] = 1;
      num_swb = num_swb_long_window[fs_index];
       /* preparation of sect_sfb_offset for long blocks */
       /* also copy the last value! */
      for (i = 0; i < max_sfb + 1; i++) {
      sect_sfb_offset[0][i]  =  swb_offset_long_window[fs_index][i];
```

```
        swb_offset[i]  =  swb_offset_long_window[fs_index][i];
        }
        break;
        case
IGHT_SHORT_SEQUENCE:
        num_windows = 8;
        num_window_groups  =  1;
        window_group_length [num_window_groups–1]  =  1;
        num_swb = num_swb_short_window[fs_index];
        for  (i  =  0,  i  < num_swb_short_window[fs_index]  +  1;
i++)
        swb_offset[i]  =  swb_offset_short_window[fs_index][i];
      for  (i  =  0;  i  < num_windows–1;  i + + )  {
      if (bit_set (scale_factor_grouping,6–i))  ==  0)  {
        num_window_groups += 1;
      window_group_length [num_window_groups–1]  =  1;
      } else
{
      window_group_length [num_window_groups–1]  += 1;}
      /* preparation  of sect_sfb_offset  for short  blocks  */
      for  (g =  0;  g < num_window_groups;  g+ + )  {
        sect_sfb  =  0;
        offset  =  0;
        for  (i  =  0;  i  <  max_sfb;  i++)  {
      width = swb_offset_short_window[fs_index][i+1]  –
      swb_offset_short_window[fs_index] [i] ;
       width  *= window_group_length [g];
       sect_sfb_offset[g][sect_sfb++]  =  offset;
       offset  += width;
      }
      sect_sfb_offset[g][sect_sfb]  =  offset;
      }
      b
  reak;
      d
efault:
      b
  reak;
```

6.3.5 Порядок следования спектральных коэффициентов в *spectral_data* ()

Для окон *ONLY_LONG_SEQUENCE* (*num_window_groups* = 1, *window_group_length* [0] = 1) спектральные данные расположены в порядке возрастания их номеров, как показано на рисунке 5.

Для окна *EIGHT_SHORT_SEQUENCE* порядок спектральных составляющих зависит от группировки следующим способом:

- группы следуют друг за другом;

- в пределах группы полоса масштабных коэффициентов состоит из спектральных данных всех сгруппированных *SHORT_WINDOW* окон соответствующей оконной полосы масштабных коэффициентов. Для примера, пусть длина группы находится в диапазоне от первого до восьмого окна *SHORT_WINDOW*;

- если есть восемь групп с единичной длиной каждая (*num_window_groups* = 8, *window_group_length* [0] = 1), результатом является последовательность их восьми спектральных значений, расположенных в порядке возрастания;

- если есть только одна группа с длиной восемь (*num_window_group* = 1, *window_group_length* [0] = 8), в результате спектральные данные всех восьми *SHORT_WINDOW* окон чередуются оконными полосами масштабных коэффициентов;

- на рисунке 6 показано расположение составляющих спектра для *EIGHT_SHORT_SEQUENCE* с группировкой *SHORT_WINDOW* согласно рисунку 4 (*num_window_groups* = 4);

- в пределах оконной полосы масштабных коэффициентов, коэффициенты расположены в порядке возрастания.

6.3.6 Длина выходного слова

Глобальное усиление для каждого звукового канала масштабируется так, чтобы целочисленная часть выхода ОМДКП могла использоваться непосредственно в качестве 16-разрядных выходных ИКМ аудиоданных, поступающих в цифро-аналоговый преобразователь (*D/A*). Это является режимом по умолчанию и обеспечивает корректные уровни громкости. Если декодер обладает цифро-аналоговым преобразователем, имеющим разрешение больше 16 разрядов, то выход ОМДКП может быть усилен так, чтобы сформировать требуемый размер слова. В этом случае уровень на выходе преобразователя будет соответствовать случаю с 16-разрядным *D/A*, однако будет обладать большим динамическим диапазоном и меньшим уровнем шумов квантования. Аналогичным образом обстоит ситуация с более короткими словами.

6.4 Канал низкой частоты (*LFE*)

Для совместимости с обычной структурой декодера *lfe_channel_element*() определяется как стандартный *individual_channel_stream* (0) элемент, равный *single_channel_element* (). Таким образом, декодирование может быть выполнено с помощью стандартной процедуры декодирования *single_channel_element* ().

Чтобы достичь большей скорости передачи и эффективности аппаратной реализации декодера *LFE*, требуется применить несколько ограничений к опциям, используемым для кодирования этого элемента:

- поле *window_shape* всегда устанавливается в 0, синусное окно (см. 4.3, таблица 15).
- поле *window_sequence* всегда устанавливается в 0 (*ONLY_LONG_SEQUENCE*) (см. 4.3, таблица 15).
- только самые нижние 12 спектральных коэффициентов любого *LFE* могут быть ненулевыми.
- временное формирование шума не используется, т.е. *tns_data_present* устанавливается в 0 (см. 4.3, таблица 16).
- предсказание не используется, *predictor_data_present* устанавливается в 0 (см. 4.3, таблицу 15).

Наличие каналов *LFE* зависит от используемого профиля.

6.5 Элемент конфигурации программы (*PCE*)

Элемент program_config_element () может присутствовать как вне данных AAC, например в *adif_header*(), так и внутри AAC как синтаксический элемент в *raw_data_block* ().

6.5.1 Функции данных

byte_alignment () — Для элементов PCE в пределах *raw_data_block* (), выравнивание относительно первого бита *raw_data_block* (). Для элементов PCE в пределах *adif_header* (), выравнивание относительно первого бита заголовка.

6.5.2 Элементы данных

profile — Двухбитный индекс профиля в таблице 31 (таблица 25).

sampling_frequency_index — Указывает на частоту дискретизации программы (и всех других программ в этом потоке битов).

num_front_channel_elements — Число синтаксических элементов аудио в передних каналах, от переднего к заднему, симметрично левые и правые, или попеременно левые и правые в случае элементов одиночного канала (таблица 25).

num_side_channel_elements — Число элементов боковых каналов (таблица 25).

num_back_channel_elements — То же, для заднего канала (таблица 25).

num_lfe_channel_elements — Число элементов *LFE* канала (таблица 25).

num_assoc_data_elements — Число элементов, связанных с этой программой (таблица 25).

num_valid_cc_elements — Число элементов *CCE*, которые могут быть добавлены к аудиоданным этой программы (таблица 25).

mono_mixdown_present — Один бит, указывающий на присутствие элемента моно сведения (таблица 25).

mono_mixdown_element_number — Количество указанных *SCE* моно сведения (таблица 25).

stereo_mixdown_present — Один бит, указывающий на присутствие стерео сведения (таблица 25).

stereo_mixdown_element_number — Число указанных *CPE* стерео сведения (таблица 25).

matrix_mixdown_idx_present — Один бит, указывающий на присутствие матричной информации средствами индекса матричного коэффициента стерео (таблица 39). Для всех конфигураций, кроме 3/2, этот бит должен быть равен '0' (таблица 25).

matrix_mixdown_idx	Двухбитное поле, определяющее индекс коэффициента сведения, для использования в объединении 5 каналов в 2. Возможные матричные коэффициенты перечислены в таблице 39 (таблица 25).
pseudo_surround_enable	Один бит, указывающий на возможность сведения для псевдопространственного окружения (таблица 25).
front_element_is_cpe	Указывает, являются ли *SCE* или *CPE* передними элементами (таблица 25). '0' соответствует *SCE*,'1' соответствует *CPE*.
front_element_tag_select	*instance_tag SCE/CPE*, адресуемых как передние элементы (таблица 25).
side_element_is_cpe	См. *front_element_is_cpe*, но для элементов стороны (таблица 25).
side_element_tag_select	См. *front_element_tag_select*, но для элементов стороны (таблица 25).
back_element_is_cpe	См. *front_element_is_cpe*, но для задних элементов (таблица 25).
back_element_tag_select	См. *front_element_tag_select*, но для задних элементов (таблица 25).
lfe_element_tag_select	*instance_tag* канала низкой частоты (таблица 25).
assoc_data_element_tag_select	*instance_tag DSE* (таблица 25).
valid_cc_element_tag_select	*instance_tag CCE* (таблица 25).
cc_element_is_ind_sw	Один бит, указывающий, что соответствующий *CCE* — независимо переключаемый спаренный канал (таблица 25).
comment_field_bytes	Длина поля комментария, в байтах (таблица 25).
comment_field_data	Данные в поле комментария (таблица 25).

Обращение к элементам *SCE* или *CPE* в пределах *PCE* происходит с помощью двух элементов синтаксиса. Во первых, элемент синтаксиса *is_cpe* выбирает, поэтому происходит обращение к *SCE* или *CPE*. Во вторых, элемент синтаксиса *tag_select* выбирает *instance_tag SCE/CPE*. Обращение к элементам *LFE*, *CCE* и *DSE* осуществляется непосредственно по их *instance_tag*.

6.5.3 Конфигурация канала

Аудио синтаксис AAC обеспечивает три способа передачи соответствия каналов физическим расположениям динамиков.

6.5.3.1 Явное соответствие каналов с настройками по умолчанию

Соответствия каналов по умолчанию определяются в таблице 42 (значения больше 0).

6.5.3.2 Явное соответствие каналов на основе *program_config_element* ()

Любая возможная конфигурация каналов может быть определена при помощи *program_config_element* (). Существует 16 доступных элементов PCE, каждый из которых может указывать на отдельную программу, которая присутствует в потоке необработанных данных. Все доступные PCE в пределах *raw_data_block*() должны следовать перед всеми другими синтаксическими элементами. Программы могут совместно использовать некоторые общие аудио синтаксические элементы, например общий *channel_pair_element* () и различные каналы комментариев на различных языках. Данный *program_config_element* () содержит информацию, имеющую отношение только к одной программе из тех, которые могут быть включены в *raw_data_stream* (). Включенный в *PCE* "список передних каналов", организован по принципу сначала центральный, затем остальные, левый и правый. В этом списке *SCE* центрального канала, если имеется, должен следовать первым, а любые другие SCE должны появляться в парах, составляя пару *LR*. Если указаны только два *SCE*, это соответствует одной стереофонической паре *LR*.

После списка передних каналов имеется "список боковых каналов", состоящих из *CPE* или из пар *SCE*. Они перечисляются в порядке от переднего к тыловому. Снова, в случае пары *SCE*, первым идет левый канал, вторым — правый канал.

После списка боковых каналов следует список тыловых каналов, перечисляемых снаружи внутрь. Любой *SCE*, кроме последнего должен быть спарен, и наличие двух *SCE* (отдельных или после *CPE*) указывает, что два SCE являются левым и правым тыловым центром соответственно.

Конфигурация, обозначенная *PCE*, вступает в силу в *raw_data_block* (), содержащем *PCE*. Количество передних, боковых и задних каналов, указанное в *PCE*, должно присутствовать в том блоке и всех последующих *raw_data_block* () блоках, пока не будет передан *raw_data_block*(), содержащий новый *PCE*.

Также определяются другие элементы. Список одного или нескольких *LFE*, список одного или нескольких (зависимых от профиля) *CCE*, для осуществления управления диалогами и различными спаренными интенсивностными потоками различных каналов, при использовании тех же основных каналов. Список потоков данных, связанных с программой, также может связывать один или более потоков данных

с программой. Элемент конфигурации программы также позволяет описать один монофонический и один стереофонической каналы, объединенные в канал сведения для одновременной передачи.

Элемент *PCE* не предназначен для быстрого изменения программы. В любой момент времени, когда данный *PCE*, в соответствии с его *element_instance_tag*, определяет новую (в противоположность повторному) программу, декодер не обязан обеспечивать непрерывность звукового сигнала.

6.5.3.3 Неявное соответствие каналов

Если нет явного соответствия каналов, следующие методы описывают неявное соответствие каналов:

1) Любое количество *SCE* может появиться (до тех пор, пока разрешено другими ограничивающими факторами, например профилем). Если это количество *SCE* нечетное, то первый *SCE* представляет передний центральный канал, и другой *SCE* представляет пары каналов *L/R*, следуя от центра в стороны и назад, к тыловому каналу.

Если количество *SCE* четное, то *SCE* интерпретируются как пары *L/R*, от переднего центра в стороны и назад, к тыловому каналу.

2) Может присутствовать любое количество *CPE* или пар *SCE*. Каждый *CPE* или пара *SCE* представляют одну пару *L/R*, после первых наборов *SCE* элементы следуют попарно до достижения тыловой центральной пары.

3) Может присутствовать любое количество *SCE*. Если это количество является четным, *L/R* пары распределены от 2) до тылового центра. Если это количество нечетное, оно распределяется по *L/R* парам, за исключением последнего *SCE*, который присваивается тыловому центру.

4) Может присутствовать любое количество *LFE*. Для случая, когда имеется несколько *LFE*, соответствие громкоговорителям не определено.

В случае такого неявного соответствия каналов, число и порядок *SCE*, *CPE* и *LFE* и получающаяся конфигурация не должны меняться в пределах потока битов без отправления *program_config_element* (), неявная смена конфигурации не разрешена.

Другие синтаксические аудио элементы, которые не подразумевают наличия дополнительных громкоговорителей, такие как спаренный *channel_element*, могут следовать за перечисленным набором синтаксических элементов. Очевидно синтаксические элементы, не относящиеся к звуковым, могут быть получены в дополнение к перечисленных синтаксическим элементам в произвольном порядке.

6.5.4 Метод матричного сведения

6.5.4.1 Описание

Метод матричного сведения применяется только для сведения конфигурации громкоговорителей 3 передних/2 задних, 5-канальной программы, в стерео или моно программу. Это не применимо к любой другой программе, кроме конфигурации 3/2.

6.5.4.2 Процесс матричного сведения

Требуемый сигнал стерео может быть сгенерирован матричным декодером при помощи одного из двух наборов уравнений.

Набор 1:

$$L' = \frac{1}{1 + 1/\sqrt{2} + A} \cdot [L + C/\sqrt{2} + A \cdot L_s]$$

$$R' = \frac{1}{1 + 1/\sqrt{2} + A} \cdot [R + C/\sqrt{2} + A \cdot R_s]$$

Набор 2:

$$L' = \frac{1}{1 + 1/\sqrt{2} + 2 \cdot A} \cdot [L + C/\sqrt{2} - A \cdot (L_s + R_s)]$$

$$R' = \frac{1}{1 + 1/\sqrt{2} + 2 \cdot A} \cdot [R + C/\sqrt{2} + A \cdot (L_s + R_s)],$$

где L, C, R, L_s и R_s — исходные сигналы, L' и R' — полученные сигналы стерео, A — матричный коэффициент, обозначенный *matrix_mixdown_idx*. Каналы *LFE* исключаются из сведения.

Если флаг *pseudo_surround_enable* не установлен, то используется только набор уравнений 1. Если *pseudo_surround_enable* установлен, то могут использоваться как набор 1, так и набор 2, в зависимости от наличия в ресивере средств пространственного синтеза.

Можно получить моно сигнал, используя следующее уравнение

$$M = \frac{1}{3 + 2 \cdot A} \cdot [L + C + R + A \cdot (L_s + R_s)].$$

6.5.4.3 Рекомендация

Матричные сведения позволяют организовать режим, выгодный для некоторых операторов при определенных обстоятельствах. Однако рекомендуется не использовать этот метод. Психоакустические принципы, на которых базируется кодирование звука, нарушаются этим методом пост-процессинга, и перцепционно верное восстановление сигнала не может быть гарантировано. В синтаксисе AAC рекомендуется использовать стерео или моно сведение для обеспечения стерео или моно, созданного в стандартной студии.

Стерео и моно каналы сведения дополнительно позволяют провайдеру контента отдельно улучшать стерео и многоканальные программы — это невозможно при использовании матричного метода.

Из-за алгоритмов, используемых для многоканального и стерео кодирования, лучшая комбинация качества и скорости передачи обычно обеспечивается при использовании стерео.

Таблица 39 — Матричные коэффициенты

matrix_mixdown_idx	*A*
0	$1/\sqrt{2}$
1	1/2
2	$1/(2\sqrt{2})$
3	0

6.6 Элемент потока данных (*DSE*)

6.6.1 Функции данных

byte_alignment () Выравнивание относительно первого бита *raw_data_block* ().

6.6.2 Элементы данных

data_byte_align_flag Один бит, указывающий, что выравнивание выполняется в пределах элемента потока данных (таблица 24).

count Начальное значение длины потока данных (таблица 24).

esc_count Инкремент длины данных или элемента дополнения (таблица 24).

data_stream_byte Байт потока данных, извлеченный из потока битов (таблица 24).

Элемент данных содержит любые дополнительные данные, например вспомогательную информацию, которая не является непосредственно частью аудиоинформации. Возможно любое количество элементов данных с одинаковым *element_instance_tag* или до 16 элементов данных с различным *element_instance_tags*. Процесс расшифровки элемента данных описывается в этом пункте.

6.6.3 Процесс декодирования

Первым считывается синтаксический элемент *data_byte_align_flag* размером 1 бит. Затем считывается 8-битное значение *count*. Он содержит начальный размер потока данных в байтах. Если *count* равен 255, его значение инкрементируется вторым 8-битным значением, *esc_count*, и это заключительное значение соответствует количеству байт в элементе потока данных. Если *data_byte_align_flag* установлен, выполняется выравнивание. Далее следуют байты потока данных.

6.7 Элемент заполнения (*FIL*)

6.7.1 Элементы данных

count Начальное значение длины *extension_payload* () (таблица 26).

esc_count Инкремент длины *extension_payload* () (таблица 26).

6.7.2 Процесс декодирования

Элементы *fill_element* () могут быть добавлены для учета различных расширений. Разрешено любое количество элементов *fill_element* ().

Синтаксический элемент **count** дает начальное значение длины данных заполнения. Если *count* равен 15, его значение инкрементируется значением *esc_count*, и это заключительное значение соответствует количеству *fill_bytes*, которое будет считано.

6.8 Расширение

6.8.1 Общие сведения

1 Элементы данных

extension_type Поле, размером 4 бита, указывающее на тип контента элемента заполнения (таблица 26).

2 Процесс декодирования

Разрешено любое количество *extension_payload* ().

Следующие символьные сокращения для значений поля extension_type определены в таблице 40.

Таблица 40 — Значения элемента данных extension_type

Символ	Значение *extension_type*	Назначение
EXT_FILL	'0000'	Заполнение потока битов
EXT_FILL_DATA	'0001'	Заполнение битами данных
EXT_DYNAMIC_RANGE	'1011'	Управление динамическим диапазоном
EXT_SBR_DATA	'1101'	Расширение *SBR*
EXT_SBR_DATA_CRC	'1110'	Расширение *SBR* с *CRC*
—	другие значения	Зарезервировано

Зарезервированные значения могут использоваться для дальнейшего расширения синтаксиса.

6.8.2 Данные заполнения и другие биты

6.8.2.1 Элементы данных

fill_nibble 4-битное поле заполнения (таблица 28).

fill_byte Байт, отбрасываемый декодером (таблица 28).

other_bits Биты, отбрасываемые декодером (таблица 28).

6.8.2.2 Процесс декодирования

Данные заполнения должны быть добавлены, если общее количество битов всех аудиоданных вместе со всеми дополнительными данными меньше, чем минимальное допустимое количество битов во фрейме, необходимое для достижения целевой скорости передачи. При нормальных условиях для заполнения резервуара битов вместо битов заполнения используются свободные биты. Биты заполнения дописываются, только если резервуар битов полон.

Следует учесть, что *fill_nibble* должен быть равен '0000', а *fill_byte* равен '10100101' (для улучшения самосинхронизации).

6.8.3 Управление динамическим диапазоном (DRC)

6.8.3.1 Элементы данных

pce_tag_present Один бит, указывающий на присутствие тега элемента программы (таблица 29).

pce_instance_tag Поле тега, указывающее, с какой программой ассоциируется информация о динамическом диапазоне (таблица 29).

drc_tag_reserved_bits В резерве (таблица 29).

excluded_chns_present Один бит, указывающий на присутствие исключенных каналов (таблица 29).

drc_bands_present Один бит, указывающий на присутствие многополосной информации DRC (таблица 29).

drc_band_incr Число диапазонов DRC, большее чем 1 (таблица 29).

drc_bands_reserved_bits В резерве (таблица 29).

drc_band_top [i]	Указывает на верхнюю часть *i*-го диапазона DRC с шагом в 4 спектральные линии (таблица 29). Если *drc_band_top [i]* = *k*, то индекс самой высокой спектральной линии *i*-го диапазона *DRC* равен k*4+3. В случае *EIGHT_SHORT_SEQUENCE window_sequence* индекс интерпретируется как указатель на составной массив из 8*128 частотных линий, соответствующих 8 коротким преобразованиям.
prog_ref_level_present	Один бит, указывающий наличие контрольного уровня (таблица 29).
prog_ref_level	Контрольный уровень. Величина уровня воспроизведения каналов (таблица 29).
prog_ref_level_reserved_bits	В резерве (таблица 29).
dyn_rng_sgn [i]	Знак управления динамическим диапазоном. Один бит, указывающий на знак dyn_rng_ctl (0, если положительный, 1, если отрицательный (таблица 29).
dyn_rng_ctl [i]	Модуль управления динамическим диапазоном (таблица 29).
exclude_mask [i]	Логический массив, указывающий на звуковые каналы программы, которые исключаются из DRC.
additional_excluded_chns [i]	Один бит, указывающий на наличие дополнительных исключенных каналов (таблица 30).

6.8.3.2 Процесс декодирования

Оценка потенциально доступной информации управления динамическим диапазоном в декодере является опциональной.

prog_ref_level_present указывает, что *prog_ref_level* передается. В этом случае *prog_ref_level* передается только по требованию (например, однажды).

prog_ref_level квантуется с шагом 0,25 дБ, используя 7 битов, и поэтому динамический диапазон равен приблизительно 32 дБ. Восстановление выполняется по формуле

$$level = 32767 \cdot T^{prog\_ref\_level/24},$$

где «полный уровень» соответствует 32767 (***prog_ref_level*** = 0).

pce_tag_present указывает, что *pce_instance_tag* передается. В этом случае *pce_instance_tag* передается только по требованию (например, однажды).

pce_instance_tag указывает, с какой программой ассоциируется информация о динамическом диапазоне. Если этот тег не присутствует, то используется программа по умолчанию. Так как в каждом потоке битов AAC обычно есть только одна программа, это является наиболее распространенным режимом. Для каждой программы многопрограммного потока битов должна передаваться своя информация о динамическом диапазоне в отдельном *extension_payload* () элемента *fill_element* (). В случае многопрограммного потока всегда должен присутствовать *pce_instance_tag*.

Бит *drc_tag_reserved_bits* заполняют дополнительные поля до целого числа байт.

Бит *excluded_chns_present* указывает, что каналы, которые должны быть *исключены* из обработки динамического диапазона, следуют сразу же после этого бита. Маски каналов должны передаваться в каждом фрейме, где есть исключенные каналы. Следующие принципы упорядочивания используются, чтобы назначить *exclude_mask* выходным каналам:

Если *PCE* присутствует, биты *exclude_mask* соответствуют звуковым каналам в элементах синтаксиса *SCE*, *CPE*, *CCE* и *LFE* в порядке их появления в *PCE*. В случае *CPE* первый переданный бит маски соответствует первому каналу в *CPE*, второй переданный бит маски — второму каналу. В случае *CCE* бит маски передается, только если спаренный канал является независимо переключаемым.

Если *PCE* не присутствует, биты *exclude_mask* соответствуют звуковым каналам в элементах синтаксиса *SCE*, *CPE* и *LFE* в порядке их появления в потоке битов, в сопровождении звуковых каналов в элементах синтаксиса *CCE* в порядке их появления в потоке битов. В случае *CPE* первый переданный бит маски соответствует первому каналу в *CPE*, второй переданный бит маски — второму каналу. В случае *CCE* бит маски передается, только если спаренный канал является независимо переключаемым.

drc_band_incr является количеством полос больше единицы, если есть многополосная информация DRC.

dyn_rng_ctl квантуется с шагом 0,25 дБ, используя 7-разрядное целое число без знака, и поэтому, совместно с *dyn_rng_sgn*, имеет диапазон ± 31,75 дБ. Это интерпретируется как значение усиления, которое должно быть применено к декодируемым отсчетам звукого сигнала текущего фрейма.

Диапазон, обеспечиваемый информацией о динамическом диапазоне, указан в таблице 41.

Таблица 41 — Диапазон DRC информации

Поле	Биты	Шаги	Шаг, дБ	Диапазон, дБ
prog_ref_level	7	128	0,25	31,75
dyn_rng_sgn и dyn_rng_ctl	1 и 7	± 127	0,25	±31,75

Процесс управления динамическим диапазоном применяется к спектральным данным spec[i] одного фрейма непосредственно перед банком фильтров синтеза. В случае *EIGHT_SHORT_SEQUENCE* window_sequence индекс *i* интерпретируется как указатель на составной массив из 8*128 частотных линий, соответствующих 8 коротким преобразованиям.

Следующий псевдокод приводится в качестве примера, демонстрируя один метод использования одного набора информации управления динамическим диапазоном с целевым звуковым каналом. Константы *ctrl1* и *ctrl2* являются константами сжатия (обычно между 0 и 1, нуль соответствует отсутствию сжатия), которые могут дополнительно использоваться, чтобы масштабировать характеристики сжатия динамического диапазона для уровней, больших или меньших контрольного уровня программы соответственно. Константа *target_level* описывает уровень на выходе, требуемый пользователем, выраженный в том же масштабе как *prog_ref_level*.

```
bottom = 0;
drc_num_bands = 1;
if (drc_bands_present)
drc_num_bands + = drc_band_incr; else
drc_band_top [0] = 1024/4 – 1;
for (bd = 0;bd <drc_num_bands;bd ++) {
  top = 4 * (drc_band_top [bd] + 1);
/* Декодировать коэффициент усиления DRC */
if (dyn_rng_sgn [bd])
factor = 2 ^ (–ctrl1*dyn_rng_ctl [bd]/24);/* компрессирование */
else
 factor = 2 ^ (ctrl2*dyn_rng_ctl [bd]/24);/* усиление */
/* Если нормализация выполняется для цифрового сигнала
* коэффициент должен быть изменен. prog_ref_level может быть передан системе для изменения уровня аналогового сигнала. Аналоговое изменение уровня лишено проблем с уменьшенным отношением СИГНАЛ-ШУМ ЦАП или клиппированием */
factor *= 0.5 ^ ((target_level–prog_ref_level)/24);
/* Применить коэффициент усиления */
for (i = bottom; i < top; i++)
spec [i]*= factor; bottom = top.
```

Следует учесть соотношение между управлением динамическим диапазоном и спаренными каналами.

Зависимо переключаемые спаренные каналы всегда объединяются с их целевыми каналами как спектральные коэффициенты до *DRC* обработки и синтезирующей фильтрации этих каналов. Поэтому сигнал зависимо переключаемых спаренных каналов, который связывается с определенным целевым каналом, будет подвергнут *DRC* обработке целевого канала.

Так как независимо переключаемые спаренные каналы связываются с их целевыми каналами во временном пространстве, каждый независимо переключаемый спаренный канал будет подвергнут *DRC* обработке и последующей синтезирущей фильтрации, отдельно от его целевых каналов. Это позволяет независимо переключаемому спаренному каналу иметь отдельную *DRC* обработку при необходимости.

6.8.3.3 Персистентность *DRC* информации

В начале потока вся *DRC* информация для всех каналов устанавливается в значение по умолчанию: контрольный уровень программы, равный контрольному уровню декодера, одна полоса *DRC*, без изменения усиления *DRC*. Если эти данные не перезаписываются, это остается в силе.

Есть два случая для персистентности *DRC* информации, которая была передана:

Контрольный уровень программы дается для каждой звуковой программы и сохраняется до тех пор, пока не будет передано новое значение (разумно периодически передавать значение во избежание ошибки).

Другая *DRC* информация сохраняется для каналов. Следует учесть что, если канал исключается соответствующим битом маски *exclude_mask[]*, то никакая информация не передается для этого канала в *dynamic_range_info ()*. Маска должна быть передана в каждом фрейме, где есть исключение каналов.

Правила сохранения поканальной *DRC* информации:

Если в данном фрейме данного канала нет *DRC* информации, используется информацию из предыдущего фрейма. (Это означает, что одна настройка может быть в силе в течение долгого времени, хотя уместно периодически передавать *DRC* информацию во избежание ошибки).

Если *DRC* информация для этого канала появляется в текущем фрейме, имеет место такая последовательность: вся *DRC* информация для канала заменяется значениями по умолчанию (одна полоса *DRC*, без изменений усиления *DRC* для этой полосы), затем информация заменяется соответствующими переданными значениями.

6.8.4 Расширение полосы частот (*SBR*)

Элементы заполнения, содержащие *extension_payload* с *extension_type EXT_SBR_DATA* или *EXT_SBR_DATA_CRC*, резервируются для данных *SBR*. В этом случае, значение поля count элемента *fill_element* должно быть равно общей длине в байтах, включая данные *SBR* плюс поле *extension_type*.

sbr_extension_data () и процесс декодирования определяются в ИСО/МЭК 14496-3 [3].

Элементы заполнения *SBR* должны быть обработаны согласно ИСО/МЭК 14496-3 [3], пп. 4.5.2.8.2.2 «Расширение *SBR* для аудио объектов основного *AAC*, *AAC SSR*, *LC AAC* и *AAC LTP*». Сигнализация *SBR* должна быть сделана не явно, в соответствии с ИСО/МЭК 14496-3 [3], пп. 1.6.5 «Сигнализация *SBR*».

6.9 Таблицы

Таблица 42 — Конфигурация каналов

Значение	Число громкоговорителей	Элементы аудио синтаксиса перечисленные в порядке появления	Соответствие элемента громкоговорителю
0		—	Определено в *program_config_element* () (см. 6.5.3.2) или задано не явно (см. 6.5.3.3)
1	1	*single_channel_element ()*	Центральные передние
2	2	*channel_pair_element ()*	Левые, правые передние
3	3	*single_channel_element(),* *channel_pair_element()*	Центральные передние Левые, правые передние
4	4	*single_channel_element (),* *channel_pair_element (),*	Центральные передние Левые, правые передние
5	5	*single_channel_element ()* *single_channel_element (),* *channel_pair_element (),* *channel_pair_element ()*	Задние окружения Центральные передние Левые, правые передние Левые, правые окружения
6	5 + 1	*single_channel_element (),* *channel_pair_element (),* *channel_pair_element (),* *lfe _element ()*	Задние окружения Центральные передние Левые, правые передние Левые, правые окружения Задние окружения НЧ
7	7 + 1	*single_channel_element (),* *channel_pair_element (),* *channel_pair_element (),* *channel_pair_element (),* *lfe_element ()*	Центральные передние Левый, правый передние Левые, правые внешние Левые, правые окружения Задние окружения НЧ

Таблица 43 — Окна преобразований (для 48 кГц)

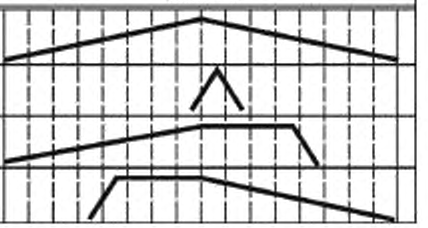
window	num_swb	#coeffs	looks like
LONG_WINDOW	49	1024	
SHORT_WINDOW	14	128	
LONG_START_WINDOW	49	1024	
LONG_STOP_WINDOW	49	1024	

Таблица 44 — Последовательность окон

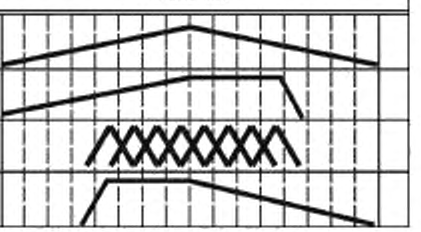
value	window_sequence	num_ windows	looks like
0	ONLY_LONG_SEQUENCE =LONG_WINDOW	1	
1	LONG_START_SEQUENCE =LONG_START_WINDOW	1	
2	EIGHT_SHORT_SEQUENCE =8 * SHORT_WINDOW	8	
3	LONG_STOP_SEQUENCE =LONG_STOP_WINDOW	1	

Таблица 45 — Полосы масштабных коэффициентов для *LONG_WINDOW*, *LONG_START_WINDOW*, *LONG_STOP_WINDOW* при 44,1 кГц и 48 кГц

Fs, кГц		44 1, 48	
num_swb_long_window		49	
swb	*swb_offset_long_window*	*swb*	*swb_offset_long_window*
0	0	25	216
1	4	26	240
2	8	27	264
3	12	28	292
4	16	29	320
5	20	30	352
6	24	31	384
7	28	32	416
8	32	33	448
9	36	34	480
10	40	35	512
11	48	36	544
12	56	37	576
13	64	38	608
14	72	39	640
15	80	40	672
16	88	41	704

Окончание таблицы 45

Fs, кГц		44,1, 48	
num_swb_long_window		49	
swb	*swb_offset_long_window*	*swb*	*swb_offset_long_window*
17	96	42	736
18	108	43	768
19	120	44	800
20	132	45	832
21	144	46	864
22	160	47	896
23	176	48	928
24	196		1024

Таблица 46 — Полосы масштабных коэффициентов для *SHORT_WINDOW* при 32 кГц, 44,1 кГц и 48 кГц

Fs, кГц		32, 44,1, 48	
num_swb_short_window		14	
swb	*swb_offset_short_window*	*swb*	*swb_offset_short_window*
0	0	8	44
1	4	9	56
2	8	10	68
3	12	11	80
4	16	12	96
5	20	13	112
6	28		128
7	36		

Таблица 47 — Полосы масштабных коэффициентов для *LONG_WINDOW*, *LONG_START_WINDOW*, *LONG_STOP_WINDOW* на 32 кГц

Fs, кГц		32	
num_swb_long_window		51	
swb	*swb_offset_long_window*	*swb*	*swb_offset_long_window*
0	0	26	240
1	4	27	264
2	8	28	292
3	12	29	320
4	16	30	352
5	20	31	384
6	24	32	416
7	28	33	448

Окончание таблицы 47

Fs, кГц		32	
num_swb_long_window		51	
swb	*swb_offset_long_window*	*swb*	*swb_offset_long_window*
8	32	34	480
9	36	35	512
10	40	36	544
11	48	37	576
12	56	38	608
13	64	39	640
14	72	40	672
15	80	41	704
16	88	42	736
17	96	43	768
18	108	44	800
19	120	45	832
20	132	46	864
21	144	47	896
22	160	48	928
23	176	49	960
24	196	50	992
25	216		1024

Таблица 48 — Полосы масштабных коэффициентов для *LONG_WINDOW*, *LONG_START_WINDOW*, *LONG_STOP_WINDOW* на 8 кГц

Fs, кГц		8	
num_swb_long_window		40	
swb	*swb_offset_long_window*	*swb*	*swb_offset_long_window*
0	0	21	288
1	12	22	308
2	24	23	328
3	36	24	348
4	48	25	372
5	60	26	396
6	72	27	420
7	84	28	448
8	96	29	476
9	108	30	508
10	120	31	544

Окончание таблицы 48

Fs, кГц		8	
num_swb_long_window		40	
swb	*swb_offset_long_window*	*swb*	*swb_offset_long_window*
11	132	32	580
12	144	33	620
13	156	34	664
14	172	35	712
15	188	36	764
16	204	37	820
17	220	38	880
18	236	39	944
19	252		1024
20	268		

Таблица 49 — Полосы масштабных коэффициентов для *SHORT_WINDOW* на 8 кГц

Fs, кГц	8
num_swb_short_window	15
swb	*swb_offset_short_window*
0	0
1	4
2	8
3	12
4	16
5	20
6	24
7	28
8	36
9	44
10	52
11	60
12	72
13	88
14	108
	128

Таблица 50 — Полосы масштабных коэффициентов для *LONG_WINDOW*, *LONG_START_WINDOW*, *LONG_STOP_WINDOW* на 11,025 кГц, 12 кГц и 16 кГц

Fs, кГц		11,025, 12, 16	
num_swb_long_window		43	
swb	*swb_offset_long_window*	*swb*	*swb_offset_long_window*
0	0	22	228
1	8	23	244
2	16	24	260
3	24	25	280
4	32	26	300
5	40	27	320
6	48	28	344
7	56	29	368
8	64	30	396
9	72	31	424
10	80	32	456
11	88	33	492
12	100	34	532
13	112	35	572
14	124	36	616
15	136	37	664
16	148	38	716
17	160	39	772
18	172	40	832
19	184	41	896
20	196	42	960
21	212		1024

Таблица 51 — Полосы масштабных коэффициентов для *SHORT_WINDOW* на 11,025 кГц, 12 кГц и 16 кГц

Fs, кГц		11,025, 12, 16	
num_swb_short_window		15	
swb	*swb_offset_short_window*	*swb*	*swb_offset_short_window*
0	0	8	32
1	4	9	40
2	8	10	48
3	12	11	60
4	16	12	72
5	20	13	88
6	24	14	108
7	28		128

Таблица 52 — Полосы масштабных коэффициентов для *LONG_WINDOW*, *LONG_START_WINDOW*, *LONG_STOP_WINDOW* на 22,05 кГц и 24 кГц

Fs, кГц		22,05 и 24	
num_swb_long_window		47	
swb	*swb_offset_long_window*	*swb*	*swb_offset_long_window*
0	0	25	172
1	4	26	188
2	8	27	204
3	12	28	220
4	16	29	240
5	20	30	260
6	24	31	284
7	28	32	308
8	32	33	336
9	36	34	364
10	40	35	396
11	44	36	432
12	52	37	468
13	60	38	508
14	68	39	552
15	76	40	600
16	84	41	652
17	92	42	704
18	100	43	768
19	108	44	832
20	116	45	896
21	124	46	960
22	136		1024
23	148		
24	160		

Таблица 53 — Полосы масштабных коэффициентов для *SHORT_WINDOW* на 22,05 кГц и 24 кГц

Fs, кГц		22,05 и 24	
num_swb_short_window		15	
swb	*swb_offset_short_window*	*swb*	*swb_offset_short_window*
0	0	8	36
1	4	9	44
2	8	10	52
3	12	11	64
4	16	12	76
5	20	13	92
6	24	14	108
7	28		128

Таблица 54 — Полосы масштабных коэффициентов для *LONG_WINDOW*, *LONG_START_WINDOW*, *LONG_STOP_WINDOW* на 64 кГц

Fs, кГц		64	
num_swb_long_window		47	
swb	*swb_offset_long_window*	*swb*	*swb_offset_long_window*
0	0	24	172
1	4	25	192
2	8	26	216
3	12	27	240
4	16	28	268
5	20	29	304
6	24	30	344
7	28	31	384
8	32	32	424
9	36	33	464
10	40	34	504
11	44	35	544
12	48	36	584
13	52	37	624
14	56	38	664
15	64	39	704
16	72	40	744
17	80	41	784
18	88	42	824
19	100	43	864
20	112	44	904
21	124	45	944
22	140	46	984
23	156		1024

Таблица 55 — Полосы масштабных коэффициентов для *SHORT_WINDOW* на 64 кГц

Fs, кГц	64
num_swb_short_window	12
swb	*swb_offset_short_window*
0	0
1	4
2	8
3	12
4	16

Окончание таблицы 55

Fs, кГц	64
num_swb_short_window	12
swb	*swb_offset_short_window*
5	20
6	24
7	32
8	40
9	48
10	64
11	92
	128

Таблица 56 — Полосы масштабных коэффициентов для *LONG_WINDOW*, *LONG_START_WINDOW*, *LONG_STOP_WINDOW* на 88,2 кГц и 96 кГц

Fs, кГц		88,2 и 96	
num_swb_long_window		41	
swb	*swb_offset_long_window*	*swb*	*swb_offset_long_window*
0	0	21	120
1	4	22	132
2	8	23	144
3	12	24	156
4	16	25	172
5	20	26	188
6	24	27	212
7	28	28	240
8	32	29	276
9	36	30	320
10	40	31	384
11	44	32	448
12	48	33	512
13	52	34	576
14	56	35	640
15	64	36	704
16	72	37	768
17	80	38	832
18	88	39	896
19	96	40	960
20	108		1024

Таблица 57 — Полосы масштабных коэффициентов для *SHORT_WINDOW* на 88,2 кГц и 96 кГц

Fs, кГц	88,2 и 96
num_swb_short_window	12
swb	*swb_offset_short_window*
0	0
1	4
2	8
3	12
4	16
5	20
6	24
7	32
8	40
9	48
10	64
11	92
	128

6.10 Рисунки

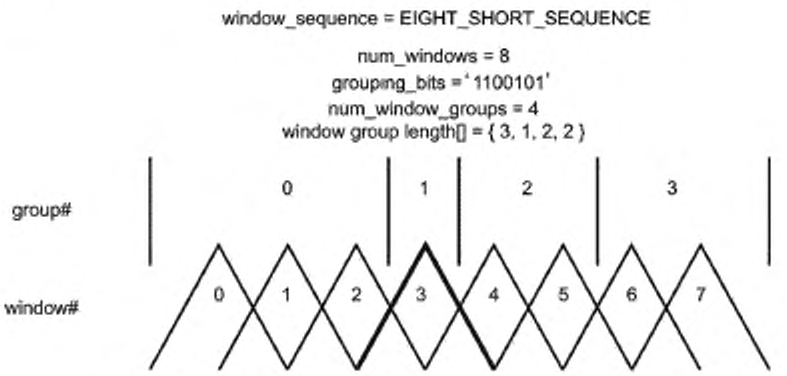
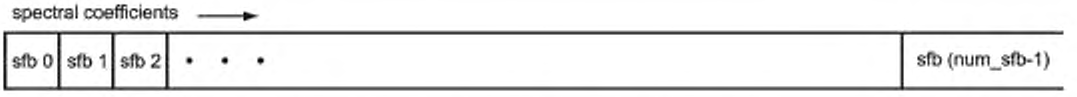


Рисунок 4 — Пример группировки коротких окон



Order of scalefactor bands for ONLY_LONG_SEQUENCE

Рисунок 5 — Порядок полос масштабных коэффициентов в случае *ONLY_LONG_SEQUENCE*

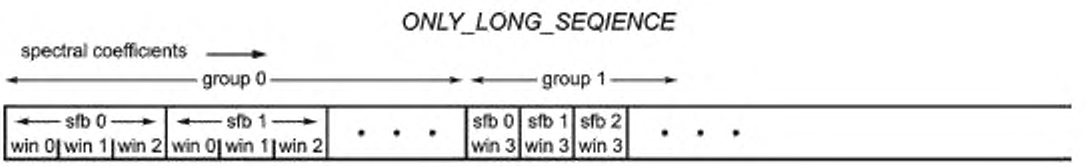


Рисунок 6 — Порядок полос масштабных коэффициентов в случае *EIGHT_SHORT_SEQUENCE*

7 Прозрачное кодирование

7.1 Описание инструмента

Прозрачное кодирование используется для дальнейшего уменьшения избыточности масштабных коэффициентов и квантованных спектральных значений каждого звукового канала.

global_gain кодируется как беззнаковое 8-битовое целое. Первый масштабный коэффициент, связанный с квантованными спектральными значениями, дифференциально кодируется относительно значения *global_gain,* и затем используется кодирование Хаффмана по кодовой книге для масштабных коэффициентов. Оставшиеся масштабные коэффициенты дифференциально кодируются относительно предыдущих масштабных коэффициентов, и затем используется кодирование Хаффмана по кодовой книге для масштабных коэффициентов.

Прозрачное кодирование квантованных спектральных значений основано на двух разделениях спектральных коэффициентов. Первым является разделение на полосы масштабных коэффициентов, которые содержат кратное четырем количество квантованных спектральных коэффициентов. См. 6.3.4 и 6.3.5.

Второе разделение, зависящее от квантованных спектральных значений, является разделением на полосы масштабных коэффициентов с целью формирования разделов. Значение раздела состоит в том, что квантованные спектральные значения в пределах раздела кодируются с использованием одной кодовой книги Хаффмана, выбранной из 11 возможных. Длина раздела и связанной с ним кодовой книги Хаффмана должна быть передана в качестве служебных данных вместе с кодированными по Хаффману спектральными значениями. Следует учесть, что длина раздела выражается в полосах масштабных коэффициентов, а не в оконных масштабных коэффициентах (см. 6.3.4). Чтобы максимизировать соответствие статистики квантованного спектра кодовой книге Хаффмана, число разделов может быть таким же, как и число полос масштабных коэффициентов. Максимальный размер раздела равен *max_sfb* полос масштабных коэффициентов.

Как показано в таблице 59, кодовые книги Хаффмана для спектральных значений могут заменять знаковые или беззнаковые комбинации из n коэффициентов. Для беззнаковых кодовых книг знаковые биты каждого ненулевого коэффициента в блоке из n коэффициентов следуют сразу же за соответствующей кодовой комбинацией.

При прозрачном кодировании есть два способа представить большие квантованные спектральные значения. Один способ состоит в том, чтобы передавать флаг escape из escape (ESC), сборник кодов Хаффмана, который сигнализирует о том, что биты, следующие сразу после указанной кодовой комбинации, плюс дополнительные знаковые биты являются escape-последовательностью, которой закодированы значения, большие чем представленные в кодовой книге Хаффмана ESC. Вторым путем является импульсный escape-метод, при котором коэффициенты относительно большой амплитуды могут быть заменены коэффициентами с меньшими амплитудами для включения в таблицу кода Хаффмана с более высокой эффективностью кодирования. Эта замена корректируется путем отправки позиции спектрального коэффициента и разности амплитуд в качестве служебной информации. Частотная информация представляется комбинацией номера полосы масштабных коэффициентов для указания на основную частоту и смещения внутри этой полосы масштабных коэффициентов.

7.2 Определения

7.2.1 Элементы данных

sect_cb [g] [i] Кодовая книга Хаффмана для спектральных значений, используемая для раздела *i* в группе *g* (см. 4.3, таблица 17).

sect_len_incr — Используется для вычисления размера раздела, равно количеству полос масштабных коэффициентов от начала раздела. Длина *sect_len_incr* составляет 3 бита, если window_sequence равно EIGHT_SHORT_SEQUENCE, и 5 битов в противном случае (см. 4.3, таблица 17).

global_gain — Глобальный коэффициент усиления квантованного спектра, передаваемый как беззнаковое целое (см. 4.3, таблица 16).

hcod_sf [] — Кодовое слово Хаффмана из кодовой таблицы, используемой для кодирования масштабных коэффициентов (см. 4.3, таблица 18).

hcod [sect_cb [g] [i]] [w] [x] [y] [z] — Кодовое слово Хаффмана из кодовой книги *sect_cb [g] [i]*, с помощью которого кодируется следующая четверка (*w*, *x*, *y*, *z*) спектральных коэффициентов, где *w*, *x*, *y*, *z* — квантованные спектральные коэффициенты. В пределах группы из *n* спектральных коэффициентов, *w*, *x*, *y*, *z* упорядочиваются согласно 6.3.5 так, чтобы *x_quant [group] [win] [sfb] [bin] = w*, *x_quant [group] [win] [sfb] [bin+1] = x*, *x_quant [group] [win] [sfb] [bin+2] = y* и *x_quant [group] [win] [sfb] [bin+3] = z*. Группы из *n* спектральных коэффициентов следуют снизу вверх по частоте в пределах текущего раздела (см. 4.3, таблица 20).

hcod [sect_cb [g] [i]] [y] [z] — Кодовое слово Хаффмана из кодовой книги *sect_cb [g] [i]*, с помощью которого кодируется следующая пара (*y*, *z*) спектральных коэффициентов, где *y*, *z* — квантованные спектральные коэффициенты. В пределах группы из *n* спектральных коэффициентов *y*, *z* упорядочиваются согласно 6.3.5 так, чтобы *x_quant [group] [win] [sfb] [bin] = y* и *x_quant [group] [win] [sfb] [bin+1] = z*. Группы из *n* спектральных коэффициентов следуют снизу вверх по частоте в пределах текущего раздела (см. 4.3, таблица 20).

quad_sign_bits — Знаковые биты для коэффициентов, отличных от нуля в спектральной четверке. '1' указывает на отрицательный коэффициент, '0' — на положительный. Биты, относящиеся к более низким по частоте спектральным коэффициентам, следуют первыми (см. 4.3, таблица 20).

pair_sign_bits — Знаковые биты для коэффициентов, отличных от нуля в спектральной паре. '1' указывает на отрицательный коэффициент, '0' — на положительный. Биты, относящиеся к более низким по частоте спектральным коэффициентам, следуют первыми (см. 4.3, таблица 20).

hcod_esc_y — Escape-последовательность для квантованного спектрального коэффициента *y* пары (*y*, *z*), связанной с предыдущим кодовым словом Хаффмана (см. 4.3, таблица 20).

hcod_esc_z — Escape-последовательность для квантованного спектрального коэффициента *z* пары (*y*, *z*), связанной с предыдущим кодовым словом Хаффмана (см. 4.3, таблица 20).

pulse_data_present — 1 бит, указывающий на использование импульсного escape. (1) — используется, (0) — не используется (см. 4.3, таблица 21). *Pulse_data_present* должен быть 0 для *EIGHT_SHORT_SEQUENCE*.

number_pulse — 2 бита, указывающие, сколько Escape импульсов используется. Число Escape импульсов от 1 до 4 (см. 4.3, таблица 21).

pulse_start_sfb — 6 бит, определяющих индекс самого низкой полосы масштабных коэффициентов, где достигается импульсный escape (см. 4.3, таблица 21).

pulse_offset [i] — 5 бит, указывающих на смещение (см. 4.3, таблица 21).

pulse_amp [i] — 4 бита, указывающие на неиспользованную амплитуду импульса (см. 4.3, таблица 21).

7.2.2 Элементы справки

sect_start [g] [i] — Смещение относительно первой полосы масштабных коэффициентов в разделе *i* из группы *g* (см. 4.3, таблица 17).

sect_end [g] [i] — Смещение относительно на единицу большей, чем последняя полоса масштабных коэффициентов в разделе *i* из группы *g* (см. 4.3, таблица 17).

num_sec [g] — Число разделов в группе *g* (см. 4.3, таблица 17).

escape_flag — Значение 16 в кодовой книге Хаффмана ESC.

escape_prefix — Последовательность битов из N единичных битов.

escape_separator — Один нулевой бит.

escape_word — Беззнаковое целое размером (N+4), msb первым.

escape_sequence	Последовательность *escape_prefix*, *escape_separator* и *escape_word*.
escape_code	2 (N+4) + escape_word.
x_quant [g] [win] [sfb] [bin]	Декодированное значение Хаффмана для группы *g*, окна *win*, полосы масштабных коэффициентов *sfb*, коэффициента *bin*.
spec [w] [k]	Спектр без чередования. *w* в диапазоне от 0 до *num_windows–1* и *k* в диапазоне от 0 до *swb_offset [num_swb]–1*.

Инструмент прозрачного кодирования требует следующих констант (см. 4.3, *spectral_data* ()).

ZERO_HCB	0
FIRST_PAIR_HCB	5
ESC_HCB	11
QUAD_LEN	4
PAIR_LEN	2
INTENSITY_HCB2	14
INTENSITY_HCB	15
ESC_FLAG	16

7.3 Процесс декодирования

Четверки или пары квантованных спектральных коэффициентов являются кодированными с помощью кодов Хаффмана и передаются в порядке от коэффициента самой низкой частоты и до коэффициента самой высокой частоты. При наличии нескольких окон в блоке (*EIGHT_SHORT_SEQUENCE*) сгруппированные и чередующиеся наборы спектральных коэффициентов обрабатываются как единый набор коэффициентов, которые следуют от низкой частоты к высокой. Чередование может быть устранено после декодирования коэффициентов (см. 6.3.5). Коэффициенты сохраняются в массиве *x_quant [g] [win] [sfb] [bin]*, и порядок передачи кодовых комбинаций Хаффмана такой, чтобы коэффициенты декодировались в порядке получения и записи в массив. *bin* — наиболее быстро меняющийся индекс, а *g* — наиболее медленный. В пределах кодовой комбинации порядок декодирования четверок — *w*, *x*, *y*, *z*; порядок декодирования пар — *y*, *z*. Набор коэффициентов делится на разделы, и информация разделения передается, начиная с самого низкого частотного раздела до самого высокого частотного раздела. Спектральная информация для разделов, которые кодируются с «нулевой» кодовой книгой, не передается, поскольку эта спектральная информация равна нулю. Точно так же, спектральная информация для разделов, кодированных «интенсивностными» кодовыми книгами, не передается. Спектральная информация для всех полос масштабных коэффициентов, выше и равных *max_sfb*, для которых нет данных раздела, равна нулю.

Существует единственная дифференциальная кодовая книга масштабных коэффициентов, которая представляет диапазон значений как показано в таблице 58. Дифференциальная кодовая книга масштабных коэффициентов дана в таблице А.1. Существует одиннадцать кодовых книг Хаффмана для спектральных данных, в соответствии с таблицей 59. Кодовые книги даны в таблицах А.2—А.12. Существует три других «кодовых книги» выше и вне фактических кодовых книг Хаффмана, в частности «нулевая» кодовая книга, указывающая, что ни масштабные коэффициенты, ни квантованные данные не будут переданы, и «интенсивностные» сборники кодов, указывающие, что этот отдельный канал является частью канальной пары, и что данные, которые обычно были бы в обычном случае масштабными коэффициентами, вместо этого являются данными для *intensity stereo*. В этом случае никакие квантованные спектральные данные не передаются. Кодовые книги 12 и 13 находятся в резерве.

Кодовые книги Хаффмана для спектральных значений кодируют пары или четверки квантованных спектральных коэффициентов без знака или со знаком, как показано в таблице 59. В этой таблице также указано самое большое абсолютное значение (*LAV*), которое может быть закодировано каждой кодовой книгой, и определен двоичный массив *unsigned_cb []*, в котором 1 соответствует беззнаковой кодовой книге и 0 — знаковой.

Результатом декодирования Хаффмана каждого кодового слова является индекс кодового слова, приведенный в первой графе таблицы А.1. Этот результат преобразовывается в требуемый дифференциальный масштабный коэффициент путем добавления *index_offset* к индексу. Значение *index_offset* равно –60, как показано в таблице 58. Аналогично результатом декодирования Хаффмана каждой группы из *n* спектральных коэффициентов является индекс кодового слова, приведенный в первой графе таблиц А.2—А.12. Этот индекс преобразовывается в *n* спектральных значений в соответствии со следующим псевдо с-кодом:
unsigned = булевое значение *unsigned_cb* [i], приведенное во второй графе таблицы 59.
dim = размерность кодовой книги, приведенная в третьем графе таблицы 59.

lav = *LAV*, приведенный в четвертом графе таблицы 59.
idx = индекс кодового слова.

```
if(unsigned){
        mod = lav + 1;
        off = 0;
}
else {
        mod = 2*lav + 1;
        off = lav;
}
if (dim == 4) {
        w = INT (idx / (mod*mod*mod)) – off;
        idx –= (w+off) * (mod*mod*mod)
        x = INT (idx / (mod*mod)) – off;
        idx –=  (x+off) * (mod*mod)
        Y = INT (idx/mod) – off;
        idx –=  (y+off) *mod
        z = idx – off;
}
else {
        Y = INT (idx/mod) – off;
        idx –=  (y+off) *mod
        z = idx – off;
}
```

Если кодовая книга Хаффмана представляет знаковые значения, декодирование *n* квантованных спектральных значений заканчивается после декодирования Хаффмана и преобразования индекса кодового слова в квантованные спектральные коэффициенты.

Если кодовая книга представляет беззнаковые значения, биты знака, связанные с ненулевыми коэффициентами, следуют сразу за кодовой комбинацией Хаффмана, '1' указывает на отрицательный коэффициент, а '0' — на положительный. Например, если кодовая комбинация Хаффмана из сборника кодов 7

hcod [7] [y] [z]

была обнаружена, то сразу после нее в потоке битов следует
pair_sign_bits, которое является полем переменной длины от 0 до 2 битов. Это может быть обнаружено в потоке битов как

```
if (y! = 0)
        if (one_sign_bit == 1)
                y =–y;
if (z! = 0)
        if (one_sign_bit == 1)
                z = – z,
```

где *one_sign_bit* — следующий бит в потоке битов, а *pair_sign_bits* — объединение *one_sign_bit* полей.

Кодовая книга *ESC* является особым случаем. Она представляет значения от 0 до 16 включительно, однако кодирует фактически значения от 0 до 15, и значение 16 является флагом *escape_flag*, который сигнализирует присутствие *hcod_esc_y* или *hcod_esc_z*, любой из которых будет обозначен как *escape_sequence*. Этот *escape_sequence* разрешает кодирование квантованных спектральных элементов с *LAV*> 15. Он состоит из *escape_prefix* из N единичных бит, *escape_separator* из одного нулевого бита, *escape_word* из (N+4) битов, представляющих беззнаковое целое. Декодированное значение *escape_sequence* равно 2 *(N+4)* + *escape_word*. Требуемый квантованный спектральный коэффициент получен, когда знак, определяемый *pair_sign_bits*, применен к значению *escape_sequence*. Другими словами, *escape_sequence* 00000 декодируется как 16, *escape_sequence* 01111 как 31, *escape_sequence* 1000000 как 32, один из 1011111 как 63, и т. д. Следует учесть, что ограничения в 10.3 диктуют, что длина *escape_sequence* всегда меньше, чем 22 бита. Для *escape* кодовых слов Хаффмана порядок элементов данных является кодовой комбинацией Хаффмана, сопровождаемой от 0 до 2 битов знака, сопровождаемых от 0 до 2 *escape*-последовательностями.

Когда *pulse_data_present* равен 1 (используется импульсный *escape*), один или несколько квантованных коэффициентов были заменены в кодере коэффициентами с меньшими амплитудами. Число замененных коэффициентов обозначается *number_pulse*. При восстановлении квантованных спектральных коэф-

фициентов *x_quant* эта замена компенсируется путем добавления или вычитания *pulse_amp* из ранее декодированных коэффициентов, индексы частоты которых обозначаются *pulse_start_sfb* и *pulse_offset*. Следует учесть, что импульсный *escape*-метод недопустим для блока, *window_sequence* которого *EIGHT_SHORT_SEQUENCE*. Процесс декодирования определяется в следующем псевдо с-коде:

```
if (pulse_data_present) {
    g = 0;
    win = 0;
    k = swb_offset [pulse_start_sfb];
    for (j = 0; j <number_pulse+1; j ++) {
        k + = pulse_offset [j];
        /* translate_pulse_parameters (); */
        for (sfb = pulse_start_sfb; sfb <num_swb; sfb ++) {
            if (k <swb_offset [sfb+1]) {
                bin = k – swb_offset [sfb];
                break;
            }
        }
        /* восстановление коэффициентов */
        if (x_quant [g] [win] [sfb] [bin]> 0)
            x_quant [g] [win] [sfb] [мусорное ведро] + = pulse_amp [j];
        else
            x_quant [g] [win] [sfb] [bin] – = pulse_amp [j];
    }
}
```

Несколько инструментов декодера (*TNS*, банк фильтров) получают доступ к спектральным коэффициентам без чередования, т. е. все спектральные коэффициенты упорядочиваются согласно номеру окна и частоте в пределах окна. Это обозначается указанием *spec [w][k]*, вместо *x_quant [g] [w] [sfb] [bin]*.

Следующий псевдо с-код указывает на соотношение между четырехмерной, с чередованием, структурой массива *x_quant* [] [] [] [] и двумерной, без чередования, структурой массива spec[] []. В последнем массиве первый индекс увеличивается по отдельным окнам в последовательности окон, и второй индекс увеличивается по спектральным коэффициентам, которые соответствуют каждому окну, внутри которого коэффициенты следуют линейно от низких частот к верхним.

```
quant_to_spec ()
    {k = 0;
    for (g = 0; g <num_window_groups; g + +) {
        j = 0;
        for (sfb = 0; sfb <num_swb; sfb ++) {
            width = swb_offset [sfb+1] – swb_offset [sfb];
            for (win = 0; win<window_group_length [g]; win ++) {
                for (bin = 0; bin <width; bin ++) {
                    spec [win+k] [bin+j] = x_quant [g] [win] [sfb] [bin];
                }
            }
            j + = width;
        }
        k + = window_group_length [g]
    }
}
```

7.4 Таблицы

Таблица 58 — Параметры кодовой книги Хаффмана для масштабных коэффициентов

Номер кодовой книги	Размерность кодовой книги	index_offset	Диапазон значений	Кодовая книга приведена в:
0	1	–60	–60 … +60	Таблица А.1

Таблица 59 — Параметры кодовой книги Хаффмана для спектральных значений

Номер кодовой книги	*unsigned_cb [i]*	Размерность кодовой книги	*LAV* кодовой книги	Кодовая книга приведена в:
0	—	—	0	—
1	0	4	1	Таблица А.2
2	0	4	1	Таблица А.3
3	1	4	2	Таблица А.4
4	1	4	2	Таблица А.5
5	0	2	4	Таблица А.6
6	0	2	4	Таблица А.7
7	1	2	7	Таблица А.8
8	1	2	7	Таблица А.9
9	1	2	12	Таблица А. 10
10	1	2	12	Таблица А. 11
11	1	2	(16) *ESC*	Таблица А. 12
12	—	—	(в резерве)	—
13	—	—	(в резерве)	—
14	—	—	Интенсивность не в фазе	—
15	—	—	Интенсивность в фазе	—

8 Квантование

8.1 Описание инструмента

Для квантования спектральных коэффициентов в кодере используется неравномерное квантование. Декодер должен выполнить обратное неравномерное квантование после декодирования Хаффмана масштабных коэффициентов (см. разделы 7 и 11) и спектральных данных (см. раздел 7).

8.2 Элементы справки

x_quant [g] [win] [sfb] [bin] Квантованный спектральный коэффициент для группы *g*, окна *win*, полосы масштабных коэффициентов *sfb*, коэффициента *bin*.

x_invquant [g] [win] [sfb] [bin] Спектральный коэффициент для группы *g*, окна *win*, полосы масштабных коэффициентов *sfb*, коэффициента *bin* после обратного квантования.

8.3 Процесс декодирования

Обратное квантование описывается следующей формулой:

$x\_invquant = Sign(x\_quant) \; |x\_quant|^{4/3} \; 3 \; V \; k.$

Максимальная допустимая абсолютная амплитуда для x_quant равна 8191. Обратное квантование выполняется следующим образом:

```
for (g = 0; g <num_window_groups; g + +) {
  for (sfb = 0; sfb <max_sfb; sfb ++) {
        width = (swb_offset [sfb+1] – swb_offset [sfb]);
        for (win = 0; win <window_group_len [g]; win ++) {;
              for (bin = 0, bin <width; bin ++) {
        x_invquant [g] [win] [sfb] [bin] = sign (x_quant [g] [win] [sfb] [bin]) *
        abs (x_quant [g] [win] [sfb] [bin]) ^4/3);
              }
        }
  }
}
```

9 Масштабные коэффициенты

9.1 Описание инструмента

Основным методом регулирования шумов квантования в частотной области является формирование шума с использованием масштабных коэффициентов. С этой целью спектр делится на несколько групп спектральных коэффициентов, называемых полосами масштабных коэффициентов, для которых используется один масштабный коэффициент (см. 6.3.4). Масштабный коэффициент представляет собой значение коэффициента усиления, которой используется для изменения амплитуды всех спектральных коэффициентов в этой полосе масштабного коэффициента. Этот механизм используется для изменения распределения шумов квантования, вызванных неравномерным квантованием, в спектральной области.

Для *window_sequences,* которые содержат окна *SHORT_WINDOW*, может быть применена группировка, т. е. для конкретного количества последовательных окон *SHORT_WINDOW* может использоваться только один набор масштабных коэффициентов. Каждый масштабный коэффициент в этом случае применяется к группе соответствующих полос масштабных коэффициентов (см. 6.3.4).

В этом инструменте масштабные коэффициенты применяются к деквантованным коэффициентам для восстановления спектральных значений.

9.2 Определения

9.2.1 Функции данных

scale_factor_data () Часть потока битов, которая содержит дифференциально-кодированные масштабные коэффициенты (см. таблицу 18)

9.2.2 Элементы данных

global_gain 8-разрядное беззнаковое целое, представляющее значение первого масштабного коэффициента. Это также начальное значение для следующих дифференциально-кодированных масштабных коэффициентов (см. таблицу 16).

hcod_sf [] Кодовая комбинация Хаффмана из таблицы кода Хаффмана, используемая для кодирования масштабных коэффициентов (см. таблицу 18 и 7.2).

9.2.3 Элементы справки

dpcm_sf [g] [sfb] Дифференциально-кодированный масштабный коэффициент группы *g*, полосы масштабных коэффициентов *sfb*.

x_rescal [] Перемасштабированные спектральные коэффициенты

sf [g] [sfb] Массив масштабных коэффициентов каждой группы

get_scale_factor_gain () Функция, которая возвращает значение усиления, соответствующего масштабному коэффициенту

9.3 Процесс декодирования

9.3.1 Полосы масштабных коэффициентов

Масштабные коэффициенты используются для формирования шумов квантования в спектральной области. С этой целью спектр делится на несколько полос масштабных коэффициентов (см. 6.3.4). Каждой полосе масштабных коэффициентов соответствует масштабный коэффициент, который представляет определенное значение коэффициента усиления, которое должно быть применено ко всем спектральным коэффициентам в этой полосе масштабных коэффициентов. В случае *EIGHT_SHORT_SEQUENCE* полоса масштабных коэффициентов может содержать многократные оконные полосы масштабных коэффициентов *SHORT_WINDOW* (см. 6.3.4 и 6.3.5).

9.3.2 Декодирование масштабных коэффициентов

Для всех масштабных коэффициентов разность с предыдущим значеним кодируется при помощи кодовой книги Хаффмана, данной в таблице А1. Начальное значение задается явным образом как 8-битный ИКМ отсчет элемента данных *global_gain*. Масштабный коэффициент не передается для полос масштабных коэффициентов, которые кодируются со сборником кодов Хаффмана *ZERO_HCB*. Если сборник кодов Хаффмана для полос масштабных коэффициентов кодируется с *INTENSITY_HCB* или *INTENSITY_HCB2*,

масштабный коэффициент используется для *intensity stereo* (см. раздел 7 и 10.2). В этом случае обычный масштабный коэффициент не существует (но инициализируется в нуль, чтобы иметь определенное значение в массиве).

Следующий псевдокод описывает декодирование масштабных коэффициентов *sf [g] [sfb]*:

```
last_sf = global_gain;
for (g = 0; g <num_window_groups; g + +) {
 for (sfb = 0; sfb <max_sfb; sfb ++) {
if (sfb_cb [g] [sfb]! = ZERO_HCB && sfb_cb [g] [sfb]! = INTENSITY_HCB
&& sfb_cb [g] [sfb]! = INTENSITY_HCB2) {
dpcm_sf = decode_huffman () – index_offset;/*см. раздел 7*/
sf [g] [sfb] = dpcm_sf + last_sf;
last_sf = sf [g] [sfb];
}
else {
sf [g] [sfb] = 0;
}
  }
}
```

Следует учесть, что масштабные коэффициенты *sf [g] [sfb].* должны быть в пределах диапазона 0…255 включительно.

9.3.3 Использование масштабных коэффициентов

Спектральные коэффициенты всех полос масштабных коэффициентов должны повторно масштабироваться в соответствии с их масштабными коэффициентами. В случае последовательности окон, которая состоит из группы коротких окон, все коэффициенты в сгруппированных полосах масштабных коэффициентов окна должны масштабироваться с использованием того же самого масштабного коэффициента.

В случае window_sequences из одного окна полосы масштабных коэффициентов их соответствующие коэффициенты расположены в порядке возрастания частоты. В случае *EIGHT_SHORT_SEQUENCE* и группировки спектральные коэффициенты сгруппированных коротких окон чередуются с полосами масштабных коэффициентов окна.

Перемасштабирование выполняется согласно следующему псевдокоду:

```
for (g = 0; g <num_window_groups; g + +) {
 for (sfb = 0; sfb <max_sfb; sfb ++) {
        width = (swb_offset [sfb+1] – swb_offset [sfb]);
        for (win = 0; win <window_group_len [g]; win ++) {;
                win = get_scale_factor_gain (sf [g] [sfb]).
                for (k = 0; k <width; k + +) {
 x_rescal [g] [window] [sfb] [k] = x_invquant [g] [window] [sfb] [k] * gain;
                }
        }
 }
}
```

Функция *get_scale_factor_gain (sf [g] [sfb])* возвращает коэффициент усиления, который соответствует масштабному коэффициенту. Возвращаемое значение получается из уравнения

$$gain = 2^{0,25\cdot(sf[g][sfb]-SF\_OFFSET)}.$$

Постоянная *SF_OFFSET* должна быть установлена равной 100.

Следующий псевдокод описывает эту операцию:

```
get_scale_factor_gain (sf [g] [sfb]) {
 SF_OFFSET = 100;
 gain = 2 ^ (0.25 * (sf [g] [sfb] – SF_OFFSET));
 return(gain);
}
```

10 Совместное кодирование

10.1 *M/S stereo*

10.1.1 Описание инструмента

Кодирование *M/S stereo* применяется к парам каналов. Каналы чаще всего объединяются так, чтобы они имели симметричное расположение относительно слушателя, например левый/правый или левый окружения/правый окружения. Первый канал в паре обозначается как «левый», а второй — как «правый». Вектор, образованный сигналами левого и правого каналов, восстанавливается или де-матрицируется с помощью единичной матрицы

$$\begin{bmatrix} l \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} l \\ r \end{bmatrix}$$

или обратной матрицы *M/S*

$$\begin{bmatrix} l \\ r \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} m \\ s \end{bmatrix}.$$

Решение о том, какая матрица должна использоваться, принимается для каждой полосы масштабных коэффициентов в соответствии с флагами ms_used. Кодирование M/S может применяться только, если common_window равно '1' (см. 6.3.1).

10.1.2 Определения

10.1.2.1 Элементы данных

ms_mask_present Двухбитное поле, определяющее MS маску
00 — Все нули
01 — Маска max_sfb полос ms_used следует за этим полем
10 — Все единицы
11 — В резерве
(см. 4.3, таблица 14),

ms_used [g] [sfb] Однобитный флаг для каждой полосы масштабных коэффициентов, указывающий, что кодирование *M/S* используется в группе окон *g* и полосе масштабных коэффициентов *sfb* (см. 4.3, таблица 14).

10.1.2.2 Элементы справки

l_spec [] Массив, содержащий спектр левого канала соответствующей пары каналов.

r_spec [] Массив, содержащий спектр правого канала соответствующей пары каналов.

is_intensity (g, sfb) Функция, возвращающая состояние интенсивности, определенная в 10.2.3.

10.1.3 Процесс декодирования

Восстановление (спектральные коэффициенты первого («левого») и второго («правого») каналов) выполняется в соответствии с *mask_present* и флагами *ms_used [] []* следующим образом:

```
if (mask_present> = 1) {
for (g = 0; g <num_window_groups; g + +) {
for (b = 0; b <window_group_length [g]; b ++) {for (sfb = 0; sfb <max_sfb; sfb ++) {
if ((ms_used [g] [sfb] || mask_present == 2) &&! is_intensity (g, sfb)) {
for (i = 0; i <swb_offset [sfb+1]–swb_offset [sfb]; i + +) {
tmp = l_spec [g] [b] [sfb] [i] – r_spec [g] [b] [sfb] [i]:
l_spec [g] [b] [sfb] [i] = l_spec [g] [b] [sfb] [i] + r_spec [g] [b] [sfb] [i]:
r_spec [g] [b] [sfb] [i] = tmp;}
}
}
```

Следует учесть, что ms_used [] [] используется также в контексте кодирования *intensity stereo*. Если кодирование *intensity stereo* было использовано для определенной полосы масштабных коэффициентов, декодирование *стерео M/S* не выполняется.

10.2 *Intensity stereo*

10.2.1 Описание инструмента

Этот инструмент используется, чтобы реализовать кодирования *intensity stereo* пары каналов. Оба выходных канала получаются из одного набора спектральных коэффициентов после обратного процесса квантования. Это выполняется выборочно на основе полос масштабных коэффициентов при активном *intensity stereo*.

10.2.2 Определения

10.2.2.1 Элементы данных

hcod_sf [] Кодовая комбинация Хаффмана из таблицы Хаффмана, используемая для кодирования масштабных коэффициентов (см. 7.2).

10.2.2.2 Элементы справки

dpcm_is_position [] []	Дифференциально-кодированная позиция intensity stereo
is_position [group] [sfb]	Позиция *intensity stereo* для каждой группы и полосы масштабных коэффициентов
l_spec []	Массив, содержащий спектр левого канала соответствующей пары канала
r_spec []	Массив, содержащий спектр левого канала соответствующей пары канала

10.2.3 Процесс декодирования

Использование кодирования *intensity stereo* сигнализируется при помощи псевдокодовых книг *INTENSITY_HCB* и *INTENSITY_HCB2* (15 и 14) в правом канале *channel_pair_element ()*, имеющем общий *ics_info () (common_window* == 1). *INTENSITY_HCB* и *INTENSITY_HCB2* сигнализируют синфазное и несовпадающее по фазе кодирование *intensity stereo* соответственно.

Кроме того, фазовое соотношение кодирования *intensity stereo* может быть реверсировано посредством поля *ms_used:* поскольку кодирование *M/S stereo* и кодирование *intensity stereo* являются взаимоисключающими для определенной полосы масштабных коэффициентов и группы, основное фазовое соотношение, обозначенное таблицами Хаффмана, изменяется от синфазного до несовпадающего по фазе или наоборот, если соответствующий бит *ms_used* устанавливается для соответствующей полосы.

Информация для декодирования *intensity stereo* представлена "позицией *intensity stereo*", указывающей на соотношение между масштабированием левого и правого каналов. Если кодирование *intensity stereo* является активным для определенной группы и полосы масштабных коэффициентов, значение позиции *intensity stereo* передается вместо масштабного коэффициента правого канала.

Позиции интенсивности кодируются точно так же, как масштабные коэффициенты, т. е. с помощью кодирования методом Хаффмана разностных значений, за исключением двух отличий:

нет первого значения, которое передается как ИКМ. Вместо этого следует дифференциальное декодирование, предполагая, что последнее значение позиции intensity stereo равно нулю;

дифференциальное декодирование выполняется отдельно для масштабных коэффициентов и позиций *intensity stereo*. Другими словами, декодер масштабных коэффициентов игнорирует значения позиции *intensity stereo* и наоборот (см. 9.3.2).

Одна и та же кодовая книга используется для кодирования позиций *intensity stereo* и масштабных коэффициентов. Две псевдофункции определены для использования в декодировании *intensity stereo*.

```
function is_intensity (group, sfb) {
+ 1 для оконных групп/полос масштабных коэффициентов с кодовой книгой правого канала
  sfb_cb [group] [sfb] == INTENSITY_HCB
– 1 для оконных групп / полос масштабных коэффициентов с кодовой книгой правого канала
  sfb_cb [group] [sfb] == INTENSITY_HCB2
0 else
}
function invert_intensity (group, sfb) {
  1–2*ms_used [group] [sfb],if (ms_mask_present == 1)
  +1      else
}
```

Декодирование *intensity stereo* для одной пары каналов определяется следующим псевдокодом:

```
p = 0;
for (g = 0; g <num_window_groups; g + +) {
/* Декодировать позиции интенсивности для этой группы */
for (sfb = 0; sfb <max_sfb; sfb ++),
 if (is_intensity (g, sfb))
is_position [g] [sfb] = p + = dpcm_is_position [g] [sfb];
/* Выполнить декодирование intensity stereo */
for (b = 0; b <window_group_length [g]; b ++) {for (sfb = 0; sfb <max_sfb; sfb ++) {
if (is_intensity (g, sfb)) {
scale = is_intensity (g, sfb) * invert_intensity (g, sfb) *0.5 ^ (0.25*is_position [g] [sfb]);
/* Масштабировать от левого к правому каналу, не трогать левый канал */
for (i = 0; i <swb_offset[sfb+1]–swb_offset [sfb]; i ++)
                r_spec [g] [b] [sfb] [i] = scale * l_spec [g] [b] [sfb] [i];
                }
        }
 }
}
```

10.2.4 Интеграция с инструментом внутриканального предсказания

Для полос масштабных коэффициентов, кодированных в *intensity stereo*, соответствующие предсказатели в правом канале выключаются для эффективного переопределения состояния, определенного маской *prediction_used*. Обновление этих предсказателей выполняется использованием декодируемых *intensity stereo* спектральных значений правого канала в качестве "последнего квантованного значения" $x_{rec}(n–1)$. Эти значения следуют из процесса масштабирования, от левого к правому каналу, как описано в псевдокоде.

10.3 Спаренный канал

10.3.1 Описание инструмента

Элементы спаренного канала обеспечивают две функции: во-первых, спаренные каналы могут использоваться для реализации обобщенного *intensity stereo* кодирования, где спектральные значения каналов могут быть совместно использованы; во-вторых, спаренные каналы могут использоваться для динамического сведения одного звукового объекта в стерео.

Следует учесть, что этот инструмент включает параметры, зависимые от определенного профиля (см. 5.1).

10.3.2 Определения

10.3.2.1 Элементы данных

ind_sw_cce_flag	*Один бит, указывающий, является ли спаренный целевой элемент синтаксиса независимо (1) или зависимо (0) переключаемым CCE (см. 4.3, таблицу 22).*
num_coupled_elements	*Количество спаренных целевых каналов равно num_coupled_elements+1. Минимальное значение 0 указывает на 1 спаренный целевой канал (см. 4.3, таблицу 22).*
cc_target_is_cpe	*Один бит, указывающий, является ли спаренный целевой элемент синтаксиса CPE (1) или SCE (0) (см. 4.3, таблицу 22).*
cc_target_tag_select	*Четырехбитное поле, определяющее element_instance_tag спаренного целевого элемента синтаксиса (см. 4.3, таблицу 22).*
cc_l	*Один бит, указывающий, что список значений gain_element применяется к левому каналу пары каналов (см. 4.3, таблицу 22).*
cc_r	*Один бит, указывающий, что список значений gain_element применяется к правому каналу пары каналов (см. 4.3, таблицу 22).*
cc_domain	*Один бит, указывающий, выполняется ли спаривание каналов до (0) или после (1) TNS декодирования спаренных целевых каналов (см. 4.3, таблицу 22).*

gain_element_sign — *Один бит, указывающий, содержат ли переданные значения gain_element информацию о совпадающем по фазе/несовпадающем по фазе спаривании (1) или нет (0) (см. 6.3, таблицу 22).*

gain_element_scale — *Определяет разрешение амплитуды cc_scale операции масштабирования согласно таблице 61 (см. 4.3, таблицу 22).*

common_gain_element_present [c] — *Один бит, указывающий, что передается кодированный по Хафману common_gain_element (1) или передаются кодированные по Хафману разностные gain_elements (0) (см. 4.3, таблицу 22).*

10.3.2.2 Элементы справки

dpcm_gain_element[][] — *Дифференцированно-кодированный элемент усиления.*

gain_element [group] [sfb] — *Элемент усиления для каждой группы и полосы масштабных коэффициентов.*

common_gain_element[] — *Элемент усиления, который используется для всех групп окон и полос масштабных коэффициентов одного целевого спаренного канала.*

spectrum_m (idx, domain) — *Указатель на спектральные данные, связанные с single_channel_element () с индексом idx. В зависимости от значения "domain" указываются спектральные коэффициенты до (0) или после (1) TNS декодирования.*

spectrum_l (idx, domain) — *Указатель на спектральные данные, связанные с левым каналом channel_pair_element () с индексом idx. В зависимости от значения "domain" указываются спектральные коэффициенты до (0) или после (1) TNS декодирования.*

spectrum_r (idx, domain) — *Указатель на спектральные данные, связанные с правым каналом channel_pair_element () с индексом idx. В зависимости от значения "domain", указываются спектральные коэффициенты до (0) или после (1) TNS декодирования.*

10.3.3 Процесс декодирования

Спаренный канал основан на встроенном *single_channel_element ()*, который объединяется для этого с некоторыми соответствующими полями.

Спаренные целевые элементы синтаксиса (*SCE* или *CPEs*) адресуются, используя два элемента синтаксиса. Во-первых, поле *cc_target_is_cpe* указывает, адресуются ли *SCE* или *CPE*. Во-вторых, поле *cc_target_tag_select* указывает *instance_tag SCE/CPE*.

Операция масштабирования, включенная в спаривание каналов, определяется значениями *gain_element*, которые описывают используемый коэффициент усиления и знак. В соответствии с процедурами кодирования для масштабных коэффициентов и позиций *intensity stereo* значения *gain_element* дифференциально кодируются с использованием таблицы Хаффмана для масштабных коэффициентов. Точно так же декодируемые коэффициенты усиления для пары относятся к оконным группам спектральных коэффициентов.

Независимо коммутируемые *CCE* и зависимо коммутируемые *CCE*

Есть два вида *CCE:* "независимо переключаемые" и "зависимо переключаемые". Независимо переключаемым *CCE* является *CCE*, в котором статус окна (т. е. *window_sequence* и *window_shape*) *CCE* не должен соответствовать статусу окна любого спаренного *SCE* или спаренных каналов *CPE*. Есть несколько важных ограничений:

Во-первых, требуется, чтобы независимо переключаемый *CCE* мог использовать только элемент *common_gain*, а не *список gain_elements*.

Во-вторых, *CCE* должен быть полностью декодирован во временном пространстве (включая банк фильтров синтеза) до масштабирования и добавления к различным *SCE* и каналам *CPE*, с которыми он спарен, в случае несоответствия статуса окна.

У зависимо переключаемого *CCE*, с другой стороны, должен быть статус окна, соответствующий всем целевым *SCE* и каналы *CPE*, которые спарены согласно списку элементов *cc_l* и *cc_r*. В этом случае *CCE* должен быть полностью декодирован в частотном пространстве и затем масштабироваться в соответствии со списком усиления до добавления к целевому *SCE* или каналам *CPE*.

Следующий псевдокод в функции *decode_ coupling_channel* () определяет декодирование для зависимо переключаемого элемента спаренного канала. Сначала спектральные коэффициенты встроенного *single_channel_element* () декодируются и помещаются во внутренний буфер. Так как элементы усиления для первой спаренной цели (*list_index* == 0) не передаются, все значения *gain_element*, связанные с этой

целью, как предполагается, равны 0, т. е. спаренный канал добавляется к спаренному целевому каналу с его масштабированием. Иначе спектральные коэффициенты масштабируются и добавляются к коэффициентам спаренных целевых каналов, используя соответствующий список значений *gain_element*.

Независимо коммутируемый *CCE* декодируется как зависимо коммутируемый *CCE*, имеющий только элементы *common_gain_element*. Однако получающийся масштабированный спектр преобразовывается обратно к его временному представлению и затем спаривается.

Следует учесть, что списки *gain_element* могут быть совместно использованы левым и правым каналами целевого элемента пары каналов. Это сигнализируется нулевыми *cc_l* и *cc_r*, как обозначено в таблице 60.

Таблица 60 — Совместное использование списков *gain_element*

cc_l, *cc_r*	Общий список усиления представлен	Левый список усиления представлен	Правый список усиления представлен
0, 0	да	нет	нет
0, 1	нет	нет	да
1, 0	нет	да	нет
1, 1	нет	да	да

decode_coupling_channel ()

{

- декодирует спектральные коэффициенты встроенного *single_channel_element* в буфер *«cc_spectrum []».*

```
/* Объединение спектральных коэффициентов по целевым каналам */
list_index = 0;
for (c = 0; c <num_coupled_elements+1; c ++) {
if (! cc_target_is_cpe [c]) {
couple_channel (cc_spectrum, spectrum_m (cc_target_tag_select [c], cc_domain), list_index ++);
}
if (cc_target_is_cpe [c]) {
if (! cc_l [c] &&! cc_r [c]) {
couple_channel (cc_spectrum,spectrum_l (cc_target_tag_select [c], cc_domain), list_index);
couple_channel (cc_spectrum, spectrum_r (cc_target_tag_select [c], cc_domain), list_index ++);
}
if (cc_l [c]) {
couple_channel (cc_spectrum,
spectrum_l (cc_target_tag_select [c], cc_domain), list_index ++));
if (cc_r [c]) {
couple_channel (cc_spectrum, spectrum_r (cc_target_tag_select [c], cc_domain), list_index ++));
couple_channel (source_spectrum [ ], dest_spectrum [ ], gain_list_index)
{
idx = gain_list_index;
a = 0;
cc_scale = cc_scale_table [gain_element_scale];
for (g = 0; g <num_window_groups; g + +) {
/* Декодировать спаренные элементы усиления для этой группы */
if (common_gain_element_present [idx]) {
for (sfb = 0; sfb <max_sfb; sfb ++) {
cc_sign [idx] [g] [sfb] = 1;
gain_element [idx] [g] [sfb] = common_gain_element [idx];}
else {
for (sfb = 0; sfb <max_sfb; sfb ++) {
if (sfb_cb [g] [sfb] == ZERO_HCB)
continue;
```

```
if (gain_element_sign) {
cc_sign [idx] [g] [sfb] = 1 – 2* (dpcm_gain_element [idx] [g] [sfb] & 0x1);
 gain_element [idx] [g] [sfb] = a+ = (dpcm_gain_element [idx] [g] [sfb]>> 1);
} else {
cc_sign [idx] [g] [sfb] = 1;
gain_element [idx] [g] [sfb] = a+ = dpcm_gain_element [idx] [g] [sfb];}
}
/* Выполнять спаривание целевых каналов */
for (b = 0; b <window_group_length [b]; b + +) {
 for (sfb = 0; sfb <max_sfb; sfb ++) {
if (sfb_cb [g] [sfb]! = ZERO_HCB) {
cc_gain [idx] [g] [sfb] = cc_sign [idx] [g] [sfb] * cc_scale^gain_element [idx] [g] [sfb];
for (i = 0; i <swb_offset [sfb+1]–swb_offset [sfb];
 dest_spectrum [g] [b] [sfb] [i] + = cc_gain [idx] [g] [sfb] * source_spectrum [g] [b] [sfb] [i].
```

10.3.4 Таблицы

Таблица 61 — Масштабирование для спаренных каналов (cc_scale_table)

Значение "gain_element_scale"	Разрешение амплитуды "cc_scale"	Шаг, дБ
0	2^(1/8)	0,75
1	2^(1/4)	1,50
2	2^(1/2)	3,00
3	2^1	6.00

11 Предсказание

11.1 Описание инструмента

Предсказание используется для улучшенного устранения избыточности, и особенно эффективно в случае более или менее стационарных участков сигнала, которые являются наиболее требовательными с точки зрения необходимой скорости передачи. Предсказание может быть применено к каждому каналу с помощью внутриканального (или моно) предсказывающего устройства, которое использует автокорреляцию спектральных составляющих последовательных фреймов. Поскольку *window_sequence* типа *EIGHT_SHORT_SEQUENCE* указывает на сигнальные изменения, т. е. нестационарность, предсказание используется, только если *window_sequence* имеет типы *ONLY_LONG_SEQUENCE*, *LONG_START_SEQUENCE* или *LONG_STOP_SEQUENCE*. Использование инструмента предсказания является зависимым профилем.

Для каждого канала предсказание применяется к спектральным компонентам, полученным в результате спектрального преобразования в банке фильтров. Для каждого спектрального компонента, вплоть до *PRED_SFB_MAX*, есть один соответствующий предсказатель, приводящий к банку предсказания, где каждый предсказатель использует автокорреляцию спектральных значений последовательных фреймов.

Полная структура кодирования с использованием банка фильтров с высокой спектральной разрешающей способностью подразумевает использование обратных адаптивных предсказателей для достижений высокой эффективности кодирования. В этом случае коэффициенты предсказания вычисляются по предыдущим квантованным спектральным значениям в кодере, так же как и в декодере, и никакая дополнительная служебная информация не требуется для передачи коэффициентов предсказания, в отличие от прямых адаптивных предсказаний. Обратно-адаптивное предсказание второго порядка с решетчатой структурой используется для каждого спектрального компонента так, чтобы каждый предсказатель работал со спектральными значениями двух предыдущих фреймов. Параметры предсказателя адаптируются к текущей сигнальной статистике от фрейма к фрейму, используя алгоритм адаптации на основе *LMS*. Если предсказание активно, на вход квантователя поступает ошибка предсказания вместо исходного спектрального компонента, что приводит к увеличению эффективности кодирования.

Чтобы сохранить требования на минимуме, переменные состояния предсказателя квантуются перед хранением.

11.2 Элементы данных

predictor_data_present 1 бит, указывающий, используется ли предсказание в текущем фрейме (1) или нет (0) (всегда представлено при *ONLY_LONG_SEQUENCE*, *LONG_START_SEQUENCE* и *LONG_STOP_SEQUENCE*, см. 6.3, таблицу 15).

predictor_reset 1 бит, указывающий, происходит ли сброс предсказателя в текущем фрейме (1) или нет (0) (только если флаг *predictor_data_present* установлен, см. 4.3, таблицу 15).

predictor_reset_group_number 5–битное число, определяющее группу, которая будет сброшена в текущем фрейме, если сброс предсказателя активирован (только если флаг *predictor_reset* установлен, см. 4.3, таблицу 15).

prediction_used 1 бит для каждой полосы масштабных коэффициентов (*sfb*), определяющий включено ли предсказание (1) или нет (0) для полосы *sfb*. Если *max_sfb* меньше, чем *PRED_SFB_MAX*, тогда, для *i* больше или равно *max_sfb*, *prediction_used [i]* не передается, и поэтому устанавливается в ноль (0) (только если флаг *predictor_data_present* установлен, см. 4.3, таблицу 15).

Таблица 62 определяет верхний предел полос масштабных коэффициентов, для которых может использоваться предсказание.

Таблица 62 — Верхний частотный предел для предсказания

Частота дискретизации, Гц	*Pred_SFB_MAX*	Количество предсказателей	Максимальная частота предсказания, Гц
96000	33	512	24000,00
88200	33	512	22050,00
64000	38	664	20750,00
48000	40	672	15750,00
44100	40	672	14470,31
32000	40	672	10500,00
24000	41	652	7640,63
22050	41	652	7019,82
16000	37	664	5187,50
12000	37	664	3890,63
11025	37	664	3574,51
8000	34	664	2593,75

Это означает, что на частоте дискретизации 48 кГц предсказание может использоваться в полосах масштабных коэффициентов от 0 до 39. Согласно таблице 46 эти 40 полос масштабных коэффициентов включают линии МДКП 0…671, следовательно, по максимуму имеется 672 предсказателя.

11.3 Процесс декодирования

Для каждого спектрального компонента вплоть до предела, определенного *PRED_SFB_MAX* каждого канала, есть один предсказатель. Предсказанием управляют на основе *single_channel_element* () или *channel_pair_element* (), на основе переданной служебной информации, в два шага: сначала для целого фрейма, а затем для каждой отдельной полосы масштабных коэффициентов (см. 11.3.1). Коэффициенты предсказания для каждого предсказателя вычисляются по предыдущим восстановленным значениям соответствующих спектральных составляющих. Детали необходимой обработки предсказателя описываются в 11.3.2. В начале процесса декодирования инициализируются все предсказатели. Механизм инициализации и сброс предсказателей описываются в 11.3.2.4.

11.3.1 Служебная информация предсказателя

Следующее описание действительно для любого одного *single_channel_element* () или одного *channel_pair_element* () и должно быть применено к каждому такому элементу. Для каждого фрейма служебная информация предсказателя должна быть извлечена из потока битов, чтобы осуществить управление дальнейшей обработкой предсказателя в декодере. В случае *single_channel_element* () управляющая информация допустима для банка предсказателей канала, связанного с этим элементом. В случае

channel_pair_element () есть два варианта. Если *common_window* = 1, есть только один набор управляющей информации, которая допустима для двух банков предсказателей двух каналов, связанных с этим элементом. Если *common_window* = 0, есть два набора управляющей информации для каждого из двух банков предсказателей двух каналов, связанных с этим элементом.

Если window_sequence имеет тип *ONLY_LONG_SEQUENCE*, *LONG_START_SEQUENCE* или *LONG_STOP_SEQUENCE*, бит *predictor_data_present* считывается. Если этот бит не установлен в (0), то предсказание выключается для всего текущего фрейма, и нет дальнейшей служебной информации о предсказании. В этом случае бит *prediction_used* для каждой полосы масштабных коэффициентов, сохраненной в декодере, должен быть обнулен. Если бит *predictor_data_present* установлен в (1), то предсказание используется для текущего фрейма, и бит *predictor_reset* считывается, определяя, применяется ли сброс предсказателя в текущем фрейме (1), или нет (0). Если *predictor_reset* установлен, то считываются следующие 5 битов, соответствующие числу, определяющему группу предсказателей, которые будут сброшены в текущем фрейме (см. также 11.3.2.4). Если *predictor_reset* не установлен, то в потоке битов указанное 5-битное число отсутствует. Затем считываются биты *prediction_used*, которые управляют использованием предсказания в каждой полосе масштабных коэффициентов индивидуально, т. е. если бит установлен для определенной полосы масштабных коэффициентов, предсказание включается для всех спектральных компонентов этой полосы масштабных коэффициентов, и квантованная ошибка предсказания каждого спектрального компонента передается вместо квантованного значения спектрального компонента. Иначе предсказание отключается для этой полосы масштабных коэффициентов, и передаются квантованные значения спектральных компонентов.

11.3.2 Обработка предсказателя

11.3.2.1 Общие сведения

Следующее описание действительно для одного предсказателя и должно быть применено к каждому предсказателю. Используется обратное адаптивное предсказание второго порядка с решетчатой структурой. На рисунке 7 показана соответствующая блок-схема предсказания в декодере. Оценка $x_{est}(n)$ текущего значения спектрального компонента $x(n)$, вычисленная по предыдущим восстановленным значениям $x_{rec}(n-1)$ и $xre_c(n-2)$, сохраняется в элементах регистра структуры предсказателя с помощью коэффициентов предсказания $k_1(n)$ и $k_2(n)$. Эта оценка затем добавляется к квантованной ошибке предсказания $e_q(n)$, восстановленной по переданным данным, которая приводит к восстановленному значению $xrec(n)$ текущего спектрального компонента $x(n)$. На рисунке 8 показана блок-схема этого процесса восстановления для одного предсказателя. Предсказатель состоит из двух основных элементов, которые располагаются каскадом. В каждом элементе части $x_{est}t_{,M}(n)$, $m = 1, 2$ оценки вычисляются согласно

$$x_{est,m}(n) = b \cdot k_m(n) \cdot r_{q,m-1}(n-1),$$

где

$r_{q,0}(n) = ax_{rec}(n)$,

$r_{q,1}(n) = a(r_{q,0}(n-1) - b \cdot k_1(n) \cdot e_{q,0}(n))$

и $e_{q,m}(n) = e_{q,m-1}(n) - x_{est,m}(n)$.

Следовательно, полная оценка:

$x_{est}(n) = x_{est,1}(n) + x_{est,2}(n)$.

Константы a и b, $0 < a$, $b \le 1$ являются коэффициентами затухания, которые применяются в целях стабилизации рекурсивной структуры. Благодаря им возможные колебания из-за ошибок передачи или дрейфа между коэффициентами предсказания из-за численной погрешности в кодере и декодере могут постепенно быть устранены или даже предотвращены.

В случае стационарных сигналов и $c = b = 1$ коэффициент предсказания элемента m вычисляется

$$k_m = \frac{E\left[e_{q,m-1}(n) \cdot r_{q,m-1}(n-1)\right]}{\frac{1}{2} \cdot \left(E\left[e^2_{q,m-1}(n)\right] + E\left[r^2_{q,m-1}(n-1)\right]\right)}, \quad m = 1,2 \text{ и } e_{q,0}(n) = r_{q,0}(n) = x_{rec}(n).$$

Чтобы адаптировать коэффициенты к текущим сигнальным свойствам, математические ожидания в приведенном выше уравнении заменяют на усредненные по времени оценки, измеренные за ограниченный прошлый сигнальный период. Должен быть принят компромисс между хорошей сходимостью и оптимальной настройкой предсказателя в течение периодов сигнала с квазистационарной характеристикой и возможностью быстрой адаптации в случае нестационарности. В этом контексте особенно интересны алгорит-

мы с итеративным улучшением оценок, т. е. от отсчета к отсчету. Здесь используется метод наименьших квадратов (*LMS*) и коэффициенты предсказания вычисляются следующим образом:

$$k_n(n+1) = \frac{COR_m(n)}{VAR_m(n)}$$

с

$$COR_m(n) = a \cdot COR_m(n-1) + r_{q,m-1}(n-1) \cdot e_{q,m-1}(n);$$

$$VAR_m(n) = a \cdot VAR_m(n-1) + 0{,}5 \cdot \left(r^2_{q,m-1}(n-1) + e^2_{q,m-1}(n)\right),$$

где a — время адаптации, постоянная, которая определяет влияние текущего отсчета на оценку математического ожидания, $a = 0{,}90625$.

Оптимальные значения коэффициентов затухания a и b должны быть определены из-за компромисса между высоким усилением предсказания и быстрым временем спада. Выбранные значения $a = b = 0{,}953125$ независимо от того, выключено ли предсказание вообще или только для определенной полосы масштабных коэффициентов, или, все предсказатели вычисляются постоянно, для постоянной адаптации коэффициентов к текущей сигнальной статистике.

Если *window_sequence* имеет типы *ONLY_LONG_SEQUENCE*, *LONG_START_SEQUENCE* и *LONG_STOP_SEQUENCE*, отличается только вычисление восстановленного значения квантованных спектральных компонентов в зависимости от значения бита *prediction_used:*

если бит установлен в (1), то квантованная ошибка предсказания, восстановленная по переданным данным, прибавляется к оценке $x_{est}(n)$ вычисленной предсказателем, образуя восстановленное значение квантованного спектрального компонента, т. е. $x_{rec}(n) = x_{est}(n) + e_q(n)$;

если бит не установлен в (0), то квантованное значение спектрального компонента восстанавливается непосредственно по переданным данным.

В случае коротких блоков, т. е. *window_sequence* имеет тип *EIGHT_SHORT_SEQUENCE*, предсказание всегда отключается, и выполняется сброс всех предсказателей во всех полосах масштабных коэффициентов, который эквивалентен инициализации (см. 11.3.2.4). Для *single_channel_element* () обработка предсказателя для одного фрейма выполняется согласно следующему псевдокоду:

(Предполагается, что восстановленное значение *y_rec* (*c*), которое является либо восстановленной квантованной ошибкой предсказания, либо восстановленным квантованным спектральным коэффициентом, доступно из предыдущей обработки.)

```
if (ONLY_LONG_SEQUENCE || LONG_START_SEQUENCE ||
LONG_STOP_SEQUENCE) {
    for (sfb = 0; sfb <PRED_SFB_MAX; sfb ++) {
        fc = swb_offset_long_window [fs_index] [sfb];
        lc = swb_offset_long_window [fs_index] [sfb+1];
        for (c = fc; c <lc; c ++) {
            x_est [c] = predict();
            if (predictor_data_present && prediction_used [sfb])
                x_rec [c] = x_est [c] + y_rec [c];
            else
                x_rec [c] = y_rec [c];
        }
else {
reset_all_predictors().
```

В случае *channel_pair_element* () с *common_window* = 1 единственное отличие заключается в том, что вычисление *x_est* и *x_rec* во внутренней петле цикла выполняется для обоих каналов, связанных с *channel_pair_element* (). В случае *channel_pair_element ()* с *common_window* = 0 у каждого канала есть предсказание в соответствии с канальной служебной информацией предсказания.

11.3.2.2 Квантование при вычислении предсказания

Для заданного предсказателя должны быть сохранены шесть параметров состояния: r_0, u, COR_1, COR_2, VAR_1 и VAR_2. Эти переменные хранятся как округленные IEEE-числа с плавающей запятой (т. е. 16 старших разрядов слова).

Предсказанное значение x_{est} округляется до 16-разрядного представления с плавающей запятой (т. е. к 7-разрядной мантиссе) до возможности использования в любом вычислении. Точный алгоритм округления показан в псевдо С-функции *flt_round_inf* (). Следует учесть, что из соображений сложности реализации вместо округления к ближайшему значению используется округление в сторону бесконечности.

Выражения (b/VAR_1) и (b/VAR_2) округляются до 16-разрядного представления с плавающей запятой (т. е. к 7-разрядной мантиссе), что позволяет вычислить эти отношения по двум небольшим таблицам. С-код для генерации таких таблиц показан в псевдо С-функции *make_inv_tables* ().

Все промежуточные результаты в каждом вычислении с плавающей запятой в алгоритме предсказания будут представлены в округленной форме с использованием фиксированной точности с плавающей запятой.

Вычислительный модуль IEEE с плавающей запятой, используемый при выполнении всех арифметических действий в инструменте предсказания, включает следующие опции:

Округление к ближайшему значению; округление к значению с младшим значащим битом, равным 0.

Исключение переполнения — значения, величина которых больше, чем самое большое возможное значение, будут установлены в бесконечность.

Исключение потери значимости — постепенная потеря значимости (денормализованные числа) будет поддерживаться; будут обнулены значения, величина которых меньше, чем самое маленькое представимое значение.

11.3.2.3 Быстрый алгоритм для округления

```
*/
static void
flt_round_inf
(float *pf).
{
int flg;
unsigned long tmp, tmp1, tmp2;
tmp = *(unsigned long*) pf;
flg = tmp & (unsigned long) 0x00008000;
tmp &= (unsigned long) 0xffff0000;
tmp1 = tmp;
/* округление 1/2 lsb к бесконечности */
if (flg) {
tmp &= (unsigned long) 0xff 800000;     /* извлечь экспоненту и знак */
tmp | = (unsigned long) 0x00010000;     /* вставить 1 lsb */
tmp2 = tmp;  /* добавить 1 lsb и игнорировать один */
tmp &= (unsigned) 0xff800000;    /* извлечь экспоненту и знак */
*pf = * (float *), &tmp1+ * (float *) &tmp2 – * (float *) &tmp;
/* вычесть один */} {
*pf = * (float *),&tmp;}
```

11.3.2.4 Генерирование округленного *b/Var*

```
static foat mnt_table [128];
static float exp_table [256];
/* функция flt_round_even () работает только для аргументов в диапазоне
*         1.0 <  * pf  <2.0 – 2 ^ –24
*/static void flt_round_even (float *pf),
{
int exp, a;
 float tmp;
frexp ((double) *pf, &exp);
 tmp = *pf * (1 <<(8–exp)); = (int) tmp;
if ((tmp–a)> = 0.5) + +;
 if ((tmp–a) == 0.5) a&= –2;
 *pf = (float) / (1 <<(8–exp));}
```

```
static void make_inv_tables (void)
{
int i;
unsigned long tmp1, tmp;
float *pf = (float *) &tmp1;
float ftmp;
*pf = 1.0;
for (i=0; i <128; i ++) {
tmp = tmp1 + (i <<16);/* float 1.m, 7 msb только */
ftmp = b / * (float *), &tmp;
flt_round_even (&ftmp);/* округление к 16 битам */
mnt_table [i] = ftmp;
для (i=0; i <256;
tmp = (i <<23);
/* float 1.0 * 2^exp */, if (* (float *) &tmp> 1.0) {
ftmp = 1.0 / * (float *), &tmp;} else {
ftmp = 0;
}
exp_table [i] = ftmp;
}
```

11.3.3 Сброс предсказателя

Инициализация предсказателя означает, что параметры состояния предсказателя устанавливаются следующим образом: $r_0 = r_1 = 0$, $COR_1 = COR_2 = 0$, $VAR_1 = VAR_2 = 1$. В начале процесса декодирования все предсказатели инициализируются.

Циклический механизм сброса применяется кодером и сообщается декодеру, в котором все предсказатели инициализируются снова в определенном временном интервале способом чередования. С одной стороны это увеличивает устойчивость предсказателя, повтороно синхронизируя предсказатели кодера и декодера, и с другой стороны это позволяет задавать определенные точки входа в потоке битов.

Весь набор предсказателей подразделяется на 30 групп сброса согласно таблице 63.

Таблица 63 — Группы сброса предсказателей

Номер группы сброса	Предсказатели группы сброса
1	P0, P30, P60, P90...
2	P1, P31, P61, P91...
3	P2, P32, P62, P92...
...	
30	P29, P59, P89, P119...

Должен ли сброс быть применен в текущем фрейме, определяется битом *predictor_reset*. Если этот бит установлен, номер группы сброса предсказателей, которая будет сброшена в текущем фрейме, определяется из *predictor_reset_group_number*. Все предсказатели, принадлежащие этой группе сброса, инициализируются как описано выше. Эта инициализация должна быть сделана после того, как была выполнена обычная обработка предсказателей для текущего фрейма. Следует учесть, что у *predictor_reset_group_number* не может быть значений 0 или 31.

Типичный цикл сброса начинается с группы сброса номер 1, и номер группы сброса увеличивается на 1, пока не достигнет 30, затем все снова начинается с 1. Однако может произойти разрыв в нумерации групп сброса, например, из-за переключения между программами (потоками битов) или вырезания и вставки. В этом случае есть три возможности для работы декодера:

- проигнорировать разрыв и продолжить нормальную обработку. Это может привести к короткому слышимому искажению из-за несоответствия (дрейф) между предсказателем в кодере и декодере. После одного полного цикла сброса (группа сброса n, n +1..., 30, 1, 2..., n –1) предсказатели повторно синхронизируются. Кроме того, возможное искажение постепенно уменьшается из-за коэффициентов затухания a и b;

- обнаружить разрыв, продолжить нормальную обработку, но отключить выходной сигнал, пока не будет выполнен один полный цикл сброса, и предсказатели не будут повторно синхронизированы;
- сбросить все предсказатели.

Каждая группа предсказателей должна быть сброшена после максимального "активного" периода длиной 240 фреймов. Сброс 30 групп сброса предсказателей может быть выполнен либо постепенно, либо за один раз, либо любым другим путем, пока максимальный период сброса 240 "активных" фреймов не нарушен. Следует учесть, что "активный" период в 240 фреймов может длиться намного дольше, чем 240 фреймов, так как фреймы с активным предсказанием могут чередоваться с произвольным числом фреймов без предсказания. Далее, группы предсказателей могут быть активны независимо друг от друга, таким образом, требуется отдельный учет "активности" каждой группы сброса предсказателей.

В случае *single_channel_element ()* или *channel_pair_element ()* с *common_window* = 0 сброс должен быть применен к банку(ам) предсказателей канала(ов), связанных с этим элементом. В случае *channel_pair_element ()* с *common_window* = 1 сброс должен быть применен к двум банкам предсказателей двух каналов, связанных с этим элементом.

В случае короткого блока (*window_sequence* типа *EIGHT_SHORT_SEQUENCE*) должны быть сброшены все предсказатели во всех полосах масштабных коэффициентов.

11.4 Схемы

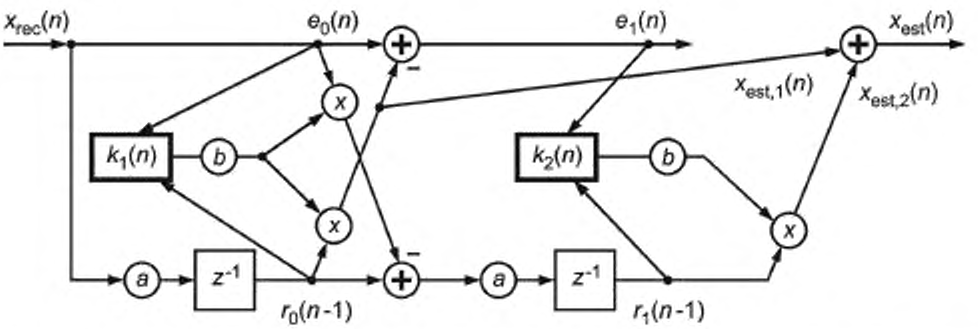
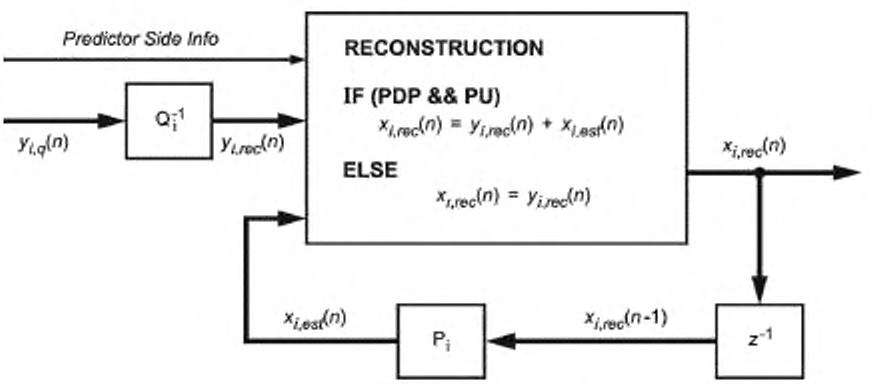


Рисунок 7 — Блок-схема внутриканального предсказателя для одного спектрального компонента в декодере. (Стрелки указывают на поток сигналов для адаптации коэффициентов предсказания).



Legend: P Predictor; Q_i^{-1} Inverse quantizer; **PDP** predictor_data_present; **PU** predictor_used

Рисунок 8 — Блок-схема модуля предсказания декодера для одного спектрального компонента

12 Временное формирование шума (*TNS*)

12.1 Описание инструмента

Временное формирование шума используется для управления временной формой шумов квантования в пределах отдельного окна преобразования. Это выполняется применением процессов фильтрации к частям спектральных данных каждого канала.

Следует учесть, что этот инструмент включает параметры, зависимые от определенного профиля (см. 5.1).

12.2 Элементы данных

n_filt [w]	Число фильтров формирования шума, используемых для окна *w* (см. 4.3, таблицу 19).
coef_res [w]	Маркер, определяющий разрешение переданных коэффициентов фильтра для окна *w*, соответствующий разрешению 3 бит (0) и 4 битам (1) (см. 4.3, таблицу 19).
length [w] [filt]	Длина области, к которой фильтр применяется в окне *w* (в полосах масштабных коэффициентов) (см. 4.3, таблицу 19).
order [w] [filt]	Порядок одного фильтра формирования шума, применяемого к окну *w* (см. 4.3, таблицу 19).
direction [w] [filt]	1 бит, указывающий, применяется ли фильтр в восходящем (0) или нисходящем (1) направлении (см. 4.3, таблицу 19).
coef_compress [w] [filt]	1 бит, указывающий, исключаются ли старшие значащие биты коэффициентов фильтра формирования шума *filt* в окне *w*, исключаются (1) или не исключаются (0) (см. 4.3, таблицу 19).
coef [w] [filt] [i]	Коэффициенты одного фильтра формирования шума, относящиеся к окну *w* (см. 4.3, таблицу 19).
spec [w] [k]	Массив, содержащий спектр для окна *w* обрабатываемого канала.

В зависимости от *window_sequence* размер следующих полей потока битов переключается для каждого окна преобразования согласно его размеру:

Имя	Окно со 128 спектральными линиями	Размер другого окна
'n_filt'	1	2
'length'	4	6
'order'	3	5

12.3 Процесс декодирования

Процесс декодирования для временного формирования шума выполняется отдельно для каждого окна текущего фрейма путем фильтрации выбранных областей спектральных коэффициентов (см. функцию *tns_decode_frame*).

Число фильтров формирования шума, применяемых к каждому окну, определяется "*n_filt*". Целевой диапазон спектральных коэффициентов выражен в полосах масштабных коэффициентов с нисходящим счетом от верхней полосы (или нижней полосы предыдущей полосы формирования шума).

Сначала происходит декодирование переданных коэффициентов фильтра, т. е. преобразование в числа со знаком, обратное квантование, преобразование в коэффициенты *LPC*, как описано в функции *tns_decode_coef* ().

Затем фильтры применяются к целевым областям частот спектральных коэффициентов канала (см. функцию *tns_ar_filter* ()). Маркер "*direction*" используется для определения направления, в котором фильтр движется относительно коэффициентов (0 = вверх, 1 = вниз).

Постоянная *TNS_MAX_BANDS* определяет максимальное количество полос масштабных коэффициентов, к которым применяется временное формирование шума. Максимально возможный порядок фильтра определяется постоянной *TNS_MAX_ORDER*. Обе константы являются параметрами, зависимыми от профиля.

Процесс декодирования для одного канала может быть описан следующим псевдокодом:

```
/* Декодирование TNS для одного канала и фрейма */
tns_decode_frame ()
{
for (w = 0; w <num_windows; w ++) {
bottom = num_swb;
for (f = 0; f <n_filt [w]; f ++) {
top = bottom;
bottom = max(top –length [w] [f], 0);
tns_order = min (order [w] [f], TNS_MAX_ORDER);
if (! tns_order),continue;
tns_decode_coef (tns_order, coef_res [w] +3, coef_compress [w] [f], coef [w] [f], lpc []);
start = swb_offset [min (bottom, TNS_MAX_BANDS, max_sfb)];
end = swb_offset [min (top, TNS_MAX_BANDS, max_sfb)];
if ((sizep =end – start) <= 0) continue;
if (direction [w] [f]) {
inc = –1;
start = end – 1;
} else {
inc = 1;
}
tns_ar_filter (&spec [w] [start], size, inc, lpc [], tns_order);
}
```

Следует учесть, что этот псевдокод использует С-интерпретацию массивов и векторов, т. е. если *coef[w] [filt] [i]* описывает коэффициенты для всех окон и фильтров, *coef [w] [filt]* является указателем на коэффициенты одного определенного окна и фильтра. Кроме того, коэффициент идентификатора используется в качестве формального параметра в функции *tns_decode_coef ()*.

```
/* Декодирование переданных коэффициентов для одного фильтра TNS */
tns_decode_coef (order, coef_res_bits, coef_compress, coef[],a[])
{
/* Некоторые внутренние таблицы *
/sgn_mask [] = {0x2, 0x4, 0x8};
neg_mask [] = {~0x3, ~0x7, ~0xf};
/* размер, используемый для передачи */
coef_res2 = coef_res_bits – coef_compress;
s_mask = sgn_mask [coef_res2 – 2]; /* маска для бита знака */
n_mask = neg_mask [coef_res2 – 2];
/*маска для того, чтобы дополнить отрицательные значения */
/* Преобразование в целое число со знаком */
for(i = 0; i <order; i ++)
tmp [i] = (coef [i] & s_mask)? (coef [i] | n_mask):coef [i];
/* Обратное квантование */
iqfac = ((1<<(coef_res_bits–1)) – 0.5) / (p/2.0);
iqfac_m = ((1 <<(coef_res_bits–1)) + 0.5) / (p/2.0);
for (i = 0; i <order; i ++) {
tmp2 [i] = sin (tmp [i] / ((tmp [i]> = 0)? iqfac: iqfac_m));}
/* Преобразование в коэффициенты LPC */
a[0] = 1;
for (m = 1; m <= order; m ++) {
for (i = 1; i <m; i ++) {
b [i] = [i] + tmp2 [m–1] * [m–i];
for (i = 1; i <m, i ++) {
a[m] = tmp2 [m–1];
}
tns_ar_filter (spectrum [ ], size, inc, lpc [ ], order)
{
```

Простой фильтр порядка "order", определенный согласно

*y (n) = x (n) – lpc [1] *y (n–1)–... – lpc [order] *y (n–order)*

Параметры состояния фильтра инициализируются в нуль каждый раз.

Выходные данные заменяют входные данные.

Входной вектор "*size*" отсчетов обрабатывается, и индекс увеличивается на "*inc*" до следующего отсчета данных.

}

13 Банк фильтров и переключение окон

13.1 Описание инструмента

Частотно-временное представление сигнала отображается во временное пространство путем подачи его коэффициентов в банк фильтров. Этот модуль состоит из ОМДКП и функции окна со сложением. Чтобы адаптировать частотно/временную разрешающую способность банка фильтров к характеристикам входного сигнала, применяется инструмент переключения окон. *N* соответствует длине окна, *N* является функцией **window_sequence**, см. 6.3.3. Для каждого канала N/2 частотно-временных значений $Xi_{,k}$ преобразовываются в *N* значений времени $xi_{,n}$ с помощью ОМДКП. После применения оконной функции для каждого канала первая половина последовательности $zi_{,n}$ складывается со второй половиной последовательности $z_{(i-1),n}$ из предыдущего блока для восстановления выходных отсчетов каждого канала $outi_{,n}$.

13.2 Определения

Элементы синтаксиса банка фильтров определяются в потоке необработанных данных для *single_channel_element ()* (см. 4.3, таблицу 13), *channel_pair_element ()* (см. 4.3, таблицу 14) и *coupling_channel* (см. 4.3, таблицу 22). Они состоят из управляющей информации *window_sequence* и *window_shape*.

Элементы данных:

window_sequence — 2 бита, указывающие, какая последовательность окон (т. е. размер блока) используется (см. 4.3, таблицу 15).

window_shape — 1 бит, указывающий, какая оконная функция выбрана (см. 4.3, таблица 15).

В таблице 44 показаны четыре window_sequences (ONLY_LONG_SEQUENCE, LONG_START_SEQUENCE, EIGHT_SHORT_SEQUENCE, LONG_STOP_SEQUENCE).

13.3 Процесс декодирования

13.3.1 ОМДКП

Аналитическое выражение для ОМДКП

$$x_{i,n} = \frac{2}{N}\sum_{k=0}^{\frac{N}{2}-1} spec[i][k]\cos\left(\frac{2\pi}{N}\left(n + n_0\right)\left(k + \frac{1}{2}\right)\right) \quad for\ 0 \le n < N,$$

где *n* — индекс отсчета;

i — индекс окна;

к — индекс спектрального коэффициента;

N — длина окна, основанная на значении *window_sequence* *n* = (N/2 + 1)/2.

Длина окна синтеза *N* для обратного преобразования является функцией элемента синтаксиса *window_sequence* и определяется следующим образом:

$$N = \begin{cases} 2048, & \text{if ONLY\_LONG\_SEQUENCE (0x0)} \\ 2048, & \text{if LONG\_START\_SEQUENCE (0x1)} \\ 256, & \text{if EIGHT\_SHORT\_SEQUENCE (0x2), (8 times)} \\ 2048, & \text{if LONG\_STOP\_SEQUENCE (0x3)} \end{cases}$$

Значимые блочные переходы:

$$\text{from ONLY\_LONG\_SEQUENCE} \quad \text{to} \quad \begin{cases} \text{ONLY\_LONG\_SEQUENCE} \\ \text{LONG\_START\_SEQUENCE} \end{cases}$$

from LONG_START_SEQUENCE to {EIGHT_SHORT_SEQUENCE, LONG_STOP_SEQUENCE}

from LONG_STOP_SEQUENCE to {ONLY_LONG_SEQUENCE, LONG_START_SEQUENCE}

from EIGHT_SHORT_SEQUENCE to {EIGHT_SHORT_SEQUENCE, LONG_STOP_SEQUENCE}

В дополнение к значимым блочным переходам следующие переходы возможны:

from ONLY_LONG_SEQUENCE to {EIGHT_SHORT_SEQUENCE, LONG_STOP_SEQUENCE}

from LONG_START_SEQUENCE to {ONLY_LONG_SEQUENCE, LONG_START_SEQUENCE}

from LONG_STOP_SEQUENCE to {EIGHT_SHORT_SEQUENCE, LONG_STOP_SEQUENCE}

from EIGHT_SHORT_SEQUENCE to {ONLY_LONG_SEQUENCE, LONG_START_SEQUENCE}

Все это обеспечивает плавный переход от одного блока к следующему.

13.3.2 Взвешивание в окне и переключение окон

В зависимости от *window_sequence* и элемента *window_shape* используются различные окна. Комбинация из половин окон, описанных ниже, обеспечивает любые возможные window_sequences.

Для *window_shape* == 1 коэффициенты окна берутся из оконной функции Кайзера — Бесселя (KBD):

$$W_{KBD\_LEFT,N}(n) = \sqrt{\frac{\sum_{p=0}^{n}[W'(p,\alpha)]}{\sum_{p=0}^{N/2}[W'(p,\alpha)]}} \quad \text{for } 0 \le n < \frac{N}{2};$$

$$W_{KBD\_RIGHT,N}(n) = \sqrt{\frac{\sum_{p=0}^{N-n-1}[W'(p,\alpha)]}{\sum_{p=0}^{N/2}[W'(p,\alpha)]}} \quad \text{for } \frac{N}{2} \le n < N,$$

где:

W' (ядро оконной функции Кайзера–Бесселя) определяется следующим образом:

$$W'(n,\alpha) = \frac{I_0\left[\pi\alpha\sqrt{1{,}0-\left(\frac{n-N/4}{N/4}\right)^2}\right]}{I_0[\pi\alpha]} \quad \text{for } 0 \le n \le \frac{N}{2};$$

$$I_0[x] = \sum_{k=0}^{\infty}\left[\frac{\left(\frac{x}{2}\right)^k}{k!}\right]^2,$$

$$\alpha = \text{kernel window alpha factor}, \alpha = \begin{cases} 4 \text{ for } N = 2048 \\ 6 \text{ for } N = 256 \end{cases}.$$

Иначе для *window_shape* == 0, используется синусное окно:

$$W_{SIN\_LEFT,N}(n) = \sin\left(\frac{\pi}{N}\left(n + \frac{1}{2}\right)\right) \text{ for } 0 \le n < \frac{N}{2},$$

$$W_{SIN\_RIGHT,N}(n) = \sin\left(\frac{\pi}{N}\left(n + \frac{1}{2}\right)\right) \text{ for } \frac{N}{2} \le n < N.$$

Длина окна *N* может быть 2048 или 256 для KBD и синусного окна. Процедура получения возможных последовательностей окон объясняется в частях а) — г) этого пункта. У всех четырех *window_sequences*, описанных ниже, общая длина равна 2048 отсчета.

Для всех видов *window_sequences window_shape* левой половины первого окна определяется формой окна предыдущего блока. Это выражается следующей формулой:

$$W_{LEFT,N}(n) = \begin{cases} W_{KBD\_LEFT,N}(n), \text{if} & window\_shape\_previous\_block = 1 \\ W_{SIN\_LEFT,N}(n), \text{if} & window\_shape\_previous\_block = 0 \end{cases},$$

где:

window_shape_previous_block: *window_shape* предыдущего (*i*–1) блока.

Для первого декодируемого блока потока битов *window_shape* левой и правой половины окна идентичны.

а) *ONLY_LONG_SEQUENCE*:

window_sequence == *ONLY_LONG_SEQUENCE* эквивалентно одному *LONG_WINDOW* (см. таблицу 44) с общей длиной окна 2048.

Для *window_shape* == 1, окно для *ONLY_LONG_SEQUENCE* задается следующим образом:

$$W(n) = \begin{cases} W_{LEFT,2048}(n), & \text{for } 0 \le n < 1024 \\ W_{KBD\_RIGHT,2048}(n), & \text{for } 1024 \le n < 2048 \end{cases}.$$

Если *window_shape* == 0, окно для *Only_Long_Sequence* может быть описано следующим образом:

$$W(n) = \begin{cases} W_{LEFT,2048}(n), & \text{for } 0 \le n < 1024 \\ W_{SIN\_RIGHT,2048}(n), & \text{for } 1024 \le n < 2048 \end{cases}.$$

После работы с окнами значения во времени могут быть выражены как:

$$z_{i,n} = w(n) \cdot x_{i,n};$$

б) *LONG_START_SEQUENCE*:

LONG_START_SEQUENCE необходима для получения корректного перекрытия и сложения для блочного перехода от *ONLY_LONG_SEQUENCE* к *EIGHT_SHORT_SEQUENCE*.

Если *window_shape* == 1, окно для *LONG_START_SEQUENCE* задается следующим образом:

$$W(n) = \begin{cases} W_{LEFT,2048}(n), & \text{for } 0 \le n < 1024 \\ 1.0, & \text{for } 1024 \le n < 1472 \\ W_{KBD\_RIGHT,256}(n + 128 - 1472), & \text{for } 1472 \le n < 1600 \\ 0.0, & \text{for } 1600 \le n < 2048 \end{cases}.$$

Если *window_shape* == 0, окно для *LONG_START_SEQUENCE*:

$$W(n) = \begin{cases} W_{LEFT,2048}(n), & \text{for } 0 \le n < 1024 \\ 1.0, & \text{for } 1024 \le n < 1472 \\ W_{SIN\_RIGHT,256}(n + 128 - 1472), & \text{for } 1472 \le n < 1600 \\ 0.0, & \text{for } 1600 \le n < 2048 \end{cases}$$

Взвешенные в окне временные значения могут быть вычислены по формуле из а).

в) *EIGHT_SHORT*

window_sequence == *EIGHT_SHORT* включает в себя восемь перекрытых и сложенных *SHORT_WINDOW* (см. таблицу 44) длиной 256 каждое. Общая длина *window_sequence* вместе со смещением и нулями равна 2048. Каждый из восьми коротких блоков в начале взвешивается в отдельном окне.

Номер короткого блока индексируется переменной $j = 0, \ldots, 7$.

window_shape предыдущего блока влияет только на первый из восьми коротких блоков ($W_0(n)$). Если *window_shape* == 1, оконные функции даны следующим образом:

$$W_0(n) = \begin{cases} W_{LEFT,256}(n), & \text{for } 0 \le n < 128 \\ W_{KBD\_RIGHT,256}(n), & \text{for } 128 \le n < 256 \end{cases}$$

$$W_{1-7}(n) = \begin{cases} W_{KBD\_LEFT,256}(n), & \text{for } 0 \le n < 128 \\ W_{KBD\_RIGHT,256}(n), & \text{for } 128 \le n < 256 \end{cases}$$

Иначе, если *window_shape* == 0, оконные функции могут быть описаны как:

$$W_0(n) = \begin{cases} W_{LEFT,256}(n), & \text{for } 0 \le n < 128 \\ W_{SIN\_RIGHT,256}(n), & \text{for } 128 \le n < 256 \end{cases}$$

$$W_{1-7}(n) = \begin{cases} W_{SIN\_LEFT,256}(n), & \text{for } 0 \le n < 128 \\ W_{SIN\_RIGHT,256}(n), & \text{for } 128 \le n < 256 \end{cases}$$

Перекрытие и сложение между *EIGHT_SHORT window_sequence*, приводящее к взвешенным временным значениям, описывается следующим образом:

$$z_{i,n} = \begin{cases} 0, & \text{for } 0 \le n < 448 \\ x_{j,n-448} \cdot W_0(n - 448), & \text{for } 448 \le n < 576 \\ x_{j,n-448} \cdot W_0(n - 448) + x_{j,n-576} \cdot W_1(n - 576), & \text{for } 576 \le n < 704 \\ x_{j,n-576} \cdot W_1(n - 576) + x_{j,n-704} \cdot W_2(n - 704), & \text{for } 704 \le n < 832 \\ x_{j,n-704} \quad W_2(n - 704) + x_{j,n-832} \quad W_3(n - 832), & \text{for } 832 \le n < 960 \\ x_{j,n-832} \cdot W_3(n - 832) + x_{j,n-960} \cdot W_4(n - 960), & \text{for } 960 \le n < 1088 \\ x_{j,n-960} \cdot W_4(n - 960) + x_{j,n-1088} \cdot W_5(n - 1088), & \text{for } 1088 \le n < 1216 \\ x_{j,n-1088} \cdot W_5(n - 1088) + x_{j,n-1216} \cdot W_6(n - 1216), & \text{for } 1216 \le n < 1344 \\ x_{j,n-1216} \cdot W_6(n - 1216) + x_{j,n-1344} \cdot W_7(n - 1344), & \text{for } 1344 \le n < 1472 \\ x_{j,n-1344} \cdot W_7(n - 1344), & \text{for } 1472 \le n < 1600 \\ 0, & \text{for } 1600 \le n < 2048 \end{cases}$$

г) *LONG_STOP_SEQUENCE*

Эта window_sequence необходима для переключения от *IGHT_SHORT_SEQUENCE* назад к *ONLY_LONG_SEQUENCE.*

Если *window_shape* == 1, окно для *LONG_STOP_SEQUENCE* задается следующим образом:

$$W(n) = \begin{cases} 0.0, & \text{for } 0 \le n < 448 \\ W_{LEFT,256}(n-448), & \text{for } 448 \le n < 576 \\ 1.0, & \text{for } 576 \le n < 1024 \\ W_{KBD\_RIGHT,2048}(n), & \text{for } 1024 \le n < 2048 \end{cases}$$

Если *window_shape* == 0, окно для *LONG_START_SEQUENCE* определяется как:

$$W(n) = \begin{cases} 0.0, & \text{for } 0 \le n < 448 \\ W_{LEFT,256}(n-448), & \text{for } 448 \le n < 576 \\ 1.0, & \text{for } 576 \le n < 1024 \\ W_{SIN\_RIGHT,2048}(n), & \text{for } 1024 \le n < 2048 \end{cases}$$

13.3.3 Перекрытие и сложение с предыдущей последовательностью окон

Помимо перекрытия и сложения в пределах *EIGHT_SHORT window_sequence* первая (левая) половина каждого *window_sequence* перекрывается и складывается со второй (правой) половиной предыдущего *window_sequence*, образуя окончательные временные значения *out*$_i$. Математическое выражение для этой операции дано ниже. Это допустимо для всех четырех возможных *window_sequences*

$$out_{i,n} = z_{i,n} + z_{i-1,n+\frac{N}{2}}; \quad \text{for } 0 \le n < \frac{N}{2},\ N = 2048.$$

14 Управление усилением

14.1 Описание инструмента

Инструмент управления усилением состоит из нескольких компенсаторов усиления, этапов перекрытия/сложения и *IPQF* (обратной полифазного квадратурной фильтрации) этапа. Этот инструмент получает неперекрытые последовательности сигналов после этапов ОМДКП, *window_sequence* и *gain_control_data* и затем воспроизводит выходные ИКМ данные. Блок-схема для инструмента управления усилением показана на рисунке 9.

Из-за характеристик *PQF* банка фильтров порядок следования коэффициентов МДКП в каждой четной полосе *PQF* должен быть реверсирован. Это выполняется путем реверсирования порядка следования спектральных коэффициентов МДКП, т. е. заменой местами более высоких по частоте коэффициентов МДКП на более низкие по частоте коэффициенты МДКП.

Если инструмент управления усиления используется, конфигурация инструмента банка фильтров изменяется следующим образом. В случае *EIGHT_SHORT_SEQUENCE window_sequence* число коэффициентов для ОМДКП — 32, вместо 128, и выполняются восемь ОМДКП. В случае других значений *window_sequence* число коэффициентов для ОМДКП — 256 вместо 1024, и выполняется одно ОМДКП. Во всех случаях на выходе инструмента банка фильтров в общей сложности 2048 неперекрытых значений на фрейм. Эти значения поступают в инструмент управления усилением как $U_{W,B}(j)$, определенный в 14.3.3.

IPQF комбинирует четыре равных полосы частот и производит декодированный выходной сигнал во времени. Компоненты элайзинга, возникшие из *PQF* в кодере, отменяются в *IPQF*.

Значения усиления для каждой полосы можно регулировать независимо за исключением самой низкой полосы частот. Размер шага регулирования усиления равен 2^*n*, где *n* — целое число.

На выходе инструмента управления усилением образуется последовательность временного сигнала *AS (n)*, определенная в 14.3.4.

14.2 Определения

14.2.1 Элементы данных

adjust_num — 3-битное поле, указывающее на количество изменений коэффициента усиления для каждой полосы *IPQF*. Максимальное количество изменений коэффициента усиления равно семи (см. 4.3, таблицу 27).

max_band — 2-битное поле, указывающее на число полос *IPQF*, коэффициент усиления которых подлежит управлению. Смысл этого значения раскрыт ниже (см. 4.3, таблицу 27):
0: нет полос с управлением усиления.
1: управляется усиление 2-ой полосы *IPQF*.
2: управляется усиление 2-ой и 3-ей полос *IPQF*.
3: управляется усиление 2-ой, 3-ей и 4-ой полос *IPQF*.

alevcode — 4-битное поле, указывающее на значение коэффициента усиления для одного изменения коэффициента усиления (см. 4.3, таблицу 27).

aloccode — 2-, 4-, или 5-битное поле, указывающее позицию для одного изменения коэффициента усиления. Продолжительность этих данных изменяется в зависимости от последовательности окон (см. 4.3, таблицу 27).

14.2.2 Элементы справки

gain control data — Служебная информация, указывающая на значения усиления и позиции, используется для изменения усиления.

Полоса IPQF — Каждая полоса IPQF.

14.3 Процесс декодирования

Следующие процессы требуются для декодирования:
- Декодирование данных управления усилением.
- Установка функции управления усилением.
- Взвешивание регулирования усиления и наложение.
- Фильтр синтеза.

14.3.1 Декодирование данных управления усилением

Данные управления усилением восстанавливаются следующим образом.

(1)

$NAD_{W,B}$ = adjust_num[*B*][*W*],

(2)

$ALOC_{W,B}(m) = AdjLoc(\text{aloccode}[B][W][m-1]), 1 \le m \le NAD_{W,B}$,

$ALEV_{W,B}(m) = 2^{AdjLev[\text{alevcode}[B][W][m-1]]}, 1 \le m \le NAD_{W,B}$,

(3)

$ALOC_{W,B}(0) = 0$

$$ALEV_{W,B}(0) = \begin{cases} 1, if\ NAD_{W,B} == 0 \\ ALEV_{W,B}(1), otherwise \end{cases},$$

(4)

$$ALOC_{W,B}(NAD_{W,B}+1) = \begin{cases} 256, W == 0\ if \text{ONLY\_LONG\_SEQUENCE} \\ \left.\begin{matrix} 112, W == 0 \\ 32, W == 1 \end{matrix}\right\} if \text{LONG\_START\_SEQUENCE} \\ 32, 0 \le W \le 7\ if \text{EIGHT\_SHORT\_SEQUENCE} \\ \left.\begin{matrix} 112, W == 0 \\ 256, W == 1 \end{matrix}\right\} if \text{LONG\_STOP\_SEQUENCE} \end{cases}$$

$ALEV_{W,B}(NAD_{W,B}+1) = 1$,

где

$NAD_{W,B}$ — номер информации об управлении усилением, целое число;

$ALOC_{W,B}$ — расположение управления усилением, целое число;

$ALEV_{W,B}$ — уровень управления усилением, целочисленное действительное число;

B — *ID* полосы, целое число от 1 до 3;

W — *ID* окна, целое число от 0 до 7;

m — целое число;

aloccode [B] [W] [m] должен быть установлен так, чтобы $\{ALOC_{W,B}(m)\}$ удовлетворял следующим условиям:

$ALOC_{W,B}(m_1) < ALOC_{W,B}(m_2), 1 \le m_1 < m_2 \le NAD_{W,B} + 1,$

AdjLoc () определяется из таблицы 64. *AdjLev()* определяется из таблицы 65.

14.3.2 Установка функции управления усилением

Функция управления усилением получается следующим образом:

(1)

$$M_{W,B,j} = \max\{m: ALOC_{W,B}(m) \le j\},$$

$$\left.\begin{array}{l} 0 \le j \le 255, W = 0 \\ 0 \le j \le 111, W = 0 \\ 0 \le j \le 31, W = 1 \end{array}\right\} \text{if LONG\_START\_SEQUENCE},$$

$$0 \le j \le 31, 0 \le W \le 7 \text{ if EIGHT\_SHORT\_SEQUENCE}$$

$$\left.\begin{array}{l} 0 \le j \le 111, W = 0 \\ 0 \le j \le 255, W = 1 \end{array}\right\} \text{if LONG\_STOP\_SEQUENCE}$$

(2)

$$FMD_{W,B}(j) = \begin{cases} Inter\begin{pmatrix} ALEV_{W,B}(M_{W,B,j}), \\ ALEV_{W,B}(M_{W,B,j}+1), \\ j - ALOC_{W,B}(M_{W,B,j}) \end{pmatrix} \\ \text{if } ALOC_{W,B}(M_{W,B,j}) \le j \le ALOC_{W,B}(M_{W,B,j}) + 7 \\ ALEV_{W,B}(M_{W,B,j}+1), \text{otherwise} \end{cases},$$

(3)

if ONLY_LONG_SEQUENCE

$$GMF_{0,B}(j) = \begin{cases} ALEV_{0,B}(0) \times PFMD_B(j), 0 \le j \le 255 \\ FMD_{0,B}(j-256), 256 \le j \le 511 \end{cases},$$

$$PFMD_B(j) = FMD_{0,B}(j),\ 0 \le j \le 255$$

if LONG_START_SEQUENCE

$$GMF_{0,B}(j) = \begin{cases} ALEV_{0,B}(0) \times ALEV_{1,B}(0) \times PFMD_B(j), 0 \le j \le 255 \\ ALEV_{1,B}(0) \times FMD_{0,B}(j-256), 256 \le j \le 367 \\ FMD_{1,B}(j-368), 368 \le j \le 399 \\ 1, 400 \le j \le 511 \end{cases},$$

$$PFMD_B(j) = FMD_{1,B}(j),\ 0 \le j \le 31,$$

если *EIGHT_SHORT_SEQUENCE*

$$GMF_{W,B}(j) = \begin{cases} ALEV_{W,B}(0) \times PFMD_B(j), W = 0, 0 \le j \le 31 \\ ALEV_{W,B}(0) \times FMD_{W-1,B}(j), 1 \le W \le 7, 0 \le j \le 31, \\ FMD_{W,B}(j-32), 0 \le W \le 7, 32 \le j \le 63 \end{cases}$$

$PFMD_B(j) = FMD_{7,B}(j), 0 \le j \le 31,$

if LONG_STOP_SEQUENCE

$$GMF_{0,B}(j) = \begin{cases} 1,0 \le j \le 111 \\ ALEV_{0,B}(0) \times ALEV_{1,B}(0) \times PFMD_B(j-112), 112 \le j \le 143 \\ ALEV_{1,B}(0) \times FMD_{0,B}(j-144), 144 \le j \le 255 \\ FMD_{1,B}(j-256), 256 \le j \le 511 \end{cases}$$

$PFMD_B(j) = FMD_{1,B}(j), 0 \le j \le 255,$

(4)

$$AD_{W,B}(j) = \frac{1}{GMF_{W,B}(j)},$$

$0 \le j \le 511, W = 0$ *if* ONLY_LONG_SEQUENCE

$0 \le j \le 511, W = 0$ *if* LONG_START_SEQUENCE

$0 \le j \le 63,0 \le W \le 7$ *if* EIGHT_SHORT_SEQUENCE,

$0 \le j \le 511, W = 0$ *if* LONG_STOP_SEQUENCE

где

$FMD_{W,B}(j)$	Функция модификации фрагмента, действительное число
$PFMD_B(j)$	Функция модификации фрагмента предыдущего фрейма, вещественное число
$GMF_{W,B}(j)$	Функция модификации усиления, вещественное число
$AD_{W,B}(j)$	Функция управления усилением, вещественное число
$ALOC_{W,B}(m)$	Расположение управления усилением, определенное в 16.3.1, целое число
$ALEV_{W,B}(m)$	Уровень управления усилением, определенный в 14.3.1, целочисленное вещественное число
B	ID полосы, целое число от 1 до 3
W	ID окна, целое число от 0 до 7
$MBT_{W,B}$	целое число
m	целое число

и $Inter(a,b,j) = 2^{\frac{(8-j)\log_2(a) + j\log_2(b)}{8}}$.

Следует учесть, что начальное значение $PFMD_B(j)$ должно быть равно 1.0.

14.3.3 Взвешивание в окне управления усилением и наложение

Данные полос получаются посредством процессов (1) и (2), приведенные ниже.

(1) Взвешивание в окне

$T_{W,B}(j) = U_{W,B}(j),$

$0 \le j \le 511, W = 0$ *if* ONLY_LONG_SEQUENCE

$0 \le j \le 511, W = 0$ *if* LONG_START_SEQUENCE

$0 \le j \le 63,0 \le W \le 7$ *if* EIGHT_SHORT_SEQUENCE

$0 \le j \le 511, W = 0$ *if* LONG_STOP_SEQUENCE

else

$T_{W,B}(j) = AD_{W,B}(j) \times U_{W,B}(j)$,

$0 \le j \le 511, W = = 0$ *if* ONLY_LONG_ SEQUENCE

$0 \le j \le 511, W = = 0$ *if* LONG_ START_ SEQUENCE

$0 \le j \le 63, 0 \le W \le 7$ *if* EIGHT_ SHORT_ SEQUENCE

$0 \le j \le 511, W = = 0$ *if* LONG_ STOP_ SEQUENCE

(2) Наложение

if ONLY_LONG_SEQUENCE

$V_B(j) = PT_B(j) + T_{0,B}(j)$, $0 \le j \le 255$

$PT_B(j) = T_{0,B}(j + 256)$, $0 \le j \le 255$

if LONG_START_SEQUENCE

$V_B(j) = PT_B(j) + T_{0,B}(j)$, $0 \le j \le 255$

$V_B(j + 256) = T_{0,B}(j + 256)$, $0 \le j \le 111$

$PT_B(j) = T_{0,B}(j + 368)$, $0 \le j \le 31$

if EIGHT_SHORT_SEQUENCE

$V_B(j) = PT_B(j) + T_{W,B}(j)$, $W = 0, 0 \le j \le 31$

$V_B(32W + j) = T_{W-1,B}(j + 32) + T_{W,B}(j)$, $1 \le W \le 7, 0 \le j \le 31$

$PT_B(j) = T_{W,B}(j + 32)$, $W = = 7, 0 \le j \le 31$

if LONG_STOP_SEQUENCE

$V_B(j) = PT_B(j) + T_{0,B}(j + 112)$, $0 \le j \le 31$

$V_B(j + 32) = T_{0,B}(j + 144)$, $0 \le j \le 111$

$PT_B(j) = T_{0,B}(j + 256)$, $0 \le j \le 255$,

где:

$U_{W,B}(j)$ — спектральные данные полосы, вещественное число;

$T_{W,B}(j)$ — данные полосы с управлением усиления, вещественное число;

$PT_B(j)$ — данные полосы с управлением усиления предыдущего фрейма, вещественное число;

$V_B(j)$ — данные полосы, вещественное число;

$AD_{W,B}(j)$ — функция управления усилением, определенная в 14.3.2, вещественное число;

B — ID полосы, целое число от 0 до 3;

W — ID окна, целое число от 0 до 7;

J — целое число.

Следует учесть, что начальное значение $PT_B()j$ должно быть равно 0.0.

14.3.4 Фильтр синтеза

Звуковые данные получаются из следующих уравнений:

(1)

$$\tilde{V}_B(j) = \begin{cases} V_B(k), if\ j = 4k, \\ 0, else \end{cases} 0 \le B \le 3,$$

(2)

$$Q_B(j) = Q(j) \cdot \cos\left(\frac{(2B + 1)(2j - 3)\pi}{16}\right), \quad 0 \le j \le 95, 0 \le B \le 3,$$

(3)

$$AS(n) = \sum_{B=0}^{3} \sum_{j=0}^{95} Q_B(j) \times \tilde{V}_B(n - j),$$

где:

$AS(n)$ — звуковые данные;

$V_B(n)$ — данные полосы, определенные в 16.3.3, вещественное число;

$\hat{V}_{\hat{A}}(j)$ — интерполированные данные полосы, вещественное число;
$QB(j)$ — коэффициенты фильтра синтеза, вещественное число;
$Q(j)$ — коэффициенты фильтра-прототипа, данные ниже, вещественное число;
B — ID полосы, целое число от 0 до 3;
W — ID окна, целое число от 0 до 7;
n — целое число;
j — целое число;
k — целое число.

Значения $Q(0) \ldots Q(47)$ даны в таблице 66. Значения $Q(48) \ldots Q(95)$ получаются из следующего уравнения:

$Q(j) = Q(95 - j)$, $48 \le j \le 95$.

14.4 Схемы

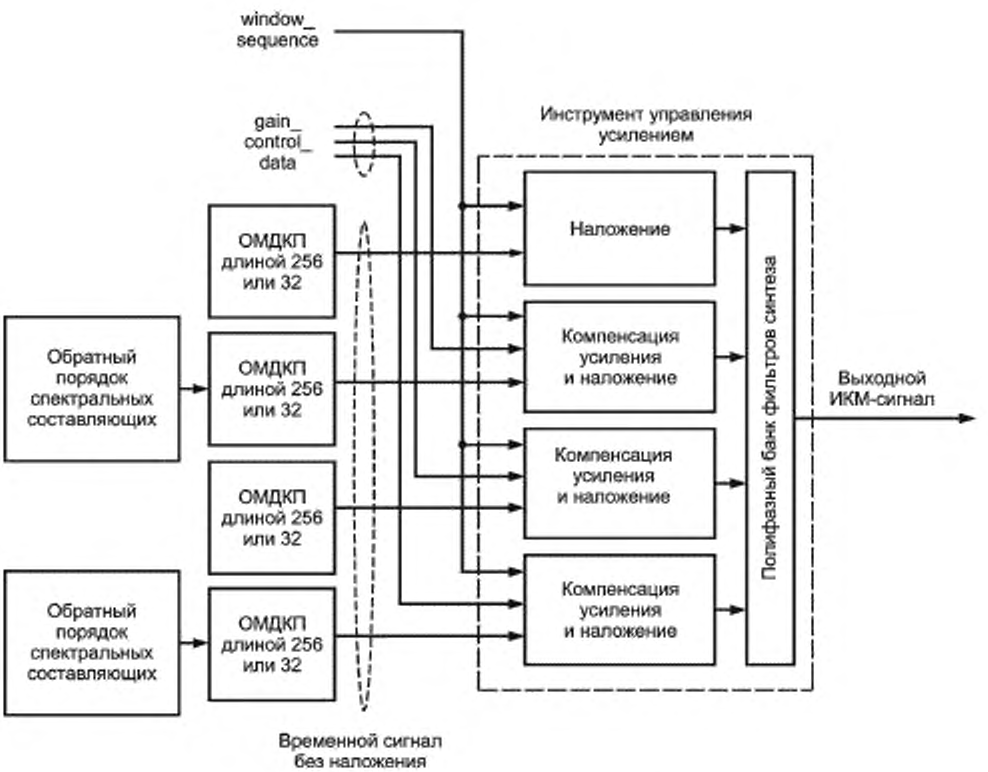


Рисунок 9 — Блок-схема инструмента управления усилением

14.5 Таблицы

Таблица 64 — *AdjLoc ()*

AC	AdjLoc (AC)	AC	AdjLoc (AC)
0	0	16	128
1	8	17	136
2	16	18	144
3	24	19	152
4	32	20	160
5	40	21	168
6	48	22	176
7	56	23	184
8	64	24	192
9	72	25	200
10	80	26	208
11	88	27	216
12	96	28	224
13	104	29	232
14	112	30	240
15	120	31	248

Таблица 65 — *AdjLev ()*

AV	AdjLev (AV)
0	–4
1	–3
2	–2
3	–1
4	0
5	1
6	2
7	3
8	4
9	5
10	6
11	7
12	8
13	9
14	10
15	11

Таблица 66 — *Q ()*

j	Q (j)	j	Q (j)
0	9,7655291007575512E–05	24	–2,2656858741499447E–02
1	1,3809589379038567E–04	25	–6,8031113858963354E–03
2	9,8400749256623534E–05	26	1,5085400948280744E–02
3	–8,6671544782335723E–05	27	3,9750993388272739E–02
4	–4,6217998911921346E–04	28	6,2445363629436743E–02
5	–1,0211814095158174E–03	29	7,7622327748721326E–02
6	–1,6772149340010668E–03	30	7,9968338496132926E–02
7	–2,2533338951411081E–03	31	6,5615493068475583E–02
8	–2,4987888343213967E–03	32	3,3313658300882690E–02
9	–2,1390815966761882E–03	33	–1,4691563058190206E–02
10	–9,5595397454597772E–04	34	–7,2307890475334147E–02
11	1,1172111530118943E–03	35	–1,2993222541703875E–01
12	3,9091309127348584E–03	36	–1,7551641029040532E–01
13	6,9635703420118673E–03	37	–1,9626543957670528E–01
14	9,5595442159478339E–03	38	–1,8073330670215029E–01
15	1,0815766540021360E–02	39	–1,2097653136035738E–01
16	9,8770514991715300E–03	40	–1,4377370758549035E–02
17	6,1562567291327357E–03	41	1,3522730742860303E–01
18	–4,1793946063629710E–04	42	3,1737852699301633E–01
19	–9,2128743097707640E–03	43	5,1590021798482233E–01
20	–1,8830775873369020E–02	44	7,1080020379761377E–01
21	–2,7226498457701823E–02	45	8,8090632488444798E–01
22	–3,2022840857588906E–02	46	1,0068321641150089E+00
23	–3,0996332527754609E–02	47	1,0737914947736096E+00

Приложение А
(обязательное)

Таблицы кодов Хаффмана

Таблица А.1 — Кодовая книга для масштабных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	18	3ffe8	41	12	ff7	82	13	1ff6
1	18	3ffe6	42	12	ff6	83	13	fef8
2	18	3ffe7	43	11	7f9	84	14	3ff8
3	18	3ffe5	44	12	ff4	85	14	3ff4
4	19	7fff5	45	11	7f8	86	16	fff0
5	19	7fff1	46	10	3f9	87	15	7ff4
6	19	7ffed	47	10	3f7	88	16	fff6
7	19	7fff6	48	10	3f5	89	15	7ff5
8	19	7ffee	49	9	1f8	90	18	3ffe2
9	19	7ffef	50	9	1f7	91	19	7ffd9
10	19	7fff0	51	8	fa	92	19	7ffda
11	19	7fffc	52	8	f8	93	19	7ffdb
12	19	7fffd	53	8	f6	94	19	7ffdc
13	19	7ffff	54	7	79	95	19	7ffdd
14	19	7fffe	55	6	3a	96	19	7ffde
15	19	7fff7	56	6	38	97	19	7ffd8
16	19	7fff8	57	5	1a	98	19	7ffd2
17	19	7fffb	58	4	b	99	19	7ffd3
18	19	7fff9	59	3	4	100	19	7ffd4
19	18	3ffe4	60	1	0	101	19	7ffd5
20	19	7fffa	61	4	a	102	19	7ffd6
21	18	3ffe3	62	4	c	103	19	7fff2
22	17	1ffef	63	5	1b	104	19	7ffdf
23	17	1fff0	64	6	39	105	19	7ffe7
24	16	fff5	65	6	3b	106	19	7ffe8
25	17	1ffee	66	7	78	107	19	7ffe9
26	16	fff2	67	7	7a	108	19	7ffea
27	16	fff3	68	8	f7	109	19	7ffeb
28	16	fff4	69	8	f9	110	19	7ffe6
29	16	fff1	70	9	1f6	111	19	7ffe0
30	15	7ff6	71	9	1f9	112	19	7ffe1
31	15	7ff7	72	10	3f4	113	19	7ffe2
32	14	3ff9	73	10	3f6	114	19	7ffe3
33	14	3ff5	74	10	3f8	115	19	7ffe4
34	14	3ff7	75	11	7f5	116	19	7ffe5
35	14	3ff3	76	11	7f4	117	19	7ffd7
36	14	3ff6	77	11	7f6	118	19	7ffec
37	14	3ff2	78	11	7f7	119	19	7fff4
38	13	1ff7	79	12	ff5	120	19	7fff3
39	13	1ff5	80	12	ff8			
40	12	ff9	81	13	1ff4			

Таблица А.2 — Кодовая книга 1 для спектральных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	11	7f8	41	5	14
1	9	1f1	42	7	65
2	11	7fd	43	5	16
3	10	3f5	44	7	6d
4	7	68	45	9	1e9
5	10	3f0	46	7	63
6	11	7f7	47	9	1e4
7	9	1ec	48	7	6b
8	11	7f5	49	5	13
9	10	3f1	50	7	71
10	7	72	51	9	1e3
11	10	3f4	52	7	70
12	7	74	53	9	1f3
13	5	11	54	11	7fe
14	7	76	55	9	1e7
15	9	1eb	56	11	7f3
16	7	6c	57	9	1ef
17	10	3f6	58	7	60
18	11	7fc	59	9	1ee
19	9	1e1	60	11	7f0
20	11	7f1	61	9	1e2
21	9	1f0	62	11	7fa
22	7	61	63	10	3f3
23	9	1f6	64	7	6a
24	11	7f2	65	9	1e8
25	9	1ea	66	7	75
26	11	7fb	67	5	10
27	9	1f2	68	7	73
28	7	69	69	9	1f4
29	9	1ed	70	7	6e
30	7	77	71	10	3f7
31	5	17	72	11	7f6
32	7	6f	73	9	1e0
33	9	1e6	74	11	7f9
34	7	64	75	10	3f2
35	9	1e5	76	7	66
36	7	67	77	9	1f5
37	5	15	78	11	7ff
38	7	62	79	9	1f7
39	5	12	80	11	7f4
40	1	0			

Таблица А.3 — Кодовая книга 2 для спектральных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	9	1f3	41	5	7
1	7	6f	42	6	1d
2	9	1fd	43	5	b
3	8	eb	44	6	30
4	6	23	45	8	ef
5	8	ea	46	6	1c
6	9	1f7	47	7	64
7	8	e8	48	6	1e
8	9	1fa	49	5	c
9	8	f2	50	6	29
10	6	2d	51	8	f3
11	7	70	52	6	2f
12	6	20	53	8	f0
13	5	6	54	9	1fc
14	6	2b	55	7	71
15	7	6e	56	9	1f2
16	6	28	57	8	f4
17	8	e9	58	6	21
18	9	1f9	59	8	e6
19	7	66	60	8	f7
20	8	f8	61	7	68
21	8	e7	62	9	1f8
22	6	1b	63	8	ee
23	8	f1	64	6	22
24	9	1f4	65	7	65
25	7	6b	66	6	31
26	9	1f5	67	4	2
27	8	ec	68	6	26
28	6	2a	69	8	ed
29	7	6c	70	6	25
30	6	2c	71	7	6a
31	5	a	72	9	1fb
32	6	27	73	7	72
33	7	67	74	9	1fe
34	6	1a	75	7	69
35	8	f5	76	6	2e
36	6	24	77	8	f6
37	5	8	78	9	1ff
38	6	1f	79	7	6d
39	5	9	80	9	1f6
40	3	0			

Таблица А.4 — Кодовая книга 3 для спектральных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	1	0	41	10	3ef
1	4	9	42	9	1f3
2	8	ef	43	9	1f4
3	4	b	44	11	7f6
4	5	19	45	9	1e8
5	8	f0	46	10	3ea
6	9	1eb	47	13	1ff c
7	9	1e6	48	8	f2
8	10	3f2	49	9	1f1
9	4	a	50	12	ffb
10	6	35	51	10	3f5
11	9	1ef	52	11	7f3
12	6	34	53	12	ffc
13	6	37	54	8	ee
14	9	1e9	55	10	3f7
15	9	1ed	56	15	7ffe
16	9	1e7	57	9	1f0
17	10	3f3	58	11	7f5
18	9	1ee	59	15	7ffd
19	10	3ed	60	13	1ffb
20	13	3ff a	61	14	3ffa
21	9	1ec	62	16	ffff
22	9	1f2	63	8	f1
23	11	7f9	64	10	3f0
24	11	7f8	65	14	3ffc
25	10	3f8	66	9	1ea
26	12	ff8	67	10	3ee
27	4	8	68	14	3ffb
28	6	38	69	12	ff6
29	10	3f6	70	12	ffa
30	6	36	71	15	7ffc
31	7	75	72	11	7f2
32	10	3f1	73	12	ff5
33	10	3eb	74	16	fffe
34	10	3ec	75	10	3f4
35	12	ff4	76	11	7f7
36	5	18	77	15	7ffb
37	7	76	78	12	ff7
38	11	7f4	79	12	ff9
39	6	39	80	15	7ffa
40	7	74			

Таблица А.5 — Кодовая книга 4 для спектральных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	4	7	41	7	6b
1	5	16	42	8	e3
2	8	f6	43	7	69
3	5	18	44	9	1f3
4	4	8	45	8	eb
5	8	ef	46	8	e6
6	9	1ef	47	10	3f6
7	8	f3	48	7	6e
8	11	7f8	49	7	6a
9	5	19	50	9	1f4
10	5	17	51	10	3ec
11	8	ed	52	9	1f0
12	5	15	53	10	3f9
13	4	1	54	8	f5
14	8	e2	55	8	ec
15	8	f0	56	11	7fb
16	7	70	57	8	ea
17	10	3f0	58	7	6f
18	9	1ee	59	10	3f7
19	8	f1	60	11	7f9
20	11	7fa	61	10	3f3
21	8	ee	62	12	fff
22	8	e4	63	8	e9
23	10	3f2	64	7	6d
24	11	7f6	65	10	3f8
25	10	3ef	66	7	6c
26	11	7fd	67	7	68
27	4	5	68	9	1f5
28	5	14	69	10	3ee
29	8	f2	70	9	1f2
30	4	9	71	11	7f4
31	4	4	72	11	7f7
32	8	e5	73	10	3f1
33	8	f4	74	12	ffe
34	8	e8	75	10	3ed
35	10	3f4	76	9	1f1
36	4	6	77	11	7f5
37	4	2	78	11	7fe
38	8	e7	79	10	3f5
39	4	3	80	11	7fc
40	4	0			

Таблица А.6 — Кодовая книга 5 для спектральных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	13	1fff	41	4	a
1	12	ff7	42	7	71
2	11	7f4	43	8	f3
3	11	7e8	44	11	7e9
4	10	3f1	45	11	7ef
5	11	7ee	46	9	1ee
6	11	7f9	47	8	ef
7	12	ff8	48	5	18
8	13	1ffd	49	4	9
9	12	ffd	50	5	1b
10	11	7f1	51	8	eb
11	10	3e8	52	9	1e9
12	9	1e8	53	11	7ec
13	8	f0	54	11	7f6
14	9	1ec	55	10	3eb
15	10	3ee	56	9	1f3
16	11	7f2	57	8	ed
17	12	ffa	58	7	72
18	12	ff4	59	8	e9
19	10	3ef	60	9	1f1
20	9	1f2	61	10	3ed
21	8	e8	62	11	7f7
22	7	70	63	12	ff6
23	8	ec	64	11	7f0
24	9	1f0	65	10	3e9
25	10	3ea	66	9	1ed
26	11	7f3	67	8	f1
27	11	7eb	68	9	1ea
28	9	1eb	69	10	3ec
29	8	ea	70	11	7f8
30	5	1a	71	12	ff9
31	4	8	72	13	1ffc
32	5	19	73	12	ffc
33	8	ee	74	12	ff5
34	9	1ef	75	11	7ea
35	11	7ed	76	10	3f3
36	10	3f0	77	10	3f2
37	8	f2	78	11	7f5
38	7	73	79	12	ffb
39	4	b	80	13	1ffe
40	1	0			

Таблица А.7 — Кодовая книга 6 для спектральных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	11	7fe	41	4	3
1	10	3fd	42	6	2f
2	9	1f1	43	7	73
3	9	1eb	44	9	1fa
4	9	1f4	45	9	1e7
5	9	1ea	46	7	6e
6	9	1f0	47	6	2b
7	10	3fc	48	4	7
8	11	7fd	49	4	1
9	10	3f6	50	4	5
10	9	1e5	51	6	2c
11	8	ea	52	7	6d
12	7	6c	53	9	1ec
13	7	71	54	9	1f9
14	7	68	55	8	ee
15	8	f0	56	6	30
16	9	1e6	57	6	24
17	10	3f7	58	6	2a
18	9	1f3	59	6	25
19	8	ef	60	6	33
20	6	32	61	8	ec
21	6	27	62	9	1f2
22	6	28	63	10	3f8
23	6	26	64	9	1e4
24	6	31	65	8	ed
25	8	eb	66	7	6a
26	9	1f7	67	7	70
27	9	1e8	68	7	69
28	7	6f	69	7	74
29	6	2e	70	8	f1
30	4	8	71	10	3fa
31	4	4	72	11	7ff
32	4	6	73	10	3f9
33	6	29	74	9	1f6
34	7	6b	75	9	1ed
35	9	1ee	76	9	1f8
36	9	1ef	77	9	1e9
37	7	72	78	9	1f5
38	6	2d	79	10	3fb
39	4	2	80	11	7fc
40	4	0			

Таблица А.8 — Кодовая книга 7 для спектральных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	1	0	32	8	f3
1	3	5	33	8	ed
2	6	37	34	9	1e8
3	7	74	35	9	1ef
4	8	f2	36	10	3ef
5	9	1eb	37	10	3f1
6	10	3ed	38	10	3f9
7	11	7f7	39	11	7fb
8	3	4	40	9	1ed
9	4	c	41	8	ef
10	6	35	42	9	1ea
11	7	71	43	9	1f2
12	8	ee	44	10	3f3
13	8	ee	45	10	3f8
14	9	1ee	46	11	7f9
15	9	1f5	47	11	7fc
16	6	36	48	10	3ee
17	6	34	49	9	1ec
18	7	72	50	9	1f4
19	8	ea	51	10	3f4
20	8	f1	52	10	3f7
21	9	1e9	53	11	7f8
22	9	1f3	54	12	ffd
23	10	3f5	55	12	ffe
24	7	73	56	11	7f6
25	7	70	57	10	3f0
26	8	eb	58	10	3f2
27	8	f0	59	10	3f6
28	9	1f1	60	11	7fa
29	9	1f0	61	11	7fd
30	10	3ec	62	12	ffc
31	10	3fa	63	12	fff

Таблица А.9 — Кодовая книга 8 для спектральных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	5	e	18	4	6
1	4	5	19	5	14
2	5	10	20	6	2e
3	6	30	21	7	69
4	7	6f	22	7	72
5	8	f1	23	8	f5
6	9	1fa	24	6	2f
7	10	3fe	25	5	11
8	4	3	26	5	13
9	3	0	27	6	2a
10	4	4	28	6	32
11	5	12	29	7	6c
12	6	2c	30	8	ec
13	7	6a	31	8	fa
14	7	75	32	7	71
15	8	f8	33	6	2b
16	5	f	34	6	2d
17	4	2	35	6	31

Окончание таблицы А.9

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
36	7	6d	50	7	73
37	7	70	51	8	ed
38	8	f2	52	8	f0
39	9	1f9	53	8	f6
40	8	ef	54	9	1f6
41	7	68	55	9	1fd
42	6	33	56	10	3fd
43	7	6b	57	8	f3
44	7	6e	58	8	f4
45	8	ee	59	8	f7
46	8	f9	60	9	1f7
47	10	3fc	61	9	1fb
48	9	1f8	62	9	1fc
49	7	74	63	10	3ff

Таблица А.10 — Кодовая книга 9 для спектральных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	1	0	35	11	7d0	70	11	7d1
1	3	5	36	12	fc7	71	11	7db
2	6	37	37	12	fd4	72	12	fd2
3	8	e7	38	12	fe4	73	11	7e0
4	9	1de	39	8	e6	74	12	fd9
5	10	3ce	40	7	70	75	12	feb
6	10	3d9	41	8	e9	76	13	1fe3
7	11	7c8	42	9	1dd	77	13	1fe9
8	11	7cd	43	9	1e3	78	11	7c4
9	12	fc8	44	10	3d2	79	9	1e5
10	12	fdd	45	10	3dc	80	10	3d7
11	13	1fe4	46	11	7cc	81	11	7c6
12	13	1fec	47	11	7ca	82	11	7cf
13	3	4	48	11	7de	83	11	7da
14	4	c	49	12	fd8	84	12	fcb
15	6	35	50	12	fea	86	12	fe3
16	7	72	51	13	1fdb	87	12	fe9
17	8	ea	52	9	1df	88	13	1fe6
18	8	ed	53	8	eb	89	13	1ff3
19	9	1e2	54	9	1dc	90	13	1ff7
20	10	3d1	55	9	1e6	91	11	7d3
21	10	3d3	56	10	3d5	92	10	3d8
22	10	3e0	57	10	3de	93	10	3e1
23	11	7d8	58	11	7cb	94	11	7d4
24	12	fcf	59	11	7dd	95	11	7d9
25	12	fd5	60	11	7dc	96	12	fd3
26	6	36	61	12	fcd	97	12	fde
27	6	34	62	12	fe2	98	13	1fdd
28	7	71	63	12	fe7	99	13	1fd9
29	8	e8	64	13	1fe1	100	13	1fe2
30	8	ec	65	10	3d0	101	13	1fea
31	9	1e1	66	9	1e0	102	13	1ff1
32	10	3cf	67	9	1e4	103	13	1ff6
33	10	3dd	68	10	3d6	104	11	7d2
34	10	3db	69	11	7c5	105	10	3d4

Окончание таблицы А.10

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
106	10	3da	127	14	3ff3	148	13	1fdc
107	11	7c7	128	14	3ff4	149	13	1fdf
108	11	7d7	129	14	3ff5	150	13	1fed
109	11	7e2	130	12	fe0	151	13	1ff5
110	12	fce	131	11	7ce	152	14	3ff9
111	12	fdb	132	11	7d5	153	14	3ffb
112	13	1fd8	133	12	fc6	154	15	7ffd
113	13	1fee	134	12	fd1	155	15	7ffe
114	14	3ff0	135	12	fe1	156	13	1fe7
115	13	1ff4	136	13	1fe0	157	12	fee
116	14	3ff2	137	13	1fe8	158	12	fd6
117	11	7e1	138	13	1ff0	159	12	fdf
118	10	3df	139	14	3ff1	160	13	1fde
119	11	7c9	140	14	3ff8	161	13	1fda
120	11	7d6	141	14	3ff6	162	13	1fe5
121	12	fca	142	15	7ffc	163	13	1ff2
122	12	fd0	143	12	fe8	164	14	3ffa
123	12	fe5	144	11	7df	165	14	3ff7
124	12	fe6	145	12	fc9	166	14	3ffc
125	13	1feb	146	12	fd7	167	14	3ffd
126	13	1fef	147	12	fdc	168	15	7fff

Таблица А.11 — Кодовая книга 10 для спектральных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	6	22	27	4	2	54	6	1f
1	5	8	28	5	6	55	6	23
2	6	1d	29	5	c	56	6	27
3	6	26	30	6	1e	57	7	59
4	7	5f	31	6	28	58	7	64
5	8	d3	32	7	5b	59	8	d8
6	9	1cf	33	8	cd	60	8	df
7	10	3d0	34	8	d9	61	9	1d2
8	10	3d7	35	9	1ce	62	9	1e2
9	10	3ed	36	9	1dc	63	10	3dd
10	11	7f0	37	10	3d9	64	10	3ee
11	11	7f6	38	10	3f1	65	8	d1
12	12	ffd	39	6	25	66	7	55
13	5	7	40	5	b	67	6	29
14	4	0	41	5	a	68	7	56
15	4	1	42	5	d	69	7	58
16	5	9	43	6	24	70	7	62
17	6	20	44	7	57	71	8	ce
18	7	54	45	7	61	72	8	e0
19	7	60	46	8	cc	73	8	e2
20	8	d5	47	8	dd	74	9	1da
21	8	dc	48	9	1cc	75	10	3d4
22	9	1d4	49	9	1de	76	10	3e3
23	10	3cd	50	10	3d3	77	11	7eb
24	10	3de	51	10	3e7	78	9	1c9
25	11	7e7	52	7	5d	79	7	5e
26	6	1c	53	6	21	80	7	5a

Окончание таблицы А.11

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
81	7	5c	111	9	1d6	140	11	7f3
82	7	63	112	10	3d1	141	11	7f9
83	8	ca	113	10	3d5	142	12	ff9
84	8	da	114	10	3f2	143	11	7f2
85	9	1c7	115	11	7ee	144	10	3ce
86	9	1ca	116	11	7fb	145	9	1e4
87	9	1e0	117	10	3e9	146	10	3cb
88	10	3db	118	9	1cd	147	10	3d8
89	10	3e8	119	9	1c8	148	10	3d6
90	11	7ec	120	9	1cb	149	10	3e2
91	9	1e3	121	9	1d1	150	10	3e5
92	8	d2	122	9	1d7	151	11	7e8
93	8	cb	123	9	1df	152	11	7f4
94	8	d0	124	10	3cf	153	11	7f5
95	8	d7	125	10	3e0	154	11	7f7
96	8	db	126	10	3ef	155	12	ffb
97	9	1c6	127	11	7e6	156	11	7fa
98	9	1d5	128	11	7f8	157	10	3ec
99	9	1d8	129	12	ffa	158	10	3df
100	10	3ca	130	10	3eb	159	10	3e1
101	10	3da	131	9	1dd	160	10	3e4
102	11	7ea	132	9	1d3	161	10	3e6
103	11	7f1	133	9	1d9	162	10	3f0
104	9	1e1	134	9	1db	163	11	7e9
105	8	d4	135	10	3d2	164	11	7ef
106	8	cf	136	10	3cc	165	12	ff8
107	8	d6	137	10	3dc	166	12	ffe
108	8	de	138	10	3ea	167	12	ffc
109	8	e1	139	11	7ed	168	12	fff
110	9	1d0						

Таблица А.12 — Кодовая книга 11 для спектральных коэффициентов

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
0	4	0	18	4	1	36	5	9
1	5	6	19	5	8	37	6	18
2	6	19	20	6	14	38	7	39
3	7	3d	21	7	37	39	7	40
4	8	9c	22	7	42	40	8	8e
5	8	c6	23	8	92	41	8	a3
6	9	1a7	24	8	af	42	8	b8
7	10	390	25	9	191	43	9	199
8	10	3c2	26	9	1a5	44	9	1ac
9	10	3df	27	9	1b5	45	9	1c
10	11	7e6	28	10	39e	46	10	3b1
11	11	7f3	29	10	3c0	47	10	396
12	12	ffb	30	10	3a2	48	10	3be
13	11	7ec	31	10	3cd	49	10	3ca
14	12	ffa	32	11	7d6	50	8	9d
15	12	ffe	33	8	ae	51	7	3c
16	10	38e	34	6	17	52	6	15
17	5	5	35	5	7	53	6	16

Продолжение таблицы А.12

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
54	6	1a	110	9	19f	166	10	3e0
55	7	3b	111	9	1af	167	11	7d2
56	7	44	112	9	1bf	168	11	7e5
57	8	91	113	10	399	169	8	b7
58	8	a5	114	10	3bf	170	11	7e3
59	8	be	115	10	3b4	171	9	1bb
60	9	196	116	10	3c9	172	9	1a8
61	9	1ae	117	10	3e7	173	9	1a6
62	9	1b9	118	8	a8	174	9	1b0
63	10	3a1	119	9	1b6	175	9	1b2
64	10	391	120	8	ab	176	9	1b7
65	10	3a5	121	8	a4	177	10	39b
66	10	3d5	122	8	aa	178	10	39a
67	8	94	123	8	b2	179	10	3ba
68	8	9a	124	8	c2	180	10	3b5
69	7	36	125	8	c5	181	10	3d6
70	7	38	126	9	198	182	11	7d7
71	7	3a	127	9	1a4	183	10	3e4
72	7	41	128	9	1b8	184	11	7d8
73	8	8c	129	10	38c	185	11	7ea
74	8	9b	130	10	3a4	186	8	ba
75	8	b0	131	10	3c4	187	11	7e8
76	8	c3	132	10	3c6	188	10	3a0
77	9	19e	133	10	3dd	189	9	1bd
78	9	1ab	134	10	3e8	190	9	1b4
79	9	1bc	135	8	ad	191	10	38a
80	10	39f	136	10	3af	192	9	1c4
81	10	38f	137	9	192	193	10	392
82	10	3a9	138	8	bd	194	10	3aa
83	10	3cf	139	8	bc	195	10	3b0
84	8	93	140	9	18e	196	10	3bc
85	8	bf	141	9	197	197	10	3d7
86	7	3e	142	9	19a	198	11	7d4
87	7	3f	143	9	1a3	199	11	7dc
88	7	43	144	9	14	200	11	7db
89	7	45	145	10	38d	201	11	7d5
90	8	9e	146	10	398	202	11	7f0
91	8	a7	147	10	3b7	203	8	d
92	8	b9	148	10	3d3	204	11	7fb
93	9	194	149	10	3d1	205	10	3c8
94	9	1a2	150	10	3db	206	10	3a3
95	9	1ba	151	11	7dd	207	10	395
96	9	1c3	152	8	b4	208	10	39d
97	10	3a6	153	10	3de	209	10	3ac
98	10	3a7	154	9	1a9	210	10	3ae
99	10	3bb	155	9	19b	211	10	3c5
100	10	3d4	156	9	19c	212	10	3d8
101	8	9f	157	9	1a1	213	10	3e2
102	9	1a0	158	9	1aa	214	10	3e6
103	8	8f	159	9	1ad	215	11	7e4
104	8	8d	160	9	1b3	216	11	7e7
105	8	90	161	10	38b	217	11	7e0
106	8	98	162	10	3b2	218	11	7e9
107	8	a6	163	10	3b8	219	11	7f7
108	8	b6	164	10	3ce	220	9	190
109	8	c4	165	10	3e1	221	11	7f2

Окончание таблицы А.12

Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)	Индекс	Длина	Кодовая комбинация (шестнадцатеричная)
222	10	393	245	10	3d0	267	11	7f5
223	9	1be	246	10	3e3	268	11	7f6
224	9	1c0	247	10	3e5	269	12	ffc
225	10	394	248	11	7e2	270	12	fff
226	10	397	249	11	7de	271	9	19d
227	10	3ad	250	11	7ed	272	9	1c2
228	10	3c3	251	11	7f1	273	8	b5
229	10	3c1	252	11	7f9	274	8	a1
230	10	3d2	253	11	7fc	275	8	96
231	11	7da	254	9	193	276	8	97
232	11	7d9	255	12	ffd	277	8	95
233	11	7df	256	10	3dc	278	8	99
234	11	7eb	257	10	3b6	279	8	a0
235	11	7f4	258	10	3c7	280	8	a2
236	11	7fa	259	10	3cc	281	8	ac
237	9	195	260	10	3cb	282	8	a9
238	11	7f8	261	10	3d9	283	8	b1
239	10	3bd	262	10	3da	284	8	b3
240	10	39c	263	11	7d3	285	8	bb
241	10	3ab	264	11	7e1	286	8	c0
242	10	3a8	265	11	7ee	287	9	18f
243	10	3b3	266	11	7ef	288	5	4
244	10	3b9						

Таблица А.13 — Окно Кайзера-Бесселя для SSR профиля EIGHT_SHORT_SEQUENCE

i	*w(i)*	*i*	*w(i)*	*i*	*w(i)*
0	0.0000875914060105	11	0.3260874142923209	22	0.9827581789763112
1	0.0009321760265333	12	0.4089316830907141	23	0.9914756203467121
2	0.0032114611466596	13	0.4959414909423747	24	0.9961964092194694
3	0.0081009893216786	14	0.5833939894958904	25	0.9984956609571091
4	0.0171240286619181	15	0.6674601983218376	26	0.9994855586984285
5	0.0320720743527833	16	0.7446454751465113	27	0.9998533730714648
6	0.0548307856028528	17	0.8121892962974020	28	0.9999671864476404
7	0.0871361822564870	18	0.8683559394406505	29	0.9999948432453556
8	0.1302923415174603	19	0.9125649996381605	30	0.9999995655238333
9	0.1848955425508276	20	0.9453396205809574	31	0.9999999961638728
10	0.2506163195331889	21	0.9680864942677585		

Таблица А.14 — Окно Кайзера-Бесселя для SSR профиля для остальных последовательностей окон

i	*w(i)*	*i*	*w(i)*	*i*	*w(i)*
0	0.0005851230124487	9	0.0051974336885144	18	0.0141233971318631
1	0.0009642149851497	10	0.0059425050016407	19	0.0154573353235409
2	0.0013558207534965	11	0.0067439602523141	20	0.0168678890600951
3	0.0017771849644394	12	0.0076040812644888	21	0.0183572550877256
4	0.0022352533849672	13	0.0085251378135895	22	0.0199276125319803
5	0.0027342299070304	14	0.0095093917383048	23	0.0215811201042484
6	0.0032773001022195	15	0.0105590986429280	24	0.0233199132076965
7	0.0038671998069216	16	0.0116765080854300	25	0.0251461009666641
8	0.0045064443384152	17	0.0128638627792770	26	0.0270617631981826

Продолжение таблицы А.14

i	*w(i)*	*i*	*w(i)*	*i*	*w(i)*
27	0.0290689473405856	84	0.3288452410620226	141	0.8046173857073919
28	0.0311696653515848	85	0.3371715818562547	142	0.8110450872887550
29	0.0333658905863535	86	0.3455670606953511	143	0.8173544143867162
30	0.0356595546648444	87	0.3540283646950029	144	0.8235441764639875
31	0.0380525443366107	88	0.3625521080463003	145	0.8296133044858474
32	0.0405466983507029	89	0.3711348353596863	146	0.8355608512093652
33	0.0431438043376910	90	0.3797730251194006	147	0.8413859912867303
34	0.0458455957104702	91	0.3884630932439016	148	0.8470880211822968
35	0.0486537485902075	92	0.3972013967475546	149	0.8526663589032990
36	0.0515698787635492	93	0.4059842374986933	150	0.8581205435445334
37	0.0545955386770205	94	0.4148078660689724	151	0.8634502346476508
38	0.0577322144743916	95	0.4236684856687616	152	0.8686552113760616
39	0.0609813230826460	96	0.4325622561631607	153	0.8737353715068081
40	0.0643442093520723	97	0.4414852981630577	154	0.8786907302411250
41	0.0678221432558827	98	0.4504336971855032	155	0.8835214188357692
42	0.0714163171546603	99	0.4594035078775303	156	0.8882276830575707
43	0.0751278431308314	100	0.4683907582974173	157	0.8928098814640207
44	0.0789577503982528	101	0.4773914542472655	158	0.8972684835130879
45	0.0829069827918993	102	0.4864015836506502	159	0.9016040675058185
46	0.0869763963425241	103	0.4954171209689973	160	0.9058173183656508
47	0.0911667569410503	104	0.5044340316502417	161	0.9099090252587376
48	0.0954787380973307	105	0.5134482766032377	162	0.9138800790599416
49	0.0999129187977865	106	0.5224558166913167	163	0.9177314696695282
50	0.1044697814663005	107	0.5314526172383208	164	0.9214642831859411
51	0.1091497100326053	108	0.5404346525403849	165	0.9250796989403991
52	0.1139529881122542	109	0.5493979103766972	166	0.9285789863994010
53	0.1188797973021148	110	0.5583383965124314	167	0.9319635019415643
54	0.1239302155951605	111	0.5672521391870222	168	0.9352346855155568
55	0.1291042159181728	112	0.5761351935809411	169	0.9383940571861993
56	0.1344016647957880	113	0.5849836462541291	170	0.9414432135761304
57	0.1398223211441467	114	0.5937936195492526	171	0.9443838242107182
58	0.1453658351972151	115	0.6025612759529649	172	0.9472176277741918
59	0.1510317475686540	116	0.6112828224083939	173	0.9499464282852282
60	0.1568194884519144	117	0.6199545145721097	174	0.9525720912004834
61	0.1627283769610327	118	0.6285726610088878	175	0.9550965394547873
62	0.1687576206143887	119	0.6371336273176413	176	0.9575217494469370
63	0.1749063149634756	120	0.6456338401819751	177	0.9598497469802043
64	0.1811734433685097	121	0.6540697913388968	178	0.9620826031668507
65	0.1875578769224857	122	0.6624380414593221	179	0.9642224303060783
66	0.1940583745250518	123	0.6707352239341151	180	0.9662713777449607
67	0.2006735831073503	124	0.6789580485595255	181	0.9682316277319895
68	0.2074020380087318	125	0.6871033051160131	182	0.9701053912729269
69	0.2142421635060113	126	0.6951678668345944	183	0.9718949039986892
70	0.2211922734956977	127	0.7031486937449871	184	0.9736024220549734
71	0.2282505723293797	128	0.7110428359000029	185	0.9752302180233160
72	0.2354151558022098	129	0.7188474364707993	186	0.9767805768831932
73	0.2426840122941792	130	0.7265597347077880	187	0.9782557920246753
74	0.2500550240636293	131	0.7341770687621900	188	0.9796581613210076
75	0.2575259686921987	132	0.7416968783634273	189	0.9809899832703159
76	0.2650945206801527	133	0.7491167073477523	190	0.9822535532154261
77	0.2727582531907993	134	0.7564342060337386	191	0.9834511596505429
78	0.2805146399424422	135	0.7636471334404891	192	0.9845850806232530
79	0.2883610572460804	136	0.7707533593446514	193	0.9856575802399989
80	0.2962947861868143	137	0.7777508661725849	194	0.9866709052828243
81	0.3043130149466800	138	0.7846377507242818	195	0.9876272819448033
82	0.3124128412663888	139	0.7914122257259034	196	0.9885289126911557
83	0.3205912750432127	140	0.7980726212080798	197	0.9893779732525968

Окончание таблицы А.14

i	*w(i)*	*i*	*w(i)*	*i*	*w(i)*
198	0.9901766097569984	218	0.9985085513687731	238	0.9999172570940037
199	0.9909269360049311	219	0.9986693885387259	239	0.9999318272557038
200	0.9916310308941294	220	0.9988157050968516	240	0.9999442511639580
201	0.9922909359973702	221	0.9989485378906924	241	0.9999547847121726
202	0.9929086532976777	222	0.9990688725744943	242	0.9999636603523446
203	0.9934861430841844	223	0.9991776444921379	243	0.9999710885561258
204	0.9940253220113651	224	0.9992757396582338	244	0.9999772592414866
205	0.9945280613237534	225	0.9993639958299003	245	0.9999823431612708
206	0.9949961852476154	226	0.9994432036616085	246	0.9999864932503106
207	0.9954314695504363	227	0.9995141079353859	247	0.9999898459281599
208	0.9958356402684387	228	0.9995774088586188	248	0.9999925223548691
209	0.9962103726017252	229	0.9996337634216871	249	0.9999946296375997
210	0.9965572899760172	230	0.9996837868076957	250	0.9999962619864214
211	0.9968779632693499	231	0.9997280538466377	251	0.9999975018180320
212	0.9971739102014799	232	0.9997671005064359	252	0.9999984208055542
213	0.9974465948831872	233	0.9998014254134544	253	0.9999990808746198
214	0.9976974275220812	234	0.9998314913952471	254	0.9999995351446231
215	0.9979277642809907	235	0.9998577270385304	255	0.9999998288155155
216	0.9981389072844972	236	0.9998805282555989		
217	0.9983321047686901	237	0.9999002598526793		

Приложение Б
(обязательное)

Информация о неиспользованных кодовых книгах

В настоящем стандарте декодер *AAC* не использует сборники кодов #12 и #13. Однако, при желании, декодер может использовать эти сборники кодов, чтобы расширить его функциональность, если это не противоречит другим стандартам MPEG, таким как ИСО/МЭК 14496-3 [3], которые используют эти сборники кодов для расширенных методов кодирования.

Пример: синтаксис в 4.3 изменился бы, как представлено в таблице Б1.

Таблица Б.1 — Расширенный синтаксис для *scale_factor_data* ()

Синтаксис	Биты	Мнемоника
scale_factor_data() {		
noise_pcm_flag = 1;		
for (g = 0; g < num_window_groups; g++) {		
for (sfb = 0; sfb < max sfb; sfb++) {		
if (sfb_cb[g][sfb] != ZERO_HCB) {		
if (is_intensity(g,sfb))		
hcod_sf[dpcm_is_position[g][sfb]];	1..19	vlclbf
else if (sfb_cb[g][sfb] == 13)		
if (noise_pcm_flag) {		
noise_pcm_flag = 0;		
dpcm_noise_nrg[g][sfb];	9	uimsbf
} else		
hcod_sf[dpcm_noise_nrg[g][sfb]];	1..19	vlclbf
else		
hcod sf[dpcm sf[g][sfb]];	1..19	vlclbf
}		
}		
}		
}		

Приложение В
(обязательное)

Кодер

В.1 Психоакустическая модель

В.1.1 Общие сведения

В этом приложении представлена психоакустическая модель для кодера ААС. Психоакустическая модель вычисляет максимальную энергию искажений, которая маскируется энергией сигнала. Эту энергию называют порогом маскирования. У процесса вычисления порога маскирования есть три входа:

1 Длина сдвига для процесса вычисления порога маскирования *iblen*. Параметр *iblen* должен оставаться постоянным. Так как необходимо вычислить пороги для двух различных длин сдвига, необходимо два процесса, каждый со своей фиксированной длиной сдвига. Для длинного быстрого преобразования Фурье (БПФ) *iblen* = 1024, для короткого БПФ *iblen* = 128.

2 Для каждого типа БПФ следующие *iblen* отсчетов сигнала с задержанными отсчетами (в банке фильтров или в блоке психоакустического расчета) располагаются таким образом, чтобы окно психоакустического расчета оказалось по центру окна частотно-временного преобразования.

3 Частота дискретизации. Существуют наборы таблиц, предусмотренные для стандартных частот дискретизации. Частота дискретизации так же, как *iblen*, должна оставаться постоянной в течение одной реализации процесса вычисления порога маскирования.

Выходы психоакустической модели:

1 Набор отношений сигнал-маска и значений порога, которые адаптируются к кодеру как описано ниже.

2 Задержанные временные данные (ИКМ-отсчеты), которые используются МДКП.

3 Тип блока МДКП (длинный, стартовый, стоповый или короткий).

4 Оценка количества битов, которое должно использоваться для кодирования в дополнение к среднему количеству доступных битов.

Задержка ИКМ отсчетов необходима, ибо алгоритм переключения обнаруживает резкую атаку, т. е. для фрейма должны использоваться короткие блоки, длинный блок перед короткими блоками должен быть изменен на стартовый.

Перед начальной реализацией модели массив с предыдущих данных БПФ и массивы *r* (*w*) и *f* (*w*) должны быть обнулены для обеспечения известной начальной точки.

В.1.2 Комментарии

В процессе вычисления порога маскирования используются три индекса для значений данных:

w — индекс частотной линии спектра БПФ. Индекс 0 соответствует постоянной составляющей, индекс 1023 соответствует линии спектра на частоте Найквиста.

b — индекс частотной полосы. Если вычисление включает свертку или сумму, *bb* используется в качестве переменной суммирования. Нумерация полос начинается с 0.

n — индекс полосы масштабных коэффициентов. Индекс 0 соответствует самой низкой полосе масштабных коэффициентов.

В.1.3 Функция маскирования

В психоакустическом расчете используется «функция маскирования». Она вычисляется следующим образом:

```
if j >= i
  tmpx = 3.0 (j–i)
else
  tmpx = 1.5 (j–i),
```

где *i* — значение, барк, определяющее расстояние от маскера, *j* — значение, барк, соответствующее положению маскера, *tmpx* — временная переменная.

$tmpz = 8 * \min((tmpx–0{,}5)^2–2\ (tmpx–0{,}5), 0)$,

где *tmpz* — временная переменная, *min(a, b)* — функция, возвращающая большее отрицательное значение из *a* и *b*.

$tmpy = 15{,}811389 + 7{,}5\ (tmpx + 0{,}474)–17{,}5\ (1{,}0 + (tmpx + 0{,}474)^2)^{0.5}$,

где *tmpy* — временная переменная.

if(*tmpy* <–100) then {*sprdngf* (i, j) =0} else {*sprdngf* (i, j) = 10 ^ ((*tmpz* + *tmpy*)/10)}

В.1.4 Шаги при вычислении порога маскирования

Следующие шаги являются необходимыми для вычисления *SMR (n)* и *xmin (n)*, используемыми в кодере для длинного и короткого БПФ.

1 Восстановить 2 * *iblen* отсчетов входного сигнала.

iblen новых отсчетов делаются доступными при каждом вызове генератора порога. Генератор порога должен хранить 2 * *iblen* — *iblen* отсчетов и объединить эти отсчеты для точного восстановления 2 * *iblen* последовательных отсчетов входного сигнала, *s* (*i*), где *i* — индекс входных отсчетов, 0 <= *i* <2 * *iblen*.

2 Вычислить комплексный спектр входного сигнала.

Во-первых, *s (i)* взвешивается окном Хана, т. е.

*sw(i) = s(i) * (0,5–0,5 * cos((pi *(i+0,5))/ iblen).*

Во-вторых, вычисляется стандартное БПФ *sw(i).*

В-третьих, вычисляется полярное представление преобразования. *r (w)* и *f (w)* соответствуют амплитудным и фазовым частям преобразованного *sw(i)* соответственно.

3 Вычислить предсказание *r (w)* и *f (w).*

Предсказанные амплитуда *r_pred (w)* и фаза *f_pred (w)* вычисляются по двум предыдущим блокам расчета порога *r(w)* и *f (w)*:

*r_pred (w) = 2,0 * r (t–1)–r (t–2)*

*f_pred (w) =2,0 * f (t–1)–f (t–2),*

где *t* — текущий номер блока, *t–1* соответствует данным предыдущего блока и *t–2* соответствует данным блока перед предыдущим.

4 Вычислить меру неопределенности *c (w).*

*c(w) = (((r (w) * cos (f (w)) – r_pred (w) * cos (f_pred (w))) ^2 + (r (w) * sin(f(w)) – r_pred(w) * sin(f_pred(w)))^2)^0.5) / (r(w) + abs (r_pred (w)).*

Эта формула используется для каждого из коротких блоков с коротким БПФ, для первых 6 строк длинных блоков мера неопределенности вычисляется по длинному БПФ, для оставшихся строк используется минимальное значение неопределенности для коротких блоков БПФ. Если необходимо уменьшить сложность вычислений, неопределенность для верхней части спектра может быть установлена равной 0,4.

5 Вычислить энергию и неопределенность в разделах вычисления порога.

Энергия каждого раздела, *e (b)*:

do for each partition b:

e(b) = 0

do from lower index to upper index w of partition b e (b) = e (b) + r (w) ^2

end *do end*

do

(*e (b)* используется в модуле *MIS* (см. Б.6.1): *e (b)* равно Xengy с 'X' = [R, L, M, S]) и взвешенная неопределенность, *c (b)*:

do for each partition b: c(b) = 0 do from lower index to upper index w of partition b

c(b) = *c (b) + r(w)^2 * c (w) end do end do*

Разделы вычисления порога обеспечивают разрешение, примерно равное одной частотной линии БПФ или 1/3 критической полосы. На нижних частотах одна линия БПФ соответствует одному разделу вычисления. На верхних частотах несколько строк будут объединены в один раздел вычисления. Набор значений линий разделов для каждой из трех частот дискретизации задан в таблицах Б.1 — Б.24. Элементы этих таблиц будут использоваться в процессе вычисления порога маскирования. В каждой таблице есть следующие графы:

1 Индекс раздела вычисления, *b*;

2 Самая низкая частотная линия раздела, *w_low (b)*;

3 Самая высокая частотная линия раздела, *w_high (b)*;

4 Среднее значение частоты раздела, барк, *bval (b)*;

5 Абсолютный порог слышимости *qsthr (b).*

6 Максимальное значение *b*, *bmax*, равное самому большому индексу, существующему для данной частоты дискретизации.

6 Вычислить свертку энергии разделов и неопределенности с функцией маскирования.

for each partition b:

ecb(b) = 0

do for each partition bb:

*ecb (b) = ecb (b) +e (bb) * sprdngf (bval (bb) , bval (b))*

end do end do do for each partition b:

ct (b) = 0

do for each partition bb:

ct(b) = ct(b) +c(bb) sprdngf (bval (bb) , bval (b))*

end do end do

Поскольку ct (b) взвешивается энергией сигнала, оно должно быть повторно нормализовано к cb (b)

cb (b) = ct (b) / ecb (b)

Аналогично, из-за ненормализованной природы функции маскирования *ecbb* должно быть повторно нормализовано и нормализованная энергия *en(b)* равна:

en (b) = ecb (b) / morm (b)

коэффициент нормализации, *morm (b):*

do for each partition b tmp (b) = 0

do for each partition bb

tmp (b) = tmp (b) + sprdngf (bval (bb) ,bval (b)) end do

morm (b) = 1/ tmp (b) end do

7 Преобразовать *cb (b)* в *tb (b)*, индекс тональности.

tb(b) = *–0.299 – 0.43* log_0 (cb(b))

Каждый *tb (b)* ограничен диапазоном *0 <tb (b)* <1.

8 Вычислить необходимое SNR в каждом разделе.

NMT (b) = 6 дБ для всех b. NMT (b) соответствует шумовому маскеру (в дБ) для раздела. TMN (b) = 18 дБ для всех b. TMN (b) соответствует тональному маскеру (в дБ). Требуемое отношение сигнал-шум, SNR (b):

*SNR (b) = tb (b) * TMN (b) + (1–tb (b)) * NMT (b).*

9 Вычислить отношение мощностей.

Отношение мощностей, *bc (b):*

bc (b) *=10 ^ (–SNR (b)/10)*

10 Вычислить фактический энергетический порог, nb (b).

*nb (b) = en (b) * bc (b)* nb (b) также используется в модуле *M/S* (см. пункт 12): nb (b) равен Xthr с 'X' = [R, L, M., S].

11 Контроль пре-эха и абсолютный порог слышимости

Чтобы избежать пре-эха, вычисляется контроль пре-эхом для коротких и длинных БПФ, абсолютный порог слышимости также учитывается здесь:

nb_l (b) является порогом раздела b для последнего блока, *qsthr (b)* является абсолютным порогом слышимости. Значения дБ *qsthr (b)* показаны на рисунке В.1

Значения таблиц В.1 — В.24 даны относительно уровня частотной линии БПФ, которой соответствует синусоидальная волна амплитудой ± *1* lsb. Значения (в дБ) должны быть преобразованы в единицы энергии после нормализации БПФ.

nb (b) = max (qsthr (b), min (nb (b), nb_l (b) *rpelev))

rpelev устанавливается в '1' для коротких блоков и '2' для длинных блоков

12 PE вычисляется для каждого типа блока из отношения e (b) / nb (b), где nb (b) — порог маскирования и e (b) — энергия для каждого порогового раздела.

PE = 0

do for threshold partition b

*PE = PE – (w_high(b)–w_low(b)) * log10 (nb(b) / (e (b) +1)) end do*

13 Принимается решение об использовании длинного или короткого блока.

if PE for long block is greater than switch_pe then

coding_block_type = short_block_type else

coding_block_type = long_block_type end if if (coding_block_type == short_block_type)

and

(last_coding_block_type == long_type) then

last coding block type = start_type else

last_coding_block_type = short_type.

Последние четыре строки необходимы, так как в AAC нет комбинированного стартового/стопового блока. *switch_pe* — постоянная, зависящая от реализации.

14 Вычислить отношения сигнал-маска, *SMR (n)* и порог маскирования *xmin (n)*

Таблицы 45 … 57 дают:

1 Индекс *swb* раздела кодера, называемого полосой масштабных коэффициентов.

2 Смещение линии МДКП для полосы масштабных коэффициентов *swb_offset_long/short_window.*

Вводится следующая переменная:

n = swb

w_low (n) = swb_offset_long/short_window (n) w_high (n) =

swb_offset_long/short_window (n+1) – 1

энергия БПФ в полосе масштабных коэффициентов, *epart (n)*:

do for each scalefactor band n

epart (n) = 0

do for w = lower index w_low (n) к n = upper index w_high (n) epart (n) = epart (n) + r (w) ^2

end do

end do

порог для одной спектральной линии вычисляется согласно:

do for each threshold partition b

thr (all line_indices in this partition b) =

thr (w_low (b)..., w_high (b)), =nb (b) / (w_high (b) +1–w_low (b))

end do

уровень шума в полосе масштабных коэффициентов на уровне БПФ, *npart (n)* вычисляется как:

do for each scalefactor band n

npart (n) = minimum (thr (w_low (n))..., thr (w_high (n)))

** (w_high (n) +1–w_low (n)), end do*

Где в этом случае minimum (a... z) — функция, возвращающая самый меньший положительный аргумент из a... z.

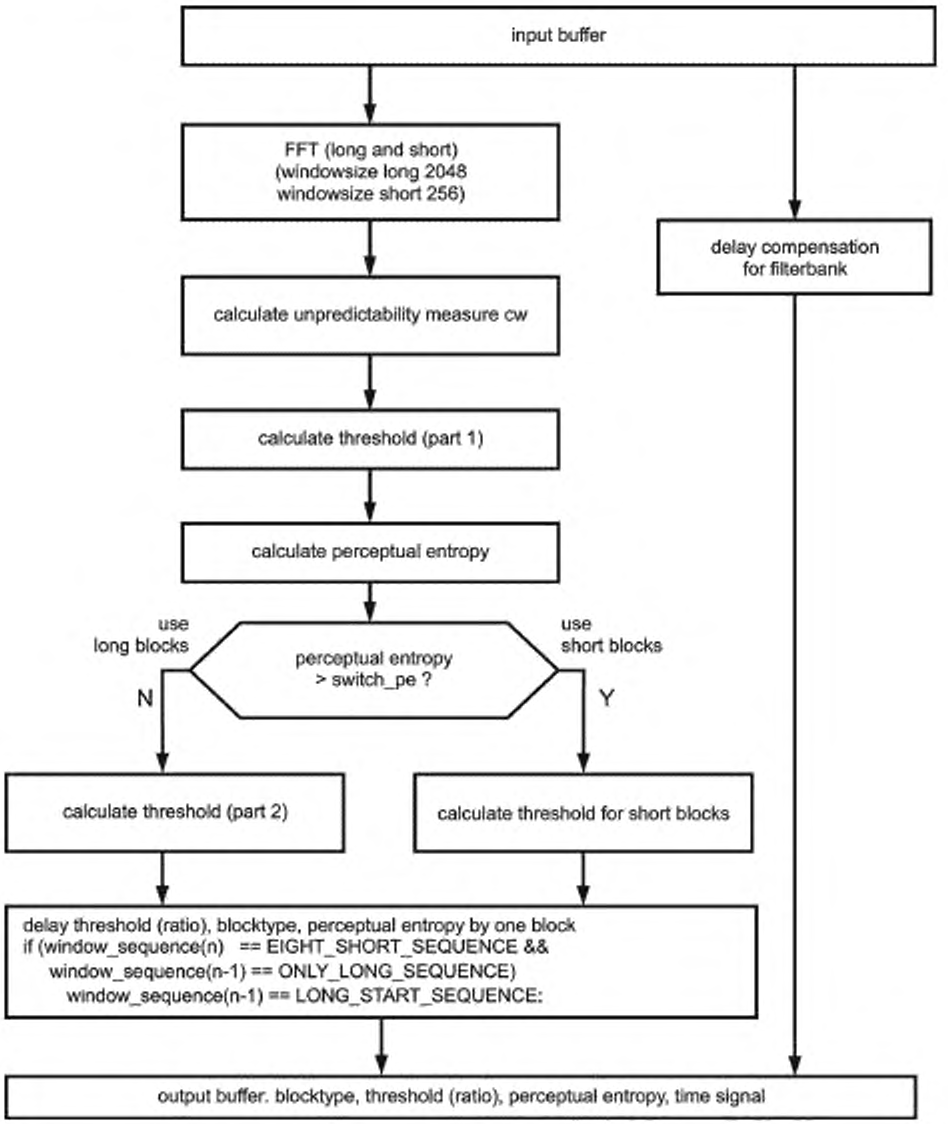


Рисунок В.1 — Блок-схема психоакустической модели

Отношения, которые будут переданы в модуль квантования, *SMR (n)*, вычисляются как:

SMR (n) = epart (n) / npart (n).

Для вычисления порогов кодера xmin (n) вычисляется энергия МДКП для каждой полосы масштабных коэффициентов:

Do for all scalefactor bands n
codec_e (n) = 0
do for lower index i to higher index i of this scalefactor band
codec_e (n) = codec_e (n) + (mdct_line (i)) ^2
end do
end do.

Затем *xmin (n)*, максимальная допустимая энергия искажений на уровне МДКП, может быть вычислена согласно этой формуле:

*xmin (n) = npart (n) * codec_e (n) / epart (n).*

15 Вычислить распределение битов из психоакустической энтропии (PE).

*bit_allocation = pew1*PE + pew2*sqrt (PE);*

для длинных блоков константы определяются как:

pew1 = 0,3, pew2 = 6,0

для коротких блоков PE восьми коротких блоков суммируются, и константы:

pew1 =0,6, pew2 = 24,

тогда *bit_allocation* ограничивается 0 <*bit_allocation* <3000, и вычисляется *more_bits:*

more_bits = bit_allocation – (mean_bits – side_info_bits).

Таблица В.1 — Психоакустические параметры для длинного БПФ, F_s = 8 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	8	9	0,18	46,82
1	9	17	9	0,53	46,82
2	18	26	9	0,89	46,82
3	27	35	9	1,24	41,82
4	36	44	9	1,59	41,82
5	45	53	9	1,94	41,82
6	54	62	9	2,29	38,82
7	63	71	9	2,63	38,82
8	72	80	9	2,98	38,82
9	81	89	9	3,31	33,82
10	90	98	9	3,65	33,82
11	99	108	10	3,99	34,28
12	109	118	10	4,35	32,28
13	119	128	10	4,71	32,28
14	129	138	10	5,05	32,28
15	139	148	10	5,39	32,28
16	149	159	11	5,74	32,69
17	160	170	11	6,10	32,69
18	171	181	11	6,45	32,69
19	182	192	11	6,79	32,69
20	193	204	12	7,13	33,07
21	205	216	12	7,48	33,07
22	217	228	12	7,82	33,07
23	229	241	13	8,17	33,42
24	242	254	13	8,51	33,42
25	255	268	14	8,85	33,74
26	269	282	14	9,20	33,74
27	283	297	15	9,54	34,04
28	298	312	15	9,88	34,04
29	313	328	16	10,22	34,32
30	329	345	17	10,56	34,58
31	346	363	18	10,91	34,83
32	364	381	18	11,25	34,83
33	382	400	19	11,58	35,06
34	401	420	20	11,91	35,29
35	421	441	21	12,24	35,50
36	442	464	23	12,58	35,89
37	465	488	24	12,92	36,08
38	489	514	26	13,26	36,43
39	515	541	27	13,59	36,59
40	542	570	29	13,93	36,90
41	571	601	31	14,26	37,19
42	602	634	33	14,60	37,46
43	635	670	36	14,93	37,84

Окончание таблицы В.1

Индекс	w_low	w_high	Ширина	bval	qsthr
44	671	708	38	15,27	38,07
45	709	749	41	15,60	38,40
46	750	793	44	15,93	38,71
47	794	841	48	16,26	39,09
48	842	893	52	16,60	39,44
49	894	949	56	16,93	39,76
50	950	1009	60	17,26	40,06
51	1010	1023	14	17,47	33,74

Таблица В.2 — Психоакустические параметры для короткого БПФ, F_s = 8 кГц

Индекс	w_low	w_high	Ширина	bval	qsthr
0	0	1	2	0,32	30,29
1	2	3	2	0,95	30,29
2	4	5	2	1,57	25,29
3	6	7	2	2,19	22,29
4	8	9	2	2,80	22,29
5	10	11	2	3,40	17,29
6	12	13	2	3,99	17,29
7	14	15	2	4,56	15,29
8	16	17	2	5,12	15,29
9	18	19	2	5,66	15,29
10	20	21	2	6,18	15,29
11	22	23	2	6,68	15,29
12	24	25	2	7,16	15,29
13	26	27	2	7,63	15,29
14	28	29	2	8,07	15,29
15	30	31	2	8,50	15,29
16	32	33	2	8,90	15,29
17	34	35	2	9,29	15,29
18	36	37	2	9,67	15,29
19	38	39	2	10,03	15,29
20	40	41	2	10,37	15,29
21	42	44	3	10,77	17,05
22	45	47	3	11,23	17,05
23	48	50	3	11,66	17,05
24	51	53	3	12,06	17,05
25	54	56	3	12,44	17,05
26	57	59	3	12,79	17,05
27	60	63	4	13,18	18,30
28	64	67	4	13,59	18,30
29	68	71	4	13,97	18,30
30	72	75	4	14,32	18,30
31	76	80	5	14,69	19,27
32	81	85	5	15,07	19,27
33	86	90	5	15,42	19,27
34	91	96	6	15,77	20,06
35	97	102	6	16,13	20,06
36	103	109	7	16,49	20,73
37	110	116	7	16,85	20,73
38	117	124	8	17,20	21,31
39	125	127	3	17,44	17,05

Таблица В.3 — Психоакустические параметры для длинного БПФ, F_s = 11,025 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	6	7	0,19	45,73
1	7	13	7	0,57	45,73
2	14	20	7	0,95	45,73
3	21	27	7	1,33	40,73
4	28	34	7	1,71	40,73
5	35	41	7	2,08	37,73
6	42	48	7	2,45	37,73
7	49	55	7	2,82	37,73
8	56	62	7	3,18	32,73
9	63	69	7	3,54	32,73
10	70	76	7	3,89	32,73
11	77	83	7	4,24	30,73
12	84	90	7	4,59	30,73
13	91	97	7	4,92	30,73
14	98	105	8	5,28	31,31
15	106	113	8	5,65	31,31
16	114	121	8	6,01	31,31
17	122	129	8	6,36	31,31
18	130	137	8	6,70	31,31
19	138	146	9	7,06	31,82
20	147	155	9	7,42	31,82
21	156	164	9	7,77	31,82
22	165	173	9	8,11	31,82
23	174	183	10	8,46	32,28
24	184	193	10	8,82	32,28
25	194	203	10	9,16	32,28
26	204	214	11	9,50	32,69
27	215	225	11	9,85	32,69
28	226	237	12	10,19	33,07
29	238	249	12	10,54	33,07
30	250	262	13	10,88	33,42
31	263	275	13	11,22	33,42
32	276	289	14	11,56	33,74
33	290	304	15	11,90	34,04
34	305	320	16	12,24	34,32
35	321	337	17	12,59	34,58
36	338	355	18	12,94	34,83
37	356	374	19	13,28	35,06
38	375	394	20	13,62	35,29
39	395	415	21	13,96	35,50
40	416	438	23	14,29	35,89
41	439	462	24	14,63	36,08
42	463	488	26	14,96	36,43
43	489	516	28	15,29	36,75
44	517	546	30	15,63	37,05
45	547	579	33	15,96	37,46
46	580	614	35	16,30	37,72
47	615	652	38	16,63	38,07
48	653	693	41	16,97	38,40
49	694	737	44	17,30	38,71
50	738	785	48	17,64	39,09
51	786	836	51	17,97	39,35
52	837	891	55	18,30	39,68
53	892	950	59	18,64	39,98
54	951	1014	64	18,97	40,34
55	1015	1023	9	19,16	31,82

Таблица В.4 — Психоакустические параметры для короткого БПФ, F_s = 11,025 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	0	1	0,00	27,28
1	1	1	1	0,44	27,28
2	2	2	1	0,87	27,28
3	3	3	1	1,30	22,28
4	4	4	1	1,73	22,28
5	5	5	1	2,16	19,28
6	6	6	1	2,58	19,28
7	7	7	1	3,00	14,28
8	8	8	1	3,41	14,28
9	9	9	1	3,82	14,28
10	10	10	1	4,22	12,28
11	11	11	1	4,61	12,28
12	12	12	1	4,99	12,28
13	13	13	1	5,37	12,28
14	14	14	1	5,74	12,28
15	15	15	1	6,10	12,28
16	16	16	1	6,45	12,28
17	17	17	1	6,79	12,28
18	18	19	2	7,44	15,29
19	20	21	2	8,05	15,29
20	22	23	2	8,64	15,29
21	24	25	2	9,19	15,29
22	26	27	2	9,70	15,29
23	28	29	2	10,19	15,29
24	30	31	2	10,65	15,29
25	32	33	2	11,08	15,29
26	34	35	2	11,48	15,29
27	36	37	2	11,86	15,29
28	38	39	2	12,22	15,29
29	40	42	3	12,64	17,05
30	43	45	3	13,10	17,05
31	46	48	3	13,53	17,05
32	49	51	3	13,93	17,05
33	52	54	3	14,30	17,05
34	55	58	4	14,69	18,30
35	59	62	4	15,11	18,30
36	63	66	4	15,49	18,30
37	67	70	4	15,84	18,30
38	71	75	5	16,21	19,27
39	76	80	5	16,58	19,27
40	81	85	5	16,92	19,27
41	86	91	6	17,27	20,06
42	92	97	6	17,62	20,06
43	98	104	7	17,97	20,73
44	105	111	7	18,32	20,73
45	112	119	8	18,67	21,31
46	120	127	8	19,02	21,31

Таблица В.5 — Психоакустические параметры для длинного БПФ, F_s = 12 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	5	6	0,18	45,06
1	6	11	6	0,53	45,06
2	12	17	6	0,89	45,06
3	18	23	6	1,24	40,06
4	24	29	6	1,59	40,06
5	30	35	6	1,94	40,06
6	36	41	6	2,29	37,06
7	42	47	6	2,63	37,06
8	48	53	6	2,98	37,06
9	54	59	6	3,31	32,06
10	60	65	6	3,65	32,06
11	66	72	7	4,00	30,73
12	73	79	7	4,38	30,73
13	80	86	7	4,75	30,73
14	87	93	7	5,11	30,73
15	94	100	7	5,47	30,73
16	101	107	7	5,82	30,73
17	108	114	7	6,15	30,73
18	115	122	8	6,51	31,31
19	123	130	8	6,88	31,31
20	131	138	8	7,24	31,31
21	139	146	8	7,58	31,31
22	147	154	8	7,92	31,31
23	155	163	9	8,27	31,82
24	164	172	9	8,62	31,82
25	173	181	9	8,96	31,82
26	182	191	10	9,31	32,28
27	192	201	10	9,66	32,28
28	202	212	11	10,01	32,69
29	213	223	11	10,36	32,69
30	224	235	12	10,71	33,07
31	236	247	12	11,06	33,07
32	248	260	13	11,41	33,42
33	261	273	13	11,75	33,42
34	274	287	14	12,09	33,74
35	288	302	15	12,43	34,04
36	303	318	16	12,77	34,32
37	319	335	17	13,11	34,58
38	336	353	18	13,46	34,83
39	354	372	19	13,80	35,06
40	373	392	20	14,13	35,29
41	393	414	22	14,47	35,70
42	415	437	23	14,81	35,89
43	438	462	25	15,14	36,26
44	463	489	27	15,48	36,59
45	490	518	29	15,81	36,90
46	519	549	31	16,15	37,19
47	550	583	34	16,48	37,59
48	584	619	36	16,82	37,84
49	620	658	39	17,15	38,19
50	659	700	42	17,48	38,51
51	701	745	45	17,81	38,81
52	746	794	49	18,14	39,18
53	795	847	53	18,48	39,52
54	848	904	57	18,81	39,83
55	905	965	61	19,15	40,13
56	966	1023	58	19,47	39,91

Таблица В.6 — Психоакустические параметры для короткого БПФ, F_s = 12 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	0	1	0,00	27,28
1	1	1	1	0,47	27,28
2	2	2	1	0,95	27,28
3	3	3	1	1,42	22,28
4	4	4	1	1,88	22,28
5	5	5	1	2,35	19,28
6	6	6	1	2,81	19,28
7	7	7	1	3,26	14,28
8	8	8	1	3,70	14,28
9	9	9	1	4,14	12,28
10	10	10	1	4,57	12,28
11	11	11	1	4,98	12,28
12	12	12	1	5,39	12,28
13	13	13	1	5,79	12,28
14	14	14	1	6,18	12,28
15	15	15	1	6,56	12,28
16	16	16	1	6,93	12,28
17	17	17	1	7,28	12,28
18	18	18	1	7,63	12,28
19	19	20	2	8,28	15,29
20	21	22	2	8,90	15,29
21	23	24	2	9,48	15,29
22	25	26	2	10,02	15,29
23	27	28	2	10,53	15,29
24	29	30	2	11,00	15,29
25	31	32	2	11,45	15,29
26	33	34	2	11,86	15,29
27	35	36	2	12,25	15,29
28	37	38	2	12,62	15,29
29	39	40	2	12,96	15,29
30	41	43	3	13,36	17,05
31	44	46	3	13,80	17,05
32	47	49	3	14,21	17,05
33	50	52	3	14,59	17,05
34	53	55	3	14,94	17,05
35	56	59	4	15,32	18,30
36	60	63	4	15,71	18,30
37	64	67	4	16,08	18,30
38	68	72	5	16,45	19,27
39	73	77	5	16,83	19,27
40	78	82	5	17,19	19,27
41	83	88	6	17,54	20,06
42	89	94	6	17,90	20,06
43	95	101	7	18,26	20,73
44	102	108	7	18,62	20,73
45	109	116	8	18,97	21,31
46	117	124	8	19,32	21,31
47	125	127	3	19,55	17,05

Таблица В.7 — Психоакустические параметры для длинного БПФ, F_s = 16 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	4	5	0,20	43,30
1	5	9	5	0,59	43,10
2	10	14	5	0,99	38,30
3	15	19	5	1,38	38,10
4	20	24	5	1,77	38,00
5	25	29	5	2,16	35,10
6	30	34	5	2,54	35,30
7	35	39	5	2,92	30,00
8	40	44	5	3,29	30,00
9	45	49	5	3,66	28,30
10	50	54	5	4,03	28,30
11	55	59	5	4,39	28,30
12	60	64	5	4,74	28,30
13	65	69	5	5,09	28,30
14	70	74	5	5,43	28,30
15	75	80	6	5,79	28,30
16	81	86	6	6,18	28,30
17	87	92	6	6,56	28,00
18	93	98	6	6,92	29,27
19	99	104	6	7,28	29,27
20	105	110	6	7,63	29,27
21	111	116	6	7,96	29,27
22	117	123	7	8,31	29,27
23	124	130	7	8,68	29,06
24	131	137	7	9,03	30,06
25	138	144	7	9,37	30,06
26	145	152	8	9,71	30,06
27	153	160	8	10,07	30,73
28	161	168	8	10,41	30,73
29	169	177	9	10,75	30,73
30	178	186	9	11,10	31,31
31	187	196	10	11,45	31,31
32	197	206	10	11,80	31,82
33	207	217	11	12,14	31,82
34	218	228	11	12,48	32,28
35	229	240	12	12,82	32,28
36	241	253	13	13,16	32,69
37	254	267	14	13,51	32,69
38	268	282	15	13,86	33,07
39	283	298	16	14,21	33,46
40	299	315	17	14,56	33,82
41	316	333	18	14,90	34,12
42	334	352	19	15,24	34,42
43	353	373	21	15,58	34,68
44	374	395	22	15,91	35,15
45	396	419	24	16,25	35,32
46	420	445	26	16,58	35,73
47	446	473	28	16,92	35,91
48	474	503	30	17,25	36,42
49	504	536	33	17,59	36,75
50	537	571	35	17,93	37,11
51	572	609	38	18,26	37,34
52	610	650	41	18,60	37,63
53	651	694	44	18,94	38,12
54	695	741	47	19,27	38,17
55	742	791	50	19,60	41,52
56	792	845	54	19,94	41,84
57	846	903	58	20,27	42,13
58	904	965	62	20,61	44,41
59	966	1023	58	20,92	44,87

Таблица В.8 — Психоакустические параметры для короткого БПФ, F_s = 16 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	0	1	0,00	27,28
1	1	1	1	0,63	27,28
2	2	2	1	1,26	22,28
3	3	3	1	1,88	22,28
4	4	4	1	2,50	19,28
5	5	5	1	3,11	14,28
6	6	6	1	3,70	14,28
7	7	7	1	4,28	12,28
8	8	8	1	4,85	12,28
9	9	9	1	5,39	12,28
10	10	10	1	5,92	12,28
11	11	11	1	6,43	12,28
12	12	12	1	6,93	12,28
13	13	13	1	7,40	12,28
14	14	14	1	7,85	12,28
15	15	15	1	8,29	12,28
16	16	16	1	8,70	12,28
17	17	17	1	9,10	12,28
18	18	18	1	9,49	12,28
19	19	19	1	9,85	12,28
20	20	20	1	10,20	12,28
21	21	22	2	10,85	15,29
22	23	24	2	11,44	15,29
23	25	26	2	11,99	15,29
24	27	28	2	12,50	15,29
25	29	30	2	12,96	15,29
26	31	32	2	13,39	15,29
27	33	34	2	13,78	15,29
28	35	36	2	14,15	15,29
29	37	39	3	14,57	17,05
30	40	42	3	15,03	17,05
31	43	45	3	15,45	17,05
32	46	48	3	15,84	17,05
33	49	51	3	16,19	17,05
34	52	55	4	16,57	18,30
35	56	59	4	16,97	18,30
36	60	63	4	17,33	18,30
37	64	68	5	17,71	19,27
38	69	73	5	18,09	19,27
39	74	78	5	18,44	19,27
40	79	84	6	18,80	20,06
41	85	90	6	19,17	20,06
42	91	97	7	19,53	20,73
43	98	104	7	19,89	20,73
44	105	112	8	20,25	24,31
45	113	120	8	20,61	24,31
46	121	127	7	20,92	23,73

Таблица В.9 — Психоакустические параметры для длинного БПФ, F_s = 22,05 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	3	4	0,22	43,30
1	4	7	4	0,65	43,30
2	8	11	4	1,09	38,30
3	12	15	4	1,52	38,30
4	16	19	4	1,95	38,30
5	20	23	4	2,37	35,30
6	24	27	4	2,79	35,30
7	28	31	4	3,21	30,30
8	32	35	4	3,62	30,30
9	36	39	4	4,02	28,30
10	40	43	4	4,41	28,30
11	44	47	4	4,80	28,30
12	48	51	4	5,18	28,30
13	52	55	4	5,55	28,30
14	56	59	4	5,92	28,30
15	60	63	4	6,27	28,30
16	64	67	4	6,62	28,30
17	68	71	4	6,95	28,30
18	72	76	5	7,32	29,27
19	77	81	5	7,71	29,27
20	82	86	5	8,10	29,27
21	87	91	5	8,46	29,27
22	92	96	5	8,82	29,27
23	97	101	5	9,16	29,27
24	102	107	6	9,52	30,06
25	108	113	6	9,89	30,06
26	114	119	6	10,25	30,06
27	120	125	6	10,59	30,06
28	126	132	7	10,95	30,73
29	133	139	7	11,31	30,73
30	140	146	7	11,65	30,73
31	147	154	8	12,00	31,31
32	155	162	8	12,35	31,31
33	163	171	9	12,70	31,82
34	172	180	9	13,05	31,82
35	181	190	10	13,40	32,28
36	191	200	10	13,74	32,28
37	201	211	11	14,07	32,69
38	212	223	12	14,41	33,07
39	224	236	13	14,76	33,42
40	237	250	14	15,11	33,74
41	251	265	15	15,46	34,04
42	266	281	16	15,80	34,32
43	282	298	17	16,14	34,58
44	299	317	19	16,48	35,06
45	318	337	20	16,82	35,29
46	338	359	22	17,16	35,70
47	360	382	23	17,50	35,89
48	383	407	25	17,84	36,26
49	408	434	27	18,17	36,59
50	435	463	29	18,51	36,90
51	464	494	31	18,84	37,19
52	495	527	33	19,17	37,46
53	528	563	36	19,51	37,84
54	564	601	38	19,84	38,07
55	602	642	41	20,17	41,40
56	643	686	44	20,50	41,71
57	687	733	47	20,84	42,00

Окончание таблицы В.9

Индекс	w_low	w_high	Ширина	bval	qsthr
58	734	784	51	21,17	44,35
59	785	839	55	21,50	44,68
60	840	898	59	21,84	44,98
61	899	962	64	22,17	50,34
62	963	1023	61	22,48	50,13

Таблица В.10 — Психоакустические параметры для короткого БПФ, F_s = 22,05 кГц

Индекс	w_low	w_high	Ширина	bval	qsthr
0	0	0	1	0,00	27,28
1	1	1	1	0,87	27,28
2	2	2	1	1,73	22,28
3	3	3	1	2,58	19,28
4	4	4	1	3,41	14,28
5	5	5	1	4,22	12,28
6	6	6	1	4,99	12,28
7	7	7	1	5,74	12,28
8	8	8	1	6,45	12,28
9	9	9	1	7,12	12,28
10	10	10	1	7,75	12,28
11	11	11	1	8,36	12,28
12	12	12	1	8,92	12,28
13	13	13	1	9,45	12,28
14	14	14	1	9,96	12,28
15	15	15	1	10,43	12,28
16	16	16	1	10,87	12,28
17	17	17	1	11,29	12,28
18	18	18	1	11,68	12,28
19	19	19	1	12,05	12,28
20	20	21	2	12,71	15,29
21	22	23	2	13,32	15,29
22	24	25	2	13,86	15,29
23	26	27	2	14,35	15,29
24	28	29	2	14,80	15,29
25	30	31	2	15,21	15,29
26	32	33	2	15,58	15,29
27	34	35	2	15,93	15,29
28	36	38	3	16,32	17,05
29	39	41	3	16,75	17,05
30	42	44	3	17,15	17,05
31	45	47	3	17,51	17,05
32	48	51	4	17,89	18,30
33	52	55	4	18,30	18,30
34	56	59	4	18,67	18,30
35	60	63	4	19,02	18,30
36	64	68	5	19,37	19,27
37	69	73	5	19,74	19,27
38	74	78	5	20,09	22,27
39	79	84	6	20,44	23,06
40	85	90	6	20,79	23,06
41	91	97	7	21,15	25,73
42	98	104	7	21,50	25,73
43	105	112	8	21,85	26,31
44	113	120	8	22,20	31,31
45	121	127	7	22,49	30,73

Таблица В.11 — Психоакустические параметры для длинного БПФ, F_s = 24 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	2	3	0,18	42,05
1	3	5	3	0,53	42,05
2	6	8	3	0,89	42,05
3	9	11	3	1,24	37,05
4	12	14	3	1,59	37,05
5	15	17	3	1,94	37,05
6	18	20	3	2,29	34,05
7	21	23	3	2,63	34,05
8	24	26	3	2,98	34,05
9	27	29	3	3,31	29,05
10	30	32	3	3,65	29,05
11	33	36	4	4,03	28,30
12	37	40	4	4,46	28,30
13	41	44	4	4,88	28,30
14	45	48	4	5,29	28,30
15	49	52	4	5,69	28,30
16	53	56	4	6,08	28,30
17	57	60	4	6,46	28,30
18	61	64	4	6,83	28,30
19	65	68	4	7,19	28,30
20	69	72	4	7,54	28,30
21	73	76	4	7,88	28,30
22	77	81	5	8,25	29,27
23	82	86	5	8,64	29,27
24	87	91	5	9,02	29,27
25	92	96	5	9,38	29,27
26	97	101	5	9,73	29,27
27	102	107	6	10,09	30,06
28	108	113	6	10,47	30,06
29	114	119	6	10,83	30,06
30	120	125	6	11,18	30,06
31	126	132	7	11,53	30,73
32	133	139	7	11,89	30,73
33	140	146	7	12,23	30,73
34	147	154	8	12,57	31,31
35	155	162	8	12,92	31,31
36	163	171	9	13,26	31,82
37	172	180	9	13,61	31,82
38	181	190	10	13,95	32,28
39	191	201	11	14,29	32,69
40	202	213	12	14,65	33,07
41	214	225	12	15,00	33,07
42	226	238	13	15,33	33,42
43	239	252	14	15,66	33,74
44	253	267	15	16,00	34,04
45	268	284	17	16,34	34,58
46	285	302	18	16,69	34,83
47	303	321	19	17,02	35,06
48	322	342	21	17,36	35,50
49	343	364	22	17,70	35,70
50	365	388	24	18,03	36,08
51	389	414	26	18,37	36,43
52	415	442	28	18,70	36,75
53	443	472	30	19,04	37,05
54	473	504	32	19,38	37,33
55	505	538	34	19,71	37,59
56	539	575	37	20,04	40,96
57	576	614	39	20,38	41,19
58	615	656	42	20,71	41,51

Окончание таблицы В.11

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
59	657	701	45	21,04	43,81
60	702	750	49	21,37	44,18
61	751	803	53	21,70	44,52
62	804	860	57	22,04	49,83
63	861	922	62	22,37	50,20
64	923	989	67	22,70	50,54
65	990	1023	34	22,95	47,59

Таблица В.12 — Психоакустические параметры для короткого БПФ, F_s = 24 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	0	1	0,00	27,28
1	1	1	1	0,95	27,28
2	2	2	1	1,88	22,28
3	3	3	1	2,81	19,28
4	4	4	1	3,70	14,28
5	5	5	1	4,57	12,28
6	6	6	1	5,39	12,28
7	7	7	1	6,18	12,28
8	8	8	1	6,93	12,28
9	9	9	1	7,63	12,28
10	10	10	1	8,29	12,28
11	11	11	1	8,91	12,28
12	12	12	1	9,49	12,28
13	13	13	1	10,03	12,28
14	14	14	1	10,53	12,28
15	15	15	1	11,01	12,28
16	16	16	1	11,45	12,28
17	17	17	1	11,87	12,28
18	18	18	1	12,26	12,28
19	19	19	1	12,62	12,28
20	20	21	2	13,28	15,29
21	22	23	2	13,87	15,29
22	24	25	2	14,40	15,29
23	26	27	2	14,88	15,29
24	28	29	2	15,32	15,29
25	30	31	2	15,71	15,29
26	32	33	2	16,08	15,29
27	34	36	3	16,49	17,05
28	37	39	3	16,94	17,05
29	40	42	3	17,35	17,05
30	43	45	3	17,73	17,05
31	46	48	3	18,07	17,05
32	49	52	4	18,44	18,30
33	53	56	4	18,83	18,30
34	57	60	4	19,20	18,30
35	61	65	5	19,57	19,27
36	66	70	5	19,96	19,27
37	71	75	5	20,31	22,27
38	76	81	6	20,67	23,06
39	82	87	6	21,04	25,06
40	88	94	7	21,41	25,73
41	95	101	7	21,77	25,73
42	102	109	8	22,13	31,31
43	110	117	8	22,48	31,31
44	118	126	9	22,82	31,82
45	127	127	1	23,01	32,28

Таблица В.13— Психоакустические параметры для длинного БПФ, F_s = 32 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	2	3	0,24	42,05
1	3	5	3	0,71	42,05
2	6	8	3	1,18	37,05
3	9	11	3	1,65	37,05
4	12	14	3	2,12	34,05
5	15	17	3	2,58	34,05
6	18	20	3	3,03	29,05
7	21	23	3	3,48	29,05
8	24	26	3	3,92	29,05
9	27	29	3	4,35	27,05
10	30	32	3	4,77	27,05
11	33	35	3	5,19	27,05
12	36	38	3	5,59	27,05
13	39	41	3	5,99	27,05
14	42	44	3	6,37	27,05
15	45	47	3	6,74	27,05
16	48	50	3	7,10	27,05
17	51	53	3	7,45	27,05
18	54	56	3	7,80	27,05
19	57	60	4	8,18	28,30
20	61	64	4	8,60	28,30
21	65	68	4	9,00	28,30
22	69	72	4	9,39	28,30
23	73	76	4	9,76	28,30
24	77	80	4	10,11	28,30
25	81	84	4	10,45	28,30
26	85	89	5	10,81	29,27
27	90	94	5	11,19	29,27
28	95	99	5	11,55	29,27
29	100	104	5	11,90	29,27
30	105	110	6	12,25	30,06
31	111	116	6	12,62	30,06
32	117	122	6	12,96	30,06
33	123	129	7	13,31	30,73
34	130	136	7	13,66	30,73
35	137	144	8	14,01	31,31
36	145	152	8	14,36	31,31
37	153	161	9	14,71	31,82
38	162	171	10	15,07	32,28
39	172	181	10	15,42	32,28
40	182	192	11	15,76	32,69
41	193	204	12	16,10	33,07
42	205	217	13	16,45	33,42
43	218	231	14	16,80	33,74
44	232	246	15	17,14	34,04
45	247	262	16	17,48	34,32
46	263	279	17	17,82	34,58
47	280	298	19	18,15	35,06
48	299	318	20	18,49	35,29
49	319	340	22	18,84	35,70
50	341	363	23	19,17	35,89
51	364	388	25	19,51	36,26
52	389	415	27	19,85	36,59
53	416	444	29	20,19	39,90
54	445	475	31	20,53	40,19
55	476	508	33	20,87	40,46
56	509	543	35	21,20	42,72
57	544	581	38	21,53	43,07
58	582	622	41	21,86	43,40

Окончание таблицы В.13

Индекс	w_low	w_high	Ширина	bval	qsthr
59	623	667	45	22,20	48,81
60	668	715	48	22,53	49,09
61	716	768	53	22,86	49,52
62	769	826	58	23,20	59,91
63	827	890	64	23,53	60,34
64	891	961	71	23,86	60,79
65	962	1023	62	24,00	65,89

Таблица В.14 — Психоакустические параметры для короткого БПФ, F_s = 32 кГц

Индекс	w_low	w_high	Ширина	bval	qsthr
0	0	0	1	0,00	27,28
1	1	1	1	1,26	22,28
2	2	2	1	2,50	19,28
3	3	3	1	3,70	14,28
4	4	4	1	4,85	12,28
5	5	5	1	5,92	12,28
6	6	6	1	6,93	12,28
7	7	7	1	7,85	12,28
8	8	8	1	8,70	12,28
9	9	9	1	9,49	12,28
10	10	10	1	10,20	12,28
11	11	11	1	10,85	12,28
12	12	12	1	11,45	12,28
13	13	13	1	12,00	12,28
14	14	14	1	12,50	12,28
15	15	15	1	12,96	12,28
16	16	16	1	13,39	12,28
17	17	17	1	13,78	12,28
18	18	18	1	14,15	12,28
19	19	20	2	14,80	15,29
20	21	22	2	15,38	15,29
21	23	24	2	15,89	15,29
22	25	26	2	16,36	15,29
23	27	28	2	16,77	15,29
24	29	30	2	17,15	15,29
25	31	32	2	17,50	15,29
26	33	35	3	17,90	17,05
27	36	38	3	18,34	17,05
28	39	41	3	18,74	17,05
29	42	44	3	19,11	17,05
30	45	48	4	19,50	18,30
31	49	52	4	19,92	18,30
32	53	56	4	20,30	21,30
33	57	60	4	20,65	21,30
34	61	65	5	21,02	24,27
35	66	70	5	21,40	24,27
36	71	75	5	21,75	24,27
37	76	81	6	22,10	30,06
38	82	87	6	22,45	30,06
39	88	94	7	22,80	30,73
40	95	102	8	23,16	41,31
41	103	110	8	23,51	41,31
42	111	119	9	23,85	41,82
43	120	127	8	24,00	60,47

Таблица В.15 — Психоакустические параметры для длинного БПФ, F_s = 44,1 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	1	2	0,22	40,29
1	2	3	2	0,65	40,29
2	4	5	2	1,09	35,29
3	6	7	2	1,52	35,29
4	8	9	2	1,95	35,29
5	10	11	2	2,37	32,29
6	12	13	2	2,79	32,29
7	14	15	2	3,21	27,29
8	16	17	2	3,62	27,29
9	18	19	2	4,02	25,29
10	20	21	2	4,41	25,29
11	22	23	2	4,80	25,29
12	24	25	2	5,18	25,29
13	26	27	2	5,55	25,29
14	28	29	2	5,92	25,29
15	30	31	2	6,27	25,29
16	32	33	2	6,62	25,29
17	34	35	2	6,95	25,29
18	36	38	3	7,36	27,05
19	39	41	3	7,83	27,05
20	42	44	3	8,28	27,05
21	45	47	3	8,71	27,05
22	48	50	3	9,12	27,05
23	51	53	3	9,52	27,05
24	54	56	3	9,89	27,05
25	57	59	3	10,25	27,05
26	60	62	3	10,59	27,05
27	63	66	4	10,97	28,30
28	67	70	4	11,38	28,30
29	71	74	4	11,77	28,30
30	75	78	4	12,13	28,30
31	79	82	4	12,48	28,30
32	83	87	5	12,84	29,27
33	88	92	5	13,22	29,27
34	93	97	5	13,57	29,27
35	98	103	6	13,93	30,06
36	104	109	6	14,30	30,06
37	110	116	7	14,67	30,73
38	117	123	7	15,03	30,73
39	124	131	8	15,40	31,31
40	132	139	8	15,76	31,31
41	140	148	9	16,11	31,82
42	149	157	9	16,45	31,82
43	158	167	10	16,79	32,28
44	168	178	11	17,13	32,69
45	179	190	12	17,48	33,07
46	191	203	13	17,83	33,42
47	204	217	14	18,18	33,74
48	218	232	15	18,52	34,04
49	233	248	16	18,87	34,32
50	249	265	17	19,21	34,58
51	266	283	18	19,54	34,83
52	284	303	20	19,88	35,29
53	304	324	21	20,22	38,50
54	325	347	23	20,56	38,89
55	348	371	24	20,90	39,08
56	372	397	26	21,24	41,43
57	398	425	28	21,57	41,75
58	426	455	30	21,91	42,05

Окончание таблицы В.15

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
59	456	488	33	22,24	47,46
60	489	524	36	22,58	47,84
61	525	563	39	22,91	48,19
62	564	606	43	23,25	58,61
63	607	653	47	23,58	59,00
64	654	706	53	23,91	59,52
65	707	765	59	24,00	69,98
66	766	832	67	24,00	70,54
67	833	908	76	24,00	71,08
68	909	996	88	24,00	71,72
69	997	1023	27	24,00	72,09

Таблица В.16 — Психоакустические параметры для короткого БПФ, F_s = 44,1 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	0	1	0,00	27,28
1	1	1	1	1,73	22,28
2	2	2	1	3,41	14,28
3	3	3	1	4,99	12,28
4	4	4	1	6,45	12,28
5	5	5	1	7,75	12,28
6	6	6	1	8,92	12,28
7	7	7	1	9,96	12,28
8	8	8	1	10,87	12,28
9	9	9	1	11,68	12,28
10	10	10	1	12,39	12,28
11	11	11	1	13,03	12,28
12	12	12	1	13,61	12,28
13	13	13	1	14,12	12,28
14	14	14	1	14,59	12,28
15	15	15	1	15,01	12,28
16	16	16	1	15,40	12,28
17	17	17	1	15,76	12,28
18	18	19	2	16,39	15,29
19	20	21	2	16,95	15,29
20	22	23	2	17,45	15,29
21	24	25	2	17,89	15,29
22	26	27	2	18,30	15,29
23	28	29	2	18,67	15,29
24	30	31	2	19,02	15,29
25	32	34	3	19,41	17,05
26	35	37	3	19,85	17,05
27	38	40	3	20,25	20,05
28	41	43	3	20,62	20,05
29	44	47	4	21,01	23,30
30	48	51	4	21,43	23,30
31	52	55	4	21,81	23,30
32	56	59	4	22,15	28,30
33	60	64	5	22,51	29,27
34	65	69	5	22,87	29,27
35	70	75	6	23,23	40,06
36	76	81	6	23,59	40,06
37	82	88	7	23,93	40,73
38	89	96	8	24,00	51,31
39	97	105	9	24,00	51,82
40	106	115	10	24,00	52,28
41	116	127	12	24,00	53,07

Таблица В.17 — Психоакустические параметры для длинного БПФ, F_s = 48 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	1	2	0,24	40,29
1	2	3	2	0,71	40,29
2	4	5	2	1,18	35,29
3	6	7	2	1,65	35,29
4	8	9	2	2,12	32,29
5	10	11	2	2,58	32,29
6	12	13	2	3,03	27,29
7	14	15	2	3,48	27,29
8	16	17	2	3,92	27,29
9	18	19	2	4,35	25,29
10	20	21	2	4,77	25,29
11	22	23	2	5,19	25,29
12	24	25	2	5,59	25,29
13	26	27	2	5,99	25,29
14	28	29	2	6,37	25,29
15	30	31	2	6,74	25,29
16	32	33	2	7,10	25,29
17	34	35	2	7,45	25,29
18	36	37	2	7,80	25,29
19	38	40	3	8,20	27,05
20	41	43	3	8,68	27,05
21	44	46	3	9,13	27,05
22	47	49	3	9,55	27,05
23	50	52	3	9,96	27,05
24	53	55	3	10,35	27,05
25	56	58	3	10,71	27,05
26	59	61	3	11,06	27,05
27	62	65	4	11,45	28,30
28	66	69	4	11,86	28,30
29	70	73	4	12,25	28,30
30	74	77	4	12,62	28,30
31	78	81	4	12,96	28,30
32	82	86	5	13,32	29,27
33	87	91	5	13,70	29,27
34	92	96	5	14,05	29,27
35	97	102	6	14,41	30,06
36	103	108	6	14,77	30,06
37	109	115	7	15,13	30,73
38	116	122	7	15,49	30,73
39	123	130	8	15,85	31,31
40	131	138	8	16,20	31,31
41	139	147	9	16,55	31,82
42	148	157	10	16,91	32,28
43	158	167	10	17,25	32,28
44	168	178	11	17,59	32,69
45	179	190	12	17,93	33,07
46	191	203	13	18,28	33,42
47	204	217	14	18,62	33,74
48	218	232	15	18,96	34,04
49	233	248	16	19,30	34,32
50	249	265	17	19,64	34,58
51	266	283	18	19,97	34,83
52	284	303	20	20,31	38,29
53	304	324	21	20,65	38,50
54	325	347	23	20,99	38,89
55	348	371	24	21,33	41,08
56	372	397	26	21,66	41,43
57	398	425	28	21,99	41,75

Окончание таблицы В.17

Индекс	w_low	w_high	Ширина	bval	qsthr
58	426	456	31	22,32	47,19
59	457	490	34	22,66	47,59
60	491	527	37	23,00	47,96
61	528	567	40	23,33	58,30
62	568	612	45	23,67	58,81
63	613	662	50	24,00	69,27
64	663	718	56	24,00	69,76
65	719	781	63	24,00	70,27
66	782	853	72	24,00	70,85
67	854	937	84	24,00	71,52
68	938	1023	86	24,00	70,20

Таблица В.18 — Психоакустические параметры для короткого БПФ, F_s = 48 кГц

Индекс	w_low	w_high	Ширина	bval	qsthr
0	0	0	1	0,00	27,28
1	1	1	1	1,88	22,28
2	2	2	1	3,70	14,28
3	3	3	1	5,39	12,28
4	4	4	1	6,93	12,28
5	5	5	1	8,29	12,28
6	6	6	1	9,49	12,28
7	7	7	1	10,53	12,28
8	8	8	1	11,45	12,28
9	9	9	1	12,26	12,28
10	10	10	1	12,96	12,28
11	11	11	1	13,59	12,28
12	12	12	1	14,15	12,28
13	13	13	1	14,65	12,28
14	14	14	1	15,11	12,28
15	15	15	1	15,52	12,28
16	16	16	1	15,90	12,28
17	17	18	2	16,56	15,29
18	19	20	2	17,15	15,29
19	21	22	2	17,66	15,29
20	23	24	2	18,13	15,29
21	25	26	2	18,54	15,29
22	27	28	2	18,93	15,29
23	29	30	2	19,28	15,29
24	31	33	3	19,69	17,05
25	34	36	3	20,14	20,05
26	37	39	3	20,54	20,05
27	40	42	3	20,92	20,05
28	43	45	3	21,27	22,05
29	46	49	4	21,64	23,30
30	50	53	4	22,03	28,30
31	54	57	4	22,39	28,30
32	58	62	5	22,76	29,27
33	63	67	5	23,13	39,27
34	68	73	6	23,49	40,06
35	74	79	6	23,85	40,06
36	80	86	7	24,00	50,73
37	87	94	8	24,00	51,31
38	95	103	9	24,00	51,82
39	104	113	10	24,00	52,28
40	114	125	12	24,00	53,07
41	126	127	1	24,00	53,07

Таблица В.19 — Психоакустические параметры для длинного БПФ, F_s = 64 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	1	2	0,32	40,29
1	2	3	2	0,95	40,29
2	4	5	2	1,57	35,29
3	6	7	2	2,19	32,29
4	8	9	2	2,80	32,29
5	10	11	2	3,40	27,29
6	12	13	2	3,99	27,29
7	14	15	2	4,56	25,29
8	16	17	2	5,12	25,29
9	18	19	2	5,66	25,29
10	20	21	2	6,18	25,29
11	22	23	2	6,68	25,29
12	24	25	2	7,16	25,29
13	26	27	2	7,63	25,29
14	28	29	2	8,07	25,29
15	30	31	2	8,50	25,29
16	32	33	2	8,90	25,29
17	34	35	2	9,29	25,29
18	36	37	2	9,67	25,29
19	38	39	2	10,03	25,29
20	40	41	2	10,37	25,29
21	42	44	3	10,77	27,05
22	45	47	3	11,23	27,05
23	48	50	3	11,66	27,05
24	51	53	3	12,06	27,05
25	54	56	3	12,44	27,05
26	57	59	3	12,79	27,05
27	60	63	4	13,18	28,30
28	64	67	4	13,59	28,30
29	68	71	4	13,97	28,30
30	72	75	4	14,32	28,30
31	76	80	5	14,69	29,27
32	81	85	5	15,07	29,27
33	86	90	5	15,42	29,27
34	91	96	6	15,77	30,06
35	97	102	6	16,13	30,06
36	103	109	7	16,49	30,73
37	110	116	7	16,85	30,73
38	117	124	8	17,20	31,31
39	125	132	8	17,54	31,31
40	133	141	9	17,88	31,82
41	142	151	10	18,23	32,28
42	152	161	10	18,58	32,28
43	162	172	11	18,91	32,69
44	173	184	12	19,25	33,07
45	185	197	13	19,60	33,42
46	198	211	14	19,94	33,74
47	212	226	15	20,29	37,04
48	227	242	16	20,63	37,32
49	243	259	17	20,97	37,58
50	260	277	18	21,31	39,83
51	278	297	20	21,64	40,29
52	298	318	21	21,98	40,50
53	319	341	23	22,31	45,89
54	342	366	25	22,65	46,26
55	367	394	28	22,98	46,75
56	395	424	30	23,32	57,05

Окончание таблицы В.19

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
57	425	458	34	23,66	57,59
58	459	495	37	23,99	57,96
59	496	537	42	24,00	68,51
60	538	584	47	24,00	69,00
61	585	638	54	24,00	69,60
62	639	701	63	24,00	70,27
63	702	774	73	24,00	70,91
64	775	861	87	24,00	71,67
65	862	966	105	24,00	72,49
66	967	1023	57	24,00	69,83

Таблица В.20 — Психоакустические параметры для короткого БПФ, F_s = 64 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	0	1	0,00	27,28
1	1	1	1	2,50	19,28
2	2	2	1	4,85	12,28
3	3	3	1	6,93	12,28
4	4	4	1	8,70	12,28
5	5	5	1	10,20	12,28
6	6	6	1	11,45	12,28
7	7	7	1	12,50	12,28
8	8	8	1	13,39	12,28
9	9	9	1	14,15	12,28
10	10	10	1	14,81	12,28
11	11	11	1	15,39	12,28
12	12	12	1	15,90	12,28
13	13	13	1	16,36	12,28
14	14	14	1	16,78	12,28
15	15	15	1	17,16	12,28
16	16	17	2	17,82	15,29
17	18	19	2	18,40	15,29
18	20	21	2	18,92	15,29
19	22	23	2	19,39	15,29
20	24	25	2	19,82	15,29
21	26	27	2	20,21	18,29
22	28	29	2	20,57	18,29
23	30	32	3	20,98	20,05
24	33	35	3	21,43	22,05
25	36	38	3	21,84	22,05
26	39	41	3	22,22	27,05
27	42	45	4	22,61	28,30
28	46	49	4	23,02	38,30
29	50	53	4	23,39	38,30
30	54	58	5	23,75	39,27
31	59	63	5	24,00	49,27
32	64	69	6	24,00	50,06
33	70	76	7	24,00	50,73
34	77	84	8	24,00	51,31
35	85	93	9	24,00	51,82
36	94	104	11	24,00	52,69
37	105	117	13	24,00	53,42
38	118	127	10	24,00	52,28

Таблица В.21 — Психоакустические параметры для длинного БПФ, F_s = 88,2 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	0	1	0,00	37,28
1	1	1	1	0,44	37,28
2	2	2	1	0,87	37,28
3	3	3	1	1,30	32,28
4	4	4	1	1,73	32,28
5	5	5	1	2,16	29,28
6	6	6	1	2,58	29,28
7	7	7	1	3,00	24,28
8	8	8	1	3,41	24,28
9	9	9	1	3,82	24,28
10	10	10	1	4,22	22,28
11	11	11	1	4,61	22,28
12	12	12	1	4,99	22,28
13	13	13	1	5,37	22,28
14	14	14	1	5,74	22,28
15	15	15	1	6,10	22,28
16	16	16	1	6,45	22,28
17	17	17	1	6,79	22,28
18	18	19	2	7,44	25,29
19	20	21	2	8,05	25,29
20	22	23	2	8,64	25,29
21	24	25	2	9,19	25,29
22	26	27	2	9,70	25,29
23	28	29	2	10,19	25,29
24	30	31	2	10,65	25,29
25	32	33	2	11,08	25,29
26	34	35	2	11,48	25,29
27	36	37	2	11,86	25,29
28	38	39	2	12,22	25,29
29	40	42	3	12,64	27,05
30	43	45	3	13,10	27,05
31	46	48	3	13,53	27,05
32	49	51	3	13,93	27,05
33	52	54	3	14,30	27,05
34	55	58	4	14,69	28,30
35	59	62	4	15,11	28,30
36	63	66	4	15,49	28,30
37	67	70	4	15,84	28,30
38	71	75	5	16,21	29,27
39	76	80	5	16,58	29,27
40	81	85	5	16,92	29,27
41	86	91	6	17,27	30,06
42	92	97	6	17,62	30,06
43	98	104	7	17,97	30,73
44	105	111	7	18,32	30,73
45	112	119	8	18,67	31,31
46	120	127	8	19,02	31,31
47	128	136	9	19,35	31,82
48	137	146	10	19,71	32,28
49	147	156	10	20,05	35,28
50	157	167	11	20,39	35,69
51	168	179	12	20,73	36,07
52	180	192	13	21,08	38,42
53	193	206	14	21,43	38,74
54	207	221	15	21,77	39,04
55	222	237	16	22,11	44,32
56	238	255	18	22,45	44,83
57	256	274	19	22,80	45,06

Окончание таблицы В.21

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
58	275	295	21	23,13	55,50
59	296	318	23	23,47	55,89
60	319	344	26	23,81	56,43
61	345	373	29	24,00	66,90
62	374	405	32	24,00	67,33
63	406	442	37	24,00	67,96
64	443	484	42	24,00	68,51
65	485	533	49	24,00	69,18
66	534	591	58	24,00	69,91
67	592	660	69	24,00	70,66
68	661	745	85	24,00	71,57
69	746	851	106	24,00	72,53
70	852	988	137	24,00	73,64
71	989	1023	35	24,00	67,72

Таблица В.22 — Психоакустические параметры для короткого БПФ, F_s = 88,2 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	0	1	0,00	27,28
1	1	1	1	3,41	14,28
2	2	2	1	6,45	12,28
3	3	3	1	8,92	12,28
4	4	4	1	10,87	12,28
5	5	5	1	12,39	12,28
6	6	6	1	13,61	12,28
7	7	7	1	14,59	12,28
8	8	8	1	15,40	12,28
9	9	9	1	16,09	12,28
10	10	10	1	16,69	12,28
11	11	11	1	17,21	12,28
12	12	12	1	17,68	12,28
13	13	13	1	18,11	12,28
14	14	14	1	18,49	12,28
15	15	15	1	18,85	12,28
16	16	17	2	19,48	15,29
17	18	19	2	20,05	18,29
18	20	21	2	20,55	18,29
19	22	23	2	21,01	20,29
20	24	25	2	21,43	20,29
21	26	27	2	21,81	20,29
22	28	29	2	22,15	25,29
23	30	32	3	22,55	27,05
24	33	35	3	22,98	27,05
25	36	38	3	23,36	37,05
26	39	42	4	23,75	38,30
27	43	46	4	24,00	48,30
28	47	51	5	24,00	49,27
29	52	56	5	24,00	49,27
30	57	62	6	24,00	50,06
31	63	69	7	24,00	50,73
32	70	77	8	24,00	51,31
33	78	87	10	24,00	52,28
34	88	99	12	24,00	53,07
35	100	115	16	24,00	54,32
36	116	127	12	24,00	53,07

Таблица В.23 — Психоакустические параметры для длинного БПФ, F_s = 96 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	0	1	0,00	37,28
1	1	1	1	0,47	37,28
2	2	2	1	0,95	37,28
3	3	3	1	1,42	32,28
4	4	4	1	1,88	32,28
5	5	5	1	2,35	29,28
6	6	6	1	2,81	29,28
7	7	7	1	3,26	24,28
8	8	8	1	3,70	24,28
9	9	9	1	4,14	22,28
10	10	10	1	4,57	22,28
11	11	11	1	4,98	22,28
12	12	12	1	5,39	22,28
13	13	13	1	5,79	22,28
14	14	14	1	6,18	22,28
15	15	15	1	6,56	22,28
16	16	16	1	6,93	22,28
17	17	17	1	7,28	22,28
18	18	18	1	7,63	22,28
19	19	20	2	8,28	25,29
20	21	22	2	8,90	25,29
21	23	24	2	9,48	25,29
22	25	26	2	10,02	25,29
23	27	28	2	10,53	25,29
24	29	30	2	11,00	25,29
25	31	32	2	11,45	25,29
26	33	34	2	11,86	25,29
27	35	36	2	12,25	25,29
28	37	38	2	12,62	25,29
29	39	40	2	12,96	25,29
30	41	43	3	13,36	27,05
31	44	46	3	13,80	27,05
32	47	49	3	14,21	27,05
33	50	52	3	14,59	27,05
34	53	55	3	14,94	27,05
35	56	59	4	15,32	28,30
36	60	63	4	15,71	28,30
37	64	67	4	16,08	28,30
38	68	72	5	16,45	29,27
39	73	77	5	16,83	29,27
40	78	82	5	17,19	29,27
41	83	88	6	17,54	30,06
42	89	94	6	17,90	30,06
43	95	101	7	18,26	30,73
44	102	108	7	18,62	30,73
45	109	116	8	18,97	31,31
46	117	124	8	19,32	31,31
47	125	133	9	19,67	31,82
48	134	143	10	20,03	35,28
49	144	153	10	20,38	35,28
50	154	164	11	20,72	35,69
51	165	176	12	21,07	38,07
52	177	189	13	21,42	38,42
53	190	203	14	21,77	38,74
54	204	218	15	22,12	44,04
55	219	234	16	22,46	44,32
56	235	252	18	22,80	44,83

Окончание таблицы В.23

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
57	253	271	19	23,14	55,06
58	272	292	21	23,47	55,50
59	293	316	24	23,81	56,08
60	317	342	26	24,00	66,43
61	343	372	30	24,00	67,05
62	373	406	34	24,00	67,59
63	407	445	39	24,00	68,19
64	446	490	45	24,00	68,81
65	491	543	53	24,00	69,52
66	544	607	64	24,00	70,34
67	608	685	78	24,00	71,20
68	686	783	98	24,00	72,19
69	784	910	127	24,00	73,31
70	911	1023	113	24,00	72,81

Таблица В.24 — Психоакустические параметры для короткого БПФ, F_s = 96 кГц

Индекс	*w_low*	*w_high*	Ширина	*bval*	*qsthr*
0	0	0	1	0,00	27,28
1	1	1	1	3,70	14,28
2	2	2	1	6,93	12,28
3	3	3	1	9,49	12,28
4	4	4	1	11,45	12,28
5	5	5	1	12,96	12,28
6	6	6	1	14,15	12,28
7	7	7	1	15,11	12,28
8	8	8	1	15,90	12,28
9	9	9	1	16,57	12,28
10	10	10	1	17,16	12,28
11	11	11	1	17,67	12,28
12	12	12	1	18,13	12,28
13	13	13	1	18,55	12,28
14	14	14	1	18,93	12,28
15	15	16	2	19,60	15,29
16	17	18	2	20,20	18,29
17	19	20	2	20,73	18,29
18	21	22	2	21,21	20,29
19	23	24	2	21,64	20,29
20	25	26	2	22,03	25,29
21	27	28	2	22,39	25,29
22	29	31	3	22,79	27,05
23	32	34	3	23,23	37,05
24	35	37	3	23,62	37,05
25	38	41	4	24,00	48,30
26	42	45	4	24,00	48,30
27	46	50	5	24,00	49,27
28	51	55	5	24,00	49,27
29	56	61	6	24,00	50,06
30	62	68	7	24,00	50,73
31	69	77	9	24,00	51,82
32	78	88	11	24,00	52,69
33	89	102	14	24,00	53,74
34	103	120	18	24,00	54,83
35	121	127	7	24,00	50,73

В.2 Управление усилением

В.2.1 Процесс кодирования

Инструмент управления усилением состоит из *PQF* (полифазного квадратурного фильтра), детекторов усиления и модификаторов усиления. Этот инструмент получает входные сигналы во времени и *window_sequence*, и затем выводит *gain_control_data* и сигнал с управляемым усилением, длина которого равна длине окна МДКП. Блок-схема для инструмента управления усилением показана на рисунке В.2.

Из-за характеристик банка *PQF*, порядок коэффициентов МДКП в каждой четной субполосе *PQF* должен быть реверсирован. Это делается путем реверсирования порядка следования спектральных коэффициентов МДКП, т. е. переменой мест более высоких частотных коэффициентов МДКП с более низкими.

Если инструмент управления усилением используется, конфигурация банка фильтров изменяется следующим образом. В случае *EIGHT_SHORT_SEQUENCE window_sequence*, число коэффициентов для МДКП 32, вместо 128, и восемь МДКП. В случае других значений *window_sequence* число коэффициентов для МДКП 256, вместо 1024, и выполняется одно МДКП. Во всех случаях инструмент банка фильтров получает в общей сложности 2048 значений сигнала с управляемым усилением на фрейм из-за перекрытия входных отсчетов.

В.2.1.1 PQF

Входной сигнал делится в PQF на четыре равных по ширине полосы. Коэффициенты в каждой полосе PQF определяются следующим образом:

$$h_i(n) = \frac{1}{4}\cos\left(\frac{(2i+1)(2n+5)\pi}{16}\right)Q(n), 0 \le n \le 95, 0 \le i \le 3,$$

где:

$Q(n) = Q(95 - n)$, $48 \le n \le 95$

и значения $Q(n)$ являются теми же самыми значениями, что и в декодере.

В.2.1.2 Детектор усиления

Детекторы усиления производят данные управления усилением, которые удовлетворяют синтаксису потока битов. Эта информация состоит из количества изменений усиления, индекса позиций изменения усиления и индекса уровня изменения усиления. Выходные данные управления усилением применяются к предыдущему входному сигналу во времени. Это означает, что у детектора усиления есть задержка на один фрейм.

Обнаружение точки изменения усиления выполняется во второй половине окна МДКП и в неперекрытой области (*LONG_START_SEQUENCE и LONG_STOP_SEQUENCE*). Таким образом, число областей равно одной для *ONLY_LONG_SEQUENCE*, двум для *LONG_START_SEQUENCE* и *LONG_STOP_SEQUNCE* и восьми для *EIGHT_SHORT_SEQUENCE*.

Отсчеты в каждой области делятся на подобласти, каждая из которых содержит количество отсчетов, кратное восьми. Затем выбирается одно значение (например, пиковое значение) в этих подобластях. Вычисляются отношения между значениями подобластей и значением последней подобласти. Если отношение лежит вне диапазона 2 ^ *n*, где *n* — целое число от –4 до 11, эти подобласти могут быть обнаружены как точки изменения усиления сигналов. Номер подобласти, которая обнаружена как точка изменения усиления, сохраняется как данные позиции. Экспонента отношения сохраняется как данные усиления. Временное разрешение управления усилением составляет приблизительно 0,7 мс на частоте дискретизации 48 кГц.

В.2.1.3 Модификатор усиления

Модификатор усиления для каждой полосы *PQF* управляет усилением каждой полосы сигнала. Дополнительный процесс управления усилением в декодере уменьшает пред–эхо и восстанавливает исходный сигнал. Оконная функция для управления усилением, функция модификации усиления (*GMF*), которая определяется в процессе декодирования, получается из значений усилений и позиций изменения усиления. Управляемые сигналы усиления получаются путем применения *GMF* к соответствующим полосным сигналам.

В.2.2 Схемы

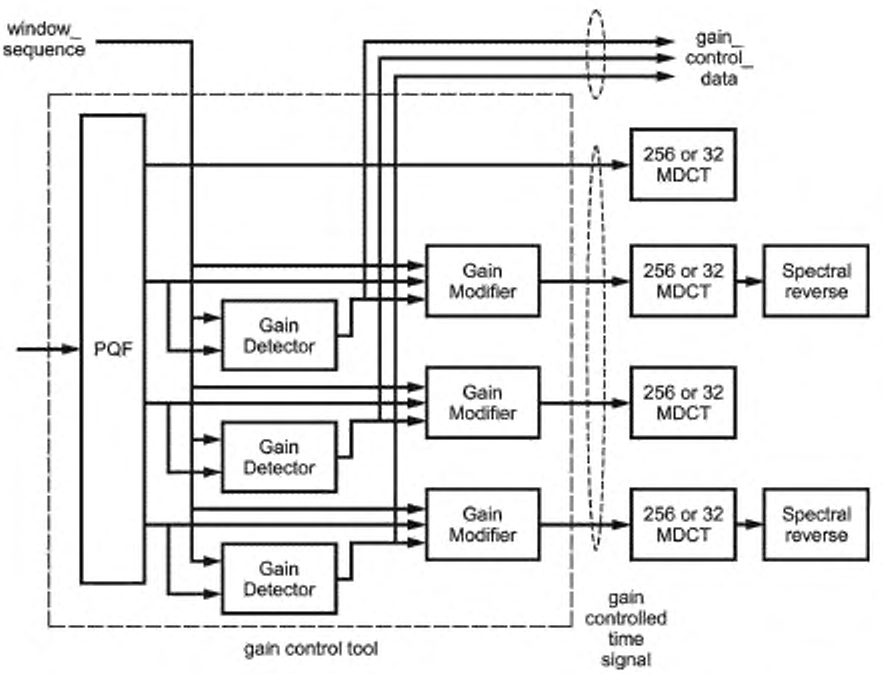


Рисунок В.2 — Блок-схема инструмента управления усилением для кодера

В.3 Банк фильтров и переключение блоков

Основным компонентом в процессе аудиокодирования является преобразование сигналов во времени в частотно-временное представление. Эту функцию выполняет модифицированное дискретное косинусное преобразование (МДКП).

В.3.1 Процесс кодирования

В банк фильтров кодера поступает соответствующий блок временных отсчетов, где они модулируются соответствующей оконной функцией, и выполняется МДКП. Каждый блок входных отсчетов перекрывается на 50 % с предыдущим и следующим блоками. Размер входного блока для преобразования N может быть установлен равным 2048 или 256 отсчетам. Так как оконная функция оказывает существенный эффект на частотную характеристику банка фильтров, последний должен быть разработан таким образом, чтобы поддерживать изменение формы окна для лучшего адаптирования к свойствам входного сигнала. Форма окна меняется одновременно в кодере и декодере, чтобы позволить банку фильтров эффективно разделять входные спектральные компоненты для разнообразных входных сигналов.

В.3.1.1 Работа с окнами и переключение блоков

Адаптация разрешающей способности банка фильтров по частоте и по времени к характеристикам входного сигнала выполняется путем переключения между длинами преобразований, равными 2048 и 256 отсчетам. Процесс переключения описан в 13.3.1.

Решение о выборе формы окна принимается кодером для каждого фрейма. Выбранная форма окна применима только ко второй половине окна, так как первая половина ограничивается соответствующей формой окна предыдущего фрейма. На рисунке В.3 показана последовательность блоков для перехода (D–E–F) к и от фрейма, использующего синусное окно. Селектор формы окна обычно производит окна большей длины, чем показано на рисунке.

2048 временных отсчетов $x'_{i,n}$ к которым должно быть применено окно, представляют собой 1024 отсчета предыдущей window_sequence, объединенные с 1024 значениями текущего блока. Следующая формула показывает этот факт:

$$x'_{i,n} = \begin{cases} x_{(i-1),(n+1024)}, & \text{for } 0 \le n < 1024 \\ x_{i,n}, & \text{for } 1024 \le n < 2048 \end{cases},$$

где: i — индекс блока, n — индекс отсчета в пределах блока. Как только форма окна выбрана, инициализируется элемент синтаксиса *window_shape*. Вместе с выбранной *window_sequence* это вся информация, необходимая для работы с окнами.

С половинами окон, описанными в 13.3.2, могут применяться все *window_sequence*.

B.3.1.2 МДКП

Спектральный коэффициент $X_{i,k}$ определяется следующим образом:

$$X_{i,k} = 2 \cdot \sum_{n=0}^{N-1} z_{i,n} \cos\left(\frac{2\pi}{N}\left(n + n_0\right)\left(k + \frac{1}{2}\right)\right) \quad \text{for } 0 \le k < N/2,$$

где:

z — входная последовательность, взвешенная в окне;

n — индекс отсчета;

k — индекс спектрального коэффициента;

i — индекс блока;

N — длина окна преобразования, соответствующая значению *window_sequence* $N_0 = (N/2+1)/2$.

Длина окна анализа N МДКП является функцией элемента синтаксиса *window_sequence* и определяется следующим образом:

$$N = \begin{cases} 2048, & \text{if ONLY\_LONG\_SEQUENCE (0x0)} \\ 2048, & \text{if LONG\_START\_SEQUENCE (0x1)} \\ 256, & \text{if EIGHT\_SHORT\_SEQUENCE (0x2), (8 times)} \\ 2048, & \text{if LONG\_STOP\_SEQUENCE (0x3)} \end{cases}$$

B.3.2 Схемы

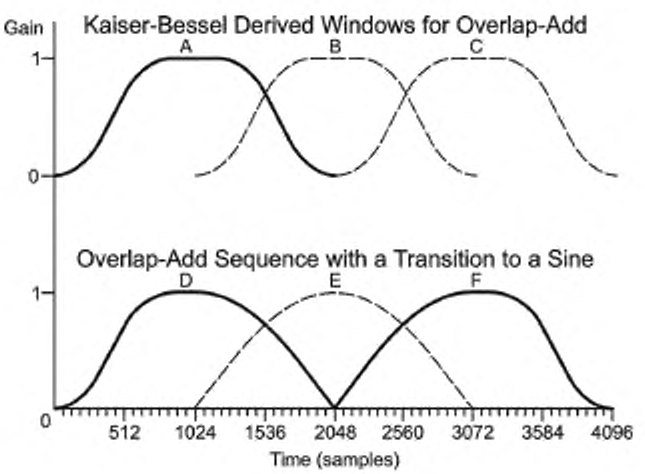


Рисунок B.3 — Пример процесса адаптации формы окна

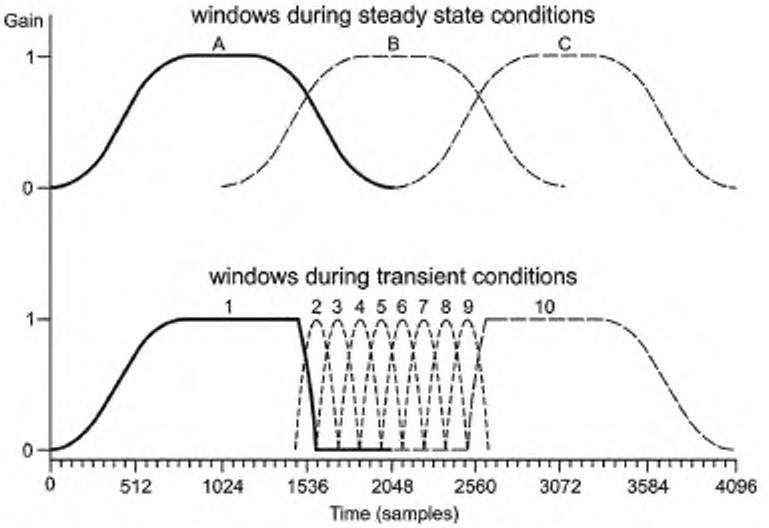


Рисунок В.4 — Пример переключения блоков при наличии резких изменений в сигнале

В.4 Предсказание

В.4.1 Описание инструмента

Так как любой предсказатель идентичен для кодера и декодера, все описания и определения, данные для декодера в разделе 13, также допустимы здесь.

Предсказание используется для улучшенного устранения избыточности и особенно эффективно в случае относительно стационарных участков сигнала, которые принадлежат к наиболее требовательным частям с точки зрения необходимой скорости передачи. Предсказание может быть применено к каждому каналу при использовани внутриканального (или моно) предсказателя, который использует автокорреляцию между спектральными компонентами последовательных фреймов. Поскольку *window_sequence* типа *EIGHT_SHORT_SEQUENCE* указывает на изменения в сигнале, т. е. нестационарные свойства сигнала, то предсказание используется только, если *window_sequence* имеет тип *ONLY_LONG_SEQUENCE*, *LONG_START_SEQUENCE* или *LONG_STOP_SEQUENCE*.

Для каждого канала предсказание применяется к спектральным компонентам, полученным в результате прохождения временных отсчетов через банк фильтров. Каждому спектральному компоненту, вплоть до *PRED_SFB_MAX*, соответствует один предсказатель, что в результате приводит к банку предсказателей, в котором каждый предсказатель использует автокорреляцию между спектральными значениями последовательных фреймов.

Общая структура кодирования с использованием банка фильтров с высокой спектральной разрешающей способностью подразумевает использование обратных адаптивных прогнозирующих устройств для достижения высокой эффективности кодирования. В этом случае коэффициенты прогнозирующего устройства вычисляются по предыдущим квантованным значениям спектральных компонентов как в кодере, так и в декодере, и никакая дополнительная служебная информация не требуется для передачи коэффициентов предсказания, как это требовалось бы для прямых адаптивных предсказаний. Обратно-адаптивное предсказание со структурой решетки второго порядка используется для каждого спектрального компонента так, чтобы каждый предсказатель работал со спектральными компонентами двух предыдущих фреймов. Параметры предсказателя адаптируются к текущей сигнальной статистике фрейма, используя алгоритм адаптации на основе *LMS*. Если предсказание активно, на квантователь поступает ошибка предсказания вместо исходных спектральных компонентов, что приводит к увеличению эффективности кодирования.

В.4.2 Процесс кодирования

Каждому спектральному компоненту каждого канала вплоть до *PRED_SFB_MAX* соответствует один предсказатель. Следующее описание допустимо для одного предсказателя и должно быть применено к остальным. Как сказано выше, каждый предсказатель идентичен в кодере и декодере. Поэтому структура предсказателя, показанная на рисунке В4, и вычисление оценки $x_{est}(n)$ текущего спектрального компонента $x(n)$, так же как вычисление и адаптация коэффициентов предсказания, идентичны тем, что были даны для декодера в 6.3.2.

Единственное отличие заключается в необходимости вычисления ошибки предсказания в кодере $e(n) = x(n) - x_{est}(n)$, которая поступает на вход блока квантования. В этом случае квантованная ошибка предсказания передается вместо квантованного спектрального компонента.

В.4.2.1 Управление предсказателем

Чтобы гарантировать использование предсказания для увеличения эффективности кодирования, требуется соответствующее управление предсказателями, и небольшое количество управляющей информации

для предсказания должно быть передано к декодеру. Для управления предсказанием предсказатели группируются в полосы масштабных коэффициентов.

Следующее описание допустимо для любого одного *single_channel_element* () или одного *channel_pair_element* () и должно быть применено к каждому такому элементу. Так как предсказание используется, только если *window_sequence* имеет типы *ONLY_LONG_SEQUENCE*, *LONG_START_SEQUENCE* или *LONG_STOP_SEQUENCE* для канала, связанного с *single_channel_element* () или для обоих каналов, связанных с *channel_pair_element* (), следующее применяется только в этих случаях.

Управляющая информация предсказания для каждого фрейма, которая должна быть передана как служебная, определяется в двух шагах. Во-первых, для каждой полосы масштабных коэффициентов определяется, приводит ли предсказание к улучшению эффективности кодирования и если да, бит *prediction_used* для соответствующей полосы масштабных коэффициентов устанавливается в '1'. После того как это было сделано для всех полос масштабных коэффициентов вплоть до *PRED_SFB_MAX*, определяется, компенсирует ли общее усиление кодирования за счет предсказания в этом фрейме, по крайней мере, дополнительные биты служебной информации, необходимые для предсказания. Если да, бит *predictor_data_present* устанавливается в '1', и полная служебная информация, включая необходимые данные для сброса предсказания (см. ниже), должна быть передана, и ошибка предсказания подается в квантователь. В противном случае бит *predictor_data_present* устанавливается в '0', все биты *prediction_used* сбрасываются и не передаются. В этом случае на вход квантователя поступают спектральные компоненты. На рисунке В.5 показана блок-схема модуля предсказания для одной полосы масштабных коэффициентов. Как описано выше, управление предсказанием сначала работает для всех предсказателей одной полосы масштабных коэффициентов, а затем, согласно второму шагу, для всех полос масштабных коэффициентов.

В случае *single_channel_element ()* или *channel_pair_element ()* с *common_window* = 0 управляющая информация вычисляется и является допустимой для банка(ов) предсказателей канала(ов), связанных с тем элементом. В случае *channel_pair_element ()* с *common_window* = 1 управляющая информация вычисляется с учетом обоих каналов, связанных с этим элементом. В этом случае управляющая информация допустима для обоих банков предсказателей двух каналов.

В.4.2.2 Восстановление квантованного спектрального компонента

Так как восстановленное значение квантованного спектрального компонента требуется в качестве входного сигнала для предсказания, оно должно быть вычислено в кодере (см. также рисунки 8 и В.5). В зависимости от значения бита *prediction_used* восстановленное значение является либо квантованным спектральным компонентом, либо квантованной ошибкой предсказания. Поэтому следующие шаги необходимы:

Если бит установлен (1), то квантованная ошибка предсказания, восстановленная из данных, которые будут переданы, добавляется к оценке $x_{est}(n)$, вычисленной в результате предсказания, что приводит к восстановленному значению квантованного спектрального компонента, т. е.

$$x_{rec}(n) = x_{est}(n) + e_q(n).$$

Если бит не установлен (0), то квантованное значение спектрального компонента идентично значению, восстановленному непосредственно из данных, которые будут переданы.

В.4.3 Схемы

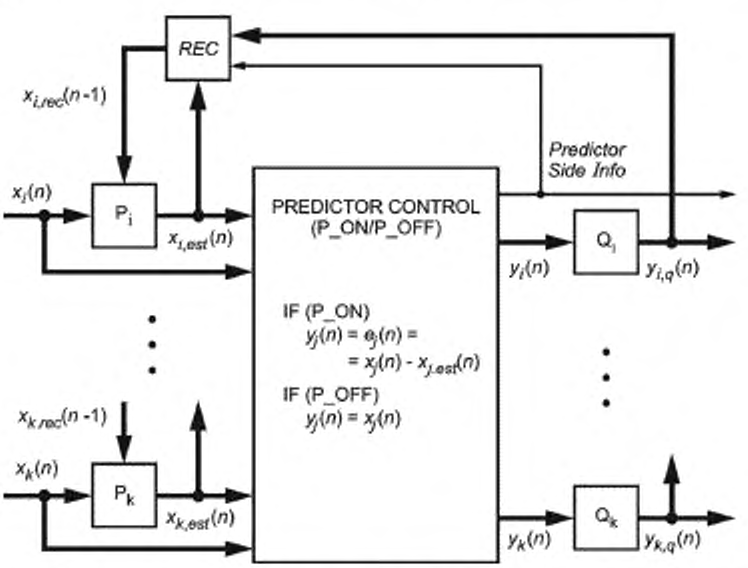


Рисунок В.5 — Блок-схема модуля предсказания для одной полосы масштабных коэффициентов

Полная обработка показана только для предсказателя P_i (*Q* — квантователь, *REC* — реконструкция последнего квантованного значения). Управление предсказанием работает для всех предсказателей P_i... P_j... P_k полосы масштабных коэффициентов и сопровождается вторым управлением всеми полосами масштабных коэффициентов.

В.5 Временное формирование шума (*TNS*)

Временное формирование шума используется для управления временной формой шумов квантования в пределах каждого окна преобразования. Это достигается в процессе фильтрации частей спектральных данных каждого канала.

Кодирование выполняется на основе окна. Следующие шаги выполняются, чтобы применить инструмент временного формирования шума к одному окну спектральных данных:

- Выбирается целевой частотный диапазон для инструмента *TNS*. Подходящий выбор состоит в том, чтобы покрыть частотный диапазон от 1,5 кГц до высшей полосы масштабных коэффициентов одним фильтром. Этот параметр (*TNS_MAX_BANDS*) зависит от профиля и частоты дискретизации, как обозначено в нормативной части.

- Далее кодирование с линейным предсказанием (*LPC*) выполняется для спектральных коэффициентов МДКП, соответствующих выбранному целевому частотному диапазону. Для лучшей устойчивости коэффициенты, соответствующие частотам ниже 2,5 кГц, могут быть исключены из этого процесса. Стандартные процедуры *LPC* могут использоваться для вычисления *LPC*, например алгоритм Левинсона-Дербина. Вычисление выполняется для максимального разрешенного порядка фильтра формирования шума (*TNS_MAX_ORDER*).

В результате вычисления *LPC* становится известен ожидаемый коэффициент усиления предсказания *gp*, так же как и коэффициенты отражения *TNS_MAX_ORDER r[]* (так называемые коэффициенты *PARCOR*).

Если усиление предсказания *gp* не превышает определенный порог *t*, временное формирование шума не используется. В этом случае бит tns_data_present устанавливается в '0', и обработка *TNS* заканчивается. Подходящее пороговое значение $t = 1.4$.

Если коэффициент усиления предсказания *gp* превышает порог *t*, временное ограничение шума используется.

На следующем шаге коэффициенты отражения квантуются, используя биты *coef_res*. Длина *coef_res* составляет 4 бита. Следующий псевдокод описывает преобразование коэффициентов отражения *r []* в индексы *index[]*, и обратное преобразование к квантованным коэффициентам отражения *rq[]*.

```
iqfac = ((1 << (coef_res–1)) – 0.5) / (π/2.0);
iqfac_m = ((1 << (coef_res–1)) + 0.5) / (π/2.0);
/* Reflection coefficient quantization */
for (i = 0; i < TNS_MAX_ORDER; i++) {
 index[i] = NINT(arcsin( r[i] ) * ((r[i] >= 0) ? iqfac : iqfac_m));
}
/* Inverse quantization */
for (i = 0; i < TNS_MAX_ORDER; i++) {
 rq[i] = sin( index[i] / ((index[i] >= 0) ? iqfac : iqfac_m) );
}
```

где *arcsin ()* соответствует функции арксинус.

Порядок используемого фильтра формирования шума определяется путем последовательного удаления всех коэффициентов отражения с "хвоста" массива коэффициентов отражения, с абсолютными значениями, меньшими чем порог *p*. Число оставшихся коэффициентов отражения является порядком фильтра формирования шума. Соответствующий порог для усечения $p = 0,1$.

Оставшиеся коэффициенты отражения *rq*[] преобразовываются в *order*+1 коэффициентов линейного предсказания *a*[].

Вычисленные коэффициенты *LPC* [] используются в качестве коэффициентов фильтра формирования шума в кодере. Этот КИХ-фильтр скользит по определенном целевому частотному диапазону по способу, который описывается в нормативной части для процесса декодирования (описание инструмента). Различие между фильтрацией при декодировании и кодировании заключается в том, что всеполосный (авторегрессивный) фильтр, используемый для декодирования, заменяется его инверсным всенулевым (усредняющим) фильтром, т.е. происходит замена уравнения фильтра декодера

$$y[n] = x[n] – a[1] * y[n – 1] – ... – a[order] * y[n – order]$$

инверсным уравнением

$$y[n] = x[n] + a[1] * x[n – 1] + ... + a[order] * x[n – order].$$

По умолчанию используется восходящее направление фильтрации.

Наконец, передается следующая служебная информация для временного формирования шума (таблица В.25).

Таблица В.25 — Служебная информация TNS

Элемент данных	Переменная или значение
n_filt	1
coef_res	*coef_res*–3
coef_compress	0
length	Число обработанных полос масштабных коэффициентов
direction	0 (вверх)
order	Порядок фильтра формирования шума
coef[]	*index[]*

Использование поля *coef_compress* позволяет сохранять 1 бит на каждый переданный коэффициент отражения, если ни один из коэффициентов отражения не использует больше половины их полного диапазона. В частности, если два старших значащих бита каждого квантованного коэффициента отражения равны '00' или '11', *coeff_compress* может быть установлен в '1', и размер переданных квантованных коэффициентов отражения будет уменьшен на единицу.

В.6 Объединенное кодирование

В.6.1 *M/S Stereo*

Решение о кодировании левых и правых коэффициентов как левый + правый (*L/R*) или как середина/сторона (*M/S*) принимается в отдельности для спектральных коэффициентов каждой из частотных полос текущего блока:

1 — Для каждой полосы кодирования вычисляются не только необработанные пороги *L* и *R*, но также и *M* = (*L*+*R*)/2 и *S* = (*L*–*R*)/2. Для необработанных порогов *M* и *S* вместо использования их степени тональности выбирается наибольший показатель тональности из *L* и *R* в каждой полосе вычисления порога. В психоакустической модели для *M* и *S* используются энергии *M* и *S* и минимальные значения *L* или *R* для $C(\omega)$ в каждой полосе вычисления порога. Значения, которые используются для процесса управления процессом обработки звуковых образов, идентифицируются в разделе психоакустической информации как *en*(*b*) (распространение нормализованной энергии) и *nb* (*b*) (необработанный порог).

2 — Необработанные пороги для *M*, *S*, *L* и *R* и энергии распространения для *M*, *S*, *L* и *R* поступают в "процесс управления звуковым образом". Получающиеся скорректированные пороги вставляются как значения для *cb* (*b*) в шаге 11 психоакустического расчета для дальнейшей обработки.

3 — Окончательные, защищенные и адаптированные полосам пороги *M*, *S*, *L* и *R*, непосредственно применяются к соответствующему спектру через квантование фактических спектральных значений *L*, *R*, *M* и *S* с соответствующим расчетным и квантованным порогом.

4 — Определяется число битов, требуемых для кодирования *M/S*, и число битов, требуемых для кодирования *L/R*.

5 — Метод, требующий наименьшее количество битов, используется в каждой полосе кодирования, и соответственно устанавливается стерео маска.

Используются следующие переменные:

Mthr, Sthr, Rthr, Lthr — необработанные пороги (*nb (b)* из шага 10 психоакустической модели)

Mengy, Sengy, Rengy, Sengy — энергия распространения порога маскирования (*en (b)* из шага 6 психоакустической модели)

Mfthr, Sfthr, Rfthr, Lfthr — окончательные (выходные) пороги (*nb (b)* из шага 11 психоакустической модели)

bmax (b) — защитное отношение BMLD, вычисляемое как

$$bmax(b) = 10^{-3} \cdot \left[0{,}5 + 0{,}5 \cdot \cos\left(\pi \cdot \frac{\min(bval(b), 15{,}5)}{15{,}5} \right) \right].$$

Используется следующий процесс управления звуковым образом:

t = Mthr/Sthr
if (t> 1)
t = 1/t
*Rfthr = max (Rthr*t, min (Rthr, bmax*Rengy)*
*Lfthr = max (Lthr*t, min (Lthr, bmax) *Lengy) t = min (Lthr, Rthr)*
*Mfthr = min (t, max (Mthr, min (Sengy*bmax, Sthr))*
*Sfthr = min (t, max (Sthr, min (Mengy*bmax, Mthr))*

В.6.2 Интенсивностное стерео кодирование

Интенсивностное стерео кодирование используется для устранения избыточности в обоих каналах стерео пары в области верхних частот. Следующая процедура описывает одну из возможных реализаций.

Кодирование выполняется отдельно для каждой группы окон. Выполняются следующие шаги:

- Интенсивностное стерео кодирование соответствующей области полос масштабных коэффициентов выполняется, начиная с граничной частоты f_0. Среднее значение f_0 = 6 кГц является подходящим для большинства типов сигналов.
- Для каждой полосы масштабных коэффициентов вычисляются энергии левого, правого и суммарного каналов путем суммирования спектральных коэффициентов в квадрате, *El[sfb]*, E_r *[sfb]*, E_s *[sfb]*. Если группа окон включает в себя несколько окон, энергии входящих в группу окон складываются.
- Для каждой полосы масштабных коэффициентов соответствующее значение позиции интенсивности вычисляется как

$$is\_position[sfb] = NINT\left(2 \cdot \log_2\left(\frac{E_l[sfb]}{E_r[sfb]}\right)\right).$$

- Затем вычисляются интенсивностные спектральные коэффициенты *speci [i]* для каждой полосы масштабных коэффициентов путем сложения спектральных отсчетов левого и правого каналов ($spec_l$ *[i]* и $spec_r$ *[i]*) и масштабированием полученных значений

$$spec_l[i] = (spec_l[i] + spec_r[i]) \cdot \sqrt{\frac{E_l[sfb]}{E_s[sfb]}}.$$

- Спектральные компоненты интенсивностного сигнала используются для замены соответствующих спектральных коэффициентов левого канала. Соответствующие спектральные коэффициенты правого канала обнуляются.

Затем, выполняется стандартный процесс квантования и кодирования спектральных данных обоих каналов. Однако состояние предсказателей в правом канале устанавливается в "выкл" для всех полос масштабных коэффициентов, кодированных в режиме интенсивностного стерео. Эти предсказатели обновляются с помощью декодируемой интенсивностной версии квантованных спектральных коэффициентов.

Наконец, перед передачей кодовой книги Хаффмана устанавливается *INTENSITY_HCB* в соответствующем разделе для всех полос масштабных коэффициентов, которые кодируются в режиме интенсивностного стерео.

B.7 Квантование

B.7.1 Введение

Описание модуля квантования AAC подразделяется на три уровня. Верхний уровень называется «программой циклов фрейма». Эта программа вызывает подпрограмму «внешний итеративный цикл», которая в свою очередь вызывает подпрограмму «внутренний итеративный цикл». Для каждого уровня дана соответствующая блок-схема. Модуль циклов квантует входной вектор спектральных данных в итеративном процессе согласно нескольким требованиям. Внутренний цикл квантует входной вектор и увеличивает размер шага квантователя, пока выходной вектор не будет кодирован с доступным числом битов. После завершения внутреннего цикла внешний цикл проверяет искажение в каждой полосе масштабных коэффициентов и, если допустимый уровень искажений превышен, полоса масштабных коэффициентов ослабляется, и снова вызывается внутренний цикл.

Вход модуля циклов AAC:

1 Вектор спектральных значений *mdct_line* (0.. 1023).

2 *xmin (sb)*.

3 *mean_bits* (среднее число битов, доступных для кодирования потока битов).

4 *more_bits*, число битов в дополнение к среднему числу битов, вычисленное психоакустическим модулем из перцепционной энтропии (PE).

5 Число и ширина полос масштабных коэффициентов (см. таблицы 45—57).

6 Для группы коротких блоков спектральные значения должны чередоваться так, чтобы спектральные линии, принадлежащие к одной полосе масштабных коэффициентов, но различным типам блоков, которые должны быть квантованы с одинаковыми масштабными коэффициентами, были соединены в одной (большей) полосе масштабных коэффициентов (для полного описания группировки см. 6.3.4),

Выход модуля циклов AAC:

1 Вектор квантованных значений *x_quant* (0.. 1023).

2 Масштабный коэффициент для каждой полосы масштабных коэффициентов (sb).

3 *common_scalefac* (информация о размере шага квантователя для всех полос масштабных коэффициентов).

4 Количество неиспользованных битов, доступных для дальнейшего использования.

B.7.2 Предварительные шаги

Сброс всех итеративных переменных:

1 Вычисляется начальное значение *common_scalefac* для квантователя так, чтобы все квантованные значения МДКП могли быть закодированы в потоке битов:

*start_common_scalefac = ceiling (16/3 * (log$_2$ ((max_mdct_line ^ (3/4))/MAX_QUANT)))*

max_mdct_line является самым большим по абсолютому значению коэффициентом МДКП, а функция *ceiling()* округляет аргумент до ближайшего целого в направлении положительной бесконечности. *MAX_QUANT* — максимальное квантованное значение, которое может быть закодировано в потоке битов, равное 8191. Во время итеративного процесса *common_scalefac* не должен стать меньше, чем *start_common_scalefac.*

2 *Scalefactor [sb]* обнуляется для всех значений *sb.*

B.7.3 Управление резервуаром битов

Биты помещаются в резервуар, если для кодирования одного фрейма используется меньше, чем *mean_bits.*

*mean_bits = bit_rate * 1024 / sampling_rate.*

Максимальное число битов, которые могут быть помещены в резервуар, называется *max_bit_reservoir*, которое вычисляется, используя процедуру в 6.2.3. Если резервуар заполнен, неиспользованные биты должны быть закодированы в потоке битов как *fillbits.* Максимальное количество битов, доступных для фрейма, является суммой *mean_bits* и битов, сохраненных в резервуаре.

Число битов, которые должны использоваться для кодирования фрейма, зависит от значения *more_bits*, которое вычисляется психоакустической моделью, и от максимального количества доступных битов. Самый простой способ управления резервуаром:

if more_bits> 0:

available_bits = mean_bits + min (more_bits, bit_reservoir_state [frame]),

if more_bits <0:

available_bits = mean_bits + max (more_bits, bit_reservoir_state [frame]

– max_bit_reservoir)

B.7.4 Квантование коэффициентов МДКП

Формула для квантования в кодере является инверсией формулы деквантования в декодере (см. также описание декодера):

*x_quant = интервал ((abs (mdct_line) * (2 ^ (–j * (sf_decoder – SF_OFFSET)))) ^ (3/4) + MAGIC_NUMBER)*

MAGIC_NUMBER равно 0.4054, *SF_OFFSET* равно 100, а *mdct_line* — одно из спектральных значений МДКП. Эти значения также вызывают «коэффициентами». Масштабный коэффициент *sf_decoder* является тем же самым, что и *sf [g] [sfb]* в разделе 9.

Для использования в итеративных циклах масштабный коэффициент *'sf_decoder'* разделяется на две переменные:

sf_decoder = common_scalefac – scalefactor + SF_OFFSET

Это следует из формулы, используемой в цикле управления искажениями:

*x_quant = int ((abs (mdct_line) * (2 ^ (–$^1/_4$ * (scalefactor – common_scalefac)))) ^ (3/4) + MAGIC_NUMBER)*

Знак масштабного коэффициента выбирается таким образом, чтобы положительное значение увеличивало величину *x_quant* и, таким образом, уменьшало искажения и увеличивало число используемых битов.

Знак *mdct_line* сохраняется отдельно и добавляется только для подсчета битов и кодирования потока битов.

B.7.4.1 Внешний итеративный цикл (цикл управления искажениями)

Внешний итеративный цикл управляет шумами квантования, которые возникают в результате квантования частотных линий в пределах внутреннего итеративного цикла. Окраска шума выполняется умножением линий в пределах полос масштабных коэффициентов на значения масштабных коэффициентов перед квантованием. Следующий псевдокод иллюстрирует умножение:

do for each scalefactor band sb:

do from lower index to upper index i of scalefactor band

*mdct_scaled(i) = abs(mdct_line(i))^(3/4) *2^(3/16 * scalefactor(sb))*

end

do end do

B.7.4.2 Вызов внутреннего итеративного цикла

Для каждого внешнего итеративного цикла (цикл управления искажениями) вызывается внутренний итеративный цикл (цикл управления уровнем). Параметрами являются масштабированные значения в частотной области (*mdct_scaled* (0 ... 1023)), начальное значение *common_scalefac* и число битов, которые доступны циклу управления уровнем. Результатом является число используемых битов, квантованные частотные линии *x_quant (i)* и новый *common_scalefac.*

Формула для вычисления квантованных коэффициентов МДКП:

x_quant (i) = int((mdct_scaled (*i*) * 2 ^ (–3/16 * common_scalefac)) + MAGIC_NUMBER).

Число битов, необходимых для кодирования квантованных значений и служебной информации (масштабные коэффициенты и т. д.) определяется согласно синтаксису потока битов, описанному в разделе 7.

В.7.4.3 Аттенюация полос масштабных коэффициентов, которые нарушают порог маскирования

Вычисление искажений (*error_energy* (*sb*)) в полосе масштабных коэффициентов выполняется следующим образом:

```
Do for each scalefactor sb:
error_energy (сурьма) =0
Do from lower index to upper index i of scalefactor band
error_energy (sb) = error_energy (sb) + (abs (mdct_line (i))
– (x_quant (i) ^ (4/3) * 2 ^ (j * (scalefactor (sb)–common_scalefac)))) ^2
end do
end do
```

Все спектральные значения полос масштабных коэффициентов, у которых есть искажение, превышающее дозволенное искажение (*xmin* (*sb*)), ослабляются согласно формуле в В.7.4.1, новые масштабные коэффициенты могут быть вычислены согласно этому псевдокоду:

```
Do for each scalefactor band sb
if (error_energy (sb)> xmin (sb)) then
scalefactor (sb) = scalefactor (sb) – 1
end if
end do.
```

В.7.4.4 Условия завершения обработки циклов

Обычно обработка циклов завершается, если отсутствуют полосы масштабных коэффициентов с искажениями, превыщающими дозволенные значения. Однако это не всегда возможно. В этом случае есть другие условия завершения внешнего цикла. Если:

- все полосы с энергией, превышающей *xmin (sb)*, уже ослаблены, или
- различие между двумя соседними масштабными коэффициентами больше, чем 60.

Обработка цикла прекращается, и восстанавленные сохраненные *scalefactors(sb)* поступают на выход. Для реализации в реальном времени возможно третье условие, которое завершает цикл в случае нехватки вычислительного времени.

Процедура, описанная выше, допустима только в том случае, когда число доступных битов равно числу необходимых битов, соответствующих перцепционной энтропии. В случае, когда число доступных битов больше или меньше, чем число необходимых битов, целью модуля циклов становится создание постоянного отношения шумов квантования к порогу маскирования во всех полосах масштабных коэффициентов (постоянное отношение шум-маска (*NMR*)). Это может быть реализовано за счет применения смещения к дозволенному искажению *xmin (sb)*, которое является одинаковым для всех полос масштабных коэффициентов до запуска модуля циклов.

В.7.4.5 Внутренний итеративный цикл (цикл управления уровнем)

Во внутреннем итеративном цикле вычисляется фактическое квантование данных частотной области (*mdct_scaled*) по следующей функции, которая использует формулу из В.7.4.2:

```
quantize_spectrum (x_quant []  ,  mdct_scaled []  ,  common_scalefac) :
 do for all MDCT coefficients i  :
        x_quant(i)  = int  ((mdct_scaled  (i)  * 2^ (–3/16  * common_scalefac))
                + MAGIC_NUMBER)
 end do
```

и затем вызывается функция *bit_count ()*. Эта функция подсчитывает число битов, требуемых для кодирования фрейма потока битов согласно разделу 6.

Внутренний итеративный цикл может быть реализован на основе последовательного приближения:

```
inner_loop():
 if (outer_loop_count == 0)
        common_scalefac = start_common_scalefac;
        quantizer_change = 32;
 else
        quantizer_change = 1;
 end if
 do
        quantize_spectrum ();
        counted_bits = bit_count();
        if (counted_bits > available_bits) then
                common_scalefac = common_scalefac + quant izer_change;
        else
                common_scalefac = common_scalefac – quantizer_change;
        end if
        quantizer_change = int (quantizer_change /2);
```

```
        if (quantizer_change == 0) && (counted_bits > available_bits)
                quantizer_change = 1;
        end if
    while (quantizer_change != 0)
```

Из-за выбора start_common_scalefac, вычисленного в В.7.2.1, после первой итерации внутреннего цикла число необходимых битов обычно превышает число доступных битов, и поэтому *common_scalefac* будут увеличены на *quantizer_change*.

В.8 Кодирование без потерь

В.8.1 Введение

В кодере AAC на вход модуля кодирования без потерь поступают 1024 квантованных спектральных коэффициента. Так как кодирование без потерь выполняется внутри квантователя внутреннего цикла, это является частью итеративного процесса, который сходится, когда полное число битов (из которых на кодирование без потерь приходится большинство) находится в пределах некоторого интервала в окрестности выделенного числа битов. В этом разделе описан процесс кодирования для одного вызова модуля кодирования без потерь.

Кодирование без потерь выполняется по следующим шагам:

- клиппирование спектра,
- предварительное кодирование методом Хаффмана с использованием максимального количества разделов,
- объединение разделов для достижения минимального количества битов.

В.8.2 Клиппирование спектра

На первом шаге кодирования без потерь реализуется метод ограничения динамического диапазона. До четырех коэффициентов с амплитудами, превышающими 1, может быть кодировано отдельно со значением ±1, оставленным в квантованном массиве коэффициентов для сохранения знака. Индекс полосы масштабных коэффициентов, содержащей самые низкочастотные, «усеченные» коэффициенты, отправляется в поток битов. Каждый из «усеченных» коэффициентов кодируется как амплитуда (сверх 1) и смещение по отношению к предыдущей полосе. Для этого длинные блоки полос масштабных коэффициентов и порядок коэффициентов в этих полосах используются независимо от последовательности окон. Одна из стратегий применения клиппирования спектра состоит в том, чтобы отсечь высокочастотные коэффициенты, абсолютные амплитуды которых превышают единицу. Так как служебная информация для усеченных коэффициентов требует несколько битов, то компрессия без потерь применяется, только если это приводит к сокращению битов.

В.8.3 Разделы

При кодировании без потерь набор из 1024 квантованных спектральных коэффициентов сегментируется на разделы так, что для каждого раздела используется одна кодовая книга Хаффмана (метод кодирования по Хаффману объясняется ниже). По причинам эффективности кодирования границы раздела должны соответствовать границам полос масштабных коэффициентов так, чтобы для каждого раздела спектра передавать его размер в полосах масштабных коэффициентов и номер кодовой книги Хаффмана, используемой для раздела.

Сегментирование на разделы является динамичным и обычно изменяется от блока к блоку, чтобы минимизировать количество битов, необходимое для представления набора квантованных спектральных коэффициентов. Это выполняется при помощи специального алгоритма, начиная с максимально возможного числа разделов, каждый из которых использует кодовую книгу Хаффмана с наименьшим индексом. Разделы объединяются, если результирующий объединенный раздел приводит к меньшему общему количеству битов, при этом объединению подлежат в первую очередь разделы с наибольшим количеством требуемых битов. Если объединяемые разделы не используют одну и ту же кодовую книгу Хаффмана, то должна использоваться кодовая книга с более высоким индексом.

Разделы часто содержат только коэффициенты с нулевыми значениями. Например, если входной звуковой сигнал ограничен в полосе 20 кГц или ниже, то самые высокие коэффициенты имеют нулевые значения. Такие разделы кодируются нулевой кодовой книгой Хаффмана, которая указывает, что все коэффициенты являются нулями и передача кодовых комбинаций Хаффмана для этого раздела не требуется.

В.8.4 Группировка и чередование

Если последовательность окон состоит из восьми коротких окон, то набор из 1024 коэффициентов представляет собой матрицу 8x128 частотных коэффициентов, представляющих частотно-временное развитие сигнала на протяжении восьми коротких окон. Несмотря на то, что механизм разделения достаточно гибок, чтобы эффективно представить 8 нулевых разделов, группировка и чередование обеспечивают большую эффективность кодирования. Как было объяснено ранее, коэффициенты, связанные с непрерывными короткими окнами, могут быть сгруппированы для совместного использования масштабных коэффициентов несколькими полосами в пределах группы. Кроме того, коэффициенты в пределах группы чередуются, обмениваясь порядком следования полос масштабных коэффициентов и окон. Перед чередованием набор из 1024 коэффициентов *c* индексируется как *c [g][w][b][k]*, где:

g — индекс группы;

w — индекс окна в пределах группы;

b — индекс полосы масштабных коэффициентов в пределах окна;

k — индекс коэффициента в пределах полосы масштабных коэффициентов (самый правый индекс изменяется наиболее быстро).

После чередования коэффициенты индексируются как *c [g][b][w][k]*.

Это обеспечивает преимущество при объединении всех нулевых разделов из-за ограничения полосы частот в пределах каждой группы.

В.8.5 Масштабные коэффициенты

При кодировании спектральных значений используется один квантователь на каждую полосу масштабных коэффициентов. Размеры шага каждого квантователя определяются как ряд масштабных коэффициентов и общее усиление, которое нормализует эти масштабные коэффициенты. Чтобы увеличить сжатие, масштабные коэффициенты, связанные с полосами, содержащими только нулевые коэффициенты, игнорируются при кодировании и не передаются. Общее усиление и масштабные коэффициенты квантуются с шагом 1,5 дБ. Общее усиление кодируется как 8-разрядное целое число без знака, а масштабные коэффициенты дифференцированно кодируются относительно предыдущих масштабных коэффициентов (либо общего усиления для первого масштабного коэффициента), затем применяется кодирование по Хаффману. Динамический диапазон общего усиления достаточен, чтобы представить полную шкалу 24-разрядных ИКМ значений.

В.8.6 Кодирование методом Хаффмана

Кодирование методом Хаффмана используется, чтобы представить *n*-кратные квантованные коэффициенты кодами Хаффмана, взятыми из одной из 11 кодовых книг. Спектральные коэффициенты в пределах *n*-кратной группы упорядочиваются (от низкого к высокому), и размер *n*-кратной группы равен двум или четырем коэффициентам. Максимальное абсолютное значение квантованных коэффициентов, которое может быть представлено каждой кодовой книгой Хаффмана, и число коэффициентов в каждой *n*-кратной группе для каждой кодовой книги даются в таблице В.26. Есть два сборника кодов для каждого максимального абсолютного значения с различной функцией распределения. Всегда выбирается ближайшее из двух распределений. Для экономии ресурсов при хранении кодовых книг (важное обстоятельство для серийно выпускаемых декодеров) большинство кодовых книг представляет значения без знака. При этом амплитуда коэффициентов кодируется методом Хаффмана, с добавлением к комбинации знакового бита каждого ненулевого коэффициента.

Таблица В.26 — Кодовые книги Хаффмана

Индекс кодовой книги	Кратность	Максимальное абсолютное значение	Наличие знака
0		0	
1	4	1	да
2	4	1	да
3	4	2	нет
4	4	2	нет
5	2	4	да
6	2	4	да
7	2	7	нет
8	2	7	нет
9	2	12	нет
10	2	12	нет
11	2	16 (ESC)	нет

Две кодовые книги требуют специального замечания: сборник кодов 0 и сборник кодов 11. Как было сказано ранее, кодовая книга 0 соответствует случаю, когда все коэффициенты в пределах раздела являются нулями. Кодовая книга 11 может представлять квантованные коэффициенты, у которых абсолютное значение больше или равно 16. Если амплитуда одного или обоих коэффициентов больше или равна 16, используется специальный escape-механизм кодирования для представления этих значений. Амплитуда коэффициентов не должна превышать 16, и соответствующая пара кодируется методом Хаффмана. Биты знака добавляются к кодовой комбинации. Для каждой амплитуды коэффициента, большей или равной 16, добавляется escape-последовательность:

escape-последовательность = <escape_prefix><escape_separator><escape_word>,

где

<escape_prefix> — последовательность из '1' бит длиной *N*,

<escape_separator> — двоичный '0',

<escape_word> — целое числом без знака, msb сначала, длиной *N*+4,

N — число, достаточно большое, чтобы амплитуда квантованного коэффициента была равна

$$2 \wedge (N+4) + \text{<escape\_word>}$$

В.9 Функция управления динамическим диапазоном AAC

Чтобы обработать исходный материал с переменными пиковыми уровнями, средними уровнями и динамическим диапазоном так, чтобы минимизировать изменения при воспроизведении, необходимо управлять уров-

нем воспроизведения таким образом, чтобы, например, уровень диалога или средний уровень музыки устанавливались в соответствии с желанием слушателя, независимо от того, какой баланс уровней был в оригинальной программе. Кроме того, не все потребители будут прослушивать программу в хороших акустических условиях (т. е. при малом окружающем шуме), без ограничений на максимальную громкость воспроизведения. Например, при прослушивании программ в условиях автомобиля, где присутствует высокий уровень фонового шума, можно ожидать, что слушатель захочет уменьшить диапазон уровней.

По указанным причинам управление динамическим диапазоном должно быть доступным в пределах спецификации AAC. Для достижения этого необходимо сопровождать сжатый звук данными, используемыми для установки и управления динамическим диапазоном элементов программы. Это управление должно быть определено относительно контрольного уровня и по отношению к важным элементам программы, например диалогам.

Функции управления динамическим диапазоном следующие:

1 Управление динамическим диапазоном является полностью опциональным. Поэтому при корректном синтаксисе нет никаких сложных изменений для тех, кто не желает использовать *DRC*.

2 Сжатые звуковые данные передаются с полным динамическим диапазоном исходного материала и с данными для осуществления управления динамическим диапазоном.

3 Информация для управления динамическим диапазоном может быть передана в каждом фрейме для уменьшения задержки в установке усиления воспроизведения до минимума.

4 Информация для управления динамическим диапазоном передается с использованием *'fill_element'* AAC.

5 Контрольный уровень соответствует полной шкале.

6 Контрольный уровень программы передается для сохранения соответствия между уровнями воспроизведения различных источников. Эта функция исходного сигнала в большей степени относится к субъективному впечатлению от громкости программы.

7 Контрольный уровень программы соответствует уровню, который может быть установлен на пользовательском устройстве воспроизведения относительно контрольного уровня. Более тихие части программы могут быть усилены по уровню, а более громкие части программы могут быть ослаблены.

8 Контрольный уровень программы определен в диапазоне от 0 до –31,75 дБ относительно контрольного уровня.

9 Контрольный уровень программы использует 7 битовых полей с шагом 0,25 дБ.

10 Управление динамическим диапазоном определено в диапазоне ±31,75 дБ.

11 Управление динамическим диапазоном использует 8 битовых полей (1 знак, 7 значений) с шагом 0,25 дБ.

12 Управление динамическим диапазоном может быть применено ко всем спектральным коэффициентам частотных полос звукового канала или к группам коэффициентов, соответствующих различным полосам масштабных коэффициентов.

13 Управление динамическим диапазоном может быть применено ко всем каналам одинаково или разделено для наборов каналов.

14 Если ожидаемый набор информации управления динамическим диапазоном отсутствует, используются последние полученные допустимые значения.

15 Не все элементы информации управления динамическим диапазоном передаются каждый раз. Например, контрольный уровень программы может передаваться в среднем только раз в 200 мс.

16 Если необходимо, обнаружение ошибок / защита обеспечивается транспортным уровнем.

17 Пользователь должен быть обеспечен средствами изменения управлением динамическим диапазоном, представленными в потоке битов и применяемыми к уровню сигнала.

Библиография

[1] ИСО/МЭК 13818-7:2006 (ISO/IEC 13818-7:2006) Информационные технологии. — Универсальное кодирование движущихся изображений и сопутствующего звука. — Часть 7: Усовершенствованное кодирование звука (AAC) (Information technology — Generic coding of moving pictures and associated audio information — Part 7: Advanced Audio Coding (AAC)

[2] ITU-R документ TG10-2/3-E:1991 Основные требования к качеству сжатия цифровых звуковых сигналов в системах вещания (ITU-R Document TG10-2/3-E only, Basic Audio Quality Requirements for Digital Audio Bit-Rate Reduction Systems for Broadcast Emission and Primary Distribution)

[3] ИСО/МЭК 14496-3:2009 (ISO/IEC 14496-3:2009) Информационная технология. Кодирование аудиовизуальных объектов. Часть 3: Аудио (Information technology — Coding of audio-visual objects — Part 3: Audio)

[4] ИСО/МЭК 13818-1:2007 (ISO/IEC 13818-1:2007) Информационные технологии. Универсальное кодирование движущихся изображений и сопутствующей звуковой информации. Часть 1. Системы (Information technology — Generic coding of moving pictures and associated audio information — Part 1: Systems)

УДК 621.396:006.354 ОКС 33.170

Ключевые слова: звуковое вещание, электрические параметры, технологии MPEG-кодирования, усовершенствованное кодирование звука, масштабирование, защита от ошибок, спектральные коэффициенты, стерео избыточность

Редактор *Е. В. Вахрушева*
Технический редактор *В. Н. Прусакова*
Корректор *Л. Я. Митрофанова*
Компьютерная верстка *Т. Ф. Кузнецовой*

Сдано в набор 15.02.2013. Подписано в печать 15.05.2013. Формат 60×84 1/8. Бумага офсетная. Гарнитура Ариал. Печать офсетная. Усл. печ. л. 17,20. Уч.-изд. л. 16,15. Тираж 75 экз. Зак. 232.

ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.
www.gostinfo.ru info@gostinfo.ru
Набрано и отпечатано в Калужской типографии стандартов, 248021 Калуга, ул. Московская, 256.

gost.e-ecolog.ru