МЕЖДУНАРОДНЫЙ СОЮЗ ЭЛЕКТРОСВЯЗИ

МСЭ-Т H.264

СЕКТОР СТАНДАРТИЗАЦИИ
ЭЛЕКТРОСВЯЗИ МСЭ

(05/2003)

СЕРИЯ H: АУДИОВИЗУАЛЬНЫЕ И МУЛЬТИМЕДИЙНЫЕ СИСТЕМЫ

Инфраструктура аудиовизуальных служб – Кодирование движущихся видеоизображений

Улучшенное кодирование видеосигнала для основополагающих аудиовизуальных услуг

Рекомендация МСЭ-Т H.264

РЕКОМЕНДАЦИИ МСЭ-Т СЕРИИ H

АУДИОВИЗУАЛЬНЫЕ И МУЛЬТИМЕДИЙНЫЕ СИСТЕМЫ

ХАРАКТЕРИСТИКИ ВИДЕОТЕЛЕФОННЫХ СИСТЕМ	H.100–H.199
ИНФРАСТРУКТУРА АУДИОВИЗУАЛЬНЫХ СЛУЖБ	
Общие положения	H.200–H.219
Мультиплексирование и синхронизация при передаче	H.220–H.229
Системные аспекты	H.230–H.239
Процедуры связи	H.240–H.259
Кодирование движущихся видеоизображений	**H.260–H.279**
Сопутствующие системные аспекты	H.280–H.299
Системы и оконечное оборудование для аудиовизуальных служб	H.300–H.349
Архитектура служб каталогов для аудиовизуальных и мультимедийных служб	H.350–H.359
Архитектура качества обслуживания для аудиовизуальных и мультимедийных служб	H.360–H.369
Дополнительные услуги для мультимедийных служб	H.450–H.499
ПРОЦЕДУРЫ МОБИЛЬНОСТИ И СОВМЕСТНОЙ РАБОТЫ	
Обзор мобильности и совместной работы, определений, протоколов и процедур	H.500–H.509
Мобильность для мультимедийных систем и служб серии H	H.510–H.519
Приложения и службы мобильной мультимедийной совместной работы	H.520–H.529
Безопасность для мобильных мультимедийных систем и служб	H.530–H.539
Безопасность для приложений и служб мобильной мультимедийной совместной работы	H.540–H.549
Процедуры мобильного взаимодействия	H.550–H.559
Процедуры взаимодействия мобильной мультимедийной совместной работы	H.560–H.569
ШИРОКОПОЛОСНЫЕ МУЛЬТИМЕДИЙНЫЕ СЛУЖБЫ И МУЛЬТИМЕДИЙНЫЕ СЛУЖБЫ В РЕЖИМЕ TRIPLE-PLAY	
Предоставление широкополосных мультимедийных услуг по VDSL	H.610–H.619

Для получения более подробной информации просьба обращаться к перечню Рекомендаций МСЭ-Т.

Рекомендация МСЭ-Т H.264

Улучшенное кодирование видеосигнала для основополагающих аудиовизуальных услуг

Резюме

Поскольку снизилась стоимость как вычислительной мощности, так и запоминающих устройств, модифицировалась сетевая поддержка кодированных видеоданных и усовершенствовалась технология кодирования видеосигналов, возросла необходимость в промышленном стандарте для сжатого представления видеосигналов с непрерывно возрастающей эффективностью кодирования и улучшенной устойчивостью к сетевому окружению.

Эта Рекомендация представляет эволюцию существующих стандартов кодирования видеосигналов (H.261, H.262 и H.263). Рекомендация была разработана в ответ на возросшую потребность в более высоком сжатии движущихся изображений для разных приложений, таких как видеоконференция, телевизионное вещание, цифровое хранение аудиовизуальных данных, потоков Интернет и связи. Она также разработана, чтобы обеспечить гибкое представление кодированного видеосигнала для разнообразных условий сетевого окружения. Использование этой Рекомендации | Международного стандарта позволит управлять движущимся изображением как видом компьютерных данных, хранить изображение на различных запоминающих устройствах, а также передавать и принимать по существующим и будущим широковещательным каналам.

Источник

Рекомендация МСЭ-Т H.264 утверждена 30 мая 2003 года 16-й Исследовательской комиссией МСЭ-Т (2001–2004 гг.) в соответствии с процедурой, изложенной в Рекомендации МСЭ-Т A.8.

Это издание включает изменения, внесенные в Рекомендацию МСЭ-Т H.264 (2003 г.) Поправкой 1 и утвержденные 16-й Исследовательской комиссией МСЭ-Т (2001–2004 гг.) 7 мая 2004 года в соответствии с процедурой, изложенной в Рекомендации МСЭ-Т A.8.

ПРЕДИСЛОВИЕ

Международный союз электросвязи (МСЭ) является специализированным учреждением Организации Объединенных Наций в области электросвязи. Сектор стандартизации электросвязи МСЭ (МСЭ-Т) – постоянный орган МСЭ. МСЭ-Т отвечает за изучение технических, эксплуатационных и тарифных вопросов и за выпуск Рекомендаций по ним с целью стандартизации электросвязи на всемирной основе.

Всемирная ассамблея по стандартизации электросвязи (ВАСЭ), которая проводится каждые четыре года, определяет темы для изучения Исследовательскими комиссиями МСЭ-Т, которые, в свою очередь, вырабатывают Рекомендации по этим темам.

Утверждение Рекомендаций МСЭ-Т осуществляется в соответствии с процедурой, изложенной в Резолюции 1 ВАСЭ.

В некоторых областях информационных технологий, которые входят в компетенцию МСЭ-Т, необходимые стандарты разрабатываются на основе сотрудничества с ИСО и МЭК.

ПРИМЕЧАНИЕ

В настоящей Рекомендации термин "администрация" используется для краткости и обозначает как администрацию электросвязи, так и признанную эксплуатационную организацию.

Соблюдение положений данной Рекомендации носит добровольный характер. Однако в Рекомендации могут содержаться определенные обязательные положения (например, для обеспечения возможности взаимодействия или применимости), и соответствие данной Рекомендации достигается в случае выполнения всех этих обязательных положений. Для выражения необходимости выполнения требований используется синтаксис долженствования и соответствующие слова (такие, как "должен" и т. п.), а также их отрицательные эквиваленты. Использование этих слов не предполагает, что соблюдение положений данной Рекомендации является обязательным для какой-либо из сторон.

ПРАВА ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

МСЭ обращает внимание на то, что практическое применение или реализация этой Рекомендации может включать использование заявленного права интеллектуальной собственности. МСЭ не занимает какую бы то ни было позицию относительно подтверждения, обоснованности или применимости заявленных прав интеллектуальной собственности, независимо от того, отстаиваются ли они членами МСЭ или другими сторонами вне процесса подготовки Рекомендации.

На момент утверждения настоящей Рекомендации МСЭ [не] получил извещения об интеллектуальной собственности, защищенной патентами, которые могут потребоваться для реализации этой Рекомендации. Однако те, кто будет применять Рекомендацию, должны иметь в виду, что это может не отражать самую последнюю информацию, и поэтому им настоятельно рекомендуется обращаться к патентной базе данных БСЭ.

СОДЕРЖАНИЕ

Стр.

СПИСОК РИСУНКОВ

СПИСОК ТАБЛИЦ

Предисловие

Международный союз электросвязи (МСЭ) является специализированным учреждением Организации Объединенных Наций в области электросвязи. Сектор стандартизации электросвязи (МСЭ-Т) – постоянный орган МСЭ. МСЭ-Т отвечает за изучение технических, эксплуатационных и тарифных вопросов и за выпуск Рекомендаций по ним с целью стандартизации электросвязи на всемирной основе. Всемирная ассамблея по стандартизации электросвязи (ВАСЭ), которая проводится каждые четыре года, определяет темы для изучения Исследовательскими комиссиями МСЭ-Т, которые, в свою очередь, вырабатывают Рекомендации по этим темам. Утверждение Рекомендаций МСЭ-Т осуществляется в соответствии с процедурой, изложенной в Резолюции 1 ВАСЭ. В некоторых областях информационных технологий, которые входят в компетенцию МСЭ-Т, необходимые стандарты разрабатываются на основе сотрудничества с ИСО и МЭК.

ИСО (Международная организация по стандартизации) и МЭК (Международная электротехническая комиссия) образуют специализированную систему стандартизации на всемирном уровне. Национальные органы, являющиеся членами ИСО и МЭК, принимают участие в разработке международных стандартов через технические комитеты, учрежденные соответствующей организацией для рассмотрения определенных областей технической деятельности. Технические комитеты ИСО и МЭК сотрудничают в областях, представляющих взаимный интерес. Другие международные организации (правительственные и неправительственные) также принимают участие в работе в сотрудничестве с ИСО и МЭК. В области информационных технологий ИСО и МЭК создали объединенный технический комитет – ИСО/МЭК/ОТК 1. Проекты международных стандартов, принятые Объединенным техническим комитетом, рассылаются национальным органам для голосования. Для публикации в качестве Международного стандарта требуется, чтобы проект утвердили по крайней мере 75% национальных органов, участвующих в голосовании.

Настоящая Рекомендация | Международный стандарт подготовлены совместно ИК16 Q.6 МСЭ-Т, известной также как VCEG (Группа экспертов по кодированию видеосигналов), и ИСО/МЭК ОТК 1/ПК29/РГ11, известной также как MPEG (Экспертная группа по движущимся изображениям). VCEG была сформирована в 1997 году для ведения разработанных ранее МСЭ-Т стандартов по кодированию видеоизображений и разработки нового стандарта (стандартов) по видеокодированию, пригодного для широкого круга речевых и неречевых служб. MPEG была создана в 1988 году для разработки стандартов по кодированию движущихся изображений и звукового сопровождения для различных применений, таких как средства хранения цифровых данных, распределение и связь.

В Приложениях A–E настоящей Рекомендации | Международного стандарта содержатся нормативные требования и они являются составной частью данной Рекомендации | Международного стандарта.

0 Введение

Этот раздел не является составной частью данной Рекомендации | Международного стандарта.

0.1 Пролог

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Поскольку снизилась стоимость как вычислительной мощности, так и запоминающих устройств, модифицировалась сетевая поддержка кодированных видеоданных и усовершенствовалась технология кодирования видеосигналов, возросла необходимость в промышленном стандарте для сжатого представления видеосигналов с непрерывно возрастающей эффективностью кодирования и улучшенной устойчивостью к сетевому окружению. С этой целью Группа экспертов по кодированию видеосигналов (VCEG) МСЭ-Т и Экспертная группа по движущимся изображениям (MPEG) ИСО/МЭК создали в 2001 г. Объединенную видеогруппу (JVT) для разработки новой Рекомендации | Международного стандарта.

0.2 Цель

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Эта Рекомендация | Международный стандарт была разработана в ответ на возросшую потребность в более высоком сжатии движущихся изображений для разных приложений, таких как видеоконференция, телевизионное вещание, цифровое хранение аудиовизуальных данных, потоков Интернета и связи. Она также разработана, чтобы обеспечить гибкое представление кодированного видеосигнала для разнообразных условий сетевого окружения. Использование этой Рекомендации | Международного стандарта позволит управлять движущимся изображением как видом компьютерных данных, хранить изображение на различных запоминающих устройствах, а также передавать и принимать по существующим и будущим широковещательным каналам.

0.3 Применения

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Эта Рекомендация | Международный стандарт разработана, чтобы охватить широкий диапазон применений видеоконтента, включая (но не ограничивая) следующие области:

CATV	Cable TV on optical networks, copper, etc.	Кабельное телевидение по оптическим и медным сетям и т. д.
DBS	Direct broadcast satellite video services	Услуги видео цифрового спутникового вещания
DSL	Digital subscriber line video services	Услуги видео по цифровым абонентским линиям
DTTB	Digital terrestrial television broadcasting	Цифровая наземная телевизионная передача
ISM	Interactive storage media (optical disks, etc.)	Интерактивные устройства хранения (оптические диски и т. д.)
MMM	Multimedia mailing	Мультимедийная почта
MSPN	Multimedia services over packet networks	Услуги мультимедиа по сетям с коммутацией пакетов
RTC	Remote video surveillance	Диалоговые услуги в реальном времени (видеоконференция, видеотелефон и т. д.)
RVS	Remote video surveillance	Удаленное видеонаблюдение
SSM	Serial storage media (digital VTR, etc.)	Серийные устройства хранения видеоинформации (цифровые видеомагнитофоны и т. д.)

0.4 Параметры и уровни

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Эта Рекомендация | Международный стандарт разработана, чтобы стать общей в том смысле, что она служит для широкого диапазона применений, скоростей передачи битов, разрешающих способностей, качеств и услуг. Кроме прочего, эти применения должны охватить цифровое хранение аудиовизуальной информации, телевизионное вещание и связь в реальном времени. В процессе создания этой спецификации были рассмотрены различные требования для типичных применений, разработаны и объединены в единый синтаксис необходимые алгоритмические элементы. Таким образом, эта спецификация облегчит обмен видеоданными между различными приложениями.

Что касается практичности применения всего синтаксиса к этой спецификации, то ограниченное число подмножеств синтаксиса обусловлено методами "параметров" и "уровней". Эти и другие, связанные с ними термины, формально определены в главе 3.

"Параметры" – это подмножество собственно синтаксиса потока битов, которое определено данной Рекомендацией | Международным стандартом. В границах этого синтаксиса для заданного набора параметров возможно все же требовать значительного разнообразия характеристик кодеров и декодеров в зависимости от значений, которые получены элементами синтаксиса в потоке битов, например заданного размера декодированных изображений. Для многих приложений сейчас нет ни практического, ни экономического смысла реализовывать декодер, способный ко всем гипотетическим применениям этого синтаксиса в рамках конкретного набора параметров.

Чтобы рассмотреть эту проблему, в каждом наборе параметров определены "уровни". Уровень – это особое множество ограничений, наложенных на значения элементов синтаксиса в потоке битов. Эти ограничения могут быть простыми ограничениями значений. Кроме того, они могут принимать форму ограничений на арифметические комбинации значений (например, на ширину изображения, умноженную на его высоту и умноженную на число изображений, декодированных за секунду).

Для кодированного видеоконтента, соответствующего данной Рекомендации | Международному стандарту, используют обычный синтаксис. Чтобы получить подмножество законченного синтаксиса, в поток битов включены флаги, параметры и другие элементы синтаксиса, сигнализирующие о присутствии или отсутствии элементов синтаксиса, которые могут позже появиться в потоке битов.

0.5 Обзор проектируемых характеристик

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Определенное в синтаксисе кодированное представление было разработано, чтобы обеспечить более высокое сжатие изображения при желательном его качестве. Этот алгоритм не является алгоритмом без потерь, поскольку точные значения образцов источника обычно не сохраняются в процессе кодирования и декодирования. Чтобы достичь высокой эффективности сжатия, можно использовать ряд методов. Для каждого изображения существует выбор между алгоритмами кодирования методами inter и intra (не определенными в этой Рекомендации | Международном стандарте) областей, имеющих форму блоков. При inter кодировании (внешнем кодировании) с помощью векторов движения производят поблочное предсказание, используя временные статистические зависимости между различными изображениями. При intra кодировании (внутреннем кодировании) различные пространственные режимы предсказания основаны на использовании статистических пространственных зависимостей в сигнале источника единичного изображения. Векторы движения и режимы intra предсказаний (внутренних предсказаний) можно определить для различных размеров блоков в изображении. В дальнейшем остаток предсказания дополнительно сжимают, используя преобразование для удаления пространственной корреляции внутри преобразованного блока перед его квантованием, тем самым выполняя необратимый процесс, при котором обычно отбрасывают менее важную визуальную информацию, и в то же время формируя близкую аппроксимацию образцов источника. Наконец, векторы движения или режимы intra предсказаний объединяют с информацией квантованных коэффициентов преобразования и кодируют, используя либо арифметическое кодирование, либо коды с переменной длиной.

0.5.1 Кодирование с предсказанием

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Вследствие противоречивых требований прямого доступа и высоко эффективного сжатия определены два основных типа кодирования. Intra кодирование выполняют без сравнения с другими изображениями. Intra кодирование может обеспечить точки доступа к кодированным последовательностям в тех местах, где декодирование можно корректно начать и продолжить. Однако, как правило, при этом демонстрируется лишь умеренная эффективность сжатия. Inter кодирование (с предсказанием или с двойным предсказанием) более эффективно и использует метод inter предсказания (внешнего предсказания) значений образцов каждого блока по некоторым ранее декодированным изображениям, выбранных кодером. В противоположность некоторым другим стандартам кодирования видеосигнала, изображения, которые кодируют методом двойного inter предсказания, можно также использовать как образцы сравнения для кодирования других изображений.

Применение этих трех методов кодирования к изображениям в виде последовательности является гибким, а порядок процесса декодирования обычно не такой, как порядок процесса фиксации источника изображения в кодере или порядок отображения на выходе декодера. Выбор оставлен кодеру и зависит от требований приложения. Порядок декодирования определен таким образом, что декодирование изображений с inter предсказанием изображения происходит позже, чем декодирование прочих изображений, которые сравнивают в процессе декодирования.

0.5.2 Кодирование последовательных или чередующихся видеосигналов

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

В этой Рекомендации | Международном стандарте определен синтаксис и процесс декодирования видеосигналов, которые появляются в форме либо сканированных последовательно, либо сканированных с чередованием. В некоторых последовательностях эти формы могут быть смешаны. Два поля чередующегося кадра разделены во время фиксации, а два поля последовательного кадра имеют общее время фиксации. Каждое поле можно кодировать раздельно или два поля можно кодировать вместе как кадр. Последовательные кадры обычно кодируют как один кадр. При чередующихся видеосигналах кодер может выбирать между кодированием кадра и кодированием поля. Кодирование кадров или кодирование полей можно адаптивно выбирать по принципу изображение за изображением, а также по более ограниченному принципу кодирования. Кодирование кадров обычно предпочтительней в тех случаях, когда видеосцены значительно детализированы, а движение ограничено. Кодирование полей обычно лучше при быстрой смене изображений.

0.5.3 Разделение изображения на макроблоки и меньшие части

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Как и в предыдущих Рекомендациях и Международных стандартах по кодированию видеосигнала в качестве основной единицы обработки в процессе декодирования видеосигнала будет использован макроблок, состоящий из блока образцов яркости 16x16 и двух соответствующих образцов цветности.

При inter предсказании макроблок может быть далее разделен на части. Выбор размера этих частей для inter предсказания является результатом компромисса между эффективностью кодирования при использовании компенсации движения с помощью меньших блоков и количеством данных, необходимых для компенсации движения. В этой Рекомендации | Международном стандарте процесс inter предсказания может формировать сегментацию для

представления движения небольшими по размеру образцами (например, образцами яркости 4x4), используя точность вектора движения с шагом сетки в одну четвертую образца яркости. Процесс inter предсказания блока образца может также включать выбор изображения из числа хранимых ранее декодированных изображений, которое следует использовать как контрольное для сравнения. Векторы движения кодируют по-разному, с учетом предсказанных значений смежных кодированных векторов движения.

Обычно кодер вычисляет соответствующие векторы движения и другие элементы данных, представленные в потоке видеоданных. Этот процесс оценки движения в кодере и выбор возможности использовать inter предсказание для представления каждой области видеоконтента в данной Рекомендации | Международном стандарте не определен.

0.5.4 Уменьшение пространственной избыточности

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Как источник изображений, так и разности от предсказаний обладают высокой пространственной избыточностью. Эта Рекомендация | Международный стандарт основаны на использовании метода поблочного преобразования для удаления пространственной избыточности. После inter предсказания по ранее декодированным образцам других изображений или на основе пространственного предсказания по ранее декодированным образцам текущего изображения результирующую разность предсказания разбивают на блоки 4x4. Эти блоки превращают в область преобразования, где их квантуют. После квантования многие из коэффициентов преобразования становятся нулями или имеют небольшую амплитуду и, таким образом, могут быть представлены небольшим количеством кодированных данных. Процессы квантования и преобразования не определены в данной Рекомендации | Международном стандарте.

0.6 Как читать эту спецификацию

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Предполагается, что читатель начнет с Раздела 1 (Область применения) и продвинется к Разделу 3 (Определения). Раздел 6 следует читать для понимания геометрических взаимодействий источника, входа и выхода декодера. Раздел 7 (Синтаксис и семантика) определяет порядок анализа элементов синтаксиса в потоке битов. См. пп. 7.1–7.3 для понимания синтаксического порядка, а п. 7.4 – семантики, т. е. области применения, условий и ограничений, которые наложены на элементы синтаксиса. Действительный анализ для большинства элементов синтаксиса определен в Разделе 9 (Процесс анализа). Наконец, в Разделе 8 (Процесс декодирования) определено, как элементы синтаксиса отражены в декодированных образцах. Во время чтения этой спецификации читателю следует при необходимости обращаться к Разделам 2 (Нормативные ссылки), 4 (Сокращения) и 5 (Условные обозначения). Приложения от A до E также являются составными частями данной Рекомендации | Международного стандарта.

В Приложении A определены три набора параметров (основные, главные и расширенные), каждый приспособлен для определенных областей приложений и определяет так называемые уровни параметров. В Приложении B определены синтаксис и семантика формата потока байтов для доставки кодированного видеосигнала в виде упорядоченного потока байтов. В Приложении C определен гипотетический контрольный декодер и его использование для контроля потока битов и соответствия этого декодера. В Приложении D определены синтаксис и семантика полезной нагрузки дополнительной расширенной информации. Наконец, в Приложении E определены синтаксис и семантика удобства использования информационных параметров видеосигнала из набора параметров последовательности.

Присутствующие в этой спецификации сообщения с преамбулой "ПРИМЕЧАНИЕ" считаются информативными и не являются составной частью данной Рекомендации | Международного стандарта.

Рекомендация МСЭ-Т H.264

Улучшенное кодирование видеосигнала для основополагающих аудиовизуальных услуг

1 Область применения

В этом документе описана Рекомендация МСЭ-Т H.264 | Международный стандарт ISO/IEC 14496-10 по кодированию видеосигнала.

2 Нормативные ссылки

Указанные ниже Рекомендации и Международные стандарты содержат положения, которые путем ссылки на них в данном тексте составляют положения настоящей Рекомендации | Международного стандарта. На момент публикации указанные издания были действующими. Все Рекомендации и стандарты могут подвергаться пересмотру, поэтому участникам соглашений, основанных на настоящей Рекомендации | Международном стандарте, предлагается изучить возможность применения последнего издания Рекомендаций и стандартов, перечисленных ниже. Члены ИСО/МЭК ведут регистры действующих в настоящее время международных стандартов. Бюро стандартизации электросвязи МСЭ ведет список действующих в настоящее время Рекомендаций МСЭ-Т.

- ITU-T Recommendation T.35 (2000), *Procedure for the allocation of ITU-T defined codes for non-standard facilities*
- ISO/IEC 11578:1996, Annex A, *Universal Unique Identifier*
- ISO/CIE 10527:1991, *Colorimetric Observers*

3 Определения

В данной Рекомендации | Международном стандарте использованы следующие определения:

3.1 **блок доступа**: Набор блоков уровней *NAL,* в которых всегда содержится только одно начальное кодированное изображение. Кроме начального кодированного изображения, блок доступа может также содержать одно или более кодированных дополнительных изображений или другие блоки *NAL,* в которых отсутствуют секции или данные разделения кодированного изображения на секции. Результатом декодирования блока доступа всегда будет декодированное изображение.

3.2 **коэффициент преобразования AC**: Любой коэффициент преобразования, для которого одно- или двухразмерный индекс частоты не является нулевым.

3.3 **процесс адаптивного арифметического бинарного декодирования**: Энтропийный *декодирующий процесс,* который извлекает значения бинов из потока битов, создаваемого процессом *адаптивного арифметического бинарного кодирования*.

3.4 **процесс адаптивного арифметического бинарного кодирования**: Энтропийный *кодирующий процесс*, не обязательно определенный этой Рекомендацией | Международным стандартом, с помощью которого кодируют бинарную последовательность и создают *поток битов*, который декодируют с помощью *процесса адаптивного арифметического бинарного декодирования*.

3.5 **произвольный порядок секций**: Порядок декодирования *секций*, в котором *адрес макроблока* первого макроблока некоторой секции изображения может быть меньше, чем адрес макроблока первого макроблока другой предшествующей секции того же самого кодированного изображения.

3.6 **B секция**: *Секция,* которую можно декодировать, используя внутреннее предсказание (intra предсказание) от декодированных образцов внутри той же *секции,* или предсказание от первоначально декодированных *контрольных изображений*, используя не более двух *векторов движения* и *контрольные индексы* для *предсказания* значений образцов каждого блока.

3.7 **бин**: Один бит из *строки бинов*.

3.8 **бинаризация**: Преобразование в двоичную форму – набор *строк бинов* для всех возможных значений *элемента синтаксиса*.

3.9 **процесс бинаризации**: Процесс преобразования в двоичную форму. Единственный процесс размещения всех возможных значений э*лемента синтаксиса* в набор *строк бинов*.

3.10 **строка бинов**: Строка бинов является промежуточным бинарным представлением значений *элементов синтаксиса* от преобразования в двоичную форму *элемента синтаксиса*.

3.11 **B предсказанная секция**: См. *B секция*.

3.12 **поток битов**: Последовательность битов, которая формирует представление *кодированных изображений* и связана с данными, формирующими одну или более *кодированных видеопоследовательностей*. Поток битов – это наиболее общий термин, используемый для обозначения либо потока *блоков уровней NAL,* либо *потока байтов*.

3.13 **блок**: Массив MxN (M-столбцов на N-строк) образцов или матрица MxN *коэффициентов преобразования*.

3.14 **нижнее поле**: Нижнее поле – одно из двух *полей*, которые составляют *кадр*. Каждый ряд нижнего поля расположен в пространстве непосредственно под рядом соответствующего верхнего поля.

3.15 **нижний макроблок (из пары макроблоков)**: Макроблок внутри пары макроблоков, который содержит образцы нижнего ряда образцов для этой *пары макроблоков.* Для *пары макроблоков поля* нижний макроблок представляет образцы из области нижнего поля кадра, который находится в пространственной области *пары макроблоков.* Для *пары макроблоков кадра* нижний макроблок представляет образцы *кадра,* который находится в нижней половине пространственной области *пары макроблоков.*

3.16 **разомкнутое звено связи**: Расположение в *потоке битов,* которое означает, что некоторые последующие *изображения* в *порядке декодирования* могут содержать серьезные визуальные дефекты вследствие неточно указанных действий, выполненных при организации *потока битов*.

3.17 **байт**: Последовательность из 8 битов, записанная и прочитанная с наиболее значащим битом слева, а наименее значащим битом справа. При представлении в последовательности битов данных наиболее значащий бит байта является первым.

3.18 **байт-синхронизованный**: Положение в *потоке битов* считают байт-синхронизованным, когда это положение является целым кратным 8 битам от позиции первого бита в *потоке битов*. Бит, *байт* или *элемент синтаксиса* считают байт-синхронизованными, если позиция, на которой они появляются в *потоке битов,* является байт-синхронизованной.

3.19 **поток байтов**: Инкапсуляция *потока блоков NAL,* в котором содержатся *префиксы кодов запуска* и *блоки NAL,* как это определено в Приложении B.

3.19.1 **может (в смысле возможности)**: Термин (can), который используют, чтобы определить действие, которое разрешено, но не обязательно требуется.

3.20 **категория**: Число, связанное с каждым *элементом синтаксиса*. Категорию используют для определения расположения *элементов синтаксиса* в *блоках NAL* для *разделения данных секции.* Этот термин можно также использовать в смысле, определенном приложением по отношению к классам *элементов синтаксиса* в понятиях, которые не определены в этой Рекомендации | Международном стандарте.

3.21 **цветовой**: Определение, указывающее, что образец массива или простой образец представляют один из двух цветоразностных сигналов, связанных с основными цветами. Символы, которые используют для цветовых массивов или образцов, обозначают как Cb и Cr.

ПРИМЕЧАНИЕ. – Термин "цветовой" используют чаще, чем термин "хроматический", чтобы избежать затруднений при использовании линейных характеристик передачи света, которые часто связывают с термином "хроматический".

3.22 **кодированное поле**: *Кодированное представление поля*.

3.23 **кодированный кадр**: *Кодированное представление кадра*.

3.24 **кодированное изображение**: *Кодированное представление изображения*. Кодированное изображение может быть либо *кодированным полем,* либо *кодированным кадром*. Кодированное изображение является собирательным термином, относящимся к *основному кодированному изображению* или к *дополнительному кодированному изображению*, но не к обоим сразу.

3.25 **буфер кодированного изображения (CPB)**: Буфер, действующий по принципу "первым пришел – первым обслужен", который содержит *блоки доступа* в *порядке декодирования*, определенном в Приложении C *гипотетическим декодером контроля*.

3.26 **кодированное представление**: Элемент данных, представленный в кодированной форме.

3.27 **кодированная видеопоследовательность**: Последовательность *блоков доступа,* состоящая (в порядке декодирования) из *блоков доступа IDR,* за которыми следуют нуль или другие блоки (но не *доступа IDR). Блоки доступа IDR* должны отсутствовать во всех последующих *блоках доступа.*

3.28 **компонент**: Массив или простой образец одного из трех массивов (одного *яркости* и двух *цветности*), которые образуют *поле* или *кадр*.

3.29 **пара дополнительных полей**: Обобщенный термин для пар *дополнительных контрольных полей* и *дополнительных неконтрольных полей*.

3.30 **пара дополнительных неконтрольных полей**: Два неконтрольных *поля,* которые расположены в последовательных *блоках доступа* в *порядке декодирования* в виде двух *кодированных полей* с *инверсным равенством*, где первое *поле* уже не является парным *полем.*

3.31 **пара дополнительных контрольных полей**: Два контрольных *поля,* которые расположены в последовательных *блоках доступа* в *порядке декодирования* в виде двух *кодированных полей* и разделяют то же значение *элемента синтаксиса* frame_num, и второе *поле* в *порядке декодирования* не является *изображением IDR* и не включает *элемент синтаксиса* memory_management_control_operation, равный 5.

3.32 **контекстная переменная**: Переменная, определенная для процесса *адаптивного арифметического бинарного декодирования* бина с помощью уравнения, в котором содержатся последние декодированные бины.

3.33 **коэффициент преобразования по постоянному току**: Коэффициент преобразования, для которого *индекс частоты* является нулем во всех координатах.

3.34 **декодированное изображение**: *Декодированное изображение,* полученное декодированием *кодированного изображения*. *Декодированное изображение* является либо декодированным *кадром*, либо декодированным *полем*. Декодированное *поле* является либо декодированным *верхним полем,* либо декодированным *нижним полем*.

3.35 **буфер декодированного изображения (DPB)**: Буфер, содержащий *декодированные изображения* для контроля, изменения порядка или задержки на выходе, как это определено в Приложении C *гипотетическим декодером контроля*.

3.36 **декодер**: Устройство для выполнения *процесса декодирования*.

3.37 **порядок декодирования**: Порядок, в котором *элементы синтаксиса* обрабатывают в *процессе декодирования*.

3.38 **процесс декодирования**: Процесс, определенный в этой Рекомендации | Международном стандарте, который считывает *поток битов* и создает из этого потока *декодированные изображения*.

3.39 **прямое предсказание**: *Внешнее предсказание (inter предсказание)* для *блока*, для которого не декодируют никакого *вектора движения*. Определены два режима прямого *предсказания*, которые обозначают как пространственное прямое *предсказание* и временное *предсказание*.

3.40 **тестируемый декодер (DUT)**: *Декодер,* тестируемый на соответствие этой Рекомендации | Международному стандарту с помощью *гипотетического планировщика потока*, который подает соответствующий *поток битов* на этот декодер и на *гипотетический контрольный декодер* и сравнивает значения и временные положения сигналов на выходах этих двух *декодеров*.

3.41 **байт предотвращения эмуляции**: *Байт,* равный 0x03, который можно представить *блоком NAL*. Присутствие байтов предотвращения эмуляции гарантирует, что последовательность следующих друг за другом *байт-синхронизированных байтов* в *блоках NAL* не содержит *префикса кода запуска*.

3.42 **кодер**: Устройство для выполнения *процесса кодирования*.

3.43 **процесс кодирования**: Процесс, не определенный в этой Рекомендации | Международном стандарте, который создает *поток битов,* соответствующих этой Рекомендации | Международному стандарту.

3.44 **поле**: Комплект чередующихся рядов *кадров*. *Кадр* состоит из двух *полей*: *верхнего* и *нижнего*.

3.45 **макроблок поля**: Макроблок, содержащий образцы из единственного *поля*. Все *макроблоки кодированного поля* являются макроблоками поля. Если используют адаптивное декодирование *макроблоков кадра/поля,* некоторые макроблоки *кодированного кадра* могут быть макроблоками поля.

3.46 **пара макроблоков поля**: *Пара макроблоков,* декодированная как два *макроблока поля.*

3.47 **сканирование полем**: Особая последовательность порядка *коэффициентов преобразования,* которая отличается от сканирования зигзагом тем, что сканирование столбцов происходит быстрее сканирования рядов. Сканирование полем используют для *коэффициентов преобразования* в *макроблоках поля.*

3.48 **флаг**: Переменная, которая может принимать одно из двух значений: 0 и 1.

3.49 **кадр**: *Кадр* содержит один массив образцов *яркости* и два соответствующих массива образцов *цветности*. *Кадр* состоит из двух *полей*: *верхнего* и *нижнего*.

3.50 **макроблок кадра**: *Макроблок,* представляющий образцы из двух *полей кодированного кадра*. Если не используют адаптивное декодирование *макроблоков кадра/поля,* все макроблоки *кодированного кадра* являются макроблоками кадра. Если используют адаптивное декодирование *макроблоков кадра/поля,* некоторые макроблоки *кодированного кадра* могут быть макроблоками кадра.

3.51 **пара макроблоков кадра**: *Пара макроблоков,* декодированная как два *макроблока кадра.*

3.52 **индекс частоты**: Одно- или двухразмерный индекс, связанный с *коэффициентом преобразования* перед этапом инверсного преобразования *процесса декодирования.*

3.53 **гипотетический контрольный декодер (HRD)**: Модель гипотетического *декодера,* которая определяет ограничения на изменчивость соответствующих *потоков блоков NAL* или соответствующих *потоков байтов,* которые могут появляться в процессе кодирования.

3.54 **гипотетический планировщик потоков (HSS)**: Гипотетический механизм подачи потоков синхронизации и данных из входного *потока битов* в *гипотетический контрольный декодер*. Планировщик HSS используют для контроля соответствия *потока битов* или *декодера*.

3.55 **I секция**: Не является *SI секцией.* Эту *секцию* декодируют, используя *предсказание,* полученное только от образцов внутри этой же *секции*.

3.55.1 **информативный**: Термин, применяемый к содержанию этой Рекомендации | Международного стандарта, который не является составной частью данной Рекомендации | Международного стандарта. Для информативного содержания не устанавливают никаких обязательных требований на соответствие данной Рекомендации | Международному стандарту.

3.56 **блок доступа мгновенного декодирующего восстановления (IDR)**: *Блок доступа*, в котором исходное *кодированное изображение* является *изображением IDR*.

3.57 **изображение мгновенного декодирующего восстановления (IDR)**: *Кодированное изображение,* содержащее только *секции* типов *I* или *SI*. При этом в *процессе декодирования* помечают все *контрольные изображения* как "неиспользованные для контроля" непосредственно после декодирования изображения IDR. После декодирования изображения IDR все последующие *кодированные изображения* в *порядке декодирования* могут быть декодированы без *внутреннего предсказания (intra предсказания)* из любого *изображения,* декодированного перед изображением IDR. Первое *изображение* каждой *кодированной видеопоследовательности* является изображением IDR.

3.58 **внешнее кодирование (inter кодирование)**: Кодирование *блока*, *макроблока*, *секции* или *изображения,* при котором используют *внешнее предсказание (inter предсказание)*.

3.59 **внешнее предсказание *(*inter предсказание*)***: *Предсказание,* полученное из декодированных образцов *контрольных изображений*, иных чем текущее *декодированное изображение*.

3.60 **внутреннее кодирование (intra кодирование)**: Кодирование *блока*, *макроблока*, *секции* или *изображения,* при котором используют *внутреннее предсказание (intra предсказание)*

3.61 **внутреннее предсказание (intra предсказание)**: *Предсказание*, полученное из декодированных образцов той же самой *декодированной секции*.

3.62 **внутренняя секция**: См. I секция.

3.63 **инверсное преобразование**: Часть *процесса декодирования,* с помощью которого набор *коэффициентов преобразования* преобразуют в пространственно распределенные значения или с помощью которого набор *коэффициентов преобразования* преобразуют в *коэффициенты преобразования DC*.

3.64 **слой**: Один из наборов синтаксических структур с неветвящейся иерархической зависимостью. Высшие слои содержат низшие слои. Кодирующими являются слои *кодированной видеопоследовательности*, *изображения*, *секции* и *макроблока*.

3.65 **уровень**: Определенный набор ограничений на значения, которые могут принимать *элементы синтаксиса* и переменные в этой Рекомендации | Международном стандарте. Этот же набор уровней определен для всех параметров пользователя. Большинство аспектов определения каждого уровня являются общими для разных параметров пользователя. Для частных применений могут существовать особые ограничения, а также может поддерживаться разный уровень для каждого разрешенного параметра пользователя. В различных контекстах уровень представляет значение степени *коэффициента преобразования* до его *масштабирования*.

3.66 **вектор движения списка 0 (списка 1)**: *Вектор движения*, связанный с индексом контроля, указывающим на *контрольное изображение списка 0* (*списка 1*).

3.67 **предсказание списка 0 (списка 1)**: *Внешнее предсказание* содержания *секции*, использующее *индекс контроля*, который указывает на *контрольное изображение списка 0* (*списка 1*).

3.68 **яркость**: Определение, которое указывает, что образец массива или простой образец представляют монохроматический сигнал, связанный с первоначальными цветами. Символ или надпись для яркости обозначают как Y или L.

ПРИМЕЧАНИЕ. – Термин "яркость" используют чаще, чем термин "сигнал яркости", чтобы избежать затруднений при использовании линейных характеристик передачи света, которые часто связывают с термином "сигнал яркости". Иногда символ L используют вместо символа Y, чтобы избежать путаницы с символом "y", который используют для обозначения вертикального направления.

3.69 **макроблок**: Блок образцов *яркости* 16x16 или двух соответствующих блоков образцов *цветности*. Деление *секции* или *пары макроблоков* на макроблоки называют *разделением на части (блоки)*.

3.70 **адаптивное декодирование макроблока кадра/поля**: *Процесс декодирования кодированных кадров,* в котором некоторые *макроблоки* могут быть декодированы как *макроблоки кадров,* а некоторые – как *макроблоки поля.*

3.71 **адрес макроблока**: Если не используют *адаптивное декодирование макроблока кадра/поля,* то адресом является индекс макроблока при *растровом сканировании макроблока изображения* начиная с верхнего левого *макроблока изображения*. Если используют *адаптивное декодирование макроблока кадра/поля,* то адресом верхнего из *пары макроблоков* является удвоенный индекс *пары макроблоков* растрового сканирования изображения, а адресом нижнего из *пары макроблоков* является адрес соответствующего верхнего *макроблока* плюс 1. Адрес верхнего *макроблока* из каждой пары макроблоков – это четное число, а адрес нижнего *макроблока* из каждой пары макроблоков – это нечетное число.

3.72 **местоположение макроблока**: Двухразмерные координаты *макроблока* в *изображении,* обозначенные как (x, y). Для верхнего левого *макроблока* в *изображении* значения (x, y) равны (0, 0). "x" изменяется на 1 для каждого столбца *макроблоков* слева направо. Если не используют *адаптивное декодирование макроблока кадра/поля*, то "y" возрастает на 1 для каждого ряда *макроблоков* сверху вниз. Если используют *адаптивное декодирование макроблока кадра/поля*, то "y" возрастает на 2 для каждого ряда *пар макроблоков* сверху вниз и дополнительно возрастает на 1, если макроблок является нижним *макроблоком*.

3.73 **пара макроблоков**: Пара вертикальных смежных *макроблоков* в *кадре,* которые сдвоены для использования при *адаптивном декодировании макроблока кадра/поля*. Деление секции на пару макроблоков – это разделение на части (*partitioning)*.

3.74 **разделение на части макроблока**: Блок образцов *яркости* и два соответствующих блока образцов *цветности*, которые получают в результате разделения на части макроблока для *внешнего предсказания (inter предсказания)*.

3.75 **отображение макроблока в группу секции**: Средство отображения *макроблоков изображения* в *группы секций*. Отображение макроблока в группу секции состоит из перечня чисел (одного для каждого кодированного *макроблока)*, определяющих группу секции, к которой принадлежит каждый кодированный *макроблок*.

3.76 **отображение блока в группу секции**: Средство отображения *блоков изображения* в *группы секций*. Отображение блока в группу секции состоит из перечня чисел (одного для каждого отображенного блока), определяющих группу секции, к которой принадлежит каждый кодированный блок, отображенный в группу секций.

3.76.1 **может (в смысле пожелания)**: Термин (may), который используют, чтобы определить действие, которое разрешено, но не обязательно требуется. В некоторых местах, в которых желают подчеркнуть дополнительные свойства описываемого поведения, для выразительности используют фразу "может быть, а может и не быть".

3.77 **операция управления памятью**: Семь операций, которые управляют разметкой *контрольного изображения*.

3.78 **вектор движения**: Двухразмерный вектор, используемый для *внешнего предсказания (inter предсказания),* который создает свиг от координат *декодированного изображения* к координатам *контрольноого изображения*.

3.78.1 **должен**: Термин **(**must), используемый для выражения наблюдения относительно требования или использования требования, которое оговорено где-либо в этой Рекомендации | Международном стандарте. Этот термин используется исключительно в информативном контексте.

3.79 **блок NAL**: Синтаксическая структура, содержащая указание типа данных, которые должны последовать, а также *байты,* содержащие эти данные в форме *RBSP*, которые передают при необходимости вместе с *байтами предотвращения эмуляции*.

3.80 **поток блоков NAL**: Последовательность блоков *NAL*.

3.81 **несдвоенное поле**: Собирательный термин для обозначения несдвоенного (непарного) контрольного *поля* или несдвоенного неконтрольного *поля*.

3.82 **несдвоенное неконтрольное поле**: Декодированное неконтрольное *поле,* которое не является частью дополнительной пары неконтрольных полей.

3.83 **несдвоенное контрольное поле**: Декодированное контрольное *поле,* которое не является частью дополнительной пары контрольных полей.

3.84 **неконтрольное поле**: *Поле,* кодированное значением nal_ref_idc, равным 0.

3.85 **неконтрольный кадр**: *Кадр,* кодированный значением nal_ref_idc, равным 0.

3.86 **неконтрольное изображение**: *Изображение,* кодированное значением nal_ref_idc, равным 0. *Неконтрольное изображение* не используют для *внешнего предсказания* других *изображений*.

3.86.1 **примечание**: Термин, используемый для присоединения информативных замечаний. Этот термин используется исключительно в информативном контексте.

3.87 **инверсное равенство**: Пример инверсного равенства: вверх – это низ и наоборот.

3.88 **порядок выхода**: Порядок, в котором *декодированные изображения* появляются на выходе буфера *декодированного изображения*.

3.89 **P секция**: *Секция,* которую можно декодировать с помощью *внутреннего предсказания (intra предсказания)* от декодированных образцов той же самой *секции* или *внешнего предсказания (inter предсказания)* от ранее декодированных *контрольных изображений*, используя не более одного *вектора движения* и *индекс контроля,* чтобы предсказать значения образцов каждого блока.

3.90 **параметр**: *Элемент синтаксиса установки параметров последовательности* или *установки параметров изображения*. Параметр используют также как часть определенного термина *параметр квантования*.

3.91 **равенство**: Равенством *поля* может быть верх или низ.

3.92 **разделение на части**: Деление множества на подмножества таким образом, что каждый элемент множества является в точности элементом этого подмножества.

3.93 **изображение**: Собирательный термин для *поля* или *кадра*.

3.93.1 **установка параметра изображения**: Синтаксическая структура, содержащая *элементы синтаксиса,* которые применимы к нулю или более законченным частям *кодированных изображений,* как это определено *элементом синтаксиса* pic_parameter_set_id, расположенным в каждом заголовке *секции.*

3.94 **счетчик последовательности изображения**: Переменная, принимающая неубывающие значения при возрастании позиции *изображения* в последовательности на выходе относительно первоначального *изображения IDR* в *порядке декодирования* или относительно первоначального *изображения,* содержащегося в операции управления памятью, которая помечает все *контрольные изображения* как "неиспользованные для контроля".

3.95 **предсказание**: Устройство для выполнения *процесса предсказания*.

3.96 **процесс предсказания**: Использование *предсказателя* для оценки значения образца или элементов данных, декодируемых в этот момент.

3.97 **предсказанная секция**: См. P секция.

3.98 **предсказатель**: Комбинация особых значений или ранее декодированных значений образцов или элементов данных, использованных в *процессе декодирования* очередных значений образцов или элементов данных.

3.99 **предварительно кодированное изображение**: Кодированное представление *изображения,* которое должно использоваться *процессом декодирования* для подтверждения соответствия потока битов этой Рекомендации | Международному стандарту. Предварительно (ранее) кодированное изображение содержит все *макроблоки изображения.* Единственные *изображения,* которые оказывают воздействие на *процесс декодирования,* – это предварительно кодированные изображения. См. также *избыточно кодированное изображение.*

3.100 **параметры пользователя**: Особое подмножество синтаксиса этой Рекомендации | Международного стандарта.

3.101 **параметр квантования**: Переменная, используемая *процессом декодирования* для масштабирования *степеней коэффициентов преобразования*.

3.102 **произвольный доступ**: Действие, запускающее процесс декодирования *потока битов* в произвольной (не начальной) точке этого потока.

3.103 **растровое сканирование**: Отображение прямоугольного двухразмерного шаблона в одноразмерный шаблон с таким свойством, что первые входы одноразмерного шаблона получены из первого верхнего ряда двухразмерного шаблона, сканированного слева направо. Далее следуют аналогичные действия для второго, третьего и т. д. рядов (сверху вниз), каждый их которых сканируют слева направо.

3.104 **полезная нагрузка последовательности исходных байтов (RBSP)**: Синтаксическая структура, в которой содержится целое число *байтов,* инкапсулированных в блок *NAL*. Нагрузка RBSP может быть либо пустой, либо иметь форму строки битов данных с *элементами синтаксиса,* за которыми следует *стоповый бит RBSP,* а далее нуль или последующие биты, равные 0.

3.105 **стоповый бит полезной нагрузки последовательности исходных байтов (RBSP)**: Равный 1 бит, представленный в *полезной нагрузке последовательности исходных байтов (RBSP),* после *строки битов данных*. Местоположение конца *строки битов данных* в *RBSP* может быть определено поиском от конца *RBSP* стопового бита *RBSP*, который является последним ненулевым битом *RBSP.*

3.106 **точка восстановления**: Точка в *потоке битов,* в которой восстановление точного или приближенного представления *декодированных изображений,* представленных *потоком битов,* достигают с помощью *произвольного доступа* или *разомкнутого звена связи*.

3.107 **избыточное кодированное изображение**: Кодированное представление *изображения* или части *изображения.* Содержание избыточного кодированного изображения не должно использоваться *процессом декодирования* для *потока битов,* соответствующих этой Рекомендации | Международному стандарту. Избыточное кодированное изображение не требует удержания всех *макроблоков* в предварительно кодированном изображении. Избыточные кодированные изображения не оказывают воздействия на *процесс декодирования*. См. также *предварительно кодированное изображение*.

3.108 **контрольное поле**: *Контрольное поле* может быть использовано для *внешнего предсказания,* если декодированы *секции P*, *SP* и *B кодированного поля* или *поля макроблоков кодированного кадра*. См. также *контрольное изображение*.

3.109 **контрольный кадр**: *Контрольный кадр* может быть использован для *внешнего предсказания,* если декодированы *секции P*, *SP* и *B кодированного кадра*. См. также *контрольное изображение*.

3.110 **индекс контроля**: Индекс в списке *контрольных изображений*.

3.111 **контрольное изображение**: *Изображение* с синтаксисом nal_ref_idc, не равным 0. *Контрольное изображение* содержит образцы, которые могут быть использованы для *внешнего предсказания (inter предсказания)* в *процессе декодирования* последующих *изображений* в *порядке декодирования*.

3.112 **список контрольных изображений**: Список краткосрочных и долгосрочных номеров изображений, которые приписаны к *контрольным изображениям*.

3.113 **список контрольных изображений 0**: *список контрольных изображений,* использованных для *внешнего предсказания секций P*, *B* или *SP*. *Список контрольных изображений 0* использует все *внешние предсказания (inter предсказания),* которые были использованы для секций *P* и *SP*. Список контрольных изображений 0 – это один из двух списков *контрольных изображений,* использованных для *внешнего предсказания B-секции* совместно с другим *списком контрольных изображений 1*.

3.114 **список контрольных изображений 1**: *Список контрольных изображений,* использованных для *внешнего предсказания B-секции*. Список контрольных изображений 1 – это один из двух списков *контрольных изображений,* использованных для *внешнего предсказания B секции* совместно с другим *списком контрольных изображений 0.*

3.115 **метка контрольного изображения**: Определяет в потоке битов, каким образом *декодированные изображения* помечены для *внешнего предсказания*.

3.116 **зарезервировано**: В разделах, определяющих некоторые особенности *элемента синтаксиса*, термин "зарезервировано" применяют для будущего использования МСЭ-Т | ИСО/МЭК. Эти значения не должны

использоваться при описании *потоков битов,* соответствующих данной Рекомендации | Международному стандарту, но могут быть использованы в будущих расширениях этой Рекомендации | Международного стандарта организациями МСЭ-Т | ИСО/МЭК.

3.117 **разность**: Разность при декодировании между *предсказанием* элемента образца или данных и его декодированным значением.

3.118 **проход**: Число последовательных элементов данных, представленных в процессе декодирования. В одном контексте это число степеней *коэффициентов преобразования* с нулевым значением, предшествующих числу степеней *коэффициентов преобразования* с ненулевым значением в перечне степеней *коэффициентов преобразования,* который получают *сканированием зигзагом* или *сканированием полем*. В других контекстах проходом называют число *макроблоков*.

3.119 **коэффициент (внешнего) вида образца**: В процессе отображения, который не определяется в этой Рекомендации | Международном стандарте, – это отношение предполагаемого расстояния между столбцами по горизонтали к предполагаемому расстоянию между рядами по вертикали в массиве образцов яркости в *кадре*. Коэффициент вида образца выражают как h:v, где h – ширина по горизонтали, а v – высота по вертикали (в произвольных единицах пространственного расстояния).

3.120 **масштабирование**: процесс умножения *степеней коэффициентов преобразования* на множитель, в результате чего получают *коэффициенты преобразования*.

3.120.1 **установка параметра последовательности**: структура синтаксиса, содержащая *элементы синтаксиса,* которые применяют к нулевым и более полным *кодированным видеопоследовательностям,* как это определено содержанием *элемента синтаксиса* seq_parameter_set_id, который находится в *установке параметра изображения,* относящейся к *элементу синтаксиса* pic_parameter_set_id, расположенному в каждом *заголовке секции.*

3.120.2 **должен**: Термин (shall) для выражения обязательности требований соответствия этой Рекомендации | Международному стандарту. Термин используют, чтобы выразить обязательность ограничений на значения *элементов синтаксиса* или на результаты, полученные действием точно определенного *процесса декодирования.* Это ограничение выполняет кодер. При использвании в ссылке на действия, выполняемые *процессом декодирования*, подразумевают любой *процесс декодирования,* который приводит к идентичным описанным здесь результатам, соответствующим требованиям этой Рекомендации | Международного стандарта к *процессу декодирования*.

3.120.3 **следует**: Термин (should) для описания поведения при реализации, которое предполагает ординарные обстоятельства, но не обязательного требования для соответствия этой Рекомендации | Международному стандарту

3.121 **SI секция**: *Секция,* которая кодирована с использованием *предсказания* только от декодированных образцов этой же секции, а также с использованием квантования предсказанных образцов. SI секция может быть кодирована таким образом, что ее декодированные образцы будут идентичны образцам *SP секции*.

3.122 **пропущенный макроблок**: *Макроблок,* для которого отсутствуют иные данные, кроме указания, что этот *макроблок* предполагают декодировать как "пропущенный". Такое указание может быть общим для нескольких макроблоков.

3.123 **секция**: Целое число *макроблоков* или *пар макроблоков,* расположенных последовательно при *растровом сканировании* в данной группе *секций.* Для *предварительно кодированного изображения* деление каждой группы *секций* на секции представляет *разделение на части*. Хотя секция содержит *макроблоки* или *пары макроблоков,* которые поступают последовательно при *растровом сканировании* в *группе секций*, эти *макроблоки* или *пары макроблоков* не обязательно поступают последовательно при *растровом сканировании изображения*. Адреса *макроблоков* находят по адресу первого *макроблока* в секции (как представленных в *заголовке секции*) и по *отображению макроблока в группу секции.*

3.124 **деление на части данных секции**: Метод деления на части выбранных *элементов синтаксиса* в структурах синтаксиса, основанных на категориях, которые связаны с каждым *элементом синтаксиса*.

3.125 **группа секций**: Подмножество *макроблоков* или *пар макроблоков изображения*. Деление *изображения* на группы секций представляет деление *изображения* на части. Деление на части определено *отображением макроблока в группу секции*.

3.126 **блоки отображения группы секций**: Блоки *отображения макроблока в группу секции.*

3.127 **заголовок секции**: Часть *кодированной секции,* содержащая элементы данных, которые принадлежат первому или всем *макроблокам,* представленным в этой секции.

3.128 **источник**: Термин, используемый для описания видеоматериала или некоторых его атрибутов перед кодированием.

3.129 **SP секция**: *Секция,* которую кодируют, используя *внешнее предсказание* от ранее декодированных *контрольных изображений.* При этом используют не более одного *вектора движения* и *индекса контроля* для *предсказания* значений образца каждого *блока.* SP секция может быть кодирована таким образом, что ее декодированные образцы будут идентичны образцам другой SP секции или *SI секции*.

3.130 **префикс кода запуска**: Особая последовательность из трех *байтов,* равных 0x000001 и включенных в *поток байтов* в виде префикса каждого *блока NAL.* Расположение префикса кода запуска может быть использовано *декодером* для идентификации начала нового блока *NAL* и конца предыдущего блока *NAL*. Эмуляцию префиксов кода запуска в блоках *NAL* предотвращают включением *байтов предотвращения эмуляции*.

3.131 **строка битов данных (SODB)**: Последовательность некоторого числа битов, представляющих *элементы синтаксиса,* расположенные в *полезной нагрузке последовательности ряда байтов (RBSP)* перед *стоповым*

битом. В строках SODB крайний левый бит считают первым и наиболее значащим битом, а крайний правый бит – последним и наименее значащим битом.

3.132 **субмакроблок**: Одна четвертая образца *макроблока*, т. е. 8x8 *блока яркости* и двух соответствующих *блоков цветности*, каждый угол которых расположен в углу *макроблока*.

3.133 **деление субмакроблока**: Образцы *блока яркости* и двух соответствующих *блоков цветности,* которые появляются в результате *деления субмакроблока* для создания *внешнего предсказания*.

3.134 **I секция переключения**: См. SI секция.

3.135 **P секция переключения**: См. SP секция.

3.136 **элемент синтаксиса**: Элемент данных, представленных в *потоке битов*.

3.137 **структура синтаксиса**: Нуль или более *элементов синтаксиса,* представленных совместно в определенном порядке в *потоке битов.*

3.138 **верхнее поле**: Одно из двух *полей,* которые составляют *кадр*. Каждый ряд *верхнего поля* пространственно располагается непосредственно над рядом *нижнего поля*.

3.139 **верхний макроблок (из пары макроблоков)**: *Макроблок* в *паре макроблоков,* который содержит образцы верхнего ряда образцов *пары макроблоков.* Для *пары макроблоков поля* верхний макроблок представляет образцы из области *верхнего поля кадра,* который расположен в пространственной области *пары макроблоков.* Для *пары макроблоков кадра* верхний макроблок представляет образцы *кадра,* который расположен в пространственной области *пары макроблоков.*

3.140 **коэффициент преобразования**: Скалярная величина в частотной области, которая связана с конкретным одно- или двухразмерным *индексом частоты* в части *инверсного преобразования процесса декодирования*.

3.141 **степень коэффициента преобразования**: Целая величина, представляющая значение, связанное с конкретным двухразмерным *индексом частоты* в *процессе декодирования* перед *масштабированием* для вычисления значения *коэффициента преобразования*.

3.142 **универсальный уникальный идентификатор (UUID)**: Идентификатор, который является уникальным в пространстве всех универсальных уникальных идентификаторов.

3.143 **неопределенный (не установленный точно)**: Термин "неопределенный" используют в разделах, описывающих некоторые значения конкретного *элемента синтаксиса*, и указывают, что эти значения не определены в понятиях данной Рекомендации | Международного стандарта, а также не будут иметь определенного значения в будущем как составляющей части этой Рекомендации | Международного стандарта.

3.144 **переменная длина кодирования (VLC)**: Обратимая процедура энтропии кодирования, присваивающая укороченные строки битов *символам*, появление которых ожидается чаще, а удлиненные строки битов – *символам*, появление которых ожидается реже.

3.145 **сканирование зигзагом**: Особая последовательность упорядочения *степеней коэффициентов преобразования* от (примерно) самой нижней пространственной частоты к самой верхней. Сканирование зигзагом используют для *степеней коэффициентов преобразования* в *кадрах макроблоков.*

4 Сокращения

4.1 **CABAC**: Основанное на контексте адаптивное арифметическое бинарное кодирование

4.2 **CAVLC**: Основанная на контексте адаптивная переменная длина кодирования

4.3 **CBR**: Передача потока битов с постоянной скоростью

4.4 **CPB**: Буфер кодированного изображения

4.5 **DPB**: Буфер декодированного изображения

4.6 **DUT**: Тестируемый декодер

4.7 **FIFO**: "Первым прибыл – первым обслужен"

4.8 **HRD**: Гипотетический контрольный декодер

4.9 **HSS**: Гипотетический планировщик потоков

4.10 **IDR**: Мгновенное декодирующее восстановление

4.11 **LSB**: Наименее значащий бит

4.12 **MB**: Макроблок

4.13 **MBAFF**: Адаптивное декодирование макроблока кадра/поля

4.14 **MSB**: Наиболее значащий бит

4.15 **NAL**: Уровень абстракции сети

4.16 **RBSP**: Полезная нагрузка последовательности исходных байтов

4.17 **SEI**: Дополнительная расширенная информация

4.18 **SODB**: Строка битов данных

4.19 **UUID**: Универсальный уникальный идентификатор

4.20 **VBR**: Передача потока битов с переменной скоростью

4.21 **VCL**: Уровень видеокодирования

4.22 **VLC**: Переменная длина кодирования

4.23 **VUI**: Удобная в использовании визуальная информация

5 Условные обозначения

ПРИМЕЧАНИЕ. – Математические операторы, использованные в Спецификации, аналогичны тем, которые использованы в языке программирования "Си". Однако операции целочисленного деления и арифметического сдвига определены особо. Условия нумерации и счета обычно начинаются с 0.

5.1 Арифметические операторы

Ниже арифметические операторы определены следующим образом:

+ Сложение.

– Вычитание (как оператор двух аргументов) или отрицание (как оператор унарного префикса).

* Умножение.

x^y Степень. Определяет x в степени y. В других контекстах это обозначение используют для надписи, не предназначенной для интерпретации в качестве степени.

/ Целочисленное деление с усечением результата в меньшую сторону. Например, 7/4 и –7/–4 усечены до 1, а –7/4 и 7/–4 усечены до –1.

÷ Используют для обозначения деления в математических уравнениях, где не предполагают усечения или округления.

$\frac{x}{y}$ Используют для обозначения деления в математических уравнениях, где не предполагают усечения или округления.

$\sum_{i=x}^{y} f(i)$ Суммирование f(i) с i, принимающим все целые значения от x до y включительно.

x % y Модуль. Остаток от деления x на y, определенный только для целых x и y при x >= 0 и y > 0.

Если порядок старшинства не выражен явно с помощью скобок, применимы следующие правила:

– операции умножения и деления предшествуют сложению и вычитанию;

– операции умножения и деления в последовательности вычисляют в порядке слева направо;

– операции сложения и вычитания в последовательности вычисляют в порядке слева направо.

5.2 Логические операторы

Ниже логические операторы определены следующим образом:

x && y Булево логическое "и" x и y.

x | | y Булево логическое "или" x и y.

! Булево логическое "не".

x ? y : z Если x – ИСТИНА или не равно 0 вычисляется значение y. Иначе вычисление значения y заменяют вычислением z.

5.3 Операторы отношений

Ниже операторы отношений определены следующим образом:

>	Больше чем.
>=	Больше чем или равно.
<	Меньше чем.
<=	Меньше чем или равно.
= =	Равно.
!=	Не равно.

5.4 Двоичные операторы

Ниже двоичные операторы определены следующим образом:

&	Двоичное "и". При операциях с целочисленными аргументами оперирует с двумя дополняющими представлениями целого значения. При операциях с двоичным аргументом, который содержит меньше битов, чем другой аргумент, к более короткому аргументу добавляют больше значащих битов, равных 0.
|	Двоичное "или". При операциях с целочисленными аргументами оперирует с двумя дополняющими представлениями целого значения. При операциях с двоичным аргументом, который содержит меньше битов, чем другой аргумент, к более короткому аргументу добавляют больше значащих битов, равных 0.
x >> y	Арифметический сдвиг вправо двух дополняющих целочисленных представлений x с помощью y бинарных знаков. Эта функция определена только для положительных целочисленных значений y. Биты, сдвинутые к битам MSB в результате сдвига вправо, должны принимать значения, равные MSB для x до операции сдвига.
x << y	Арифметический сдвиг влево двух дополняющих целочисленных представлений x с помощью y бинарных знаков. Эта функция определена только для положительных целочисленных значений y. Биты, сдвинутые к битам LSB в результате сдвига влево, должны принимать значения, равные 0.

5.5 Операторы присвоения

Ниже арифметические операторы определены следующим образом:

=	Оператор присвоения.
+ +	Приращение, т. е. x+ + эквивалентно $x = x + 1$ при использовании в индексе массива (матрицы), вычисляют по значению переменной до операции приращения.
– –	Декремент, т. е. x– – эквивалентно $x = x - 1$ при использовании в индексе массива (матрицы), вычисляют по значению переменной до операции уменьшения.
+=	Приращение на оговоренную величину, т. е. x += 3 эквивалентно x = x + 3, а x += (–3) эквивалентно x = x + (–3).
–=	Декремент на оговоренную величину, т. е. x –= 3 эквивалентно x = x – 3, а x –= (–3) эквивалентно x = x – (–3).

5.6 Обозначение ранга

Для определения ранга значений использованы следующие обозначения:

x = y .. z x принимает целые значения, начиная от y до z включительно, где x, y и z – целые числа.

5.7 Математические функции

Ниже математические функции определены следующим образом:

$$\text{Abs}(x) = \begin{cases} x & ; \quad x >= 0 \\ -x & ; \quad x < 0 \end{cases} . \tag{5-1}$$

Ceil(x) наименьшее целое, большее или равное x. (5-2)

Clip1(x) = Clip3(0, 255, x). (5-3)

$$\text{Clip3}(x, y, z) = \begin{cases} x & ; \quad z < x \\ y & ; \quad z > y \\ z & ; \quad \text{иначе} \end{cases}. \tag{5-4}$$

Floor(x)наибольшее целое, меньшее или равное x. (5-5)

$$\text{Инверсного RasterScan}(a, b, c, d, e) = \begin{cases} (a\%(d/b)) * b; & e == 0 \\ (a/(d/b)) * c; & e == 1 \end{cases}. \tag{5-6}$$

Log2(x) логарифм от x по основанию 2. (5-7)

Log10(x) логарифму от x по основанию 10. (5-8)

Luma4x4BlkScan(x, y) = (x / 2) * 4 + (y / 2) * 8 + RasterScan(x % 2, y % 2, 2). (5-9)

Median(x, y, z) = x + y + z – Min(x, Min(y, z)) – Max(x, Max(y, z)). (5-10)

$$\text{Min}(x, y) = \begin{cases} x & ; \quad x <= y \\ y & ; \quad x > y \end{cases}. \tag{5-11}$$

$$\text{Max}(x, y) = \begin{cases} x & ; \quad x >= y \\ y & ; \quad x < y \end{cases}. \tag{5-12}$$

RasterScan(x, y, n_x) = x + y * n_x. (5-13)

Round(x) = Sign(x) * Floor(Abs(x) + 0,5). (5-14)

$$\text{Sign}(x) = \begin{cases} 1 & ; \quad x >= 0 \\ -1 & ; \quad x < 0 \end{cases}. \tag{5-15}$$

$$\text{Sqrt}(x) = \sqrt{x}. \tag{5-16}$$

5.8 Переменные, элементы синтаксиса и таблицы

Элементы синтаксиса в потоке битов представлены жирным шрифтом. Каждый элемент синтаксиса описан по его названию (все строчные буквы с подчеркнутыми знаками), по одной или двум категориям синтаксиса и одному или двум дескрипторам метода кодированного представления. Процесс декодирования протекает соответственно значению элемента синтаксиса и значениям предыдущих декодированных элементов синтаксиса. Если значение элемента синтаксиса использовано в таблицах синтаксиса или в тексте, его тип считают стандартным (т. е. не выделяют жирным шрифтом).

В некоторых случаях в таблицах синтаксиса могут использоваться значения других переменных, полученных из значений элементов синтаксиса. Такие переменные, которые появляются в таблицах синтаксиса или в тексте, обозначены комбинированными строчными и заглавными буквами без подчеркнутых знаков. Переменные, которые начинаются со строчных букв, извлекают для декодирования текущей структуры синтаксиса и всех зависимых структур синтаксиса. Переменные, которые начинаются с заглавных букв, могут быть использованы в процессе декодирования для последующих структур синтаксиса, упоминающих первоначальную структуру синтаксиса этой переменной. Переменные, которые начинаются со строчных букв, используют только в том подразделе, в котором они введены.

В некоторых случаях "мнемонические" названия значений элементов синтаксиса или значений переменных используют поочередно с их численными значениями. Иногда "мнемонические" названия используют без всякой ассоциации с численными значениями. Ассоциация значений и названий определена в тексте. Названия создают из одной или более групп букв, разделенных знаком подчеркивания. Каждая группа начинается с заглавной буквы и может содержать много заглавных букв.

ПРИМЕЧАНИЕ. – Синтаксис описан способом, который близок синтаксическим структурам языка программирования Си.

Функции описывают по их названиям, которые создают в виде названий элементов синтаксиса, заключенных в круглые скобки, включая нуль и другие названия переменных (для определения) или значений (для использования), разделенных запятыми (если имеется более одной переменной).

Квадратные скобки используют для индексации списков или массивов. Списки или массивы могут быть либо элементами синтаксиса, либо переменными. Двухразмерные массивы иногда также описывают, используя матричное обозначение с подстрочными знаками для индексации.

ПРИМЕЧАНИЕ. – Для порядка индекса в двухразмерных массивах используют квадратные скобки, а подстрочный индекс меняют местами. Пример с положением x по горизонтали и y по вертикали двухразмерного массива образца обозначают как s[x, y], что соответствует матричной записи s_{yx}.

Двоичное обозначение указывают, заключая строку значений битов в однократные кавычки. Например, '01000001' представляет строку из восьми битов, в которой только второй и последний бит равны 1.

Шестнадцатеричная запись, отмеченная префиксом с шестнадцатеричным числом "0x", может быть использована вместо двоичной записи, если число битов четное и кратно 4. Например, 0x41 представляет строку из восьми битов, в которой только второй и последний бит равен 1.

Численные значения, не заключенные в однократные кавычки и без префикса "0x", являются децимальными значениями.

Значение, равное 0, в сообщении тестирования представляет ЛОЖЬ. Значение ИСТИНА представлено любыми другими отличными от нуля значениями.

5.9 Описание логических операций

В этом тексте логические операции будут описаны как бы в псевдокоде следующим образом:

```
если ( условие 0 )
утверждение 0
если еще (условие 1 )
утверждение 1
…
еще /* информативная ремарка остающегося условия */
утверждение n
```

может быть описано следующим образом:

... как следует / ... применимо следующее:

- Если условие 0, утверждение 0
- Иначе, если условие 1, утверждение 1
- …
- Иначе (информативная ремарка остающегося условия), утверждение n.

Каждое утверждение "Если...Иначе, если...Иначе, ..." в этом тексте вводят с помощью "... как следует" или "... применимо следующее", которые непосредственно следуют за "Если ... ". Последнее условие "Если...Иначе, если...Иначе, ..." всегда является "Иначе, ...". Чередующиеся утверждения "Если...Иначе, если...Иначе, ..." можно описать подходящим "... как следует " или "... применимо следующее" с окончанием "Иначе, ...".

В этом тексте логические операции будут описаны как бы в псевдокоде следующим образом:

```
если ( условие 0a && условие 0b )
утверждение 0
если еще (условие 1a | | условие 1b )
утверждение 1
…
еще
утверждение n
```

может быть описано следующим образом:

... как следует / ... применимо следующее:

- Если все следующие условия истинны, то утверждение 0
  - условие 0a
  - условие 0b.
- Иначе, если любое из следующих условий истинно, то утверждение 1
  - условие 1a
  - условие 1b.
- …
- Иначе утверждение n.

В этом тексте логические операции будут описаны как бы в псевдокоде следующим образом:

```
Если (условие 0 )
  утверждение 0
Если (условие 1 )
  утверждение 1
```

может быть описано следующим образом:

Когда условие 0, утверждение 0

Когда условие 1, утверждение 1.

5.10 Процессы

Процессы используют для описания декодирования элементов синтаксиса. Процесс имеет отдельную спецификацию и осуществление. Все элементы синтаксиса и заглавных переменных, которые принадлежат текущей структуре синтаксиса и зависимых структур синтаксиса, доступны в процессе спецификации и осуществления. Процесс спецификации может также иметь строчные переменные, точно описанные на входе. Каждый процесс спецификации имеет однозначно описанный выход. Выход – это переменная, которая может быть либо заглавной, либо строчной переменной.

Назначение переменных описывают следующим образом:

– В процессе осуществления переменным однозначно присваивают строчные знаки переменных на входе и выходе спецификации процесса в случае, если эти переменные не имеют одних и тех же названий.

– Иначе (если переменные при осуществлении и спецификации имеют одни и те же названия), вводят присвоение.

При спецификации процесса конкретный макроблок может рассматриваться по названию переменной, имеющей значение, равное адресу этого конкретного макроблока.

6 Форматы источника, кодированных, декодированных и выходных данных, процессы сканирования и взаимоотношения смежности

6.1 Форматы потоков битов

В этом подразделе описаны взаимоотношения между потоком блоков NAL и потоком байтов, причем каждый из них рассматривают как поток битов.

Поток битов может иметь один из двух форматов: формат потока блоков NAL или формат потока байтов. Формат потока блоков NAL концептуально является более "основным" типом. Он состоит из последовательности структур синтаксиса, которые называются блоками NAL. Эта последовательность расположена в порядке декодирования. Для потока блоков NAL на порядок декодирования (и содержание) блоков NAL введен ряд ограничений.

Формат потока байтов может быть создан из формата потока блоков NAL расстановкой блоков NAL в порядке декодирования и заданием префиксов каждого блока NAL с помощью префикса кода старта, нуля и дополнительных байтов с нулевым значением для формирования потока байтов. Формат потока блоков NAL можно извлечь из формата потока байтов отысканием расположения особого образца префикса кода старта среди этого потока байтов. Методы кадрирования блоков NAL способом, отличным от того, который используют для формата потока байтов, выходит за рамки этой Рекомендации | Международного стандарта. Формат потока байтов описан в Приложении B.

6.2 Форматы источника, декодированного и выходного изображений

В этом подразделе описаны взаимосвязи между источником и декодированными кадрами и полями, которые рассмотрены с помощью потока битов.

Источник видеосигнала, который представлен потоком битов, является последовательностью как кадров, так и полей (собирательно называемых изображениями) в порядке их декодирования.

Каждый источник и декодированные изображения (кадры или поля) состоят из трех массивов образцов: одного массива яркости и двух – цветности.

Переменная ChromaFormatFactor определена в таблице 6-1 в зависимости от структуры выбранного формата яркости. Значение ChromaFormatFactor должно быть, очевидно, равно 1,5, указывая на метод выборки 4:2:0. При монохромной выборке имеется только один массив образцов, который можно номинально рассматривать как массив яркости. При выборке 4:2:0 каждый из двух массивов яркости занимает половину высоты и половину ширины массива яркости. При выборке 4:2:2 каждый из двух массивов цветности занимает ту же высоту и половину ширины массива яркости. При выборке 4:4:4 каждый из двух массивов цветности занимает ту же высоту и ширину, что и массив яркости.

ПРИМЕЧАНИЕ. – Для будущей версии этой Рекомендации | Международного стандарта могут быть действительны другие значения.

Таблица 6-1 – Значения ChromaFormatFactor

Формат цветности	ChromaFormatFactor
монохромный	1
4:2:0	1,5
4:2:2	2
4:4:4	3

Эта Рекомендация | Международный стандарт представляет цвет последовательностями, используя цветовую выборку 4:2:0. Ширина массива образца яркости каждого изображения – это целое число, кратное 16. Ширина массива образца цветности каждого изображения – это целое число, кратное 8. Высота массива образца яркости каждого кодированного изображения (как для кодированного кадра, так и для кодированного поля) – это целое число, кратное 16, а высота массива образца цветности для этих изображений – это целое число, кратное 8. Если же любые кодированные кадры представлены в кодированной видеопоследовательности, которая содержит кодированные поля или кадры, использующие адаптивное кодирование макроблока кадра/поля, высота массива образца яркости всех кодированных кадров в кодированной видеопоследовательности – это целое число, кратное 32. Высота каждого массива образца цветности этих кадров – это целое число, кратное 16. Ширина или высота изображений выхода процесса декодирования не должна быть целым числом, кратным 16 и не может быть описана с использованием масштабированного прямоугольника.

Ширина полей, кодированных по специальной установке параметров этой последовательности, будет такой же, как и кадров, кодированных по той же установке параметров этой последовательности (см. ниже). Высота полей, кодированных по специальной установке параметров этой последовательности составляет половину от высоты кадров, кодированных по той же установке параметров этой последовательности (см. ниже).

Номинальное вертикальное и горизонтальное относительное расположение образцов яркости и цветности в кадрах показано на рисунке 6-1. Альтернативные относительные расположения образцов цветности можно найти в удобной в использовании визуальной информации (см. Приложение E).

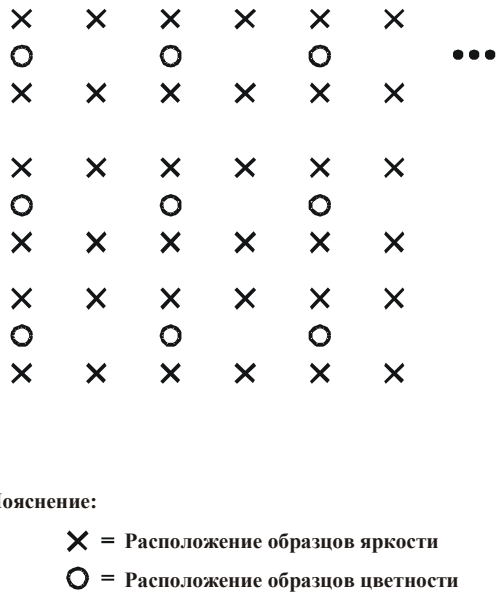

Рисунок 6-1 – Номинальное вертикальное и горизонтальное расположение образцов яркости и цветности 4:2:0 в кадре

Кадр состоит из двух полей, как описано ниже. Кодированное изображение может представлять кодированный кадр или отдельное кодированное поле. Кодированная видеопоследовательность, соответствующая этой Рекомендации | Международному стандарту, может содержать произвольные комбинации кодированных кадров и кодированных

полей. Процесс декодирования также описан способом, который позволяет кодировать малые области кодированного кадра как область либо кадра, либо поля, используя адаптивное кодирование макроблока кадра/поля.

Источник и декодированные поля бывают двух типов: верхнее поле и нижнее поле. Если два поля появляются на выходе в одно и то же время или их комбинируют, чтобы использовать как контрольный кадр (см. ниже), эти два поля (каждое должно быть типа инверсного равенства) чередуют. Первый (т. е. верхний), третий, пятый и т. д. ряды декодированных кадров являются верхними рядами полей. Второй, четвертый, шестой и т. д. ряды декодированного кадра являются нижними рядами полей. Первый (т. е. верхний) ряд нумеруют как ряд номер 0. Второй ряд нумеруют как ряд номер 1. Третий ряд нумеруют как ряд номер 2 и т. д. Верхнее поле состоит только из верхних рядов полей кадра, а нижнее поле – только из нижних рядов полей кадра. Если верхнее или нижнее поле декодированного кадра используют как контрольное поле (см. ниже), то используют только четно пронумерованные ряды (для верхнего поля) или нечетно пронумерованные ряды (для нижнего поля) декодированного кадра.

Номинальное вертикальное и горизонтальное относительное расположение образцов яркости и цветности в верхних и нижних полях показано на рисунке 6-2. Номинальные вертикальные выборки относительного расположения образцов цветности в верхнем поле описаны как сдвинутые на одну четверть высоты образца яркости относительно сетки выборки полей. Расположение вертикальной выборки образцов цветности в нижнем поле описано как сдвинутое вниз на одну четверть высоты образца яркости относительно сетки выборки полей. Альтернативные относительные расположения образцов цветности можно найти в удобной в использовании визуальной информации (см. Приложение E).

ПРИМЕЧАНИЕ. – Сдвиг образцов цветности происходит в порядке выравнивания этих образцов вертикально к обычному расположению относительно сетки выборки полного кадра, как показано на рисунке 6-1.

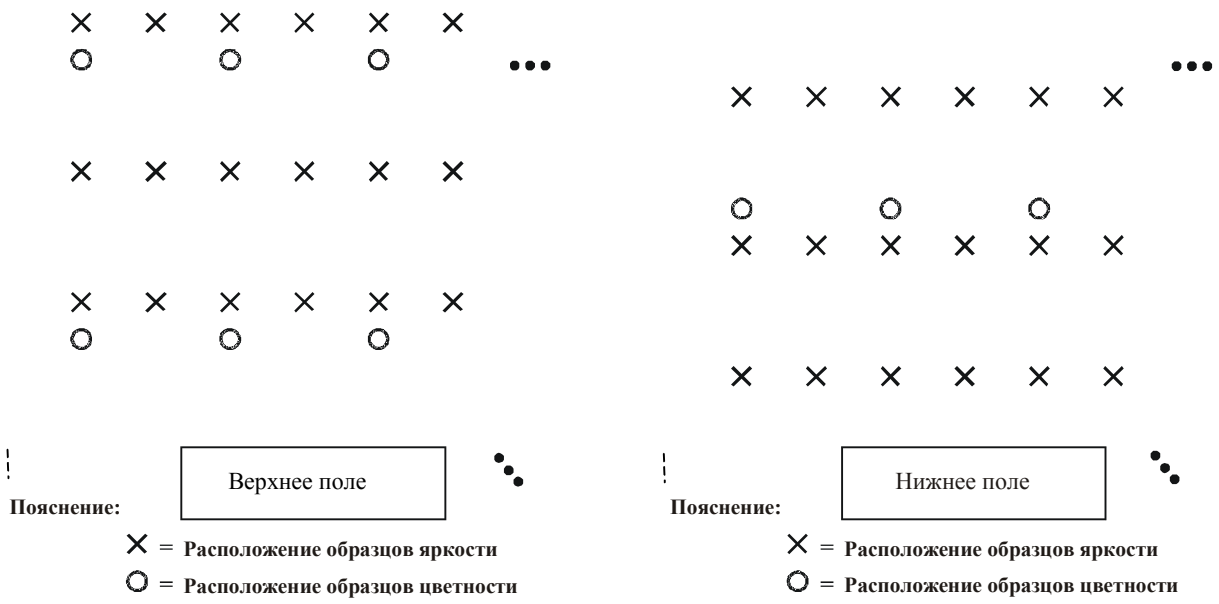

Рисунок 6-2 – Номинальное вертикальное и горизонтальное расположение выборки образцов верхних и нижних полей

6.3 Пространственное разделение на части изображений и секций

В этом подразделе описано, как изображение разделяют на секции и макроблоки. Изображение делят на секции. Секция – это последовательность макроблоков или, при использовании адаптивного декодирования макроблока кадра/поля, это последовательность пары макроблоков.

Каждый макроблок включает один массив образца яркости 16x16 и два 8x8 цветности. Если адаптивное декодирование макроблока кадра/поля не используют, каждый макроблок представляет пространственную прямоугольную область изображения. Например, изображение может быть разделено на две секции, как показано на рисунке 6-3.

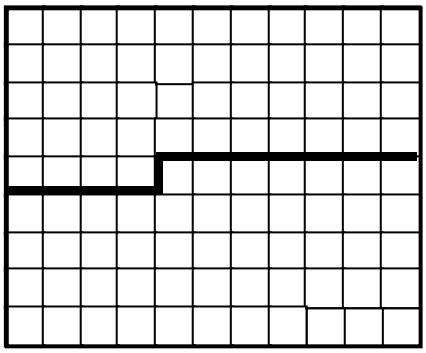

Рисунок 6-3 – Изображение с 11 на 9 макроблоков, которые разделены на две секции

Если используют адаптивное декодирование макроблока кадра/поля, изображение разделяют на секции, содержащие целое число пар макроблоков, как показано на рисунке 6-4. Каждая пара макроблоков состоит из двух макроблоков.

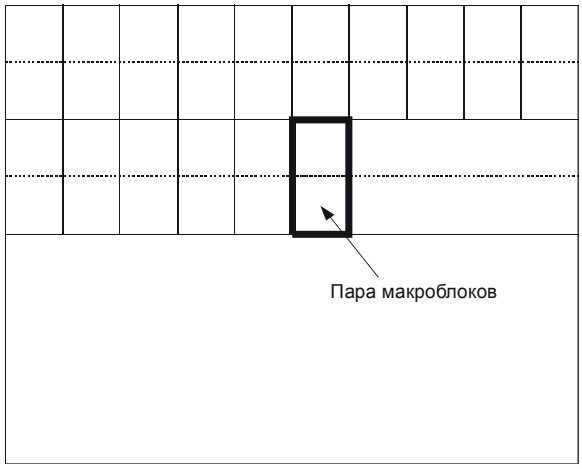

Рисунок 6-4 – Разделение декодированного кадра на пару макроблоков

6.4 Процессы инверсного сканирования и процессы образования смежных частей

В этом подразделе описаны процессы инверсного сканирования, т. е. размещение индексов на их места и процессы ответвления для смежных блоков.

6.4.1 Процесс инверсного сканирования макроблока

Вход этого процесса – адрес макроблока mbAddr.

Выход этого процесса – местоположение (x, y) левого верхнего образца яркости макроблока с адресом mbAddr относительно левого верхнего образца изображения.

Процесс инверсного сканирования макроблока описан следующим образом:

– Если MbaffFrameFlag равен 0,

$$x = \text{InverseRasterScan}(\text{mbAddr}, 16, 16, \text{PicWidthInSamples}_L, 0) \quad (6\text{-}1)$$

$$y = \text{InverseRasterScan}(\text{mbAddr}, 16, 16, \text{PicWidthInSamples}_L, 1). \quad (6\text{-}2)$$

– Иначе (MbaffFrameFlag равно 1), применяют следующее:

$$xO = \text{InverseRasterScan}(\text{mbAddr} / 2, 16, 32, \text{PicWidthInSamples}_L, 0) \quad (6\text{-}3)$$

yO = InverseRasterScan(mbAddr / 2, 16, 32, $PicWidthInSamples_L$, 1). (6-4)

В зависимости от типа текущего макроблока применяют следующее:

- Если текущий макроблок – макроблок кадра,

  x = xO (6-5)

  y = yO + (mbAddr % 2) * 16. (6-6)

- Иначе (текущий макроблок – макроблок поля),

  x = xO (6-7)

  y = yO + (mbAddr % 2). (6-8)

6.4.2 Процесс инверсного сканирования при разделении макроблока и субмакроблока

Макроблоки или субмакроблоки могут быть разделены на части, а части сканированы для внешнего предсказания, как показано на рисунке 6-5. Внешние прямоугольники относят соответственно к образцам макроблока или субмакроблока. Прямоугольники относят к частям. Число в каждом прямоугольнике определяет индекс инверсно сканированной части макроблока или субмакроблока.

Функции MbPartWidth( ), MbPartHeight( ), SubMbPartWidth( ) и SubMbPartHeight( ), описывающие ширину и высоту разделенной части макроблока и субмакроблока, указаны в таблице 7-10, таблице 7-11, таблице 7-14 и таблице 7-15. Функции MbPartWidth( ) и MbPartHeight( ) устанавливают соответствующие значения для каждого макроблока в зависимости от его типа. Функции SubMbPartWidth( ) и SubMbPartHeight( ) устанавливают соответствующие значения для каждого субмакроблока макроблока с типом, равным P_8x8, P_8x8ref0 или B_8x8 в зависимости от типа субмакроблока.

Рисунок 6-5 – Разделенные части макроблоков и субмакроблоков. Сканированные разделенные части макроблоков и субмакроблоков

6.4.2.1 Процесс инверсного сканирования при разделении макроблока

Вход этого процесса – индекс разделения макроблока mbPartIdx.

Выход этого процесса – местоположение (x, y) левого верхнего образца яркости разделенного макроблока mbPartIdx относительно левого верхнего образца этого макроблока.

Процесс инверсного сканирования макроблока описывают следующим образом:

x = InverseRasterScan(mbPartIdx, MbPartWidth(mb_type), MbPartHeight(mb_type), 16, 0) (6-9)

y = InverseRasterScan(mbPartIdx, MbPartWidth(mb_type), MbPartHeight(mb_type), 16, 1). (6-10)

6.4.2.2 Процесс инверсного сканирования при разделении субмакроблока

Входы этого процесса – индексы разделенной части макроблока mbPartIdx и индекс разделенной части субмакроблока.

Выход этого процесса – местоположение (x, y) левого верхнего образца яркости разделенного субмакроблока subMbPartIdx относительно левого верхнего образца этого субмакроблока.

Процесс инверсного сканирования субмакроблока описывают следующим образом:

– Если mb_type равен P_8x8, P_8x8ref0 или B_8x8, то

x = InverseRasterScan(subMbPartIdx, SubMbPartWidth(sub_mb_type[mbPartIdx]), SubMbPartHeight(sub_mb_type[mbPartIdx]), 8, 0) (6-11)

y = InverseRasterScan(subMbPartIdx, SubMbPartWidth(sub_mb_type[mbPartIdx]), SubMbPartHeight(sub_mb_type[mbPartIdx]), 8, 1). (6-12)

– Иначе

x = InverseRasterScan(subMbPartIdx, 4, 4, 8, 0) (6-13)

y = InverseRasterScan(subMbPartIdx, 4, 4, 8, 1). (6-14)

6.4.3 Процесс инверсного сканирования блока яркости 4x4

Входы этого процесса – индексы блока яркости 4x4 luma4x4BlkIdx.

Выход этого процесса – местоположение (x, y) левого верхнего образца блока яркости 4x4 с индексом яркости 4x4BlkIdx относительно левого верхнего образца этого макроблока.

Рисунок 6-6 показывает сканирование для блоков яркости 4x4.

0	1	4	5
2	3	6	7
8	9	12	13
10	11	14	15

Рисунок 6-6 – Сканирование блоков яркости 4x4

Процесс инверсного сканирования блока яркости 4x4 описывают следующим образом:

x = InverseRasterScan(luma4x4BlkIdx / 4, 8, 8, 16, 0) + InverseRasterScan(luma4x4BlkIdx % 4, 4, 4, 8, 0) (6-15)

y = InverseRasterScan(luma4x4BlkIdx / 4, 8, 8, 16, 1) + InverseRasterScan(luma4x4BlkIdx % 4, 4, 4, 8, 1). (6-16)

6.4.4 Процесс создания доступного адреса макроблока

Вход этого процесса – индексы адреса макроблока mbAddr.

Выход этого процесса – доступный адрес макроблока mbAddr.

ПРИМЕЧАНИЕ. – Процесс доступности определяется при реализации.

Макроблок отмечают как доступный до тех пор, пока не будет истинным одно из следующих условий, при которых макроблок должен быть отмечен как недоступный:

– mbAddr < 0

– mbAddr > CurrMbAddr

– адрес макроблока mbAddr принадлежит к другой, а не к текущей секции.

6.4.5 Процесс создания адреса смежного макроблока и его доступность

Этот процесс может быть реализован при MbaffFrameFlag, равном 0.

Выходы этого процесса:

– mbAddrA: адрес и статус доступности макроблока слева от текущего макроблока;

– mbAddrB: адрес и статус доступности макроблока выше от текущего макроблока;

– mbAddrC: адрес и статус доступности макроблока справа от текущего макроблока;

– mbAddrD: адрес и статус доступности макроблока над левым текущим макроблоком.

Рисунок 6-7 показывает относительные пространственные расположения макроблоков с адресами mbAddrA, mbAddrB, mbAddrC и mbAddrD относительно текущего макроблока с адресом CurrMbAddr.

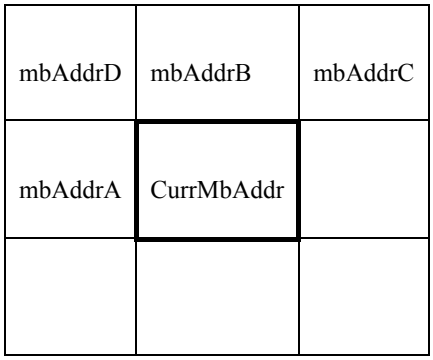

Рисунок 6-7 – Смежные макроблоки по отношению к заданному макроблоку

Вход процесса согласно п. 6.4.4 – это mbAddrA = CurrMbAddr – 1, а выход (в случае доступности) – макроблок mbAddrA. Кроме того, mbAddrA отмечают как недоступный, если CurrMbAddr % PicWidthInMbs равно 0.

Вход процесса согласно п. 6.4.4 – это mbAddrB = CurrMbAddr – PicWidthInMbs, а выход (в случае доступности) – макроблок mbAddrB.

Вход процесса согласно п. 6.4.4 – это mbAddrC = CurrMbAddr – PicWidthInMbs + 1, а выход (в случае доступности) – макроблок mbAddrC. Кроме того, mbAddrC отмечают как недоступный, если (CurrMbAddr + 1) % PicWidthInMbs равно 0.

Вход процесса согласно п. 6.4.4 – это mbAddrD = CurrMbAddr – PicWidthInMbs – 1, а выход (в случае доступности) – макроблок mbAddrD. Кроме того, mbAddrD отмечают как недоступный, если CurrMbAddr % PicWidthInMbs равно 0.

6.4.6 Процесс создания адреса смежного макроблока и его доступность в кадрах MBAFF

Этот процесс может быть реализован при MbaffFrameFlag, равном 1.

Выходы этого процесса:

– mbAddrA: адрес и статус доступности верхнего макроблока из пары макроблоков слева от текущей пары макроблоков;

– mbAddrB: адрес и статус доступности верхнего макроблока из пары макроблоков над текущей парой макроблоков;

– mbAddrC: адрес и статус доступности верхнего макроблока из пары макроблоков сверху справа от текущей пары макроблоков;

– mbAddrD: адрес и статус доступности верхнего макроблока из пары макроблоков сверху слева от текущей пары макроблоков.

Рисунок 6-8 показывает относительные пространственные расположения макроблоков с адресами mbAddrA, mbAddrB, mbAddrC и mbAddrD относительно текущего макроблока с адресом CurrMbAddr.

Адреса mbAddrA, mbAddrB, mbAddrC и mbAddrD имеют идентичные значения независимо от того, находится ли текущий макроблок наверху, внизу или в паре макроблоков.

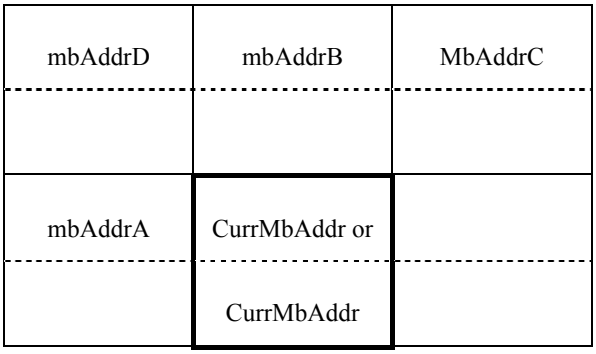

Рисунок 6-8 – Смежные макроблоки по отношению к заданному макроблоку в кадрах MBAFF

Вход процесса согласно п. 6.4.4 – это mbAddrA =2 * (CurrMbAddr / 2 – 1), а выход (в случае доступности) – макроблок mbAddrA. Кроме того, mbAddrA отмечают как недоступный, если (CurrMbAddr / 2) % PicWidthInMbs равно 0.

Вход процесса согласно п. 6.4.4 – это mbAddrB =2 * (CurrMbAddr / 2 – PicWidthInMbs), а выход (в случае доступности) – макроблок mbAddrB.

Вход процесса согласно п. 6.4.4 – это mbAddrC = 2 * (CurrMbAddr / 2 – PicWidthInMbs + 1) а выход (в случае доступности) – макроблок mbAddrC. Кроме того, mbAddrC отмечают как недоступный, если (CurrMbAddr / 2 + 1) % PicWidthInMbs равно 0.

Вход процесса согласно п. 6.4.4 – это mbAddrD = 2 * (CurrMbAddr / 2 – PicWidthInMbs – 1), а выход (в случае доступности) – макроблок mbAddrD. Кроме того, mbAddrD отмечают как недоступный, если (CurrMbAddr / 2) % PicWidthInMbs равно 0.

6.4.7 Процессы создания смежных макроблоков, блоков и разделения на части

Пункт 6.4.7.1 определяет процесс создания смежных макроблоков.

Пункт 6.4.7.2 определяет процесс создания смежных блоков яркости 8x8.

Пункт 6.4.7.3 определяет процесс создания смежных блоков яркости 4x4.

Пункт 6.4.7.4 определяет процесс создания смежных блоков цветности 4x4.

Пункт 6.4.7.5 определяет процесс разделения на части смежных блоков.

В таблице 6-2 приведены значения разности расположения яркости (xD, yD) для входа и замены N в параметрах mbAddrN, mbPartIdxN, subMbPartIdxN, luma8x8BlkIdxN, luma4x4BlkIdxN и chroma4x4BlkIdxN на выходе. Эти присвоения входа и выхода взяты из пп. 6.4.7.1–6.4.7.5. Переменную predPartWidth описывают при ссылках на таблицу 6-2.

Таблица 6-2 – Спецификация присвоений входов и выходов в пп. 6.4.7.1–6.4.7.5

N	xD	yD
A	–1	0
B	0	–1
C	predPartWidth	–1
D	–1	–1

Рисунок 6-9 иллюстрирует расположение смежных макроблоков, блоков или разделение на части A, B, C и D текущего макроблока, частей или блоков, если текущий макроблок, часть или блок появляются в режиме кодирования кадра.

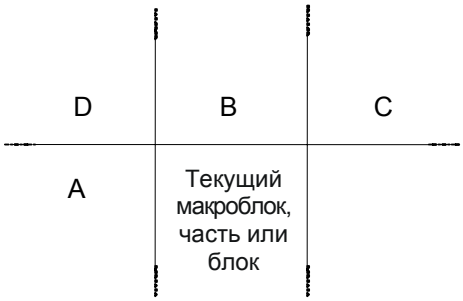

Рисунок 6-9 – Определение смежных макроблоков, блоков и разделенных частей (информативное)

6.4.7.1 Процесс создания смежных макроблоков

Выходы этого процесса:

– mbAddrA: адрес макроблока слева от текущего макроблока и статус его доступности; и

– mbAddrB: адрес макроблока над текущим макроблоком и статус его доступности.

mbAddrN (при N, равном A или B) создают следующим образом:

– Разность расположения яркости (xD, yD) устанавливают в соответствии с таблицей 6-2.

– Процесс создания смежных расположений, описанный в п. 6.4.8, реализуют для расположения яркости с (xN, yN), равным (xD, yD), а выходу приписывают mbAddrN.

6.4.7.2 Процесс создания смежных блоков яркости 8x8

Вход этого процесса – блок яркости 8x8 с индексом luma8x8BlkIdx.

Индекс luma8x8BlkIdx определяет блок яркости 8x8 макроблока при растровом сканировании.

Выходы этого процесса:

– mbAddrA: либо равен CurrMbAddr, либо адресу макроблока слева от текущего макроблока и статусу его доступности;

– luma8x8BlkIdxA: индекс блок яркости 8x8 слева от блока 8x8 с индексом luma8x8BlkIdx и статус его доступности;

– mbAddrB: либо равен CurrMbAddr, либо адресу макроблока над текущим макроблоком и статусу его доступности;

– luma8x8BlkIdxB: индекс блок яркости 8x8 над блоком 8x8 с индексом luma8x8BlkIdx и статус его доступности.

mbAddrN и luma8x8BlkIdxN (при N, равном A или B) создают следующим образом:

– Разность расположения яркости (xD, yD) устанавливают в соответствии с таблицей 6-2.

– Расположение яркости (xN, yN) описывают следующим образом:

$$xN = (luma8x8BlkIdx \% 2) * 8 + xD \quad (6\text{-}17)$$

$$yN = (luma8x8BlkIdx / 2) * 8 + yD. \quad (6\text{-}18)$$

– Процесс создания смежных блоков, как это описано в п. 6.4.8, реализуют при местоположении яркости в (xN, yN), поскольку входу и выходу присвоены параметры mbAddrN и (xW, yW).

– Переменную luma8x8BlkIdxN определяют следующим образом:

  – Если mbAddrN недоступно, то luma8x8BlkIdxN помечают как недоступное.

  – Иначе (mbAddrN доступно), блоку яркости 8x8 в макроблоке mbAddrN, перекрывающем местоположение яркости (xW, yW), должен быть присвоен параметр luma8x8BlkIdxN.

6.4.7.3 Процесс создания смежных блоков яркости 4x4

Вход этого процесса – блок яркости 4x4 с индексом luma4x4BlkIdx.

Выходы этого процесса:

– mbAddrA: либо равен CurrMbAddr, либо адресу макроблока слева от текущего макроблока и статусу его доступности;

– luma4x4BlkIdxA: индекс блок яркости 4x4 слева от блока 4x4 с индексом luma4x4BlkIdx и статус его доступности;

– mbAddrB: либо равен CurrMbAddr, либо адресу макроблока над текущим макроблоком и статусу его доступности;

– luma4x4BlkIdxB: индекс блок яркости 4x4 над блоком 4x4 с индексом luma4x4BlkIdx и статус его доступности.

mbAddrN и luma4x4BlkIdxN (при N равном A или B) создают следующим образом:

– Разность расположения яркости (xD, yD) устанавливают в соответствии с таблицей 6-2.

– Процесс инверсного сканирования блока яркости 4x4, как это описано в п. 6.4.3, реализуют с яркостью 4x4BlkIdx на входе и координатами (x, y) на выходе.

– Местоположение яркости (xN, yN) описано следующим образом:

$$xN = x + xD \tag{6-19}$$

$$yN = y + yD. \tag{6-20}$$

– Процесс создания смежных блоков, как это описано в п. 6.4.8, реализуют при местоположении яркости в (xN, yN), поскольку входу и выходу присвоены параметры mbAddrN и (xW, yW).

– Переменную luma4x4BlkIdxN создают следующим образом:

  – Если mbAddrN недоступно, то luma8x8BlkIdxN помечают как недоступное.

  – Иначе (mbAddrN доступно), блоку яркости 4x4 в макроблоке mbAddrN, покрывающим местоположение яркости (xW, yW), должен быть присвоен параметр luma4x4BlkIdxN.

6.4.7.4 Процесс создания смежных блоков цветности 4x4

Вход этого процесса – блок цветности 4x4 с индексом chroma4x4BlkIdx.

Выходы этого процесса:

– mbAddrA: либо равен CurrMbAddr, либо адресу макроблока слева от текущего макроблока и статусу его доступности;

– chroma4x4BlkIdxA: индекс блока цветности 4x4 слева от блока 4x4 с индексом chroma4x4BlkIdx и статус его доступности;

– mbAddrB: либо равен CurrMbAddr, либо адресу макроблока над текущим макроблоком и статусу его доступности,

– chroma4x4BlkIdxB: индекс блока цветности 4x4 над блоком цветности 4x4 с индексом chroma4x4BlkIdx и статус его доступности.

Процесс создания смежных блоков яркости 8x8 реализуют при luma8x8BlkIdx = chroma4x4BlkIdx на входе и при mbAddrA, chroma4x4BlkIdxA = luma8x8BlkIdxA, mbAddrB и chroma4x4BlkIdxB = luma8x8BlkIdxB на выходе.

6.4.7.5 Процесс создания смежных блоков при разделении

Входы этого процесса:

– индекс разделенного макроблока mbPartIdx;

– тип текущего субмакроблока currSubMbType;

– индекс разделенного субмакроблока subMbPartIdx;

Выходы этого процесса:

– mbAddrA\mbPartIdxA\subMbPartIdxA: определяющий разделенные макроблок или субмакроблок слева от текущего макроблока и статус доступности или разделенную часть субмакроблока CurrMbAddr\mbPartIdx\subMbPartIdx и статус доступности;

– mbAddrB\mbPartIdxB\subMbPartIdxB: определяющий разделенные макроблок или субмакроблок над текущим макроблоком и статус доступности или разделенную часть субмакроблока CurrMbAddr\mbPartIdx\subMbPartIdx и статус доступности;

– mbAddrC\mbPartIdxC\subMbPartIdxC: определяющий разделенные макроблок или субмакроблок справа сверху над текущим макроблоком и статус доступности или разделенную часть субмакроблока CurrMbAddr\mbPartIdx\subMbPartIdx и статус доступности;

– mbAddrD\mbPartIdxD\subMbPartIdxD: определяющий разделенные макроблок или субмакроблок слева сверху над текущим макроблоком и статус доступности или разделенную часть субмакроблока CurrMbAddr\mbPartIdx\subMbPartIdx и статус доступности.

mbAddrN, mbPartIdxN и subMbPartIdx (при N, равном A, B, C или D) определяют следующим образом:

– Процесс инверсного сканирования разделенного макроблока, как описано в п. 6.4.2.1, реализован с mbPartIdx на входе и (x, y) на выходе.

– Местоположение левого верхнего образца яркости внутри разделенного макроблока (xS, yS) определяют следующим образом:

  – Если mb_type равно P_8x8, P_8x8ref0 или B_8x8, процесс инверсного сканирования разделенного субмакроблока, как описано в п. 6.4.2.2, реализуют с subMbPartIdx на входе и (xS, yS) на выходе.

  – Иначе (xS, yS) устанавливают на (0, 0).

– Переменная predPartWidth в таблице 6-2 описана следующим образом:

  – Если mb_type равно P_Skip, B_Skip или B_Direct_16x16, то predPartWidth = 16.

  – Иначе, если mb_type равно B_8x8, применяют следующее:

    – Если currSubMbType равно B_Direct_8x8, то predPartWidth = 16.

      ПРИМЕЧАНИЕ. – Если currSubMbType равно B_Direct_8x8, а direct_spatial_mv_pred_flag равно 1, то предсказанный вектор движения является предсказанным вектором движения для всего макроблока.

    – Иначе predPartWidth = SubMbPartWidth(sub_mb_type[mbPartIdx]).

  – Иначе, если mb_type равно P_8x8 или P_8x8ref0,
    predPartWidth = SubMbPartWidth(sub_mb_type[mbPartIdx]).

  – Иначе predPartWidth = MbPartWidth(mb_type).

– Разность расположения яркости (xD, yD) устанавливают в соответствии с таблицей 6-2.

– Расположение смежных блоков яркости (xN, yN) описывают следующим образом:

$$xN = x + xS + xD \quad (6\text{-}21)$$

$$yN = y + yS + yD. \quad (6\text{-}22)$$

– Процесс создания смежных местоположений, как это описано в п. 6.4.8, реализуют для местоположений яркости (xN, yN) на входе и присваивают выходу mbAddrN и (xW, yW).

– В зависимости от mbAddrN применяют следующее:

  – Если mbAddrN недоступно, то разделенные макроблок или субмакроблок mbAddrN\mbPartIdxN\subMbPartIdxN помечают как недоступные.

  – Иначе (mbAddrN доступно), применяют следующее:

    – Разделенной части макроблока в макроблоке mbAddrN, перекрывающей местоположение яркости (xW, yW), должно быть присвоено mbPartIdxN, а разделенной части субмакроблока внутри разделенной части макроблока mbPartIdxN, перекрывающей местоположение образца (xW, yW) в макроблоке mbAddrN, должно быть присвоено subMbPartIdxN.

    – Если разделение, заданное mbPartIdxN и subMbPartIdxN, еще не декодировано, разделенные части макроблока mbPartIdxN и субмакроблока subMbPartIdxN помечают как недоступные.

ПРИМЕЧАНИЕ. – Последнее условие существует, например, для случая, когда mbPartIdx = 2, subMbPartIdx = 3, xD = 4, yD = –1, т. е. когда запрашивают смежный блок C из последнего блока яркости 4x4 третьего субмакроблока.

6.4.8 Процесс создания смежных местоположений

Вход этого процесса – местоположение яркости или цветности (xN, yN), выраженное относительно верхнего левого угла текущего макроблока.

Выходы этого процесса:

– mbAddrN: равно либо CurrMbAddr, либо адресу соседнего макроблока, который содержит (xN, yN), и статус его доступности;

– (xW, yW): местоположение (xN, yN), выраженное относительно левого верхнего угла макроблока mbAddrN (в большей степени, чем относительно левого верхнего угла текущего макроблока).

Положим, что maxWH – переменная, определяющая максимальное значение компонентов местоположения xN, yN, xW и yW. Тогда maxWH определяют следующим образом:

– Если этот процесс реализован для смежных местоположений яркости, то

$$maxWH = 16. \tag{6-23}$$

– Иначе (этот процесс реализован для смежных местоположений цветности),

$$maxWH = 8. \tag{6-24}$$

В зависимости от переменной MbaffFrameFlag смежные расположения яркости находят следующим образом:

– Если MbaffFrameFlag равен 0, применяют спецификацию для смежных местоположений яркости в полях и кадрах (но не кадрах MBAFF), как описано в п. 6.4.8.1.

– Иначе (MbaffFrameFlag равен 1), применяют спецификацию для смежных местоположений в полях и кадрах MBAFF, как описано в п. 6.4.8.2.

6.4.8.1 Спецификация для смежных местоположений яркости в полях и кадрах (но не кадрах MBAFF)

Спецификацию этого пункта применяют, когда MbaffFrameFlag равен 0.

В п. 6.4.5 процесс создания адресов смежных макроблоков и их доступность реализуют с помощью mbAddrA, mbAddrB, mbAddrC и mbAddrD так же, как и статус доступности на выходе.

Таблица 6-3 определяет mbAddrN в зависимости от (xN, yN).

Таблица 6-3 – Спецификация mbAddrN

xN	**yN**	**mbAddrN**
< 0	< 0	mbAddrD
< 0	0 .. maxWH – 1	mbAddrA
0 .. maxWH – 1	< 0	mbAddrB
0 .. maxWH – 1	0 .. maxWH – 1	CurrMbAddr
> maxWH – 1	< 0	mbAddrC
> maxWH – 1	0 .. maxWH – 1	недоступно
	> maxWH – 1	недоступно

Смежное местоположение яркости (xW, yW) относительно левого верхнего угла макроблока mbAddrN находят как

$$xW = (xN + maxWH) \% maxWH \tag{6-25}$$

$$yW = (yN + maxWH) \% maxWH. \tag{6-26}$$

6.4.8.2 Спецификация для смежных местоположений яркости в полях и кадрах MBAFF

Спецификацию этого пункта применяют, когда MbaffFrameFlag равен 1.

В п. 6.4.6 процесс создания адресов смежных макроблоков и их доступность реализуют с помощью mbAddrA, mbAddrB, mbAddrC и mbAddrD так же, как и статус доступности на выходе.

Таблица 6-4 определяет адреса макроблоков mbAddrN и yM двумя этапами:

1. Спецификация адрес макроблока mbAddrX в зависимости от (xN, yN) и следующих переменных:
   - Переменную currMbFrameFlag находят следующим образом:
     - Если макроблок с адресом CurrMbAddr – макроблок кадра, то currMbFrameFlag устанавливают на 1,
     - Иначе (макроблок с адресом CurrMbAddr – макроблок поля), currMbFrameFlag устанавливают на 0.
   - Переменную mbIsTopMbFlag находят следующим образом:
     - Если макроблок с адресом CurrMbAddr – верхний макроблок (CurrMbAddr % 2 равно 0), то mbIsTopMbFlag устанавливают на 1;
     - Иначе (макроблок с адресом CurrMbAddr – нижний макроблок, CurrMbAddr % 2 равно 1), mbIsTopMbFlag устанавливают на 0.
2. В зависимости от доступности mbAddrX применяют следующее:
   - Если mbAddrX не доступен, то mbAddrN помечают как недоступный.
   - Иначе (mbAddrX доступен), mbAddrN помечают как доступный, а в таблице 6-4 указывают mbAddrN и yM, зависящие от (xN, yN), currMbFrameFlag, mbIsTopMbFlag и переменную mbAddrXFrameFlag, которую находят следующим образом:
     - Если макроблок mbAddrX – макроблок кадра, то mbAddrXFrameFlag устанавливают на 1,
     - Иначе (макроблок mbAddrX – макроблок поля), mbAddrXFrameFlag устанавливают на 0.

Значения вышеприведенных флагов в таблице 6-4 как na (неопределенные) обозначают, что значения соответствующих флагов не существенны для данного ряда таблицы.

Таблица 6-4 – Спецификация mbAddrN и yM

xN	yN	currMbFrameFlag	mbIsTopMbFlag	mbAddrX	mbAddrXFrameFlag	дополнительное условие	mbAddrN	yM
< 0	< 0	1	1	mbAddrD			mbAddrD + 1	yN
			0	mbAddrA	1		mbAddrA	yN
					0		mbAddrA + 1	(yN + maxWH) >> 1
		0	1	mbAddrD	1		mbAddrD + 1	2*yN
					0		mbAddrD	yN
			0	mbAddrD			mbAddrD + 1	yN
< 0	0 .. maxWH – 1	1	1	mbAddrA	1		mbAddrA	yN
					0	yN % 2 = = 0	mbAddrA	yN >> 1
						yN % 2 != 0	mbAddrA + 1	yN >> 1
			0	mbAddrA	1		mbAddrA + 1	yN
					0	yN % 2 = = 0	mbAddrA	(yN + maxWH) >> 1
						yN % 2 != 0	mbAddrA + 1	(yN + maxWH) >> 1
		0	1	mbAddrA	1	yN < (maxWH / 2)	mbAddrA	yN <<1
						yN >= (maxWH / 2)	mbAddrA + 1	(yN <<1) – maxWH
					0		mbAddrA	yN
			0	mbAddrA	1	yN < (maxWH / 2)	mbAddrA	(yN <<1) + 1
						yN >= (maxWH / 2)	mbAddrA + 1	(yN <<1) + 1 – maxWH
					0		mbAddrA + 1	yN
0 .. maxWH – 1	< 0	1	1	mbAddrB			mbAddrB + 1	yN
			0	CurrMbAddr			CurrMbAddr – 1	yN
		0	1	mbAddrB	1		mbAddrB + 1	2 * yN
					0		mbAddrB	yN
			0	mbAddrB			mbAddrB + 1	yN
0 .. maxWH – 1	0 .. maxWH – 1			CurrMbAddr			CurrMbAddr	yN
> maxWH – 1	<0	1	1	mbAddrC			mbAddrC + 1	yN
			0	недоступно			недоступно	na
		0	1	mbAddrC	1		mbAddrC + 1	2 * yN
					0		mbAddrC	yN
			0	mbAddrC			mbAddrC + 1	yN
> maxWH – 1	0 .. maxWH – 1			недоступно			недоступно	na
	> maxWH – 1			недоступно			недоступно	na

Смежные местоположения яркости (xW, yW) относительно левого верхнего угла макроблока mbAddrN находят как

$$xW = (xN + maxWH) \% maxWH \quad (6\text{-}27)$$

$$yW = (yM + maxWH) \% maxWH. \quad (6\text{-}28)$$

7 Синтаксис и семантика

7.1 Метод описания синтаксиса в табличной форме

Таблицы синтаксиса описывают надмножество синтаксиса всех разрешенных потоков битов. Дополнительные ограничения на синтаксис могут быть описаны в других разделах.

ПРИМЕЧАНИЕ. – Реальный декодер должен использовать средства для идентификации входных точек в потоке битов и идентифицировать и обрабатывать несоответствующие потоки битов. Методы идентификации и обработки ошибок, а также другие аналогичные ситуации здесь не описываются.

В следующей таблице перечислены примеры псевдокодов, использованных для описания синтаксиса. Если появляется элемент синтаксиса, он указывает, что элемент данных прочтен (извлечен) из потока битов и указателя потока битов.

	C	Дескриптор
/* Может быть сделано утверждение элемента синтаксиса с соответствующей категорией синтаксиса и дескрипторов, или может быть сделано представление, использованное для описания условий существования, типа и количества элементов синтаксиса (**syntax_element)** как в двух следующих примерах */		
syntax_element	3	ue(v)
Условное утверждение		
/* Группа утверждений, заключенных в фигурные скобки, представляет сложное утверждение и рассматривается функционально как простое утверждение. */		
{		
утверждение		
утверждение		
…		
}		
/* Структура "пока" описывает проверку, действительно ли это условие – истина, и если истина, описывает оценку утверждения (или сложного утверждения) повторно до тех пор, пока это условие не перестанет быть истиной */		
Пока (условие)		
утверждение		
/*"Действует … пока" структура, описывающая оценку утверждения однократно, за которой следует проверка истинности, и если оно истинно, повторно описывает оценку до тех пор, пока это условие не перестанет быть истиной */		
действует		
утверждение		
while(условие)		
/* Структура "если … еще" описывает проверку, действительно ли это условие – истина, и если истина, описывает оценку первичного утверждения, иначе описывает оценку альтернативного утверждения. "Еще" – часть структуры и последующее альтернативное утверждение опускают, если не требуется никакой оценки альтернативного утверждения */		
if(условие)		
предварительное утверждение		
еще		
альтернативное утверждение		
/* Структура "для" определяет оценку первоначального утверждения, за которым следует проверка условия, и если оно истинно, повторно описывает оценку, за которой следует последующее утверждение, до тех пор, пока это условие не перестанет быть истиной. */		
for(первоначальное утверждение; условие; последующее утверждение)		
первоначальное утверждение		

7.2 Спецификация синтаксиса функций, категорий и дескрипторов

Представленные здесь функции используют для синтаксического описания. Эти функции предполагают присутствие указателя потока битов с указанием положения следующего бита, который должен быть считан процессом декодирования из потока битов.

byte_aligned( ) описан следующим образом:

- Если текущая позиция в потоке битов попадает на пограничный байт, т. е. следующий бит в потоке битов – это первый бит в байте, то возвратное значение byte_aligned( ) будет ИСТИНА.
- Иначе возвратное значение byte_aligned( ) будет ЛОЖЬ.

Параметр more_data_in_byte_stream( ), который используют только в структуре синтаксиса потока байтов блоков NAL, описанной в Приложении B, описывают следующим образом:

- Если в потоке байтов имеются дополнительные данные, возвратное значение more_data_in_byte_stream( ) будет ИСТИНА.
- Иначе возвратное значение more_data_in_byte_stream( ) будет ЛОЖЬ.

more_rbsp_data( ) описан следующим образом:

- Если в RBSP имеются дополнительные данные перед rbsp_trailing_битов( ), возвратное значение more_rbsp_data( ) будет ИСТИНА.
- Иначе возвратное значение more_rbsp_data( ) будет ЛОЖЬ.

Метод, который дает возможность определить, действительно ли имеются дополнительные данные в RBSP, описан приложением (или в Приложении B для приложений, которые используют формат потока байтов).

more_rbsp_trailing_data( ) описан следующим образом:

- Если в RBSP имеются дополнительные данные, возвратное значение more_rbsp_trailing_data( ) будет ИСТИНА.
- Иначе возвратное значение more_rbsp_trailing_data( ) будет ЛОЖЬ.

next_bits(n) создает следующие биты в потоке битов для целей сравнения без помощи указателя потока битов. Формирует вид этих следующих n битов в потоке битов с n аргументами. При использовании в потоке байтов, как это определено в Приложении B, возвратное значение next_bits(n) будет 0, если меньше чем n битов остается в потоке байтов.

read_bits(n) считывает следующие n битов из потока битов и продвигает указатель потока битов в положение на n битов. Если n равно 0, read_bits(n) описан возвратным значением, равным 0, и не и продвигает указатель потока битов.

Категории (отмеченные в таблицах как **C**) описывают разделение на части данных секции при разделении не более, чем на три части данных секций. Разделение данных секции A содержит все элементы синтаксисов категории 2. Разделение данных секции B содержит все элементы синтаксисов категории 3. Разделение данных секции C содержит все элементы синтаксисов категории 4. Значение других категорий не описано. Для некоторых элементов синтаксисов используют две категории значений, разделенные вертикальной чертой. В этих случаях значения категорий, разделенных вертикальной чертой, будет x. В таких случаях значение категории, которая должна быть использована в дальнейшем, описана в тексте. Для структур синтаксиса, которые использованы в других структурах синтаксиса, категории для всех элементов синтаксиса, расположенных во включенной структуре синтаксиса, перечисляют, разделяя вертикальной чертой. Элемент синтаксиса или структуры синтаксиса с категорией, отмеченной как "Все", представлен во всех структурах синтаксисов, которые включают этот элемент синтаксиса или структуру синтаксиса. Для структур синтаксиса, использованных внутри других структур синтаксиса, численное значение категории, приведенной в таблице синтаксиса в месте включения элемента в структуре синтаксиса, отмеченной как "Все", применительно к этому элементу считают как "Все".

Следующие дескрипторы определяют процесс анализа каждого элемента синтаксиса. Для некоторых элементов синтаксиса используют два дескриптора, разделенные вертикальной чертой. В таких случаях левые дескрипторы применяют тогда, когда флаг entropy_coding_mode_flag равен 0, а правый – когда entropy_coding_mode_flag равен 1.

- ae(v): элемент синтаксиса энтропийно-кодированный и контекстно-адаптивный. Процесс анализа для этого дескриптора описан в п. 9.3.
- b(8): байт, содержащий строку битов любого шаблона с возвратным значением функции (8 битов). Процесс анализа для этого дескриптора описан возвратным значением функции read_bits(8).
- ce(v): элемент синтаксиса энтропийно-кодированный переменной длины и контекстно-адаптивный, у которого первый бит левый. Процесс анализа для этого дескриптора описан в п. 9.2.
- f(n): строка битов с закрепленным шаблонам, в которой используют n битов, записанных (слева направо) с первым левым битом. Процесс анализа для этого дескриптора описан возвратным значением функции read_bits(n).
- i(n): целое со знаком с использованием n битов. Если n – это "v" в таблице синтаксиса, то число битов изменяется способом, который зависит от значения других элементов синтаксиса. Процесс анализа для этого дескриптора описан возвратным значением функции read_bits(n), интерпретируемой как два дополнительных целых представления с наиболее значимым битом, записанным первым.
- me(v): элемент синтаксиса, отображенный кодированием Exp-Golomb с левым первым битом. Процесс анализа для этого дескриптора описан в п. 9.1.
- se(v): элемент синтаксиса, кодированный Exp-Golomb целым со знаком с левым первым битом. Процесс анализа для этого дескриптора описан в п. 9.1.
- te(v): элемент синтаксиса, кодированный усеченным методом Exp-Golomb с левым первым битом. Процесс анализа для этого дескриптора описан в п. 9.1.

– u(n): целое без знака с использованием n битов. Если n – это "v" в таблице синтаксиса, то число битов изменяется способом, который зависит от значения других элементов синтаксиса. Процесс анализа для этого дескриптора описан возвратным значением функции read_bits(n), интерпретированным как бинарное представление целого без знака с наиболее значимым битом, записанным первым.

– ue(v): элемент синтаксиса, кодированный Exp-Golomb целым без знака с левым первым битом. Процесс анализа для этого дескриптора описан в п. 9.1.

7.3 Синтаксис в табличной форме

7.3.1 Синтаксис блока NAL

nal_unit(NumBytesInNALunit) {	C	Дескриптор
forbidden_zero_bit	Все	f(1)
nal_ref_idc	Все	u(2)
nal_unit_type	Все	u(5)
NumBytesInRBSP = 0		
for(i = 1; i < NumBytesInNALunit; i++) {		
if(i + 2 < NumBytesInNALunit && next_bits(24) = = 0x000003) {		
rbsp_byte[NumBytesInRBSP++ **]**	Все	b(8)
rbsp_byte[NumBytesInRBSP++ **]**	Все	b(8)
i += 2		
emulation_prevention_three_byte /* равное 0x03 */	Все	f(8)
} else		
rbsp_byte[NumBytesInRBSP++ **]**	Все	b(8)
}		
}		

7.3.2 Синтаксис последовательностей полезной нагрузки исходных байтов и концевых битов RBSP

7.3.2.1 Синтаксис последовательности установки параметра RBSP

seq_ parameter _set_rbsp( ) {	C	Дескриптор
profile_idc	0	u(8)
constraint_set0_flag	0	u(1)
constraint_set1_flag	0	u(1)
constraint_set2_flag	0	u(1)
reserved_zero_5bits /* равен 0 */	0	u(5)
level_idc	0	u(8)
seq_parameter_set_id	0	ue(v)
log2_max_frame_num_minus4	0	ue(v)
pic_order_cnt_type	0	ue(v)
if(pic_order_cnt_type = = 0)		
log2_max_pic_order_cnt_lsb_minus4	0	ue(v)
else if(pic_order_cnt_type = = 1) {		
delta_pic_order_always_zero_flag	0	u(1)
offset_for_non_ref_pic	0	se(v)
offset_for_top_to_bottom_field	0	se(v)
num_ref_frames_in_pic_order_cnt_cycle	0	ue(v)
for(i = 0; i < num_ref_frames_in_pic_order_cnt_cycle; i++)		
offset_for_ref_frame[i **]**	0	se(v)
}		
num_ref_frames	0	ue(v)
gaps_in_frame_num_value_allowed_flag	0	u(1)
pic_width_in_mbs_minus1	0	ue(v)
pic_hight_in_map_units_minus1	0	ue(v)
frame_mbs_only_flag	0	u(1)
if(!frame_mbs_only_flag)		
mb_adaptive_frame_field_flag	0	u(1)
direct_8x8_inference_flag	0	u(1)
frame_cropping_flag	0	u(1)
if(frame_cropping_flag) {		
frame_crop_left_offset	0	ue(v)
frame_crop_right_offset	0	ue(v)
frame_crop_top_offset	0	ue(v)
frame_crop_bottom_offset	0	ue(v)
}		
vui_parameters_present_flag	0	u(1)
if(vui_parameters_present_flag)		
vui_parameters( )	0	
rbsp_trailing_bits( )	0	
}		

7.3.2.2 Синтаксис установки параметра RBSP

pic_parameter_set_rbsp( ) {	C	Дескриптор
pic_parameter_set_id	1	ue(v)
seq_parameter_set_id	1	ue(v)
entropy_coding_mode_flag	1	u(1)
pic_order_present_flag	1	u(1)
num_slice_groups_minus1	1	ue(v)
if(num_секция_groups_minus1 > 0) {		
slice _group_map_type	1	ue(v)
if(slice _group_map_type = = 0)		
for(iGroup = 0; iGroup <= num_ slice _groups_minus1; iGroup++)		
run_length_minus1[iGroup **]**	1	ue(v)
else if(slice _group_map_type = = 2)		
for(iGroup = 0; iGroup < num_ slice _groups_minus1; iGroup++) {		
top_left[iGroup **]**	1	ue(v)
bottom_right[iGroup **]**	1	ue(v)
}		
else if(slice_group_map_type = = 3 | | slice_group_map_type = = 4 | | slice_group_map_type = = 5) {		
slice_group_change_direction_flag	1	u(1)
slice_group_change_rate_minus1	1	ue(v)
} else if(slice_group_map_type = = 6) {		
pic_size_in_map_units_minus1	1	ue(v)
for(i = 0; i <= pic_size_in_map_units_minus1; i++)		
slice_group_id[i **]**	1	u(v)
}		
}		
num_ref_idx_l0_active_minus1	1	ue(v)
num_ref_idx_l1_active_minus1	1	ue(v)
weighted_pred_flag	1	u(1)
weighted_bipred_idc	1	u(2)
pic_init_qp_minus26 /* относительно 26 */	1	se(v)
pic_init_qs_minus26 /* относительно 26 */	1	se(v)
chroma_qp_index_offset	1	se(v)
deblocking_filter_control_present_flag	1	u(1)
constrained_intra_pred_flag	1	u(1)
redundant_pic_cnt_present_flag	1	u(1)
rbsp_trailing_bits( )	1	
}		

7.3.2.3 Синтаксис дополнительной расширенной информации RBSP

sei_rbsp( ) {	C	Дескриптор
do		
sei_message( )	5	
while(more_rbsp_data( ))		
rbsp_trailing_битов( )	5	
}		

7.3.2.3.1 Синтаксис сообщения дополнительной расширенной информации

sei_message( ) {	C	Дескриптор
payloadType = 0		
while(next_bits(8) = = 0xFF) {		
ff_byte /* равное 0xFF */	5	f(8)
payloadType += 255		
}		
last_payload_type_byte	5	u(8)
payloadType += last_payload_type_byte		
payloadSize = 0		
while(next_bits(8) = = 0xFF) {		
ff_byte /* равное 0xFF */	5	f(8)
payloadSize += 255		
}		
last_payload_size_byte	5	u(8)
payloadSize += last_payload_size_byte		
sei_payload(payloadType, payloadSize)	5	
}		

7.3.2.4 Синтаксис доступа к блоку разграничителя RBSP

access_unit_delimiter_rbsp( ) {	C	Дескриптор
primary_pic_type	6	u(3)
rbsp_trailing_bits( )	6	
}		

7.3.2.5 Синтаксис конца последовательности RBSP

end_of_seq_rbsp( ) {	C	Дескриптор
}		

7.3.2.6 Синтаксис конца потока RBSP

end_of_stream_rbsp( ) {	C	Дескриптор
}		

7.3.2.7 Синтаксис заполнителя данных RBSP

filler_data_rbsp(NumBytesInRBSP) {	C	Дескриптор
while(next_bits(8) = = 0xFF)		
ff_byte /* равное 0xFF */	9	f(8)
rbsp_trailing_ bits ( )	9	
}		

7.3.2.8 Синтаксис слоя секции без разделения RBSP

slice_layer_without_partitioning_rbsp( ) {	C	Дескриптор
slice_header( )	2	
slice_data( ) /* все категории синтаксиса slice_data( ) */	2 | 3 | 4	
rbsp_slice_trailing_ bits ( )	2	
}		

7.3.2.9 Синтаксис деления данных секции RBSP

7.3.2.9.1 Синтаксис деления данных секции A RBSP

slice_data_partition_a_layer_rbsp( ) {	C	Дескриптор
slice_header( )	2	
slice_id	2	ue(v)
slice_data( ) /* только части категории 2 синтаксиса slice_data( ) */	2	
rbsp_slice_trailing_bits( )	2	
}		

7.3.2.9.2 Синтаксис деления данных секции B RBSP

slice_data_partition_b_layer_rbsp( ) {	C	Дескриптор
slice_id	3	ue(v)
if(redundant_pic_cnt_present_flag)		
redundant_pic_cnt	3	ue(v)
slice_data( ) /* только части категории 3 синтаксиса slice_data( ) */	3	
rbsp_slice_trailing_bits( )	3	
}		

7.3.2.9.3 Синтаксис деления данных секции C RBSP

slice_data_partition_c_layer_rbsp( ) {	C	Дескриптор
slice_id	4	ue(v)
if(redundant_pic_cnt_present_flag)		
redundant_pic_cnt	4	ue(v)
slice_data( ) /* только части категории 4 синтаксиса slice_data( ) */	4	
rbsp_slice_trailing_ bits( )	4	
}		

7.3.2.10 Синтаксис концевых битов секции RBSP

rbsp_slice_trailing_bits( ) {	C	Дескриптор
rbsp_trailing_bits( )	Все	
if(entropy_coding_mode_flag)		
while(more_rbsp_trailing_data( ))		
cabac_zero_word /* равно 0x0000 */	Все	f(16)
}		

7.3.2.11 Синтаксис концевых битов RBSP

rbsp_trailing_bits( ) {	C	Дескриптор
rbsp_stop_one_bit /* равно 1 */	Все	f(1)
while(!byte_aligned( ))		
rbsp_alignment_zero_bit /* равно 0 */	Все	f(1)
}		

7.3.3 Синтаксис заголовка секции

slice_header( ) {	C	Дескриптор
first_mb_in_slice	2	ue(v)
slice_type	2	ue(v)
pic_parameter_set_id	2	ue(v)
frame_num	2	u(v)
if(!frame_mbs_only_flag) {		
field_pic_flag	2	u(1)
if(field_pic_flag)		
bottom_field_flag	2	u(1)
}		
if(nal_unit_type = = 5)		
idr_pic_id	2	ue(v)
if(pic_order_cnt_type = = 0) {		
pic_order_cnt_lsb	2	u(v)
if(pic_order_present_flag && !field_pic_flag)		
delta_pic_order_cnt_bottom	2	se(v)
}		
if(pic_order_cnt_type = = 1 && !delta_pic_order_always_zero_flag) {		
delta_pic_order_cnt[0 **]**	2	se(v)
if(pic_order_present_flag && !field_pic_flag)		
delta_pic_order_cnt[1 **]**	2	se(v)
}		
if(redundant_pic_cnt_present_flag)		
redundant_pic_cnt	2	ue(v)
if(slice_type = = B)		
direct_spatial_mv_pred_flag	2	u(1)
if(slice_type = = P | | slice_type = = SP | | slice_type = = B) {		
num_ref_idx_active_override_flag	2	u(1)
if(num_ref_idx_active_override_flag) {		
num_ref_idx_l0_active_minus1	2	ue(v)
if(slice_type = = B)		
num_ref_idx_l1_active_minus1	2	ue(v)
}		
}		
ref_pic_list_reordering( )	2	
if((weighted_pred_flag && (slice_type = = P | | slice_type = = SP)) | | (weighted_bipred_idc = = 1 && slice_type = = B))		
pred_weight_table( )	2	
if(nal_ref_idc != 0)		
dec_ref_pic_marking( )	2	
if(entropy_coding_mode_flag && slice_type != I && slice_type != SI)		
cabac_init_idc	2	ue(v)
slice_qp_delta	2	se(v)
if(slice_type = = SP | | slice_type = = SI) {		
if (slice_type = = SP)		
sp_for_switch_flag	2	u(1)
slice_qs_delta	2	se(v)

}		
if(deblocking_filter_control_present_flag) {		
disable_ deblocking _filter_idc	2	ue(v)
if(disable_deblocking_filter_idc != 1) {		
slice_alpha_c0_offset_div2	2	se(v)
slice_beta_offset_div2	2	se(v)
}		
}		
if(num_slice_groups_minus1 > 0 && slice_group_map_type >= 3 && slice_group_map_type <= 5)		
slice_group_change_cycle	2	u(v)
}		

7.3.3.1 Синтаксис изменения порядка списка контрольного изображения

ref_pic_list_reordering( ) {	**C**	**Дескриптор**
if(slice_type != I && slice_type != SI) {		
ref_pic_list_reordering_flag_l0	2	u(1)
if(ref_pic_list_reordering_flag_l0)		
do {		
reordering_of_pic_nums_idc	2	ue(v)
if(reordering_of_pic_nums_idc = = 0 | | reordering_of_pic_nums_idc = = 1)		
abs_diff_pic_num_minus1	2	ue(v)
else if(reordering_of_pic_nums_idc = = 2)		
long_term_pic_num	2	ue(v)
} while(reordering_of_pic_nums_idc != 3)		
}		
if(slice_type = = B) {		
ref_pic_list_reordering_flag_l1	2	u(1)
if(ref_pic_list_reordering_flag_l1)		
do {		
reordering_of_pic_nums_idc	2	ue(v)
if(reordering_of_pic_nums_idc = = 0 | | reordering_of_pic_nums_idc = = 1)		
abs_diff_pic_num_minus1	2	ue(v)
else if(reordering_of_pic_nums_idc = = 2)		
long_term_pic_num	2	ue(v)
} while(reordering_of_pic_nums_idc != 3)		
}		
}		

7.3.3.2 Синтаксис таблицы веса предсказания

pred_weight_table( ) {	C	Дескриптор
luma_log2_weight_denom	2	ue(v)
chroma_log2_weight_denom	2	ue(v)
for(i = 0; i <= num_ref_idx_l0_active_minus1; i++) {		
luma_weight_l0_flag	2	u(1)
if(luma_weight_l0_flag) {		
luma_weight_l0[i **]**	2	se(v)
luma_offset_l0[i **]**	2	se(v)
}		
chroma_weight_l0_flag	2	u(1)
if(chroma_weight_l0_flag)		
for(j =0; j < 2; j++) {		
chroma_weight_l0[i **][** j **]**	2	se(v)
chroma_offset_l0[i **][** j **]**	2	se(v)
}		
}		
if(slice_type = = B)		
for(i = 0; i <= num_ref_idx_l1_active_minus1; i++) {		
luma_weight_l1_flag	2	u(1)
if(luma_weight_l1_flag) {		
luma_weight_l1[i **]**	2	se(v)
luma_offset_l1[i **]**	2	se(v)
}		
chroma_weight_l1_flag	2	u(1)
if(chroma_weight_l1_flag)		
for(j = 0; j < 2; j++) {		
chroma_weight_l1[i **][** j **]**	2	se(v)
chroma_offset_l1[i **][** j **]**	2	se(v)
}		
}		
}		

7.3.3.3 Синтаксис разметки декодированного контрольного изображения

dec_ref_pic_marking( ) {	C	Дескриптор
if(nal_unit_type = = 5) {		
no_output_of_prior_pics_flag	2 | 5	u(1)
long_term_reference_flag	2 | 5	u(1)
} else {		
adaptive_ref_pic_marking_mode_flag	2 | 5	u(1)
if(adaptive_ref_pic_marking_mode_flag)		
do {		
memory_management_control_operation	2 | 5	ue(v)
if(memory_management_control_operation = = 1 | | memory_management_control_operation = = 3)		
difference_of_pic_nums_minus1	2 | 5	ue(v)
if(memory_management_control_operation = = 2)		
long_term_pic_num	2 | 5	ue(v)
if(memory_management_control_operation = = 3 | | memory_management_control_operation = = 6)		
long_term_frame_idx	2 | 5	ue(v)
if(memory_management_control_operation = = 4)		
max_long_term_frame_idx_plus1	2 | 5	ue(v)
} while(memory_management_control_operation != 0)		
}		
}		

7.3.4 Синтаксис данных секции

slice_data( ) {	C	Дескриптор
if(entropy_coding_mode_flag)		
while(!byte_aligned( ))		
cabac_alignment_one_bit	2	f(1)
CurrMbAddr = first_mb_in_slice * (1 + MbaffFrameFlag)		
moreDataFlag = 1		
prevMbSkipped = 0		
do {		
if(slice_type != I && slice_type != SI)		
if(!entropy_coding_mode_flag) {		
mb_skip_run	2	ue(v)
prevMbSkipped = (mb_skip_run > 0)		
for(i=0; i<mb_skip_run; i++)		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
moreDataFlag = more_rbsp_data( )		
} else {		
mb_skip_flag	2	ae(v)
moreDataFlag = !mb_skip_flag		
}		
if(moreDataFlag) {		
if(MbaffFrameFlag && (CurrMbAddr % 2 = = 0 | | (CurrMbAddr % 2 = = 1 && prevMbSkipped)))		
mb_field_decoding_flag	2	u(1) | ae(v)
macroblock_layer( )	2 | 3 | 4	
}		
if(!entropy_coding_mode_flag)		
moreDataFlag = more_rbsp_data( )		
else {		
if(slice_type != I && slice_type != SI)		
prevMbSkipped = mb_skip_flag		
if(MbaffFrameFlag && CurrMbAddr % 2 = = 0)		
moreDataFlag = 1		
else {		
end_of_slice_flag	2	ae(v)
moreDataFlag = !end_of_slice_flag		
}		
}		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
} while(moreDataFlag)		
}		

7.3.5 Синтаксис слоя макроблока

макроблок_layer( ) {	C	Дескриптор
mb_type	2	ue(v) | ae(v)
if(mb_type = = I_PCM) {		
while(!byte_aligned())		
pcm_alignment_zero_bit	2	f(1)
for(i = 0; i < 256 * ChromaFormatFactor; i++)		
pcm_byte[i **]**	2	u(8)
} else {		
if(MbPartPredMode(mb_type, 0) != Intra_4x4 && MbPartPredMode(mb_type, 0) != Intra_16x16 && NumMbPart(mb_type) = = 4)		
sub_mb_pred(mb_type)	2	
else		
mb_pred(mb_type)	2	
if(MbPartPredMode(mb_type, 0) != Intra_16x16)		
coded_блок_pattern	2	me(v) | ae(v)
if(CodedBlockPatternLuma > 0 | | CodedBlockPatternChroma > 0 | | MbPartPredMode(mb_type, 0) = = Intra_16x16) {		
mb_qp_delta	2	se(v) | ae(v)
residual( )	3 | 4	
}		
}		
}		

7.3.5.1 Синтаксис предсказания макроблока

mb_pred(mb_type) {	C	Дескриптор
if(MbPartPredMode(mb_type, 0) = = Intra_4x4 | | MbPartPredMode(mb_type, 0) = = Intra_16x16) {		
if(MbPartPredMode(mb_type, 0) = = Intra_4x4)		
for(luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++) {		
prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]**	2	u(1) | ae(v)
if(!prev_intra4x4_pred_mode_flag**[** luma4x4BlkIdx **]**)		
rem_intra4x4_pred_mode[luma4x4BlkIdx **]**	2	u(3) | ae(v)
}		
intra_chroma_pred_mode	2	ue(v) | ae(v)
} else if(MbPartPredMode(mb_type, 0) != Direct) {		
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if((num_ref_idx_l0_active_minus1 > 0 | | mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if((num_ref_idx_l1_active_minus1 > 0 | | mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(MbPartPredMode (mb_type, mbPartIdx) != Pred_L1)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
}		
}		

7.3.5.2 Синтаксис предсказания субмакроблока

sub_mb_pred(mb_type) {	C	Дескриптор
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
sub_mb_type[mbPartIdx **]**	2	ue(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if((num_ref_idx_l0_active_minus1 > 0 | | mb_field_decoding_flag) && mb_type != P_8x8ref0 && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if((num_ref_idx_l1_active_minus1 > 0 | | mb_field_decoding_flag) && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
}		

7.3.5.3 Синтаксис данных остатка

residual( ) {	C	Дескриптор
if(!entropy_coding_mode_flag)		
residual_block = residual_block_cavlc		
else		
residual_block = residual_block_cabac		
if(MbPartPredMode(mb_type, 0) = = Intra_16x16)		
residual_block(Intra16x16DCLevel, 16)	3	
for(i8x8 = 0; i8x8 < 4; i8x8++) /* каждый блок яркости 8x8 */		
for(i4x4 = 0; i4x4 < 4; i4x4++) /* каждый субблок 4x4 блока */		
if(CodedBlockPatternLuma & (1 << i8x8)) {		
if(MbPartPredMode(mb_type, 0) = = Intra_16x16)		
residual_block(Intra16x16ACLevel[i8x8 * 4 + i4x4], 15)	3	
else		
residual_block(LumaLevel[i8x8 * 4 + i4x4], 16)	3 | 4	
} else {		
if(MbPartPredMode(mb_type, 0) = = Intra_16x16)		
for(i = 0; i < 15; i++)		
Intra16x16ACLevel[i8x8 * 4 + i4x4][i] = 0		
else		
for(i = 0; i < 16; i++)		
LumaLevel[i8x8 * 4 + i4x4][i] = 0		
}		
for(iCbCr = 0; iCbCr < 2; iCbCr++)		
if(CodedBlockPatternChroma & 3) /* представлен остаток цветности DC */		
residual_block(ChromaDCLevel[iCbCr], 4)	3 | 4	
else		
for(i = 0; i < 4; i++)		
ChromaDCLevel[iCbCr][i] = 0		
for(iCbCr = 0; iCbCr < 2; iCbCr++)		
for(i4x4 = 0; i4x4 < 4; i4x4++)		
if(CodedBlockPatternChroma & 2) /* представлен остаток цветности AC */		
residual_block(ChromaACLevel[iCbCr][i4x4], 15)	3 | 4	
else		
for(i = 0; i < 15; i++)		
ChromaACLevel[iCbCr][i4x4][i] = 0		
}		

7.3.5.3.1 Синтаксис блока остатка CAVLC

residual_block_cavlc(coeffLevel, maxNumCoeff) {	C	Дескриптор
for(i = 0; i < maxNumCoeff; i++)		
coeffLevel[i] = 0		
coeff_token	3 | 4	ce(v)
if(TotalCoeff(coeff_token) > 0) {		
if(TotalCoeff(coeff_token) > 10 && TrailingOnes(coeff_token) < 3)		
suffixLength = 1		
else		
suffixLength = 0		
for(i = 0; i < TotalCoeff(coeff_token); i++)		
if(i < TrailingOnes(coeff_token)) {		
trailing_ones_sign_flag	3 | 4	u(1)
level[i] = 1 – 2 * trailing_ones_sign_flag		
} else {		
level_prefix	3 | 4	ce(v)
levelCode = (level_prefix << suffixLength)		
if(suffixLength > 0 | | level_prefix >= 14) {		
level_suffix	3 | 4	u(v)
levelCode += level_suffix		
}		
if(level_prefix = = 15 && suffixLength = = 0)		
levelCode += 15		
if(i = = TrailingOnes(coeff_token) && TrailingOnes(coeff_token) < 3)		
levelCode += 2		
if(levelCode % 2 = = 0)		
level[i] = (levelCode + 2) >> 1		
else		
level[i] = (–levelCode – 1) >> 1		
if(suffixLength = = 0)		
suffixLength = 1		
if(Abs(level[i]) > (3 << (suffixLength – 1)) && suffixLength < 6)		
suffixLength++		
}		
if(TotalCoeff(coeff_token) < maxNumCoeff) {		
total_zeros	3 | 4	ce(v)
zerosLeft = total_zeros		
} else		
zerosLeft = 0		
for(i = 0; i < TotalCoeff(coeff_token) – 1; i++) {		
if(zerosLeft > 0) {		
run_before	3 | 4	ce(v)
run[i] = run_before		
} else		
run[i] = 0		
zerosLeft = zerosLeft – run[i]		
}		

run[TotalCoeff(coeff_token) – 1] = zerosLeft		
coeffNum = -1		
for(i = TotalCoeff(coeff_token) – 1; i >= 0; i--) {		
coeffNum += run[i] + 1		
coeffLevel[coeffNum] = level[i]		
}		
}		
}		

7.3.5.3.2 Синтаксис блока остатка CABAC

residual_block_cabac(coeffLevel, maxNumCoeff) {	C	Дескриптор
coded_block_flag	3 | 4	ae(v)
if(coded_block_flag) {		
numCoeff = maxNumCoeff		
i = 0		
do {		
significant_coeff_flag[i **]**	3 | 4	ae(v)
if(significant_coeff_flag[i]) {		
last_significant_coeff_flag[i **]**	3 | 4	ae(v)
if(last_significant_coeff_flag[i]) {		
numCoeff = i + 1		
for(j = numCoeff; j < maxNumCoeff; j++)		
coeffLevel[j] = 0		
}		
}		
i++		
} while(i < numCoeff-1)		
coeff_abs_level_minus1[numCoeff-1 **]**	3 | 4	ae(v)
coeff_sign_flag[numCoeff-1 **]**	3 | 4	ae(v)
coeffLevel[numCoeff-1] = (coeff_abs_level_minus1[numCoeff – 1] + 1) * (1 – 2 * coeff_sign_flag[numCoeff – 1])		
for(i = numCoeff-2; i >= 0; i--) {		
if(significant_coeff_flag[i]) {		
coeff_abs_level_minus1[i **]**	3 | 4	ae(v)
coeff_sign_flag[i **]**	3 | 4	ae(v)
coeffLevel[i] = (coeff_abs_level_minus1[i] + 1) * (1 – 2 * coeff_sign_flag[i])		
} else		
coeffLevel[i] = 0		
}		
} else		
for(i = 0; i < maxNumCoeff; i++)		
coeffLevel[i] = 0		
}		

7.4 Семантика

7.4.1 Семантика блока NAL

ПРИМЕЧАНИЕ. – Уровень VCL описан, чтобы эффективно представить содержание видеоданных. Уровень NAL описан, чтобы сформатировать эти данные и создать информационный заголовок способом, соответствующим передаче по разным каналам связи и хранению медиа информации. Все данные содержатся в блоках NAL, каждый из которых содержит целое число байтов. Блок NAL определяет групповой формат для использования как в пакетно-ориентированных системах, так и в системах с потоками битов. Формат блоков NAL для обеих видов систем идентичен, за исключением того, что каждому блоку NAL может предшествовать префикс кода запуска и дополнительное заполнение байтами в формате потока байтов.

NumBytesInNALunit определяет размер блока NAL в байтах. Эта величина требуется для декодирования блока NAL. Некоторые формы разграничения блоков NAL необходимы для возможности введения NumBytesInNALunit. Один из методов разграничения описан в Приложении B для формата потока байтов. Другие методы могут быть описаны вне этой Рекомендации | Международного стандарта.

forbidden_zero_bit должен быть равным 0.

nal_ref_idc, не равное 0, определяет, что блок NAL содержит набор параметров последовательности или набор параметров изображения, или секцию контрольного изображения, или секцию данных разделения контрольного изображения.

nal_ref_idc равно 0 для блока NAL, содержащего секцию или разделенную часть данных секции, и указывает, что секция или часть секции разделенных данных не являются частью неконтрольного изображения.

nal_ref_idc не должно быть равным 0 для набора параметров последовательности или набора параметров изображения блоков NAL. Если nal_ref_idc равно 0 для одной секции или секции разделенных данных блока NAL конкретного изображения, то оно должно быть равным 0 для всей секции и секции разделенных данных блока NAL изображения.

nal_ref_idc не должно быть равным 0 для блоков NAL IDR, т. е. блоков NAL с nal_unit_type равным 5.

nal_ref_idc должен быть равным 0 для всех блоков NAL, имеющих nal_unit_type равным 6, 9, 10, 11 или 12.

nal_unit_type определяет тип структуры данных RBSP, содержащуюся в блоке NAL, как описано в таблице 7-1. Блоки NAL VCL описаны как блоки NAL, имеющие nal_unit_type равным от 1 до 5 включительно. Все оставшиеся блоки NAL называют блоками NAL не уровня VCL.

В столбце, помеченном "C" в таблице 7-1, перечисляются категории элементов синтаксиса, которые могут быть представлены в блоке NAL. Кроме того, элементы синтаксиса с категорией синтаксиса "Все" могут быть представлены как определенные синтаксисом и семантикой структуры данных RBSP. Присутствие или отсутствие любого из элементов синтаксиса конкретно перечисленной категории определяют из синтаксиса и семантики связанных структур данных RBSP. nal_unit_type не должно быть равно 3 или 4, пока по крайней мере один элемент синтаксиса не будет представлен в структуре данных RBSP, имеющую значение категории элемента синтаксиса, равную значению nal_unit_type и не попавшую в категорию "Все".

Таблица 7-1 – Коды типов блока NAL

nal_unit_type	Структура синтаксиса содержания блока NAL и RBSP	C
0	Не определено	
1	Кодированное не IDR изображение секции slice_layer_without_partitioning_rbsp( )	2, 3, 4
2	Кодированная часть A разделенных данных секции slice_data_partition_a_layer_rbsp( )	2
3	Кодированная часть B разделенных данных секции slice_data_partition_b_layer_rbsp( )	3
4	Кодированная часть C разделенных данных секции slice_data_partition_c_layer_rbsp( )	4
5	Кодированное изображение IDR секции slice_layer_without_partitioning_rbsp( )	2, 3
6	Дополнительная расширенная информация (SEI) sei_rbsp( )	5
7	Набор параметров последовательности seq_parameter_set_rbsp( )	0
8	Набор параметров изображения pic_parameter_set_rbsp( )	1
9	Разграничитель блока доступа access_unit_delimiter_rbsp( )	6
10	Конец последовательности end_of_seq_rbsp( )	7
11	Конец потока end_of_stream_rbsp( )	8
12	Данные заполнения filler_data_rbsp( )	9
13..23	Зарезервировано	
24..31	Не определено	

Блоки NAL, которые используют nal_unit_type, равное 0 или в диапазоне 24..31 включительно, не должны влиять на процесс декодирования, описанный в этой Рекомендации | Международном стандарте.

ПРИМЕЧАНИЕ. – Типы блоков NAL 0 и 24..31 могут быть использованы, как определено приложением. Никакие процессы декодирования для этих значений nal_unit_type не определены в этой Рекомендации | Международном стандарте.

Декодеры должны игнорировать (удалять из потока битов и аннулировать) содержание всех блоков NAL, которые используют зарезервированные значения nal_unit_type.

ПРИМЕЧАНИЕ. – Это требование допускает будущее определение расширений совместимости с этой Рекомендацией | Международным стандартом.

В данном тексте кодированную секцию блока NAL собирательно считают кодированной секцией блока NAL IDR-изображения.

Если значение nal_unit_type равно 5 для блока NAL, содержащего секцию кодированного изображения, то значение nal_unit_type должно быть 5 для всех других VCL блоков NAL этого же кодированного изображения. Такое изображение считают изображением IDR.

ПРИМЕЧАНИЕ. – Секция разделения данных не может быть использована для IDR изображений.

rbsp_byte[i **]** – это i-й байт RBSP. Нагрузка RBSP описана как упорядоченная последовательность байтов следующим образом.

RBSP содержит SODB при таких условиях:

– Если SODB – пустое (т. е. длиной нуль битов), RBSP также пустое.

– Иначе RBSP содержит SODB следующим образом:

1) Первый байт RBSP содержит (наиболее значащих, крайних левых) восемь битов SODB; следующий байт RBSP должен содержать следующие восемь SODB и т. д., пока не останется меньше восьми битов SODB.

2) rbsp_trailing_bits( ) представлен после SODB следующим образом:

   i) первые (наиболее значащие, крайние левые) биты последнего байта RBSP содержат оставшиеся биты SODB, (если таковые имеются)

   ii) следующий бит состоит из единственного rbsp_stop_one_bit, равного 1, и

   iii) если rbsp_stop_one_bit не последний бит байт-синхронизированного байта, то одно или более значений rbsp_alignment_zero_bit представлено в результате синхронизации байтов.

3) Одно или более значений cabac_zero_word 16-ти битовых элементов синтаксиса, равных 0x0000, могут быть представлены в некоторых нагрузках RBSP после rbsp_trailing_bits( ) в конце RBSP.

Структуры синтаксиса, имеющие эти свойства RBSP, описывают в таблицах синтаксиса, используя суффикс "_rbsp". Эти структуры должны находиться в блоках NAL как содержание rbsp_byte[i] данных байтов. Связь структур синтаксиса RBSP с блоками NAL должна быть такой, как это описано в таблице 7-1.

ПРИМЕЧАНИЕ. – Если границы RBSP известны, декодер может извлекать SODB из RBSP, объединяя биты байтов RBSP и отбрасывая значение rbsp_stop_one_bit, которое будет последним (наименее значащим, крайним правым) битом, равным 1, а также отбрасывая любые следующие за ним (менее значащие, ближе к правому краю) биты, которые равны 0. Данные, необходимые для этого процесса декодирования, содержатся в части SODB нагрузки RBSP.

emulation_prevention_three_byte – это байт, равный 0x03. Если в блоке NAL представлена эмуляция emulation_prevention_three_byte, то она должна быть отброшена процессом декодирования.

Последний байт блока NAL не должен быть равным 0x00.

В блоке NAL, следующие трехбайтовые последовательности не должны находиться в любых байт-синхронизированных положениях:

– 0x000000

– 0x000001

– 0x000002.

В блоке NAL любая четырехбайтовая последовательность, которая начинается с 0x000003 и иная, чем нижеперечисленные последовательности, не должна находиться в любом байт-синхронизированном положении:

– 0x00000300

– 0x00000301

– 0x00000302

– 0x00000303.

7.4.1.1 Инкапсуляция SODB в RBSP (информативное)

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Формирование инкапсуляции SODB в RBSP и пользование emulation_prevention_three_byte для инкапсуляции RBSP в блок NAL описано для следующих целей:

– предотвратить эмуляцию стартовых кодов (кодов запуска) в блоки NAL, в то же время позволяя представить любую строку SODB в блоке NAL,

– обеспечить идентификацию конца SODB в блоке NAL, отыскивая RBSP для значения rbsp_stop_one_bit, которое начинается в конце RBSP, и

– обеспечить, чтобы при тех же условиях блок NAL имел размер больше, чем SODB (используя одно или более значений cabac_zero_word).

Кодер может создать блок NAL из RBSP следующей процедурой:

Отыскивают строку данных RBSP для байт-синхронизированных битов бинарных шаблонов:

'00000000 00000000 000000xx' (где xx представляет любой 2-х битовый шаблон: 00, 01, 10 или 11),

извлекают байт, равный 0x03, для замены этих битов шаблонов на шаблоны

'00000000 00000000 00000011 000000xx',

наконец, если последний байт данных RBSP равен 0x00 (что может случиться, если RBSP оканчивается на cabac_zero_word), то конечный, равный 0x03 байт присоединяют к концу данных.

Затем результирующую последовательность байтов ставят впереди первого байта блока NAL, где имеется указание типа структуры данных RBSP, которую он содержит. Это является результатом создания собственно блока NAL.

Этот процесс допускает любое представление SODB в блоке NAL, в то же время обеспечивая, чтобы

– байт-синхронизированный префикс кода запуска не эмулировался в блоке NAL, и
– последовательность 8 нулевых значений битов, которая следует за префиксом кода запуска (независимо от байт-синхронизации), не эмулировалась в блоке NAL.

7.4.1.2 Порядок блоков NAL и связь с кодированными изображениями, блоками доступа и видеопоследовательностями

В этом пункте описаны ограничения на порядок блоков NAL в потоке битов. Любой порядок блоков NAL в потоке битов, подчиняющийся этим ограничениям, в этом тексте считают порядком декодирования блоков NAL. В блоке NAL синтаксис в пп. 7.3, D.1 и E.1 определяет порядок декодирования элементов синтаксиса. Соответствующие этой Рекомендации | Международному стандарту декодеры должны быть в состоянии принимать блоки NAL и их элементы синтаксиса в порядке декодирования.

7.4.1.2.1 Порядок последовательности, установка параметров изображения строк RBSP и их активация

ПРИМЕЧАНИЕ. – Последовательность и механизм установки параметров изображения нарушает передачу редко изменяемой информации при передаче кодированных данных макроблока. Установки последовательности и параметры изображения для некоторых приложений можно передавать "вне полосы", используя надежный транспортный механизм.

Установка параметров изображения строк RBSP включает параметры, которые можно считать кодированными секциями блоков NAL или кодированными секциями разделенных данных типа A блоков NAL одного или более кодированных изображений. Каждую RBSP установку параметра изображения первоначально считают неактивированной в начале операции процесса декодирования. Принято, что не более одной RBSP установки параметра изображения можно считать активной в любой момент во время операции процесса декодирования, а активация установки любого конкретного RBSP параметра изображения приводит к деактивации исходного действующей RBSP установки параметра изображения (если таковая была сделана).

Если RBSP установка параметра изображения (с конкретным значением pic_parameter_set_id) не действует и считается кодированной секцией блока NAL или кодированной секцией разделенных данных A блока NAL (при использовании этого значения pic_parameter_set_id), то она активирована. Эту RBSP установку параметра изображения называют активной RBSP установкой параметра изображения до тех пор, пока она не будет деактивирована другой RBSP установкой параметра изображения. RBSP установка параметра изображения с конкретным значением pic_parameter_set_id должна быть доступна процессу декодирования до ее активации.

Любая установка параметра изображения блока NAL со значением pic_parameter_set_id для действующей RBSP установки параметра изображения должна иметь то же самое содержание, что и действующая RBSP установка параметра изображения, пока за ней не последует последний блок NAL с VCL кодированным изображением и предшествующий первый блок NAL с VCL другого кодированного изображения.

Последовательность RBSP установки параметра включает параметры, которые можно считать одной или более RBSP установкой параметров изображения или одним или более блоков NAL SEI, содержащих сообщение SEI о периоде буферизации. Каждая последовательность RBSP установки параметра в исходном положении считается не действующей в начале операции процесса декодирования. Принято, что не более одной RBSP установки параметра изображения можно считать активной в любой момент во время операции процесса декодирования, а активация установки любого конкретного RBSP параметра изображения приводит к деактивации исходной действующей RBSP установки параметра изображения (если таковая была сделана).

Если последовательность RBSP установки параметра (с конкретным значением seq_parameter_set_id) уже не действует и считается активированной RBSP установкой параметра изображения (при использовании значения seq_parameter_set_id) или считается блоком NAL SEI, содержащим сообщение SEI о периоде буферизации (при использовании этого же значения seq_parameter_set_id), то эта последовательность активирована. Такую последовательность RBSP установки параметра называют активной последовательностью RBSP установки параметра до тех пор, пока она не будет деактивирована другой последовательностью RBSP установки параметра. Последовательность RBSP установки параметра с этим конкретным значением seq_parameter_set_id должна быть доступна процессу декодирования до ее активации. Активированная последовательность RBSP установки параметра должна оставаться активной для всей кодированной видеопоследовательности.

ПРИМЕЧАНИЕ. – Вследствие того, что блок доступа IDR начинает новую кодированную видеопоследовательность, а активированная последовательность RBSP установки параметра должна оставаться активной для всей кодированной видеопоследовательности, последовательность RBSP установки параметра может быть активирована только сообщением SEI о периоде буферизации, если сообщение SEI о периоде буферизации является частью блока доступа IDR.

Любая последовательность RBSP установки параметра блока NAL со значением seq_parameter_set_id для действующей последовательности RBSP установки параметра должна иметь то же самое содержание, что и действующая RBSP установка параметра до тех пор, пока за ней не последует последний блок кодированной видеопоследовательности и предшествующий первый блок NAL с VCL, а также первый блок NAL SEI с сообщением SEI о периоде буферизации (если такое имеется) другой кодированной видеопоследовательности.

ПРИМЕЧАНИЕ. – Если RBSP установка параметра изображения или последовательность RBSP установки параметра передают в потоке битов, эти ограничения накладывают ограничение на порядок блоков NAL, которые содержат соответственно RBSP установки параметра изображения или последовательность RBSP установки параметра. Иначе (RBSP установки параметра изображение или последовательность RBSP установки параметра передают другими средствами, не описанными в этой Рекомендации | Международном стандарте), они должны быть доступны процессу декодирования вовремя, так чтобы эти ограничения выполнялись.

Все ограничения, которые выражают взаимодействие между значениями элементов синтаксиса (и значения переменных, полученных из этих элементов синтаксиса) в последовательности установки параметров, установки параметров изображения и другие элементы синтаксиса выражают ограничения, которые наложены только на активную последовательность установки параметров и на активную установку параметров изображения. Если представлена любая, не активированная в потоке битов последовательность RBSP установки параметров, элементы синтаксиса этой последовательности должны иметь значения, которые должны соответствовать определенным ограничениям в случае, если активация сделана с помощью контроля потока битов, подчиняющегося другим

требованиям соответствия. Если представлена любая, исходно не активированная в потоке битов RBSP установка параметров изображения, элементы синтаксиса этой последовательности должны иметь значения, которые должны соответствовать определенным ограничениям в случае, если активация сделана с помощью контроля потока битов, подчиняющегося другим требованиям соответствия.

Во время операции процесса декодирования (см. раздел 8) должны учитываться значения параметров действующей установки параметров изображения и действующей последовательности установки параметров. Для интерпретации сообщений SEI значения параметров установки параметра изображения и последовательности установки параметра, которые действуют в процессе декодирования для блоков NAL с VCL исходного кодированного изображения в том же блоке доступа, должны учитываться, пока иное не будет определено в семантике сообщения SEI.

7.4.1.2.2 Порядок блоков доступа и связь с кодированными видеопоследовательностями

Поток битов, соответствующий этой Рекомендации | Международному стандарту, состоит из одной или более кодированных видеопоследовательностей.

Кодированная видеопоследовательность состоит из одного или более блоков доступа. Порядок блоков NAL и кодированных изображений и их связь с блоками доступа описаны в п. 7.4.1.2.3.

Первый блок доступа каждой кодированной видеопоследовательности – это блок доступа IDR. Все последующие блоки доступа в кодированной видеопоследовательности – это не блоки доступа IDR.

Значения порядка счета изображений для кодированных изображений в последующих блоках доступа в порядке декодирования, содержащего неконтрольные изображения, должны быть неубывающими.

Если такой присутствует, то блок доступа, следующий за блоком доступа, который содержит конечную последовательность блоков NAL, должен быть блоком доступа IDR.

Если блок NAL с SEI содержит данные, которые принадлежат более чем одному блоку доступа (например, если блок NAL с SEI предназначен для кодированной видеопоследовательности), этот блок должен находиться в первом блоке доступа, в котором он будет использован.

Если в блоке доступа представлен конец потока блоков NAL, то этот блок должен быть последним блоком доступа в потоке битов, а концом потока блоков NAL должен быть последний блок NAL в этом блоке доступа.

7.4.1.2.3 Порядок блоков NAL и кодированных изображений и их связь с блоками доступа

Блок доступа состоит из одного исходного кодированного изображения, нуля или более соответствующих дополнительных кодированных изображений и нуля или более блоков NAL без VCL. Связь блоков NAL VCL с исходными или дополнительно кодированными изображениями описана в п. 7.4.1.2.5.

Первый из любых, следующих за блоками NAL (за последним блоком NAL VCL) исходных кодированных изображений, определяет характеристики начала нового блока доступа:

– разграничитель блок доступа блока NAL (если имеется);

– установку параметра последовательности блока NAL (если имеется);

– установку параметра изображения блока NAL (если имеется);

– блок NAL с SEI (если имеется);

– блоки NAL с nal_unit_type в диапазоне от 13 до 18 включительно;

– первый блок NAL VCL исходного кодированного изображения (имеется всегда).

Ограничения на обнаружение первого блока NAL VCL исходного кодированного изображения описаны в п. 7.4.1.2.4.

Следующие ограничения должны подчиняться в блоке доступа порядку кодированных изображений и блоков NAL без VCL.

– Если имеется разграничитель блока доступа блока NAL, он должен быть первым блоком NAL. В каждом блоке доступа должно быть не более одного разграничителя блока доступа блока NAL.

– Если имеются любые блоки NAL с SEI, они должны предшествовать исходному кодированному изображению.

– Если имеется блок NAL с SEI, содержащий сообщение SEI о периоде буферизации, это сообщение должно быть первой полезной нагрузкой сообщения SEI первого блока NAL SEI в блоке доступа.

– Ранее кодированному изображению должны предшествовать соответствующие дополнительные кодированные изображения.

– Если имеются дополнительные кодированные изображения, они должны располагаться в возрастающем порядке значения с redundant_pic_cnt.

– Если представлен конец последовательности блоков NAL, за ним должно следовать исходное кодированное изображение и все дополнительно кодированные изображения (если таковые имеются).

– Если представлен конец потока блоков NAL, то это должен быть последний блок NAL.

– Блоки NAL с nal_unit_type, равным 0, 12 или в диапазоне от 19 до 31 включительно, не должны предшествовать первому блоку NAL VCL исходного кодированного изображения.

ПРИМЕЧАНИЕ. – Установки параметров последовательности блоков NAL или установки параметров изображения блоков NAL могут быть представлены в блоке доступа, но не могут следовать за последним блоком NAL VCL исходного кодированного изображения в блоке доступа, поскольку это условие должно указывать на начало нового блока доступа.

ПРИМЕЧАНИЕ. – Если в блоке доступа имеется блок NAL с nal_unit_type, равным 7 или 8, он может быть или не быть отнесен к кодированным изображениям блока доступа, в котором он представлен, а может быть отнесен к кодированным изображениям последующих блоков доступа.

Структура блоков доступа, не содержащих никаких блоков NAL со значениями nal_unit_type, равными 0, 7, 8 или в диапазоне от 12 до 31 включительно, показана на рисунке 7-1.

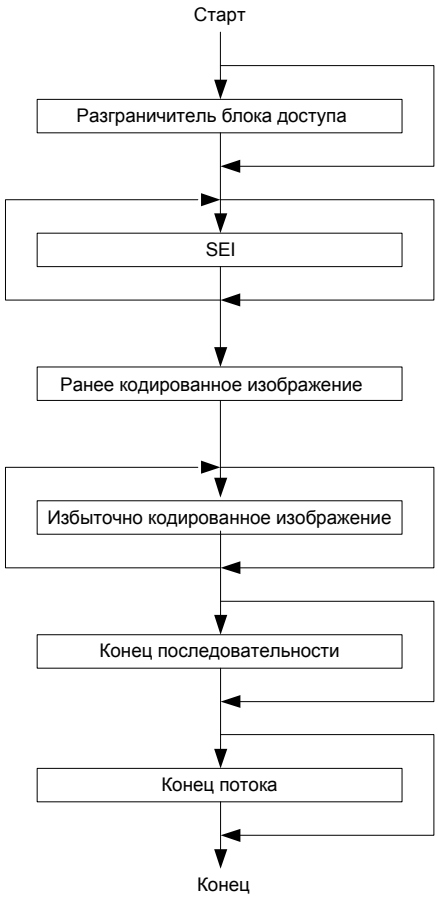

Рисунок 7-1 – Структура блока доступа, не содержащего никаких блоков NAL со значениями nal_unit_type, равными 0, 7, 8 или в диапазоне от 12 до 31 включительно

7.4.1.2.4 Обнаружение первого блока NAL VCL исходного кодированного изображения

В этом подразделе описаны ограничения на синтаксис блока NAL VCL, которые достаточны, чтобы обеспечить обнаружение первого блока NAL VCL каждого раннее кодированного изображения.

Любой блок NAL кодированной секции или блок NAL кодированной секции с разделенными данными типа A исходного кодированного изображения текущего блока доступа следует отличить от любого блока NAL кодированной секции или блока NAL кодированной секции с разделенными данными исходного кодированного изображения предыдущего блока доступа одним или некоторыми из следующих способов:

– frame_num различается по значению. Значение frame_num используют для проверки, что это условие является тем значением frame_num, которое появляется в синтаксисе заголовка секции независимо от того, считают ли это значение равным 0 для последующего использования в процессе декодирования вследствие присутствия равного 5 значения memory_management_control_operation,

ПРИМЕЧАНИЕ. – Последствие вышеприведенного утверждения заключается в том, что исходное кодированное изображение с frame_num, равным 1, не может содержать memory_management_control_operation, равным 5, пока не будет выполнены некоторые другие перечисленные ниже условия для следующего исходного кодированного изображения, которое последует за этим (если последует).

– pic_parameter_set_id отличается по величине,
– field_pic_flag отличается по величине,
– bottom_field_flag присутствует в обоих и отличается по величине,
– nal_ref_idc отличается по величине от nal_ref_idc, равного 0,
– pic_order_cnt_type равно 0 для обоих, и каждое значение pic_order_cnt_lsb отличается по величине либо отличается по величине значение delta_pic_order_cnt_bottom,
– pic_order_cnt_type равно 1 для обоих, и каждое значение delta_pic_order_cnt[0] отличается по величине либо отличается по величине delta_pic_order_cnt[1],
– nal_unit_type отличается по величине от значения nal_unit_type, равного 5,
– nal_unit_type равно 5 для обоих и отличается по величине от idr_pic_id.

ПРИМЕЧАНИЕ. – Некоторые блоки NAL VCL в дополнительных кодированных изображениях или некоторые блоки NAL без VCL (например, разграничитель блока NAL блока доступа) можно также использовать для обнаружения границ между блоками доступа, и поэтому эти блоки могут помогать обнаруживать начало нового исходного кодированного изображения.

7.4.1.2.5 Порядок блоков NAL VCL и связь с кодированными изображениями

Каждый блок NAL VCL – это часть кодированного изображения.

Порядок блоков NAL VCL в кодированном IDR изображении ограничен следующим образом:

– Если допустим произвольный порядок секций, как это определено в Приложении A, то кодированные секции блоков NAL IDR изображения могут иметь любой порядок относительно друг друга.
– Иначе (произвольный порядок секций не допустим), кодированные секции блоков NAL IDR изображения должны располагаться в возрастающем порядке адреса макроблока для первого макроблока каждой кодированной секции блока NAL IDR изображения.

Порядок блоков NAL VCL в кодированном не IDR изображении ограничен следующим образом:

– Если допустим произвольный порядок секций, как это определено в Приложении A, то кодированные секции блоков NAL не IDR изображения или A кодированные секции блоков NAL с разделенными данными типа A могут иметь любой порядок относительно друг друга. Кодированная секция блоков NAL разделенных данных A с конкретным значением slice_id должна предшествовать любой представленной кодированной секции блоков NAL разделенных данных B с тем же самым значением slice_id. Кодированная секция блоков NAL разделенных данных A с конкретным значением slice_id должна предшествовать любой представленной кодированной секции блоков NAL разделенных данных C с тем же самым значением slice_id. Если представлена кодированная секция блоков NAL разделенных данных B с конкретным значением slice_id, то она должна предшествовать любой представленной кодированной секции блока NAL разделенных данных C с тем же самым значением slice_id.
– Иначе (произвольный порядок секций не допустим), кодированные секции блоков NAL не IDR изображения или кодированные секции блоков NAL разделенных данных A должны располагаться в возрастающем порядке адреса макроблока для первого макроблока каждой кодированной секции блоков NAL не IDR изображения или кодированный секции блоков NAL разделенных данных A. Кодированная секция блоков NAL разделенных данных A с конкретным значением slice_id должна непосредственно предшествовать любой представленной кодированной секции блоков NAL разделенных данных B с тем же самым значением slice_id. Кодированная секция блоков NAL разделенных данных A с конкретным значением slice_id должна непосредственно предшествовать любой представленной кодированной секции блоков NAL разделенных данных C с тем же самым значением slice_id, если не представлена кодированная секции блоков NAL разделенных данных B с тем же самым значением slice_id. Если представлена кодированная секции блоков NAL разделенных данных B с тем же самым значением slice_id, она должна непосредственно предшествовать любой представленной кодированной секции блоков NAL разделенных данных C с тем же самым значением slice_id.

Блоки NAL с nal_unit_type, равным 12, могут присутствовать в блоке доступа, но не должны предшествовать первому блоку NAL VCL исходного кодированного изображения в блоке доступа.

Блоки NAL с nal_unit_type, равным 0 или в диапазоне от 24 до 31 включительно, которые не определены, могут присутствовать в блоке доступа, но не должны предшествовать первому блоку NAL VCL исходного кодированного изображения в блоке доступа.

Блоки NAL с nal_unit_type в диапазоне от 19 до 23 включительно, которые были зарезервированы, не должны предшествовать первому блоку NAL VCL исходного кодированного изображения в блоке доступа (если они описаны МСЭ-Т | ИСО/МЭК для будущих применений).

7.4.2 Полезная нагрузка последовательности исходных байтов и семантика концевых битов RBSP

7.4.2.1 Семантика установки параметров последовательности RBSP

profile_idc and **level_idc** указывает характеристики и уровень, которым соответствует поток битов, как это определено в Приложении A.

constraint_set0_flag, равный 1, указывает, что поток битов подчиняется всем ограничениям, описанным в п. A.2.1. constraint_set0_flag, равное 0, указывает, что поток битов может подчиняться или не подчиняться всем ограничениям, описанным в п. A.2.1.

constraint_set1_flag, равный 1, указывает, что поток битов подчиняется всем ограничениям, описанным в п. A.2.2. constraint_set1_flag, равное 0, указывает, что поток битов может подчиняться или не подчиняться всем ограничениям, описанным в п. A.2.2.

constraint_set2_flag, равный 1, указывает, что поток битов подчиняется всем ограничениям, описанным в п. A.2.3. constraint_set2_flag, равное 0, указывает, что поток битов может подчиняться или не подчиняться всем ограничениям, описанным в п. A.2.3.

ПРИМЕЧАНИЕ. – Если более одного из ограничений constraint_set0_flag, constraint_set1_flag или constraint_set2_flag равно 1, то поток битов подчиняется ограничениям всех указанных пунктов в п. A.2.

reserved_zero_5bits должен быть равен 0 в потоке битов, соответствующих этой Рекомендации | Международному стандарту. Прочие значения reserved_zero_5bits могут быть описаны в будущем МСЭ-Т | ИСО/МЭК. Декодеры должны игнорировать значение reserved_zero_5bits.

seq_parameter_set_id указывает установку параметров последовательности, на которую ссылается установка параметров изображения. Значение seq_parameter_set_id должно находиться в диапазоне от 0 до 31 включительно.

ПРИМЕЧАНИЕ. – В случае реализуемости кодеры должны использовать различающиеся значения seq_parameter_set_id, если значения других элементов синтаксиса установки параметров последовательности отличаются больше, чем изменяющиеся значения элементов синтаксиса, связанных с особым значением seq_parameter_set_id.

log2_max_frame_num_minus4 определяет значение переменной MaxFrameNum, которую используют в кадре frame_numrelated, полученных следующим образом:

$$\text{MaxFrameNum} = 2^{(\text{log2\_max\_frame\_num\_minus4} + 4)}. \quad (7\text{-}1)$$

Значение log2_max_frame_num_minus4 должно быть в диапазоне от 0 to 12 включительно.

pic_order_cnt_type определяет метод вычисления порядка декодирования изображения (как описано в п. 8.2.1). Значение pic_order_cnt_type должно быть в диапазоне от 0 до 2 включительно.

pic_order_cnt_type не должно быть равно 2 в кодированной видеопоследовательности, в которой содержится любое из следующих:

- блок доступа с неконтрольным кадром, за которым сразу следует блок доступа с неконтрольным изображением;
- два блока доступа, каждый содержащий поле с двумя полями, вместе формирующее дополнительную пару неконтрольных полей, за которыми сразу следует блок доступа с неконтрольным изображением;
- блок доступа с неконтрольным полем, за которым сразу следует блок доступа с другим неконтрольным изображением, которое не формирует дополнительную пару неконтрольных полей с первым из двух блоков доступа.

log2_max_pic_order_cnt_lsb_minus4 определяет значение переменной MaxPicOrderCntLsb, которую используют в процессе вычисления порядока декодирования изображения, как описано в п. 8.2.1, следующим образом:

$$\text{MaxPicOrderCntLsb} = 2^{(\text{log2\_max\_pic\_order\_cnt\_lsb\_minus4} + 4)}. \quad (7\text{-}2)$$

Значение log2_max_pic_order_cnt_lsb_minus4 должно быть в диапазоне от 0 до 12 включительно.

delta_pic_order_always_zero_flag, равный 1, определяет, что delta_pic_order_cnt[0] и delta_pic_order_cnt[1] отсутствуют в заголовках секции последовательности и должны считаться равными 0. delta_pic_order_always_zero_flag, равный 0, определяет, что delta_pic_order_cnt[0] присутствует в заголовках секции последовательности, а delta_pic_order_cnt[1] может находиться в заголовках секции последовательности.

offset_for_non_ref_pic используют для вычисления в порядке изображения неконтрольного изображения, как описано в п. 8.2.1. Значение offset_for_non_ref_pic должно быть в диапазоне от 2^{31} до $2^{31} – 1$ включительно.

offset_for_top_to_bottom_field используют для вычисления в порядке изображения нижнего поля в кадре, как описано в 8.2.1. Значение offset_for_top_to_bottom_field должно быть в диапазоне от $–2^{31}$ до $2^{31} – 1$ включительно.

num_ref_frames_in_pic_order_cnt_cycle используют в процессе декодирования для учета порядка изображения, как описано в п. 8.2.1. Значение num_ref_frames_in_pic_order_cnt_cycle должно быть в диапазоне от 0 до 255 включительно.

offset_for_ref_frame[i **]** – это элемент списка значений num_ref_frames_in_pic_order_cnt_cycle, использованных в процессе декодирования для учета порядка изображения, как описано в п. 8.2.1. Значение offset_for_ref_frame[i] должно быть в диапазоне от $–2^{31}$ до $2^{31} – 1$ включительно.

num_ref_frames определяет максимальное число сокращенных и удлиненных контрольных кадров, дополняющих пар контрольных полей и непарных(несдвоенных) контрольных полей, которые могут быть использованы процессом декодирования для внешнего предсказания любого изображения в последовательности. num_ref_frames также определяет размер действия скользящего окна, как описано в п. 8.2.5.3. Значение num_ref_frames должно быть в диапазоне от 0 до MaxDpbSize включительно (как описано в п. A.3.1).

gaps_in_frame_num_value_allowed_flag определяет разрешенные значения frame_num, как описано в п. 7.4.3, и процесс декодирования в случае предполагаемого расхождения между значениями frame_num, как описано в п. 8.2.5.2.

pic_width_in_mbs_minus1 плюс 1 определяет ширину каждого декодированного изображения в блоках макроблоков.

Переменную ширины изображения в блоках макроблоков находят следующим образом:

$$PicWidthInMbs = pic\_width\_in\_mbs\_minus1 + 1. \quad (7\text{-}3)$$

Переменную ширины изображения компонента яркости находят следующим образом:

$$PicWidthInSamples_L = PicWidthInMbs * 16. \quad (7\text{-}4)$$

Переменную ширины изображения компонентов цветности находят следующим образом:

$$PicWidthInSamples_C = PicWidthInMbs * 8. \quad (7\text{-}5)$$

pic_height_in_map_units_minus1 плюс 1 определяет высоту блоков секции отображения группы декодированного кадра или поля.

Переменные PicHeightInMapUnits и PicSizeInMapUnits находят следующим образом:

$$PicHeightInMapUnits = pic\_height\_in\_map\_units\_minus1 + 1 \quad (7\text{-}6)$$

$$PicSizeInMapUnits = PicWidthInMbs * PicHeightInMapUnits. \quad (7\text{-}7)$$

frame_mbs_only_flag, равный 0, определяет, что кодированные изображения кодированной видеопоследовательности могут быть либо кодированными полями, либо кодированными кадрами. frame_mbs_only_flag, равный 1, определяет, что каждое кодированное изображение кодированной видеопоследовательности – это кодированный кадр, содержащий только макроблоки кадра.

Разрешенный диапазон значений для pic_width_in_mbs_minus1, pic_hieght_in_map_units_minus1 и frame_mbs_only_flag описан в ограничениях Приложения A.

В зависимости от значения frame_mbs_only_flag значения семантики pic_hieght_in_map_units_minus1 присваивают следующим образом:

– Если frame_mbs_only_flag равно 0, то pic_hieght_in_map_units_minus1 плюс 1 – это высота поля в блоках макроблоков.

– Иначе (frame_mbs_only_flag равно 1), pic_hieght_in_map_units_minus1 плюс 1 – это высота кадра в блоках макроблоков.

Переменную FrameHeightInMbs находят следующим образом:

$$FrameHeightInMbs = (2 - frame\_mbs\_only\_flag) * PicHeightInMapUnits. \quad (7\text{-}8)$$

mb_adaptive_frame_field_flag, равное 0, определяет отсутствие переключения между кадром и полем макроблоков в изображении. mb_adaptive_frame_field_flag, равное 1, определяет возможное переключение между макроблоками кадра и поля в кадрах. Если mb_adaptive_frame_field_flag отсутствует, его считают равным 0.

direct_8x8_inference_flag определяет метод, использованный в процессе определения векторов движения яркости для значений B_Skip, B_Direct_16x16 и B_Direct_8x8, как описано в п. 8.4.1.2. Если frame_mbs_only_flag равно 0, direct_8x8_inference_flag должен быть равен 1.

frame_cropping_flag, равное 1, определяет, что параметры сдвига обрезки (изменения масштаба) кадра поступают после следующей установки параметров последовательности. frame_cropping_flag, равный 0, определяет, что параметры сдвига обрезки отсутствуют.

frame_crop_left_offset, **frame_crop_right_offset**, **frame_crop_top_offset**, **frame_crop_bottom_offset** определяют образцы кадра в прямоугольнике следующим образом:

– Если frame_mbs_only_flag равен 1, обрезанный прямоугольник содержит образцы яркости с горизонтальными координатами от 2 * frame_crop_left_offset до $PicWidthInSamples_L$ – (2 * frame_crop_right_offset + 1) и вертикальными координатами от 2 * frame_crop_top_offset до (FrameHeightInMbs * 16) – (2 * frame_crop_bottom_offset + 1) включительно. В этом случае значение frame_crop_left_offset должно быть в диапазоне от 0 до 8 * PicWidthInMbs – (frame_crop_right_offset + 1) включительно; значение

frame_crop_top_offset должно быть в диапазоне от 0 до 8 * FrameHeightInMbs – (frame_crop_bottom_offset + 1) включительно.

– Иначе (frame_mbs_only_flag равен 0), обрезанный прямоугольник содержит образцы яркости с горизонтальными координатами от 2 * frame_crop_left_offset до $PicWidthInSamples_L$ – (2 * frame_crop_right_offset + 1) и вертикальными координатами от 4 * frame_crop_top_offset до (FrameHeightInMbs * 16) – (4 * frame_crop_bottom_offset + 1) включительно. В этом случае значение frame_crop_left_offset должно быть в диапазоне от 0 до 8 * PicWidthInMbs – (frame_crop_right_offset + 1) включительно; значение frame_crop_top_offset должно быть в диапазоне от 0 до 4 * FrameHeightInMbs – (frame_crop_bottom_offset + 1) включительно.

Если frame_cropping_flag равно 0, должны подразумеваться следующие значения: frame_crop_left_offset = 0, frame_crop_right_offset = 0, frame_crop_top_offset = 0 и frame_crop_bottom_offset = 0.

Определенные образцы двух массивов цветности – это образцы с координатами кадра (x / 2, y / 2), где (x, y) – координаты кадра определенных образцов яркости.

Для декодированных полей описанные образцы декодированных полей являются такими, которые попадают в прямоугольник, определенный в координатах кадра.

vui_parameters_present_flag, равное 1, определяет, что структура синтаксиса vui_parameters( ), описанная в Приложении E, будет следующей в потоке битов. vui_parameters_present_flag, равный 0, определяет, что структура синтаксиса vui_parameters( ), описанная в Приложении E, не будет следующей в потоке битов.

7.4.2.2 Семантика установки RBSP параметров изображения

pic_parameter_set_id указывает на установку параметров изображения, на которую есть ссылки в заголовке секции. Значение pic_parameter_set_id должно быть в диапазоне от 0 до 255 включительно.

seq_parameter_set_id ссылается на действующую установку параметров последовательности. Значение seq_parameter_set_id должно быть в диапазоне от 0 до 31 включительно.

entropy_coding_mode_flag выбирает энтропию метода декодирования, использованного для элементов синтаксиса, в таблицах которых два дескриптора включены следующим образом:

– Если entropy_coding_mode_flag равен 0, используют метод, описанный левым дескриптором в таблице синтаксиса (кодированный Exp-Golomb, см. п. 9.1, или CAVLC, см. п. 9.2).

– Иначе (entropy_coding_mode_flag равен 1), используют метод, описанный правым дескриптором в таблице синтаксиса (CABAC, см. п. 9.3).

pic_order_present_flag, равный 1, определяет, что отсчеты порядка изображения относительно элементов синтаксиса находятся в заголовке секций, как описано в п. 7.3.3. pic_order_present_flag, равный 0, определяет, что отсчеты порядка изображения относительно элементов синтаксиса отсутствуют в заголовке секций.

num_slice_groups_minus1 плюс 1 определяет число групп секций для изображения. Если num_slice_groups_minus1 равно 0, все секции изображения принадлежат той же группе секций. Разрешенный диапазон num_slice_groups_minus1 описан в Приложении A.

slice_group_map_type определяет характеристики кодирования отображения группы секции и отображения блоков в группах секций. Значение slice_group_map_type должно быть в диапазоне от 0 до 6 включительно.

slice_group_map_type, равное 0, определяет чередование групп секций.

slice_group_map_type, равное 1, определяет распределенное отображение группы секций.

slice_group_map_type, равное 2, определяет одну или более "приоритетных" групп секций и "остаток" групп секций.

значения slice_group_map_type, равные 3, 4, и 5, определяют изменение групп секций. Если num_slice_groups_minus1 на равно 1, то значения slice_group_map_type не должны быть равны 3, 4 или 5.

slice_group_map_type, равное 6, определяет явно заданное присвоение группы секций каждому блоку отображения группы секций.

Блоки отображения группы секции характеризуют следующим образом:

– Если frame_mbs_only_flag равен 0, а mb_adaptive_frame_field_flag равен 1 и кодированное изображение – это кадр, то блоки отображения группы представляют пару блоков из макроблоков.

– Иначе, если frame_mbs_only_flag равен 1, или если кодированное изображение – это поле, то блоки отображения группы секций представляют блоки из макроблоков.

– Иначе (frame_mbs_only_flag равен 0, а mb_adaptive_frame_field_flag равен 0, а кодированное изображение – это кадр), блоки отображения группы секций представляют блоки из двух макроблоков, которые являются вертикально-смежными, как пара макроблоков кадра MBAFF.

run_length_minus1[i **]** используют для спецификации числа последовательных блоков отображения группы секций, которые должны быть присвоены i-ой группе секций в порядке растрового сканирования блоков отображения группы секций. Значение run_length_minus1[i] должно быть в диапазоне от 0 до PicSizeInMapUnits – 1 включительно.

top_left[i **]** и **bottom_right[** i **]** определяют левый верхний и нижний правый углы прямоугольника соответственно. top_left[i] и bottom_right[i] – это положения блоков отображения группы секций при растровом сканировании изображения блоков отображения группы секций. Для каждого прямоугольника i все следующие ограничения должны быть выдержаны значениями элементов синтаксиса top_left[i] и bottom_right[i]:

– top_left[i] должен быть меньше или равен bottom_right[i], а bottom_right[i] должен быть меньше чем PicSizeInMapUnits,

– (top_left[i] % PicWidthInMbs) должен быть меньше или равен значению (bottom_right[i] % PicWidthInMbs).

slice_group_change_direction_flag используют вместе с slice_group_map_type для спецификации уточненного типа отображения, если slice_group_map_type равен 3, 4 или 5.

slice_group_change_rate_minus1 используют для спецификации переменной SliceGroupChangeRate. SliceGroupChangeRate определяет кратное число блоков отображения группы секций, при которых размер группы секций может изменяться от одного изображения к другому. Значение slice_group_change_rate_minus1 должно быть в диапазоне от 0 до PicSizeInMapUnits – 1 включительно. Переменную SliceGroupChangeRate описывают следующим образом:

SliceGroupChangeRate = slice_group_change_rate_minus1 + 1. (7-9)

pic_size_in_map_units_minus1 используют для спецификации числа блоков отображения группы секций в изображении. pic_size_in_map_units_minus1 должно быть равно PicSizeInMapUnits – 1.

slice_group_id[i **]** определяет группу секций i-ого блока отображения группы секций в порядке растрового сканирования. Размер элемента синтаксиса slice_group_id[i] – это Ceil(Log2(num_slice_groups_minus1 + 1)) битов. Значение of slice_group_id[i] должно быть в диапазоне от 0 до num_slice_groups_minus1 включительно.

num_ref_idx_l0_active_minus1 определяет максимум индекса контроля для контрольного изображения списка 0, который должен быть использован для декодирования каждой секции изображения. Для этого изображения используют список 0, если для этой секции num_ref_idx_active_override_flag равно 0. Если MbaffFrameFlag равно 1, то num_ref_idx_l0_active_minus1 – это максимум индекса для декодирования макроблоков кадра, а 2 * num_ref_idx_l0_active_minus1 + 1 – это максимум индекса для декодирования макроблоков поля. Значение num_ref_idx_l0_active_minus1 должно быть в диапазоне от 0 до 31 включительно.

num_ref_idx_l1_active_minus1 имеет ту же семантику, что и num_ref_idx_l0_active_minus1 с заменой l0 и list 0 на l1 и list 1 соответственно.

weighted_pred_flag, равное 0, определяет, что взвешенное предсказание не должно использоваться в секциях P и SP. Флаг weighted_pred_flag, равный 1, определяет, что взвешенное предсказание должно использоваться в секциях P и SP.

weighted_bipred_idc, равное 0, определяет, что взвешенное предсказание по умолчанию должно использоваться в секциях B. Значение weighted_bipred_idc, равное 2, определяет, что подразумеваемое взвешенное предсказание должно использоваться в секциях B. Значение weighted_bipred_idc должно быть в диапазоне от 0 до 2 включительно.

pic_init_qp_minus26 определяет начальное значение минус 26 $SliceQP_Y$ для каждой секции. Начальное значение модифицируют в слое секции, если декодировано ненулевое значение slice_qp_delta, и модифицируют в дальнейшем, если декодировано ненулевое значение mb_qp_delta в слое макроблока. Значение pic_init_qp_minus26 должно быть в диапазоне от –26 до +25 включительно.

pic_init_qs_minus26 определяет начальное значение минус 26 $SliceQS_Y$ для всех макроблоков в секциях SP или SI. Начальное значение модифицируют в слое секции, если декодировано ненулевое значение slice_qs_delta. Значение pic_init_qs_minus26 должно быть в диапазоне от –26 до +25 включительно.

chroma_qp_index_offset определяет сдвиг, который должен быть добавлен к QP_Y и QS_Y для адресации в таблице значений QP_C. Значение chroma_qp_index_offset должно быть в диапазоне от –12 до +12 включительно.

deblocking_filter_control_present_flag, равное 1, определяет, что установка элементов синтаксиса, контролирующих характеристики фильтра разделения на блоки, представлена в заголовках секций. deblocking_filter_control_present_flag, равный 0, определяет что установка элементов синтаксиса, контролирующих характеристики фильтра разделения на блоки, не представлена в заголовках секций, а действуют их предполагаемые значения.

constrained_intra_pred_flag, равное 0, определяет, что внутреннее предсказание допускает использование оставшихся данных и декодированных образцов смежных макроблоков, кодированных с использованием режимов внешнего предсказания макроблоков, для предсказания макроблоков, кодированных с использованием внутренних режимов предсказания. constrained_intra_pred_flag, равный 1, определяет ограниченное внутреннее предсказание. В этом случае предсказание макроблоков, кодированных с использованием внутренних режимов предсказания макроблоков, будет использовать только оставшиеся данные и декодированные образцы от I или SI типов макроблоков.

redundant_pic_cnt_present_flag, равное 0, определяет, что элемент синтаксиса redundant_pic_cnt не представлен в заголовках секций, разделенных участках данных B и разделенных участках данных C, которые считают (либо непосредственно, либо по ассоциации с соответствующим разделением данных A) установкой параметров изображения. Значение redundant_pic_cnt_present_flag, равное 1, определяет, что элемент синтаксиса redundant_pic_cnt представлен во всех заголовках секций, разделенных участков данных B и разделенных участков данных C, которые считают (либо непосредственно, либо по ассоциации с соответствующим разделением данных A) установкой параметров изображения.

7.4.2.3 Семантика дополнительного расширения информации RBSP

Дополнительное расширение информации (SEI) содержит информацию, которая не является необходимой для декодирования образцов кодированных изображений из блоков NAL VCL.

7.4.2.3.1 Семантика сообщения дополнительного расширения информации

Блок NAL с SEI содержит одно или более сообщений SEI. Каждое сообщение SEI состоит из переменных, определяющих тип, payloadType, и размер payloadSize полезной нагрузки SEI. Полезная нагрузка SEI описана в Приложении D. Полученный размер полезной нагрузки, payloadSize, определен в байтах и должен быть равен числу байтов в полезной нагрузке SEI.

ff_byte – байт, равный 0xFF, определяет необходимость дальнейшего представления использованной структуры синтаксиса.

last_payload_type_byte – последний байт типа полезной нагрузки сообщения SEI.

last_payload_size_byte – последний байт размера сообщения SEI.

7.4.2.4 Семантика разграничителя RBSP блока доступа

Разграничитель блока доступа может быть использован для указания типа секций, представленных в исходном кодированном изображении, и для упрощения определения границы между блоками доступа. Не существует нормативного процесса декодирования, связанного с разграничителем блока доступа.

primary_pic_type указывает, что значения slice_type для всех секций исходного кодированного изображения являются элементами набора, который перечислен в таблице 7-2 для всех заданных значений primary_pic_type.

Таблица 7-2 – Значения primary_pic_type

primary_pic_type	Значения slice_type, которые могут присутствовать в исходном кодированном изображении
0	I
1	I, P
2	I, P, B
3	SI
4	SI, SP
5	I, SI
6	I, SI, P, SP
7	I, SI, P, SP, B

7.4.2.5 Семантика конца последовательности RBSP

Конец последовательности RBSP определяет, что последующий блок доступа в потоке битов в порядке декодирования (если такой имеется) должен быть блоком доступа IDR. Содержание синтаксиса SODB и RBSP для конца последовательности RBSP пусто. Для конца последовательности RBSP отсутствует нормативный процесс декодирования.

7.4.2.6 Семантика конца потока битов RBSP

Конец потока битов RBSP указывает, что никаких дополнительных блоков NAL не должно присутствовать в потоке битов, который продвигается к концу потока RBSP в порядке декодирования. Содержание синтаксиса SODB RBSP конца потока RBSP является пустым. Для конца потока RBSP нормативный процесс декодирования не определен.

7.4.2.7 Семантика данных заполнителя RBSP

Данные заполнителя RBSP содержат байты, значение которых должно быть равным 0xFF. Для данных заполнителя RBSP нормативный процесс декодирования не определен.

ff_byte – байт, равный 0xFF.

7.4.2.8 Семантика RBSP слоя без разделения

Слой секции без разделения RBSP состоит из заголовка секции и данных секции.

7.4.2.9 Семантика RBSP деления на части данных секции

7.4.2.9.1 Семантика RBSP деления на части данных A секции

Если используют деление на части данных секции, кодированные данные для простой секции делят на три разные части. Часть A содержит все элементы синтаксиса категории 2.

Элементы синтаксиса категории 2 включают все элементы синтаксиса в заголовке секции, и структуры синтаксиса данных секции иные, чем элементы синтаксиса в оставшейся( ) структуре синтаксиса.

slice_id идентифицирует секцию, связанную с делением данных. Каждая секция должна иметь особое значение slice_id в кодированном изображении, которое содержит эта секция. Если произвольный порядок секции не допустим, как это определено в Приложении A, первая секция кодированного изображения в порядке декодирования должна иметь slice_id равное 0, значение slice_id должно возрастать на единицу для каждой последующей секции кодированного изображения в порядке декодирования.

Диапазон slice_id описан следующим образом:

– Если MbaffFrameFlag равно 0, slice_id должно быть в диапазоне от 0 до PicSizeInMbs – 1 включительно.

– Иначе (MbaffFrameFlag равно 1), slice_id должно быть в диапазоне от 0 до PicSizeInMbs / 2 – 1 включительно.

7.4.2.9.2 Семантика RBSP деления на части данных B секции

Если используют деление на части данных секции, кодированные данные для простой секции делят на три разные части. Часть разделенных данных B содержит все элементы синтаксиса категории 3.

Элементы синтаксиса категории 4 включают все элементы синтаксиса в оставшейся( ) структуре синтаксиса и в структурах синтаксиса, использованных для общих типов макроблоков I и SI, как описано в таблице 7-7.

slice_id имеет такую же семантику, которая описана в п. 7.4.2.9.1.

redundant_pic_cnt должен быть равным 0 для секций и секций с разделенными данными, принадлежащими к исходному кодированному изображению. Значение redundant_pic_cnt должно быть больше 0 для кодированных секций и разделенных частей данных кодированных секций в оставшихся кодированных изображениях. Если redundant_pic_cnt отсутствует, это значение должно считаться равным 0. Значение redundant_pic_cnt должно быть в диапазоне от 0 до 127 включительно.

Наличие секции RBSP с разделенными данными B определено следующим образом:

– Если элементы синтаксиса секции RBSP с разделенными данными A указывают на наличие любого элемента синтаксиса категории 3 в данных секции, секция RBSP с разделенными данными B должна присутствовать с тем же самым значением slice_id и redundant_pic_cnt, что и в секции RBSP с разделенными данными A.

– Иначе (элементы синтаксиса секции RBSP с разделенными данными A не указывают на наличие любого элемента синтаксиса категории 3 в данных секции), секция RBSP с разделенными данными B не должна присутствовать с тем же самым значением slice_id and redundant_pic_cnt, что и в секции RBSP с разделенными данными A.

7.4.2.9.3 Семантика RBSP деления на части данных C секции

Если используют деление на части данных секции, кодированные данные для простой секции делят на три разные части. Часть разделенных данных C содержит все элементы синтаксиса категории 4.

Элементы синтаксиса категории 4 включают все элементы синтаксиса в оставшейся( ) структуре синтаксиса и в структурах синтаксиса, использованных для общих типов макроблоков P и B, как описано в таблице 7-7.

slice_id имеет такую же семантику, которая описана в п. 7.4.2.9.1.

redundant_pic_cnt имеет такую же семантику, которая описана в п. 7.4.2.9.2.

Присутствие секции RBSP с разделенными данными C определено следующим образом:

– Если элементы синтаксиса секции RBSP с разделенными данными A указывают на наличие любого элемента синтаксиса категории 4 в данных секции, секция RBSP с разделенными данными C должна присутствовать с тем же самым значением slice_id и redundant_pic_cnt, что и в секции RBSP с разделенными данными A.

– Иначе (элементы синтаксиса секции RBSP с разделенными данными A не указывают на наличие любого элемента синтаксиса категории 4 в данных секции), секция RBSP с разделенными данными C не должна присутствовать с тем же самым значением slice_id и redundant_pic_cnt, что и в секции RBSP с разделенными данными A.

7.4.2.10 Семантика секции концевых битов RBSP

cabac_zero_word – это байт-синхронизированная последовательность из двух байтов, равных 0x0000.

Допустим NumBytesInVclNALunits – сумма значений NumBytesInNALunit для всех блоков NAL VCL кодированного изображения.

Допустим BinCountsInNALunits – число раз, с которым процесс анализа функции DecodeBin(), описанный в п. 9.3.3.2, запускают для декодирования содержания всех блоков NAL VCL кодированного изображения. Если entropy_coding_mode_flag равен 1, то BinCountsInNALunits не должно превышать (32 ÷ 3) * NumBytesInVclNALunits + 96 * PicSizeInMbs.

ПРИМЕЧАНИЕ. – Ограничение по максимуму числа бинов, возникающих при декодировании содержания слоя секций блоков NAL, может быть выполнено введением числа элементов синтаксиса cabac_zero_word для увеличения значения NumBytesInVclNALunits. Каждое значение cabac_zero_word представлено в блоке NAL трехбайтовыми последовательностями 0x000003 (как результат ограничения на содержание блока NAL, которое возникает при требуемом включении значения emulation_prevention_three_byte для каждого значения cabac_zero_word).

7.4.2.11 Семантика RBSP концевых битов

rbsp_stop_one_bit – единичный бит, равный 1.

rbsp_alignment_zero_bit единичный бит, равный 0.

7.4.3 Семантика заголовка секции

В случае присутствия значения элементов синтаксиса заголовка секции pic_parameter_set_id значения frame_num, field_pic_flag, bottom_field_flag, idr_pic_id, pic_order_cnt_lsb, delta_pic_order_cnt_bottom, delta_pic_order_cnt[0], delta_pic_order_cnt[1], sp_for_switch_flag и slice_group_change_cycle должны быть одинаковыми во всех заголовках секций кодированного изображения.

first_mb_in_slice определяет адрес первого макроблока в секции. Если произвольный порядок секций не разрешен, как это определено в Приложении A, значение first_mb_in_slice не должно быть меньше значения first_mb_in_slice для любой другой секции текущего изображения, которое предшествует текущей секции в порядке декодирования.

Первый адрес макроблока секции находят следующим образом:

– Если MbaffFrameFlag равно 0, first_mb_in_slice – это адрес макроблока, первого в секции, а значение first_mb_in_slice должно быть в диапазоне от 0 до PicSizeInMbs – 1 включительно.

– Иначе (MbaffFrameFlag равно 1), first_mb_in_slice * 2 – это адрес макроблока, первого в секции, который является верхним макроблоком из первой пары макроблоков в этой секции, а значение first_mb_in_slice должно быть в диапазоне от 0 до PicSizeInMbs / 2 – 1 включительно.

slice_type определяет тип кодирования секции, согласно таблице **7-**3.

Таблица 7-3 – Наименование связи с slice_type

slice_type	Наименование связи slice_type
0	P (P секция)
1	B (B секция)
2	I (I секция)
3	SP (SP секция)
4	SI (SI секция)
5	P (P секция)
6	B (B секция)
7	I (I секция)
8	SP (SP секция)
9	SI (SI секция)

Кроме типа кодирования текущей секции, значения slice_type в диапазоне 5..9 определяют, что все другие секции текущего кодированного изображения должны иметь значение slice_type, равное текущему значению slice_type или текущему значению slice_type – 5.

Если nal_unit_type равно 5 (IDR изображение), slice_type должен быть равен 2, 4, 7 или 9.

Если num_ref_frames равно 0, slice_type должен быть равен 2, 4, 7, или 9.

pic_parameter_set_id определяет используемую установку параметров изображения. Значение pic_parameter_set_id должно быть в диапазоне от 0 до 255 включительно.

frame_num используют как идентификатор изображения, и это значение должно быть представлено log2_max_frame_num_minus4 + 4 битами в потоке битов. Значение frame_numis ограничено следующим образом:

Переменную PrevRefFrameNum определяют следующим образом:

– Если текущее изображение – это IDR изображение, то PrevRefFrameNum устанавливают равным 0.

  – Иначе (текущее изображение – это не IDR изображение), PrevRefFrameNum устанавливают равным значению frame_numfor предыдущего блока доступа в порядке декодирования, который содержит контрольное изображение.

Значение frame_numis ограничено следующим образом:

– Если текущее изображение – это IDR изображение, то frame_num должно быть равным 0.

– Иначе (текущее изображение – это IDR изображение), ссылаясь на исходное кодированное изображение в предыдущем блоке доступа в порядке декодирования, который содержит контрольное изображение в качестве предыдущего контрольного изображения, значение frame_numfor текущего изображения не должно быть равно PrevRefFrameNum до тех пор, пока все следующие условия не будут истинными:

  – текущее изображение и предыдущее контрольное изображение принадлежат последующим блокам доступа в порядке декодирования;

  – текущее изображение и предыдущее контрольное изображение – это контрольные поля, имеющие инверсное равенство;

  – одно или более из следующих условий является истинным:

    – предыдущее контрольное изображение – это IDR изображение,

    – предыдущее контрольное изображение включает значение элемента синтаксиса memory_management_control_operation, равное 5,

      ПРИМЕЧАНИЕ. – Если предыдущее контрольное изображение включает значение элемента синтаксиса memory_management_control_operation, равное 5, то PrevRefFrameNum равно 0.

    – имеется исходное кодированное изображение, которое предшествует предыдущему контрольному изображению, а исходное кодированное изображение, которое предшествует предыдущему контрольному изображению, не содержит frame_num, равное PrevRefFrameNum,

    – имеется исходное кодированное изображение, которое предшествует предыдущему контрольному изображению, а исходное кодированное изображение, которое предшествует предыдущему контрольному изображению, не является контрольным изображением.

Если значение frame_numis не равно PrevRefFrameNum, то применяют следующее:

– В порядке декодирования не должно быть никаких предыдущих полей или кадров, которые отмечены как "использованные для укороченного контроля" со значением frame_num, равным любому значению, полученному переменной UnusedShortTermFrameNum в следующих случаях:

$$\begin{aligned} &\text{UnusedShortTermFrameNum} = (\text{PrevRefFrameNum} + 1) \,\%\, \text{MaxFrameNum} \\ &\text{while}(\text{UnusedShortTermFrameNum} \mathrel{!}= \text{frame\_num}) \\ &\quad \text{UnusedShortTermFrameNum} = (\text{UnusedShortTermFrameNum} + 1) \,\%\, \text{MaxFrameNum}. \end{aligned} \tag{7-10}$$

– Значение frame_numis ограничено следующим образом:

  – Если gaps_in_frame_num_value_allowed_flag равно 0, значение frame_num для текущего изображения должно быть равно (PrevRefFrameNum + 1) % MaxFrameNum.

  – Иначе (gaps_in_frame_num_value_allowed_flag равно 1), используют следующее:

    – Если frame_numis больше чем PrevRefFrameNum, то не должно быть никаких неконтрольных изображений в потоке битов, который следует за предыдущим контрольным изображением и предшествует текущему изображению в порядке декодирования, в котором любое из следующих условий является истинным:

      – Значение frame_numfor неконтрольного изображения меньше, чем PrevRefFrameNum.

      – Значение frame_numfor неконтрольного изображения больше, чем значение frame_numfor текущего изображения.

– Иначе (frame_numis меньше чем PrevRefFrameNum), не должно быть никаких неконтрольных изображений в потоке битов, который следует за предыдущим контрольным изображением и предшествует текущему изображению в порядке декодирования, в котором любое из следующих условий является истинным:
  – Значение frame_numfor неконтрольного изображения меньше, чем PrevRefFrameNum.
  – Значение frame_numfor неконтрольного изображения больше, чем frame_numfor текущего изображения.

Изображение, включающее memory_management_control_operation, равное 5, должно иметь ограничения frame_num, как описано выше, а после декодирования текущего изображения и обработки операций управления памятью считают, что изображение должно иметь значение frame_num, равное 0, для всех последующих использований в процессе декодирования, за исключением описанных в п. 7.4.1.2.4.

ПРИМЕЧАНИЕ. – Если исходное кодированное изображение не является IDR изображением и не содержит элемент синтаксиса memory_management_control_operation, равное 5, значение frame_num соответствующего дополнительного кодированного изображения будет таким же, как значение frame_numin исходного кодированного изображения. Альтернативно, дополнительно кодированное изображение включает элемент синтаксиса memory_management_control_operation, равный 5, а соответствующее исходное кодированное изображение является IDR изображением.

field_pic_flag, равный 1, определяет, что данная секция – это секция кодированного поля. field_pic_flag, равный 0, определяет, что данная секция – это секция кодированного кадра. Если отсутствует field_pic_flag, то считают, что это значение должно быть равным 0.

Переменную MbaffFrameFlag определяют следующим образом:

MbaffFrameFlag = (mb_adaptive_frame_field_flag && !field_pic_flag). (7-11)

Переменную для изображения высоты в блоках макроблоков определяют следующим образом:

PicHeightInMbs = FrameHeightInMbs / (1 + field_pic_flag). (7-12)

Переменную для изображения высоты компонента яркости определяют следующим образом:

$PicHeightInSamples_L$ = PicHeightInMbs * 16. (7-13)

Переменную для изображения высоты компонента цветности определяют следующим образом:

$PicHeightInSamples_C$ = PicHeightInMbs * 8. (7-14)

Переменную PicSizeInMbs для текущего изображения определяют согласно равенству:

PicSizeInMbs = PicWidthInMbs * PicHeightInMbs. (7-15)

Переменную MaxPicNum определяют следующим образом:

– Если field_pic_flag равен 0, MaxPicNum устанавливают равным MaxFrameNum.

– Иначе (field_pic_flag равен 1), MaxPicNum устанавливают равным 2*MaxFrameNum.

Переменную CurrPicNum определяют следующим образом:

– Если field_pic_flag равен 0, CurrPicNum устанавливают равным frame_num.

– Иначе (field_pic_flag равен 1), CurrPicNum устанавливают равным 2 * frame_num + 1.

bottom_field_flag, равный 1, определяет, что эта секция составляет часть нижнего кодированного поля. bottom_field_flag, равный 0, определяет, что изображение – это кодированное верхнеее поле. Если этот элемент синтаксиса не представлен в текущей секции, он должен считаться равным 0.

idr_pic_id определяет IDR изображение. Значение idr_pic_id во всех секциях IDR изображения должно оставаться неизменным. Если оба из двух последующих блоков доступа в порядке декодирования являются IDR блоками доступа, значение idr_pic_id в секциях первого такого IDR блока доступа должно отличаться от idr_pic_id второго такого IDR блока доступа. Значение idr_pic_id должно быть в диапазоне от 0 до 65535 включительно.

pic_order_cnt_lsb определяет модуль счета порядка изображения MaxPicOrderCntLsb для верхнего поля кодированного кадра или кодированного поля. Размер элемента синтаксиса pic_order_cnt_lsb – это log2_max_pic_order_cnt_lsb_minus4 + 4 битов. Значение pic_order_cnt_lsb должно быть в диапазоне от 0 до MaxPicOrderCntLsb – 1 включительно.

delta_pic_order_cnt_bottom определяет разность счета в порядке изображения между нижним и верхним полями кодированного кадра следующим образом:

– Если текущее изображение включает memory_management_control_operation, равное 5, значение delta_pic_order_cnt_bottom должно быть в диапазоне от (1 – MaxPicOrderCntLsb) до $2^{31} – 1$ включительно.

– Иначе (текущее изображение не включает memory_management_control_operation, равное 5), значение delta_pic_order_cnt_bottom должно быть в диапазоне от $–2^{3T}$ до $2^{31} – 1$ включительно.

Если этот элемент синтаксиса не представлен в потоке битов для текущей секции, его полагают равным 0.

delta_pic_order_cnt[0 **]** определяет разность счета в порядке изображения от ожидаемого порядка изображения для верхнего поля кодированного кадра или кодированного поля, как описано в п. 8.2.1. Значение delta_pic_order_cnt[0] должно быть в диапазоне от $–2^{31}$ до $2^{31} – 1$ включительно. Если этот элемент синтаксиса не представлен в потоке битов для текущей секции, его полагают равным 0.

delta_pic_order_cnt[1 **]** определяет разность счета в порядке изображения от ожидаемого порядка изображения для нижнего поля кодированного кадра или кодированного поля, как описано в п. 8.2.1. Значение delta_pic_order_cnt[1] должно быть в диапазоне от $–2^{31}$ до $2^{31} – 1$ включительно. Если этот элемент синтаксиса не представлен в потоке битов для текущей секции, его полагают равным 0.

redundant_pic_cnt должен быть равным 0 для секций и частей секций с разделенными данными, принадлежащими исходному кодированному изображению. Значение redundant_pic_cnt должно быть больше, чем 0 для кодированных секций или кодированных секций с разделенными данными дополнительного кодированного изображения. Если redundant_pic_cnt не представлен в потоке битов, его полагают равным 0. Значение redundant_pic_cnt должно быть в диапазоне от 0 до 127 включительно.

ПРИМЕЧАНИЕ. – Не должно быть заметного различия между любой областью декодированного первичного изображения и соответствующей областью, которая является результатом применения процесса декодирования, описанного в главе 8 для любого дополнительного изображения в том же блоке доступа.

Значение pic_parameter_set_id в кодированной секции или кодированной секции с разделенными данными дополнительного кодированного изображения должно быть таким, чтобы значение pic_order_present_flag в установке параметров изображения при использовании дополнительного кодированного изображения было равно значению pic_order_present_flag при текущей установке параметров изображения, соответствующей исходному кодированному изображению.

В представленном исходном кодированном изображении и любом дополнительном кодированном изображении, следующий за элементами синтаксиса элемент должен иметь те же значения: field_pic_flag, bottom_field_flag и idr_pic_id.

Если значение nal_ref_idc в блоке NAL VCL блока доступа равно 0, значение nal_ref_idc во всех других блоках NAL VCL того же блока доступа должно быть равным 0.

ПРИМЕЧАНИЕ. – Вышеприведенное ограничение имеет также следующий смысл. Если значение nal_ref_idc для блоков NAL VCL исходного кодированного изображения равно 0, то значения nal_ref_idc для блоков NAL VCL любого соответствующего дополнительного кодированного изображения равны 0; иначе (значение nal_ref_idc для блоков NAL VCL исходного кодированного изображения больше 0), значения nal_ref_idc для блоков NAL VCL любого соответствующего дополнительного кодированного изображения также больше 0.

Статус разметки контрольных изображений и значение frame_numafter после того, как процесс разметки декодированного контрольного изображения (согласно п. 8.2.5) был запущен для исходного кодированного изображения или любого дополнительного кодированного изображения того же блока доступа, должен оставаться тем же самым независимо от того, были ли декодированы исходное кодированное изображение или дополнительное кодированное изображение (вместо исходного кодированного изображения) этого блока доступа.

ПРИМЕЧАНИЕ. – Вышеприведенное ограничение имеет также следующий смысл.

Если исходное кодированное изображение – это не IDR изображение, содержание структуры синтаксиса dec_ref_pic_marking( ) должно быть идентичным во всех заголовках секций исходного кодированного изображения и всех дополнительных кодированных изображений, соответствующих этому исходному кодированному изображению.

Иначе (исходное кодированное изображение – это IDR изображение), применяют следующее.

Если дополнительное кодированное изображение, соответствующее исходному кодированному изображению, – это не IDR изображение, то содержание структуры синтаксиса dec_ref_pic_marking( ) должно быть идентичным во всех заголовках секций исходного кодированного изображения и во всех дополнительных кодированных изображениях, соответствующих исходному кодированному изображению.

Иначе (дополнительное кодированное изображение, соответствующее исходному кодированному изображению, – это IDR изображение), все заголовки секций дополнительного изображения должны содержать структуру синтаксиса( ), включающую элемент синтаксиса memory_management_control_operation, равный 5, а далее применяют следующее.

Если значение long_term_reference_flag в исходном кодированном изображении равно 0, то структура синтаксиса dec_ref_pic_marking дополнительного кодированного изображения не должна включать элемент синтаксиса memory_management_control_operation, равный 6.

Иначе (значение long_term_reference_flag в исходном кодированном изображении равно 1), структура синтаксиса dec_ref_pic_marking дополнительного кодированного изображения должна включать элемент синтаксиса memory_management_control_operation, равный 5, 4 и 6 в порядке декодирования, значение max_long_term_frame_idx_plus1 должно быть равно 1, а значение long_term_frame_idx должно быть равно 0.

Значения TopFieldOrderCnt и BottomFieldOrderCnt (если они применимы), которые являются результатом завершения процесса декодирования любого дополнительного кодированного изображения или исходного кодированного изображения того же блока доступа должны быть идентичными независимо от того, будут ли декодированы исходное кодированное изображение или любое дополнительное кодированное изображение (вместо исходного кодированного изображения) блока доступа.

Не существует обязательного процесса декодирования для кодированной секции (или для кодированной секции с разделенными данными) дополнительного кодированного изображения. Если значение redundant_pic_cnt заголовка кодированной секции больше 0, декодер может отбросить эту кодированную секцию. Однако кодированная секция

(или кодированная секция с разделенными данными) любого дополнительного кодированного изображения должна подчиняться тем же ограничениям, что и кодированная секция (или кодированной секции с разделенными данными) исходного изображения.

ПРИМЕЧАНИЕ. – Если некоторые образцы в исходном декодированном изображении не могут быть корректно декодированы из-за ошибок или потерь в передаче последовательности, а кодированная дополнительная секция может быть декодирована корректно, декодер должен заменить образцы декодированного исходного изображения соответствующими образцами декодированной дополнительной секции. Если более чем одна дополнительная секция перекрывает существенную область исходного изображения, должна быть использована дополнительная секция, имеющая наименьшее значение redundant_pic_cnt.

Дополнительные секции и секции разделенных данных, имеющих такое же значение redundant_pic_cnt, принадлежат к этому же дополнительному изображению. Декодированные секции в этом же дополнительном изображении не должны покрывать и перекрывать всю площадь изображения.

direct_spatial_mv_pred_flag определяет метод, использованный в процессе декодирования для получения векторов движения и индексов контроля для внешнего предсказания следующим образом:

– Если direct_spatial_mv_pred_flag равен 1, процесс отыскания векторов движения яркости для B_Skip, B_Direct_16x16 и B_Direct_8x8 в п. 8.4.1.2 должен использовать режим прямого пространственного предсказания, как описано в п. 8.4.1.2.2.

– Иначе (direct_spatial_mv_pred_flag равен 0), процесс отыскания векторов движения яркости для B_Skip, B_Direct_16x16 и B_Direct_8x8 в п. 8.4.1.2 должен использовать режим временного прямого предсказания, как описано в п. 8.4.1.2.3.

num_ref_idx_active_override_flag, равный 0, определяет, что действуют значения элементов синтаксиса num_ref_idx_l0_active_minus1 и num_ref_idx_l1_active_minus1, которые определены в установке параметров изображения. num_ref_idx_active_override_flag, равный 1, определяет, что значения num_ref_idx_l0_active_minus1 и num_ref_idx_l1_active_minus1, которые определены в установке параметров изображения, отменены для текущей секции (и только для текущей секции) следующими значениями в заголовке секции:

Если текущая секция – это секция P, SP или B, а field_pic_flag равен 0 и значение num_ref_idx_l0_active_minus1 в установке параметров изображения превышает 15, то num_ref_idx_active_override_flag должен быть равен 1.

Если текущая секция – это секция B, а field_pic_flag равен 0 и значение num_ref_idx_l1_active_minus1 в установке параметров изображения превышает 15, то num_ref_idx_active_override_flag должен быть равен 1.

num_ref_idx_l0_active_minus1 определяет максимум индекса контроля для контрольного изображения списка 0, который должен быть использован для декодирования секции.

Диапазон num_ref_idx_l0_active_minus1 описан следующим образом:

– Если field_pic_flag равен 0, num_ref_idx_l0_active_minus1 должен быть в диапазоне от 0 до 15 включительно. Если MbaffFrameFlag равен 1, то num_ref_idx_l0_active_minus1 – это максимальное значение индекса для декодирования кадров макроблоков, а 2 * num_ref_idx_l0_active_minus1 + 1 – это максимальное значение индекса для декодирования полей макроблоков.

– Иначе (field_pic_flag равен 1), num_ref_idx_l0_active_minus1 должен быть в диапазоне от 0 до 31 включительно.

num_ref_idx_l1_active_minus1 имеет ту же семантику, что и num_ref_idx_l0_active_minus1 с заменой l0 и список 0, на l1 и список 1 соответственно.

cabac_init_idc описывает индекс для определения таблицы инсталляции, использованной в процессе инсталляции переменных контекста. Значение cabac_init_idc должно быть в диапазоне от 0 до 2 включительно.

slice_qp_delta определяет начальное значение QP_Y, которое следует использовать для всех макроблоков в секции, которые до тех пор модифицировали значением mb_qp_delta в слое макроблока. Начальный параметр квантования QP_Y для секции вычисляют следующим образом:

$$SliceQP_Y = 26 + pic\_init\_qp\_minus26 + slice\_qp\_delta. \quad (7\text{-}16)$$

Значение slice_qp_delta должно быть ограничено таким образом, чтобы $SliceQP_Y$ находилось в диапазоне от 0 до 51 включительно.

sp_for_switch_flag определяет процесс декодирования, который следует использовать для декодирования макроблоков P в секции SP следующим образом:

– Если sp_for_switch_flag равен 0, макроблоки P в секции SP должны быть декодированы, используя процесс декодирования SP для непереключающихся изображений, как описано в п. 8.6.1.

– Иначе (sp_for_switch_flag равен 1), макроблоки P в секции SP должны быть декодированы, используя процесс декодирования SP и SI для переключающихся изображений, как описано в п. 8.6.2.

slice_qs_delta определяет значение QS_Y для всех макроблоков в секциях SP и SI. Параметр квантования QS_Y для секции вычисляют следующим образом:

$$QS_Y = 26 + pic\_init\_qs\_minus26 + slice\_qs\_delta. \quad (7\text{-}17)$$

Значение slice_qs_delta должно быть ограничено таким образом, чтобы QS_Y находилось в диапазоне от 0 до 51 включительно. Значение QS_Y используют для декодирования всех макроблоков в секциях SI с mb_type, равным SI, и для всех макроблоков в секциях SP с режимом внешнего предсказания (inter предсказания).

disable_deblocking_filter_idc определяет, действительно ли операция разблокирования (разделения на блоки) фильтра должна быть аннулирована для некоторых краев блока секции, а также определяет, для каких краев аннулируют фильтрацию. Если disable_deblocking_filter_idc отсутствует в заголовке секции, значение disable_deblocking_filter_idc должно быть принято равным 0.

Значение disable_deblocking_filter_idc должно быть в диапазоне от 0 до 2 включительно.

slice_alpha_c0_offset_div2 определяет сдвиг, использованный для доступа к таблицам разблокирования фильтра α и t_{C0}, при операциях фильтрации, которыми управляют макроблоки в секции. Исходя из этого значения следует использовать сдвиг в тех случаях, когда адресацию к этим таблицам вычисляют как:

FilterOffsetA = slice_alpha_c0_offset_div2 << 1. (7-18)

Значение slice_alpha_c0_offset_div2 должно быть в диапазоне от –6 до +6 включительно. Если slice_alpha_c0_offset_div2 отсутствует в заголовке секции, значение slice_alpha_c0_offset_div2 должно считаться равным 0.

slice_beta_offset_div2 определяет сдвиг, использованный для доступа к таблицам разблокирования фильтра β при операциях фильтрации, которыми управляют макроблоки в секции. Исходя из этого значения, следует использовать сдвиг в тех случаях, когда адресацию таблице β вычисляют как :

FilterOffsetB = slice_beta_offset_div2 << 1. (7-19)

Значение slice_beta_offset_div2 должно быть в диапазоне от –6 до +6 включительно. Если slice_beta_offset_div2 отсутствует в заголовке секции, значение slice_beta_offset_div2 должно считаться равным 0.

slice_group_change_cycle используют для отыскания числа блоков отображения группы секции в группу секций 0, если тип slice_group_map_type равен 3, 4 или 5, как это определено равенством:

MapUnitsInSliceGroup0 = Min(slice_group_change_cycle * SliceGroupChangeRate, PicSizeInMapUnits). (7-20)

Значение slice_group_change_cycle представлено в потоке битов следующим числом битов:

Ceil(Log2(PicSizeInMapUnits ÷ SliceGroupChangeRate + 1)). (7-21)

Значение slice_group_change_cycle должно быть в диапазоне от 0 до Ceil(PicSizeInMapUnits÷SliceGroupChangeRate) включительно.

7.4.3.1 Семантика изменения порядка в списке контрольного изображения

Элементы синтаксиса reordering_of_pic_nums_idc, abs_diff_pic_num_minus1 и long_term_pic_num определяют изменение списка от первоначального контрольного изображения на список контрольного изображения, который следует использовать для декодирования секции.

ref_pic_list_reordering_flag_l0, равный 1, определяет, что элемент синтаксиса reordering_of_pic_nums_idc представлен для описания списка 0 контрольного изображения. ref_pic_list_reordering_flag_l0, равный 0, определяет, что этот элемент синтаксиса отсутствует.

Если ref_pic_list_reordering_flag_l0 равен 1, то число раз, при котором значение reordering_of_pic_nums_idc, не равное 3, следующее за значением ref_pic_list_reordering_flag_l0, не должно превышать num_ref_idx_l0_ active_minus1 + 1.

Если значение RefPicList0[num_ref_idx_l0_active_minus1] в полученном исходном списке контрольных изображений (как описано в п. 8.2.4.2) равно "неконтрольному изображению", то ref_pic_list_reordering_flag_l0 должен быть равен 1, а reordering_of_pic_nums_idc не должно быть равно 3 до тех пор, пока значение RefPicList0[num_ref_idx_l0_ active_minus1] в преобразованном списке, полученное как описано в п. 8.2.4.3, не станет равным "неконтрольному изображению".

ref_pic_list_reordering_flag_l1, равное 1, определяет, что элемент синтаксиса reordering_of_pic_nums_idc представлен для описания списка 1 контрольного изображения. ref_pic_list_reordering_flag_l1, равное 0, определяет, что этот элемент синтаксиса отсутствует.

Если ref_pic_list_reordering_flag_l1 равен 1, то число раз, при котором значение reordering_of_pic_nums_idc, равное 3, следующее за значением ref_pic_list_reordering_flag_l1, не должно превышать num_ref_idx_l1_active_minus1 + 1.

Если декодируют секцию B и значение RefPicList1[num_ref_idx_l1_active_minus1] в полученном исходном списке контрольных изображений (как описано в п. 8.2.4.2) равно "неконтрольному изображению", то ref_pic_list_reordering_flag_l1 должен быть равен 1, а reordering_of_pic_nums_idc не должно быть равно 3 до тех пор, пока значение RefPicList1[num_ref_idx_l1_active_minus1] в списке с измененным порядком, как описано в п. 8.2.4.3, не станет равным "неконтрольному изображению".

reordering_of_pic_nums_idc вместе с abs_diff_pic_num_minus1 или long_term_pic_num определяет, какое из контрольных изображений отображается повторно. Значения reordering_of_pic_nums_idc описаны в таблице 7-4. Первое значение reordering_of_pic_nums_idc, которое следует после ref_pic_list_reordering_flag_l0 или ref_pic_list_reordering_flag_l1, не должно быть равно 3.

Таблица 7-4 – Списки контрольных изображений изменения порядка операций reordering_of_pic_nums_idc

reordering_of_pic_nums_idc	Характеристики изменения порядка
0	abs_diff_pic_num_minus1 представлен и соответствует разности вычитания из номера изображения значения предсказания
1	abs_diff_pic_num_minus1 представлен и соответствует разности сложения значения предсказания с номером изображения
2	long_term_pic_num представлен и определяет номер долгосрочного изображения для контрольного изображения
3	Конец цикла преобразования порядка в списке первоначального контрольного изображения

abs_diff_pic_num_minus1 плюс 1 определяет абсолютную разность между номером изображения, индекс которого изменяют в списке на текущий, и предсказанным значением номера. Значение abs_diff_pic_num_minus1 должно быть в диапазоне от 0 до MaxPicNum – 1. Допустимые значения abs_diff_pic_num_minus1 далее ограничены, как это описано в п. 8.2.4.3.1.

long_term_pic_num определяет долгосрочный номер изображения из тех, индекс которых изменяют в списке на текущий. При декодировании кодированного кадра long_term_pic_num должен быть равен значению LongTermPicNum, присвоенному одному из контрольных кадров или дополнительной паре контрольных полей, помеченных как "использованные для долгосрочного контроля". При декодировании кодированного поля long_term_pic_num должен быть равен значению LongTermPicNum, присвоенному одному из контрольных полей, "использованные для долгосрочного контроля".

7.4.3.2 Семантика таблицы веса предсказания

luma_log2_weight_denom – это логарифм по основанию 2 от знаменателя для всех факторов взвешивания яркости. Значение luma_log2_weight_denom должно быть в диапазоне от 0 до 7 включительно.

chroma_log2_weight_denom – это логарифм по основанию 2 от знаменателя для всех факторов взвешивания цветности. Значение chroma_log2_weight_denom должно быть в диапазоне от 0 до 7 включительно.

luma_weight_l0_flag равное 1, определяет, что присутствуют факторы взвешивания для компонента яркости списка предсказаний 0. Значение luma_weight_l0_flag, равное 0, определяет, что факторы взвешивания отсутствуют.

luma_weight_l0[i **]** – это фактор взвешивания для предсказания значения яркости по списку предсказаний 0 с использованием RefPicList0[i]. Если luma_weight_l0_flag равно 1, значение luma_weight_l0[i] должно быть в диапазоне от –128 до 127 включительно. Если luma_weight_l0_flag равно 0, luma_weight_l0[i] должно быть принято равным $2^{luma\_log2\_weight\_denom}$ для RefPicList0[i].

luma_offset_l0[i **]** – это аддитивный сдвиг для предсказания значения яркости по списку предсказаний 0 с использованием RefPicList0[i]. Значение luma_offset_l0[i] должно быть в диапазоне от –128 до 127 включительно. Если luma_weight_l0_flag равно 0, luma_offset_l0[i] должно быть принято равным 0 для RefPicList0[i].

chroma_weight_l0_flag, равное 1, определяет, что присутствуют факторы взвешивания для предсказания значения цветности по списку предсказаний 0. Значение chroma_weight_l0_flag, равное 0, определяет, что эти факторы взвешивания отсутствуют.

chroma_weight_l0[i **][** j **]** – это фактор взвешивания для предсказания значение цветности по списку предсказаний 0 с использованием RefPicList0[i] при j равным 0 для Cb и j равным 1 для Cr. Если chroma_weight_l0_flag равно 1, значение chroma_weight_l0[i][j] должно быть в диапазоне от –128 до 127 включительно. Если chroma_weight_l0_flag равно 0, chroma_weight_l0[i][j] должно быть принято равным $2^{chroma\_log2\_weight\_denom}$ для RefPicList0[i].

chroma_offset_l0[i **][** j **]** – это аддитивный сдвиг для предсказания значения цветности по списку предсказаний 0 с использованием RefPicList0[i] при j равным 0 для Cb и j равным 1 для Cr. Значение chroma_offset_l0[i][j] должно быть в диапазоне от –128 до 127 включительно. Если chroma_weight_l0_flag равно 0, chroma_offset_l0[i][j] должно быть принято равным 0 для RefPicList0[i].

luma_weight_l1_flag, **luma_weight_l1**, **luma_offset_l1**, **chroma_weight_l1_flag**, **chroma_weight_l1**, **chroma_offset_l1** имеют ту же семантику, что и luma_weight_l0_flag, luma_weight_l0, luma_offset_l0, chroma_weight_l0_flag, chroma_weight_l0, chroma_offset_l0 соответственно с заменой l0, list0, и List0 на l1, list 1, и List1.

7.4.3.3 Семантика разметки декодированного контрольного изображения

Элементы синтаксиса no_output_of_prior_pics_flag, long_term_reference_flag, adaptive_ref_pic_marking_mode_flag, memory_management_control_operation, difference_of_pic_nums_minus1, long_term_frame_idx, long_term_pic_num, и max_long_term_frame_idx_plus1 определяют разметку контрольных изображений.

Разметками контрольного изображения могут быть: "неиспользовано для контроля", "использовано для краткосрочного контроля" или "использовано для долгосрочного контроля", но только одно из трех. Если контрольное изображение полагают помеченным как "использовано для контроля", это в собирательном смысле считают изображением, помеченным как "использовано для краткосрочного контроля" или "использовано для долгосрочного контроля", но не оба вместе.

Элемент синтаксиса adaptive_ref_pic_marking_mode_flag и содержание структуры синтаксиса разметки декодированного контрольного изображения должны быть идентичны для всех кодированных секций кодированного изображения.

Категория синтаксиса разметки декодированного контрольного изображения должна присваиваться следующим образом:

– Если структура синтаксиса разметки декодированного контрольного изображения находится в заголовке секции, категория синтаксиса для структуры синтаксиса разметки декодированного контрольного изображения должна полагаться равной 2.

– Иначе (структура синтаксиса разметки декодированного контрольного изображения находится в повторном сообщении SEI о разметке декодированного контрольного изображения, как это определено в Приложении D), категория синтаксиса для структуры синтаксиса разметки декодированного контрольного изображения должна полагаться равной 5.

no_output_of_prior_pics_flag определяет, как обрабатывают ранее декодированные изображения в буфере декодированного изображения после декодирования IDR изображения. См. Приложение C. Если IDR изображение – это первое IDR изображение в потоке битов, значение no_output_of_prior_pics_flag не оказывает влияния на процесс декодирования. Если IDR изображение – это не первое IDR изображение в потоке битов, значения PicWidthInMbs, FrameHeightInMbs или max_dec_frame_buffering, которые находят из действующей установки параметров последовательности, отличаются от значений PicWidthInMbs, FrameHeightInMbs или max_dec_frame_buffering, которые находят из предыдущей действующей установки параметров последовательности, а no_output_of_prior_pics_flag равное 1, может быть интерпретировано декодером независимо от действующего значения no_output_of_prior_pics_flag.

long_term_reference_flag равное 0, определяет, что переменная MaxLongTermFrameIdx установлена в положение, равное "индексы недолгосрочных кадров", и что IDR изображение помечено как "использовано для краткосрочного контроля". long_term_reference_flag, равное 1, определяет, что переменная MaxLongTermFrameIdx установлена в положение, равное 0, и что текущее IDR изображение помечено "использовано для долгосрочного контроля" и задано значением LongTermFrameIdx, равным 0. Если num_ref_frames равно 0, значение long_term_reference_flag должно быть равно 0.

adaptive_ref_pic_marking_mode_flag выбирает режим разметки контрольного изображения текущего декодированного изображения так, как это описано в таблице 7-5. Значение adaptive_ref_pic_marking_mode_flag должно быть равно 1, если число кадров, дополнительных пар полей и непарных полей, которые в данный момент помечены как "использовано для долгосрочного контроля", равно Max(num_ref_frames, 1).

Таблица 7-5 – Интерпретация значения adaptive_ref_pic_marking_mode_flag

adaptive_ref_pic_marking_mode_flag	Описание режима разметки контрольного изображения
0	Режим разметки контрольного изображения скользящим окном: Режим разметки краткосрочных контрольных изображений, основанный на механизме "первый пришел – первым обслужен" (first-in first-out).
1	Адаптивный режим разметки контрольного изображения: Режим разметки контрольного изображения, создающий элементы синтаксиса для спецификации разметки контрольных изображений как "неиспользованных для контроля" и присваивающий долгосрочные индексы кадров.

memory_management_control_operation определяет, что для разметки контрольного изображения следует использовать операцию управления. Элемент синтаксиса memory_management_control_operation следует за данными, необходимыми для описания операции memory_management_control_operation. Значения и операции управления, связанные с memory_management_control_operation, описаны в таблице 7-6. Элементы синтаксиса memory_management_control_operation обрабатывают процесс декодирования в том порядке, в котором они появляются в заголовке секции, а семантику ограничения, выраженную для каждого значения memory_management_control_operation, используют в особой позиции в том порядке, в котором обрабатывают отдельные значения memory_management_control_operation.

Значение memory_management_control_operation в заголовке секции не должно быть равным 1, если заданное краткосрочное изображение не помечено в данный момент как "использовано для контроля" и этому изображению еще не присвоен долгосрочный индекс кадра, а также долгосрочный индекс кадра не присвоен в той же самой структуре синтаксиса разметки декодированного контрольного изображения.

Значение memory_management_control_operation в заголовке секции не должно быть равным 2, если номер специального долгосрочного изображения не отсылает к кадру или к полю, которые в данный момент помечены как "использовано для контроля" и если значение memory_management_control_operation не обрабатывается процессом декодирования.

Значение memory_management_control_operation в заголовке секции не должно быть равным 3, если заданное краткосрочное контрольное изображение не помечено в данный момент как "использовано для контроля" при обработке значения memory_management_control_operation процессом декодирования, а заданному краткосрочному контрольному изображению еще не был ранее присвоен долгосрочный индекс кадра, а также не был присвоен никакой другой долгосрочный индекс кадра в той же самой структуре синтаксиса разметки декодированного контрольного изображения.

Значение memory_management_control_operation не должно быть равным 3 или 6, если значение переменной MaxLongTermFrameIdx равно значению "нет долгосрочных индексов кадра" и если memory_management_control_operation обрабатывают процессом декодирования.

В заголовке секции должно быть не более одного значения memory_management_control_operation, равного 4.

В заголовке секции значение memory_management_control_operation не должно быть равно 5, если в диапазоне от 1 до 3 отсутствует значение memory_management_control_operation в той же самой структуре синтаксиса разметки декодированного контрольного изображения.

В заголовке секции должно быть не более одного значения memory_management_control_operation, равного 6.

Если имеется значение memory_management_control_operation, равное 6, любые значения memory_management_control_operation, равные 2, 3 или 4, которые следуют за memory_management_control_operation, равным 6 в том же заголовке секции, не должны описывать текущее изображение, которое следует пометить как "не использовано для контроля".

В заголовке секции значение memory_management_control_operation, равное 6, не должно предшествовать memory_management_control_operation, равному 5.

ПРИМЕЧАНИЕ. – Эти ограничения запрещают любые комбинации многочисленных элементов синтаксиса memory_management_control_operation, которые могли бы определить текущее изображение, помеченное как "не использовано для контроля". Однако разрешены некоторые другие комбинации элементов синтаксиса memory_management_control_operation, которые могут влиять на статус разметки других контрольных изображений более одного раза в том же заголовке секции. В частности, это разрешено для значения memory_management_control_operation, равного 3, определяющего индекс долгосрочного кадра, который следует присвоить конкретному краткосрочному контрольному изображению, следующему в том же заголовке секции за memory_management_control_operation, равным 2, 3 или 4, которые определяют это же контрольное изображение, чтобы в дальнейшем пометить его как "не использовано для контроля".

В заголовке секции должно быть представлено не более одного значения memory_management_control_operation, которое определяет одно и то же действие.

Таблица 7-6 – Значения операций управления памятью (memory_management_control_operation)

memory_management_control_operation	Операции управления памятью
0	Конец цикла memory_management_control_operation
1	Разметка краткосрочного изображения как "не использовано для контроля"
2	Разметка кадра или поля с номером долгосрочного изображения как "не использовано для контроля"
3	Присвоение индекса долгосрочного кадра индексу краткосрочного изображения
4	Определяет максимальный индекс долгосрочного кадра
5	Разметка всех контрольных изображений как "не использовано для контроля" и установка переменной MaxLongTermFrameIdx на "индексы недолгосрочных кадров"
6	Присвоение индекса долгосрочного кадра текущему декодированному изображению

Если во время декодирования поля команда memory_management_control_operation, равная 3, присваивает индекс долгосрочного кадра полю, которое является частью краткосрочного контрольного кадра или паре краткосрочных дополнительных полей, то другая команда memory_management_control_operation, которая присваивает тот же индекс долгосрочного кадра другому полю этого же кадра или паре дополнительных полей, должна быть представлена в структуре синтаксиса разметки того же декодированного контрольного изображения.

Если первое поле (в порядке декодирования) пары дополнительных контрольных полей включает значение long_term_reference_flag, равное 1, или команду memory_management_control_operation, равную 6, то структура синтаксиса разметки декодированного контрольного изображения для другого поля пары дополнительных контрольных полей должна содержать команду memory_management_control_operation, равную 6, которая присваивает тот же индекс долгосрочного кадра другому полю.

difference_of_pic_nums_minus1 используют (со значением memory_management_control_operation, равным 3 или 1), чтобы присвоить индекс долгосрочного кадра краткосрочному контрольному изображению или чтобы пометить краткосрочное контрольное изображение как "не использовано для контроля". Если значение memory_management_control_operation обрабатывают процессом декодирования, то номер результирующего изображения, который находят из difference_of_pic_nums_minus1, должен быть номером изображения, присвоенным одному из контрольных изображений, помеченных как "использовано для контроля", а не ранее присвоенному индексу долгосрочного кадра.

Номер результирующего изображения ограничен следующим образом:

– Если field_pic_flag равно 0, номер результирующего изображения должен быть одним из установки номеров изображений, присвоенных контрольным кадрам или парам дополнительных контрольных полей.

  ПРИМЕЧАНИЕ. – Если field_pic_flag равно 0, номер результирующего изображения должен быть номером изображения, присвоенным паре дополнительных контрольных полей, в которых оба поля помечены как "использовано для контроля", или кадра, в котором оба поля помечены как "использовано для контроля". В частности, если field_pic_flag равно 0, на разметку непарных полей или кадра, в котором единственное поле помечено как "использовано для контроля", не может влиять значение memory_management_control_operation, равное 1.

– Иначе (field_pic_flag равно 1), номер результирующего изображения должен быть одним из установки номеров изображений, присвоенных контрольным полям.

long_term_pic_num используют (со значением memory_management_control_operation, равным 2), чтобы пометить долгосрочное контрольное изображение как "не использовано для контроля". Если значение memory_management_control_operation обрабатывают процессом декодирования, long_term_pic_num должно быть равно номеру долгосрочного изображения, присвоенному одному из контрольных изображений, которое в данный момент помечено как "использовано для долгосрочного контроля".

Номер результирующего долгосрочного изображения ограничен следующим образом:

– Если field_pic_flag равно 0, номер результирующего долгосрочного изображения должен быть одним из установки номеров изображений, присвоенных контрольным кадрам или парам дополнительных контрольных полей.

  ПРИМЕЧАНИЕ. – Если field_pic_flag равно 0, номер результирующего долгосрочного изображения должен быть номером изображения, присвоенным паре дополнительных контрольных полей, в которых оба поля помечены как "использовано для контроля", или кадра, в котором оба поля помечены как "использовано для контроля". В частности, если field_pic_flag равно 0, на разметку непарных полей или кадра, в котором единственное поле помечено как "использовано для контроля", не может влиять значение memory_management_control_operation, равное 2.

– Иначе (field_pic_flag равно 1), номер результирующего долгосрочного изображения должен быть одним из установки номеров изображений, присвоенных контрольным полям.

long_term_frame_idx используют (со значением memory_management_control_operation, равным 3 или 6), чтобы присвоить индекс долгосрочного кадра изображению. Если memory_management_control_operation обрабатывают процессом декодирования, значение long_term_frame_idx должно быть в диапазоне от 0 до MaxLongTermFrameIdx включительно.

max_long_term_frame_idx_plus1 минус 1 определяет максимальное значение индекса долгосрочного кадра, разрешенного для долгосрочных контрольных изображений (до получения другого значения max_long_term_frame_idx_plus1). Значение max_long_term_frame_idx_plus1 должно быть в диапазоне от 0 до num_ref_frames включительно.

7.4.4 Семантика данных секции

cabac_alignment_one_bit – это бит, равный 1.

mb_skip_run определяет число последовательных пропущенных макроблоков, для которых при декодировании секции P или SP значение mb_type должно быть обозначено P_Skip, а тип макроблока обозначен как макроблок типа P или для которых при декодировании секции B значение mb_type должно быть обозначено B_Skip, а тип макроблока обозначен как макроблок типа B. Значение mb_skip_run должно быть в диапазоне от 0 до PicSizeInMbs – CurrMbAddr включительно.

mb_skip_flag, равное 1, определяет, что для данного макроблока при декодировании секции P или SP значение mb_type должно быть обозначено P_Skip, а тип макроблока обозначен как макроблок типа P, или при декодировании

секции B значение mb_type должно быть обозначено B_Skip, а тип макроблока обозначен как макроблок типа B. Значение mb_skip_flag, равное 0, указывает, что текущий макроблок не пропущен.

mb_field_decoding_flag, равное 0, определяет, что текущая пара макроблоков – это пара макроблоков кадра. Значение mb_field_decoding_flag, равное 1, определяет, что эта пара макроблоков – пара макроблоков поля. Оба макроблока из пары макроблоков кадра в данном тексте считают макроблоками кадра, несмотря на то что оба макроблока из пары макроблоков поля в данном тексте считают макроблоками поля.

Если значение mb_field_decoding_flag отсутствует в любом из пары макроблоков, значение mb_field_decoding_flag находят следующим образом:

– Если имеется смежная пара макроблоков непосредственно слева от текущей пары в той же секции, значение mb_field_decoding_flag должно быть принято равным значению mb_field_decoding_flag для смежной пары макроблоков непосредственно слева от текущей пары макроблоков.
– Иначе, если в той же секции нет смежной пары макроблоков непосредственно слева от текущей пары, но есть смежная пара макроблоков непосредственно над текущей парой, значение mb_field_decoding_flag должно быть принято равным значению mb_field_decoding_flag для смежной пары макроблоков непосредственно над текущей парой макроблоков.
– Иначе (если в этой же секции нет смежной пары макроблоков непосредственно слева и над текущей парой макроблоков), значение mb_field_decoding_flag должно быть принято равным 0.

end_of_slice_flag, равное 0, указывает, что другой макроблок – это следующий в данной секции. Значение end_of_slice_flag, равное 1, указывает на конец секции и отсутствие последующих макроблоков.

Функция NextMbAddress( ), которую используют в таблице синтаксиса данных секции, описана в п. 8.2.2.

7.4.5 Семантика слоя макроблока

mb_type определяет тип макроблока. Семантика mb_type зависит от типа секции.

Таблицы и семантика описаны для разных типов макроблоков секций I, SI, P, SP и B. Каждая таблица представляет: значение mb_type, наименование mb_type, число используемых разделенных частей макроблоков (которые дает функция NumMbPart(mb_type), режим предсказания макроблока (если он не разделен) или первого разделения (которые дает функция MbPartPredMode(mb_type, 0) и режим предсказания второго разделения (которые дает функция MbPartPredMode(mb_type, 1). Если значение не применимо, его обозначают "na". В данном тексте значение mb_type можно считать типом макроблока, а значение X в MbPartPredMode( ) в данном тексте можно считать "режимом предсказания (разделения) макроблока X" или "X предсказанием макроблоков".

В таблице 7-7 показаны разрешенные типы макроблоков для каждого значения slice_type.

ПРИМЕЧАНИЕ. – Существуют некоторые типы макроблоков с режимом предсказания Pred_L0, которые относят к типам B макроблоков.

Таблица 7-7 – Разрешенные типы макроблоков для значения slice_type

slice_type	Разрешенные типы макроблоков
I (секция)	I (см. таблицу 7-8) (типы макроблоков)
P (секция)	P (см. таблицу 7-10) и I (см. таблицу 7-8) (типы макроблоков)
B (секция)	B (см. таблицу 7-11) и I (см. таблицу 7-8) (типы макроблоков)
SI (секция)	SI (см. таблицу 7-9) и I (см. таблицу 7-8) (типы макроблоков)
SP (секция)	P (см. таблицу 7-10) и I (см. таблицу 7-8) (типы макроблоков)

Типы макроблоков, которые можно отнести к макроблокам типа I, показаны в таблице 7-8.

Типы макроблоков для секций I – это все типы макроблоков I.

Таблица 7-8 – Типы макроблоков для секций I

mb_type	Наименование mb_type	MbPartPredMode (mb_type, 0)	Intra16x16PredMode	CodedBlockPatternChroma	CodedBlockPatternLuma
0	I_4x4	Intra_4x4	na	Равенство 7-22	Равенство 7-22
1	I_16x16_0_0_0	Intra_16x16	0	0	0
2	I_16x16_1_0_0	Intra_16x16	1	0	0
3	I_16x16_2_0_0	Intra_16x16	2	0	0
4	I_16x16_3_0_0	Intra_16x16	3	0	0
5	I_16x16_0_1_0	Intra_16x16	0	1	0
6	I_16x16_1_1_0	Intra_16x16	1	1	0
7	I_16x16_2_1_0	Intra_16x16	2	1	0
8	I_16x16_3_1_0	Intra_16x16	3	1	0
9	I_16x16_0_2_0	Intra_16x16	0	2	0
10	I_16x16_1_2_0	Intra_16x16	1	2	0
11	I_16x16_2_2_0	Intra_16x16	2	2	0
12	I_16x16_3_2_0	Intra_16x16	3	2	0
13	I_16x16_0_0_1	Intra_16x16	0	0	15
14	I_16x16_1_0_1	Intra_16x16	1	0	15
15	I_16x16_2_0_1	Intra_16x16	2	0	15
16	I_16x16_3_0_1	Intra_16x16	3	0	15
17	I_16x16_0_1_1	Intra_16x16	0	1	15
18	I_16x16_1_1_1	Intra_16x16	1	1	15
19	I_16x16_2_1_1	Intra_16x16	2	1	15
20	I_16x16_3_1_1	Intra_16x16	3	1	15
21	I_16x16_0_2_1	Intra_16x16	0	2	15
22	I_16x16_1_2_1	Intra_16x16	1	2	15
23	I_16x16_2_2_1	Intra_16x16	2	2	15
24	I_16x16_3_2_1	Intra_16x16	3	2	15
25	I_PCM	na	na	na	na

Следующая семантика присвоена типам макроблоков в таблице 7-8:

I_4x4: макроблок кодирован как макроблок режима предсказания Intra_4x4.

I_16x16_0_0_0, I_16x16_1_0_0, I_16x16_2_0_0, I_16x16_3_0_0, I_16x16_0_1_0, I_16x16_1_1_0, I_16x16_2_1_0, I_16x16_3_1_0, I_16x16_0_2_0, I_16x16_1_2_0, I_16x16_2_2_0, I_16x16_3_2_0, I_16x16_0_0_1, I_16x16_1_0_1, I_16x16_2_0_1, I_16x16_3_0_1, I_16x16_0_1_1, I_16x16_1_1_1, I_16x16_2_1_1, I_16x16_3_1_1, I_16x16_0_2_1, I_16x16_1_2_1, I_16x16_2_2_1, I_16x16_3_2_1: макроблок кодирован как макроблок режима предсказания Intra_16x16.

Каждому макроблоку режима предсказания Intra_16x16 присваивают значение Intra16x16PredMode, которое определяет режим предсказания Intra_16x16. CodedBlockPatternChroma содержит значение образца кодированного блока цветности, как описано в таблице 7-12. CodedBlockPatternLuma определяет, присутствуют ли ненулевые уровни AC коэффициентов преобразования для компонента яркости. CodedBlockPatternLuma, равное 0, указывает, что все уровни AC коэффициентов преобразования в компоненте яркости макроблока равны 0. CodedBlockPatternLuma, равное 15, указывает, что по крайней мере один уровень AC коэффициентов преобразования в компоненте яркости макроблока – ненулевой, и требуется сканирование уровней AC коэффициентов преобразования для всех 16 блоков 4x4 в блоке 16x16.

Intra_4x4 определяет режим предсказания макроблока, а также указывает, что запущен процесс предсказания Intra_4x4, как это описано в п. 8.3.1. Intra_4x4 – это режим Intra предсказания макроблока.

Intra_16x16 определяет режим предсказания макроблока, а также указывает, что запущен процесс предсказания Intra_16x16, как это описано в п. 8.3.2. Intra_16x16 – это режим Intra предсказания макроблока.

Для макроблока, кодированного значением mb_type, равным I_PCM, должен быть принят режим Intra предсказания макроблока.

Тип A – это тип макроблока, который можно отнести к типу макроблока SI, описанному в таблице 7-9.

Тип макроблоков секций SI описан в таблице 7-9 и таблице 7-8. Нулевое значение mb_type описано в таблице 7-9, а значения mb_type от 1 до 26 описаны в таблице 7-8, индексированной вычитанием 1 из значения mb_type.

Таблица 7-9 – Нулевые значения типа макроблока для секций SI

mb_type	Наименование mb_type	MbPartPredMode (mb_type, 0)	Intra16x16PredMode	CodedBlockPatternChroma	CodedBlockPatternLuma
0	SI	Intra_4x4	na	Равенство 7-22	Равенство 7-22

Следующая семантика присвоена типу макроблока в таблице 7-9. Макроблок SI кодирован как макроблок предсказания Intra_4x4.

Типы макроблоков, которые можно отнести к типу P, показаны в таблице 7-10.

Типы макроблоков для секций P и SP показаны в таблице 7-10 и таблице 7-8. Значения mb_type от 0 до 4 показаны в таблице 7-10, значения mb_type от 5 до 30 показаны в таблице 7-8, индексированной вычитанием 5 из значения mb_type.

Таблица 7-10 – Значения типа макроблока от 0 до 4 для секций P и SP

mb_type	Наименование mb_type	NumMbPart (mb_type)	MbPartPredMode (mb_type, 0)	MbPartPredMode (mb_type, 1)	MbPartWidth (mb_type)	MbPartHeight (mb_type)
0	P_L0_16x16	1	Pred_L0	na	16	16
1	P_L0_L0_16x8	2	Pred_L0	Pred_L0	16	8
2	P_L0_L0_8x16	2	Pred_L0	Pred_L0	8	16
3	P_8x8	4	na	na	8	8
4	P_8x8ref0	4	na	na	8	8
полученные	P_Skip	1	Pred_L0	na	16	16

Следующая семантика присвоена типам макроблоков в таблице 7-10.

– P_L0_16x16: образцы макроблоков предсказаны для одного разделенного макроблока образцов яркости размером 16x16, объединенных с образцами цветности.

– P_L0_L0_MxN, с заменой MxN на 16x8 или 8x16: образцы макроблоков предсказаны с использовнием двух разделенных частей яркости размером MxN, равным 16x8, или двух разделенных частей яркости размером MxN, равным 8x16, и соответственно объединенных с образцами цветности.

– P_8x8: представлен в потоке битов для каждого субмакроблока дополнительного элемента синтаксиса (sub_mb_type), который определяет тип соответствующего субмакроблока (см. п. 7.4.5.2).

– P_8x8ref0: представлен в потоке битов. Имеет ту же семантику, что и P_8x8, но без элементов синтаксиса для контроля индекса (ref_idx_l0), а ref_idx_l0[mbPartIdx] должен считаться равным 0 для всех субмакроблоков этого макроблока (с индексами mbPartIdx, равными 0..3).

– P_Skip: в потоке битов отсутствуют дополнительные данные для этого макроблока.

Следующая семантика присвоена режимам предсказания макроблоков (MbPartPredMode( )) из таблицы 7-10.

– Pred_L0: определяет, что запущен процесс Inter предсказания (внешнего предсказания) с использованием списка предсказаний 0. Pred_L0 – это режим предсказания макроблока Inter.

Типы макроблоков, которые в общем можно отнести к типу макроблоков B, описаны в таблице 7-11.

Типы макроблоков для секций B описаны в таблице 7-11 и таблице 7-8. Значения mb_type от 0 до 22 описаны в таблице 7-11, а значения mb_type от 23 до 48 описаны в таблице 7-8, проиндексированной вычитанием 23 из значения mb_type.

Таблица 7-11 – Значения от 0 до 22 типов макроблоков для секций B

mb_type	Наименование mb_type	NumMbPart (mb_type)	MbPartPredMode (mb_type, 0)	MbPartPredMode (mb_type, 1)	MbPartWidth (mb_type)	MbPartHeight (mb_type)
0	B_Direct_16x16	na	Direct	na	8	8
1	B_L0_16x16	1	Pred_L0	na	16	16
2	B_L1_16x16	1	Pred_L1	na	16	16
3	B_Bi_16x16	1	BiPred	na	16	16
4	B_L0_L0_16x8	2	Pred_L0	Pred_L0	16	8
5	B_L0_L0_8x16	2	Pred_L0	Pred_L0	8	16
6	B_L1_L1_16x8	2	Pred_L1	Pred_L1	16	8
7	B_L1_L1_8x16	2	Pred_L1	Pred_L1	8	16
8	B_L0_L1_16x8	2	Pred_L0	Pred_L1	16	8
9	B_L0_L1_8x16	2	Pred_L0	Pred_L1	8	16
10	B_L1_L0_16x8	2	Pred_L1	Pred_L0	16	8
11	B_L1_L0_8x16	2	Pred_L1	Pred_L0	8	16
12	B_L0_Bi_16x8	2	Pred_L0	BiPred	16	8
13	B_L0_Bi_8x16	2	Pred_L0	BiPred	8	16
14	B_L1_Bi_16x8	2	Pred_L1	BiPred	16	8
15	B_L1_Bi_8x16	2	Pred_L1	BiPred	8	16
16	B_Bi_L0_16x8	2	BiPred	Pred_L0	16	8
17	B_Bi_L0_8x16	2	BiPred	Pred_L0	8	16
18	B_Bi_L1_16x8	2	BiPred	Pred_L1	16	8
19	B_Bi_L1_8x16	2	BiPred	Pred_L1	8	16
20	B_Bi_Bi_16x8	2	BiPred	BiPred	16	8
21	B_Bi_Bi_8x16	2	BiPred	BiPred	8	16
22	B_8x8	4	na	na	8	8
полученные	B_Skip	na	Direct	na	8	8

Типам макроблоков в таблице 7-11 присваивают следующую семантику:

– B_Direct_16x16: в потоке битов отсутствуют разности векторов движения или индексов контроля для макроблока. Функции MbPartWidth(B_Direct_16x16) и MbPartHeight(B_Direct_16x16) используют в процессе отыскания векторов движения и индексов контрольных кадров (по п. 8.4.1) для режима прямого предсказания.

– B_X_16x16 с заменой X на L0, L1 или Bi: образцы макроблока предсказаны с одним разделенным макроблоком яркости с размером образцов яркости 16x16 и объединеным с образцами цветности. Для макроблока типа B_X_16x16 с заменой X либо на L0, либо на L1 в потоке битов макроблока присутствуют одна разность векторов

движения и один индекс контроля. Для макроблока типа B_X_16x16 с заменой X на Bi в потоке битов макроблока присутствуют две разности векторов движения и два индекса контроля.

– B_X0_X1_MxN с заменой X0 и X1, которые считают первыми и вторыми разделенными частями макроблока, на L0, L1или Bi, а также с заменой MxN на 16x8 или 8x16: образцы макроблока предсказаны с использованием двух разделенных частей яркости размером MxN, равным 16x8, или двух разделенных частей яркости размером MxN, равным 8x16, и объединенных с образцами цветности соответственно. Для разделенных частей макроблока X0 или X1 с заменой X0 или X1 на L0 или L1 в потоке битов представлены одна разность векторов движения и один индекс контроля. Для разделенных частей макроблока X0 или X1 с заменой X0 или X1 на Bi в потоке битов представлены две разности векторов движения и два индекса контроля.

– B_8x8: в потоке битов для каждого субмакроблока представлен дополнительный элемент синтаксиса (sub_mb_type), который определяет тип соответствующего субмакроблока (см. п. 7.4.5.2).

– B_Skip: в потоке битов отсутствуют данные для этого макроблока. Функции MbPartWidth(B_Skip) и MbPartHeight(B_Skip) используют в процессе отыскания векторов движения и индексов контрольных кадров (по п. 8.4.1) для режима прямого предсказания.

Следующие элементы семантики присваивают режимам предсказания макроблоков (MbPartPredMode( )) из таблицы 7-11:

– Direct: в потоке битов отсутствуют разности векторов движения или индексов контроля для макроблока (случай B_Skip или B_Direct_16x16). Direct – это режим Inter предсказания макроблока.

– Pred_L0: см. семантику для таблицы 7-10.

– Pred_L1: определяет, что запущен процесс Inter предсказания с использованием списка предсказаний 1. Pred_L1 – это режим Inter предсказания макроблока.

– BiPred: определяет, что запущен процесс Inter предсказания с использованием списков предсказаний 0 и 1. BiPred – это режим Inter предсказания макроблока.

pcm_alignment_zero_bit – бит, равный 0.

pcm_byte[i] – значение образца. pcm_byte[i] не должно быть равным 0. Первые 256 значений pcm_byte[i] представляют значения образцов яркости при растровом сканировании макроблока. Следующие значения (256 * (ChromaFormatFactor – 1)) / 2 pcm_byte[i] представляют значения образца Cb при растровом сканировании макроблока. Последние значения (256 * (ChromaFormatFactor – 1)) / 2 pcm_byte[i] представляют значения образца Cr при растровом сканировании макроблока.

coded_блок_pattern определяет, какой из шести блоков яркости 8x8 и цветности может содержать ненулевые уровни коэффициентов преобразования. Для макроблоков с режимом предсказания, не равным Intra_16x16, образец coded_block_pattern представлен в потоке битов, а переменные CodedBlockPatternLuma и CodedBlockPatternChroma находят следующим образом:

CodedBlockPatternLuma = coded_block_pattern % 16
CodedBlockPatternChroma = coded_block_pattern / 16. (7-22)

Если coded_block_pattern представлен, переменная CodedBlockPatternLuma выбирает для каждого из четырех блоков яркости 8x8 макроблока один из следующих случаев:

– Все уровни коэффициентов преобразования четырех блоков яркости 4x4 в блоке яркости 8x8 равны нулю.

– Один или более уровней коэффициентов преобразования одного из четырех блоков яркости 4x4 в блоке яркости 8x8 должны иметь ненулевые значения.

Значения CodedBlockPatternChroma показаны в таблице 7-12.

Таблица 7-12 – Спецификация значений CodedBlockPatternChroma

CodedBlockPatternChroma	Описание
0	Все уровни коэффициентов преобразования цветности равны 0.
1	Один или более уровней коэффициентов преобразования цветности DC должны иметь ненулевые значения. Все уровни коэффициентов преобразования цветности AC равны 0.
2	Нуль или более уровней коэффициентов преобразования цветности DC имеют ненулевые значения. Один или более уровней коэффициентов преобразования цветности AC должны иметь ненулевые значения.

mb_qp_delta может менять значение QP_Y в слое макроблока. Декодированное значение mb_qp_delta должно быть в диапазоне от –26 до +25 включительно. Значение mb_qp_delta должно быть принято равным 0, если оно не представлено ни для одного макроблока (включая типы макроблоков P_Skip и B_Skip).

Значение QP_Y находят как

$$QP_Y = (QP_{Y,PREV} + mb\_qp\_delta + 52) \% 52, \quad (7\text{-}23)$$

где $QP_{Y,PREV}$ – параметр квантования яркости, а QP_Y взято из предыдущего макроблока в порядке декодирования в текущей секции. Для первого макроблока в секции $QP_{Y,PREV}$ представляет первоначальную установку, равную $SliceQP_Y$, найденную из равенства 7-16 в начале каждой секции.

7.4.5.1 Семантика предсказания макроблока

Все образцы макроблока предсказаны. Режимы предсказания находят, используя следующие элементы синтаксиса.

prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]** и **rem_intra4x4_pred_mode[** luma4x4BlkIdx **]** определяют предсказание Intra_4x4 блока яркости 4x4 с индексом luma4x4BlkIdx = 0..15.

intra_chroma_pred_mode определяет тип пространственного предсказания, использованного для цветности в макроблоках с предсказанием Intra_4x4 или Intra_16x16, как показано в таблице 7-13.

Таблица 7-13 – Взаимосвязь между режимами пространственных предсказаний intra_chroma_pred_mode

intra_chroma_pred_mode	Режим Intra предсказания цветности
0	DC
1	Горизонтальное
2	Вертикальное
3	Плоское

ref_idx_l0[mbPartIdx **]**, если имеется, определяет индекс в списке 0 контрольного изображения, которое используют для предсказания.

Диапазон ref_idx_l0[mbPartIdx], индекс в списке 0 контрольного изображения и, если применимо, четность поля в контрольном изображении, использованного для предсказания, описаны следующим образом:

– Если MbaffFrameFlag равно 0 или mb_field_decoding_flag равно 0, то значение ref_idx_l0[mbPartIdx] должно быть в диапазоне от 0 до num_ref_idx_l0_active_minus1 включительно.

– Иначе (MbaffFrameFlag равно 1 и mb_field_decoding_flag равно 1), значение ref_idx_l0[mbPartIdx] должно быть в диапазоне от 0 до 2 * num_ref_idx_l0_active_minus1 + 1 включительно.

Если для внешнего предсказания использовано только одно контрольное изображение, то значение ref_idx_l0[mbPartIdx] должно быть принято равным 0.

ref_idx_l1[mbPartIdx **]** имеет ту же семантику, что и ref_idx_l0 с заменой l0 и списка 0 на l1 и список 1 соответственно.

mvd_l0[mbPartIdx **][** 0 **][** compIdx **]** определяет разность между компонентом вектора, который должен быть использован, и его предсказанием. Индекс mbPartIdx определяет, какой разделенной части макроблока присвоено значение mvd_l0. Разделение макроблока описано значением mb_type. Разность компонентов горизонтального вектора движения декодируют первой в порядке декодирования и присваивают значение CompIdx = 0. Компонент вертикального вектора движения декодируют вторым в порядке декодирования и присваивают значение CompIdx = 1. Диапазон компонентов mvd_l0[mbPartIdx][0][compIdx] определен ограничениями на вектор движения переменным значением, полученным из этого диапазона, как это указано в Приложении A.

mvd_l1[mbPartIdx **][** 0 **][** compIdx **]** имеет ту же семантику, что и mvd_l0 с заменой l0 и L0 на l1 и L1 соответственно.

7.4.5.2 Семантика предсказания субмакроблока

sub_mb_type[mbPartIdx **]** определяет подтип макроблоков.

Таблицы и семантика описаны для различных подтипов макроблоков секций P, SP и B. Каждая таблица представляет значение sub_mb_type, наименование sub_mb_type, число используемых разделенных частей субмакроблока (заданных функцией NumSubMbPart(sub_mb_type) и режимом предсказания субмакроблока (заданным функцией

SubMbPredMode(sub_mb_type). В данном тексте значение sub_mb_type можно рассматривать как "подтип макроблока", а значение SubMbPredMode( ) – как "субрежим предсказания макроблока".

Подтип макроблоков для макроблоков типа P показан в таблице 7-14.

Таблица 7-14 – Подтип макроблоков для макроблоков типа P

sub_mb_type[mbPartIdx]	наименование sub_mb_type[mbPartIdx]	NumSubMbPart (sub_mb_type[mbPartIdx])	SubMbPredMode (sub_mb_type[mbPartIdx])	SubMbPartWidth (sub_mb_type[mbPartIdx])	SubMbPartHeight (sub_mb_type[mbPartIdx])
0	P_L0_8x8	1	Pred_L0	8	8
1	P_L0_8x4	2	Pred_L0	8	4
2	P_L0_4x8	2	Pred_L0	4	8
3	P_L0_4x4	4	Pred_L0	4	4

Следующую семантику присваивают подтипам макроблоков в таблице 7-14.

– P_L0_MxN, с заменой MxN на 8x8, 8x4, 4x8 и 4x4: образцы субмакроблока, предсказанные с использованием одной разделенной части яркости размером MxN, равным 8x8, двух разделенных частей яркости размером MxN, равным 8x4, или двух разделенных частей яркости размером MxN, равным 4x8, или четырех разделенных частей яркости размером MxN, равным 4x4, и соответственно объединенных с образцами цветности.

Следующую семантику присваивают субрежимам предсказания макроблоков (SubMbPredMode( )) в таблице 7-14.

– Pred_L0: см. семантику для таблицы 7-10.

Подтипы макроблоков для макроблоков типа B приведены в таблице 7-15.

Таблица 7-15 – Подтипы макроблоков для макроблоков типа B

sub_mb_type[mbPartIdx]	наименование sub_mb_type[mbPartIdx]	NumSubMbPart (sub_mb_type[mbPartIdx])	SubMbPredMode (sub_mb_type[mbPartIdx])	SubMbPartWidth (sub_mb_type[mbPartIdx])	SubMbPartHeight (sub_mb_type[mbPartIdx])
na	B_Skip	na	Direct	4	4
na	B_Direct_16x16	na	Direct	4	4
0	B_Direct_8x8	na	Direct	4	4
1	B_L0_8x8	1	Pred_L0	8	8
2	B_L1_8x8	1	Pred_L1	8	8
3	B_Bi_8x8	1	BiPred	8	8
4	B_L0_8x4	2	Pred_L0	8	4
5	B_L0_4x8	2	Pred_L0	4	8
6	B_L1_8x4	2	Pred_L1	8	4
7	B_L1_4x8	2	Pred_L1	4	8
8	B_Bi_8x4	2	BiPred	8	4
9	B_Bi_4x8	2	BiPred	4	8
10	B_L0_4x4	4	Pred_L0	4	4
11	B_L1_4x4	4	Pred_L1	4	4
12	B_Bi_4x4	4	BiPred	4	4

Следующую семантику присваивают типам макроблоков по таблице 7-15:

– B_Skip и B_Direct_16x16: в потоке битов отсутствуют разности векторов движения или индексов контроля для субмакроблока. Функции SubMbPartWidth( ) и SubMbPartHeight( ) используют в процессе отыскания векторов движения и индексов контрольных кадров (по п. 8.4.1) для режима прямого предсказания.

– B_Direct_8x8: в потоке битов отсутствуют разности векторов движения или индексов контроля для субмакроблока. Функции SubMbPartWidth(B_Direct_8x8) и SubMbPartHeight(B_Direct_8x8) используют в процессе отыскания векторов движения и индексов контрольных кадров (по п. 8.4.1) для режима прямого предсказания.

– B_X_MxN, с заменой X на L0, L1 и Bi, а MxN на 8x8, 8x4, 4x8 или 4x4: образцы субмакроблока предсказаны с использованием одной разделенной части яркости размером MxN, равным 8x8, или образцы субмакроблока предсказаны с использованием двух разделенных частей яркости размером MxN, равным 8x4, или образцы субмакроблока предсказаны с использованием двух разделенных частей яркости размером MxN, равным 4x8, или образцы субмакроблока предсказаны с использованием четырех разделенных частей яркости размером MxN, равным 4x4, и соответственно объединенных с образцами цветности. Все части субмакроблока имеют один и тот же индекс контроля. Для разделенной части субмакроблока MxN в субмакроблоке со значением sub_mb_type типа B_X_MxN с заменой X на L0 или L1 в потоке битов существует одна разность векторов движения. Для разделенной части субмакроблока MxN в субмакроблоке со значением sub_mb_type типа B_Bi_MxN в потоке битов существует две разности векторов движения.

Следующую семантику присваивают субрежимам предсказания макроблоков (SubMbPredMode( )) в таблице 7-15.

– Direct: см. семантику для таблицы 7-11.

– Pred_L0: см. семантику для таблицы 7-10.

– Pred_L1: см. семантику для таблицы 7-11.

– BiPred: см. семантику для таблицы 7-11.

ref_idx_l0[mbPartIdx **]** имеет ту же семантику, что и ref_idx_l0 в п. 7.4.5.1.

ref_idx_l1[mbPartIdx **]** имеет ту же семантику, что и ref_idx_l1 в п. 7.4.5.1.

mvd_l0[mbPartIdx **][** subMbPartIdx **][** compIdx **]** имеет ту же семантику, что и mvd_l0 в п. 7.4.5.1, за исключением того, что это применяют к индексам разделения субмакроблока со значением subMbPartIdx. Индексы mbPartIdx и subMbPartIdx определяют, каким разделенным частям макроблока и субмакроблока присвоено значение mvd_l0.

mvd_l1[mbPartIdx **][** subMbPartIdx **][** compIdx **]** имеет ту же семантику, что и mvd_l1 в п. 7.4.5.1.

7.4.5.3 Семантика данных остатка

Структуру синтаксиса residual_block ( ), которую используют для анализа уровней коэффициентов преобразования, присваивают следующим образом:

– Если значение entropy_coding_mode_flag равно 0, то residual_block устанавливают равным значению residual_ block_cavlc, которое используют при анализе элементов синтаксиса для отыскания уровней коэффициентов преобразования.

– Иначе (entropy_coding_mode_flag равно 1), residual_block устанавливают равным значению residual_block_cabac, которое используют при анализе элементов синтаксиса для отыскания уровней коэффициентов преобразования.

В зависимости от mb_type, яркости или цветности структуру синтаксиса residual_block (coeffLevel, maxNumCoeff) используют с аргументом coeffLevel, который представляет список, содержащий уровни коэффициентов преобразования maxNumCoeff, которые анализируют в блоке residual_block( ) и в maxNumCoeff следующим образом:

– В зависимости от MbPartPredMode(mb_type, 0) применяют следующее:

  – Если MbPartPredMode(mb_type, 0) равно Intra_16x16, то уровни коэффициентов преобразования анализируют в списке Intra16x16DCLevel из 16 списков Intra16x16ACLevel[i]. Intra16x16DCLevel содержит 16 уровней коэффициентов преобразования DC с уровнями для каждого блока яркости 4x4. Для каждого из 16 блоков яркости 4x4, индексированных с i = 0..15, 15 уровней коэффициентов преобразования AC i-ого блока анализируют в i-ом списке Intra16x16ACLevel[i].

  – Иначе (MbPartPredMode(mb_type, 0) не равно Intra_16x16), каждый из 16 блоков яркости 4x4, индексированных с i = 0..15, 16 уровней коэффициентов преобразования i-ого блока анализируют в i-ом списке LumaLevel[i].

– Для каждого компонента цветности, индексированного с iCbCr = 0..1, 4 уровня коэффициентов преобразования DC блоков цветности 4x4 анализируют в iCbCr-ом списке ChromaDCLevel[iCbCr].

– Для каждого из блоков цветности 4x4, индексированных с i4x4 = 0..3, каждого компонента цветности, индексированного с iCbCr = 0..1, 15 уровней коэффициентов преобразования AC анализируют в i4x4-ом списке iCbCr-ого компонента цветности ChromaACLevel[iCbCr][i4x4].

7.4.5.3.1 Семантика остатка блока CAVLC

Функция TotalCoeff(coeff_token), которую используют в п. 7.3.5.3.1, возвращает число ненулевых уровней коэффициентов преобразования, полученных из coeff_token.

Функция TrailingOnes(coeff_token), которую используют в п. 7.3.5.3.1, возвращает концевые единицы, полученные из coeff_token.

coeff_token определяет общее число ненулевых уровней коэффициентов преобразования и число концевых единиц уровней коэффициентов преобразования при сканировании уровней коэффициентов преобразования. Уровень коэффициентов преобразования концевой единицы – это один из трех последовательных ненулевых уровней коэффициентов преобразования, которые имеют абсолютное значение, равное 1, в конце сканированных ненулевых уровней коэффициентов преобразования. Диапазон coeff_token описан в п. 9.2.1.

trailing_ones_sign_flag определяет знак уровня коэффициента преобразования концевой единицы следующим образом:

– Если trailing_ones_sign_flag равно 0, то соответствующий уровень коэффициентов преобразования будет декодирован как +1.

– Иначе (trailing_ones_sign_flag равно 1), соответствующий уровень коэффициентов преобразования будет декодирован как –1.

level_prefix и **level_suffix** определяет значение ненулевого уровня коэффициентов преобразования. Диапазон level_prefix и level_suffix описан в п. 9.2.2.

total_zeros определяет общее число уровней коэффициентов преобразования с нулевыми значениями, которые расположены перед позицией последнего ненулевого уровня коэффициентов преобразования при сканировании уровней коэффициентов преобразования. Диапазон total_zeros описан в п. 9.2.3.

run_before определяет число последовательных уровней коэффициентов преобразования при сканировании с нулевыми значениями перед уровнем коэффициентов преобразования с ненулевыми значениями. Диапазон run_before описан в п. 9.2.3.

coeffLevel содержит maxNumCoeff уровней коэффициентов преобразования для текущего списка уровней коэффициентов преобразования.

7.4.5.3.2 Семантика остатка блока CABAC

coded_block_flag определяет, содержит ли блок ненулевые уровни коэффициентов преобразования, следующим образом:

– Если coded_block_flag равно 0, блок не содержит ненулевые уровни коэффициентов преобразования.

– Иначе (coded_block_flag равно 1), блок содержит по крайней мере один ненулевой уровень коэффициентов преобразования.

significant_coeff_flag[i **]** определяет, является ли ненулевым уровень коэффициентов преобразования на позиции сканирования i, следующим образом:

– Если significant_coeff_flag[i] равно 0, уровень коэффициентов преобразования на позиции сканирования i установлен равным 0;

– Иначе (significant_coeff_flag[i] равно 1), уровень коэффициентов преобразования на позиции сканирования i имеет ненулевое значение.

last_significant_coeff_flag[i **]** определяет для позиции сканирования i, являются ли ненулевыми уровни коэффициентов преобразования для последующих позиций сканирования от i + 1 до maxNumCoeff – 1, следующим образом:

– Если last_significant_coeff_flag[i] равно 1, все уровни, следующие за уровнями коэффициентов преобразования (в порядке сканирования) блока, имеют значения, равные 0.

– Иначе (last_significant_coeff_flag[i] равно 0), на пути сканирования имеются дополнительные ненулевые уровни коэффициентов преобразования.

coeff_abs_level_minus1[i **]** – это абсолютное значение уровня коэффициентов преобразования минус 1. Значение coeff_abs_level_minus1 ограничено пределами, указанными в п. 8.5.

coeff_sign_flag[i **]** определяет знак уровня коэффициентов преобразования следующим образом:

– Если coeff_sign_flag равно 0, соответствующий уровень коэффициентов преобразования имеет положительное значение.

– Иначе (coeff_sign_flag равно 1), соответствующий уровень коэффициентов преобразования имеет отрицательное значение.

coeffLevel содержит maxNumCoeff уровней коэффициентов преобразования для текущего списка уровней коэффициентов преобразования.

8 Процесс декодирования

Выходами этого процесса являются декодированные образцы текущего изображения (иногда ссылаются на переменную CurrPic).

Этот раздел описывает процесс декодирования, заданные элементы синтаксиса и обозначенные заглавными буквами переменные из раздела 7.

Процесс декодирования определен таким образом, что все декодеры должны обеспечить идентичные результаты. Любой процесс декодирования, который обеспечивает идентичные результаты для описываемого здесь процесса, соответствует требованиям к процессу декодирования данной Рекомендации | Международного стандарта.

Каждое изображение в этом разделе считают исходным. Каждую секцию в этом разделе считают секцией исходного изображения. Каждую секцию разделенных данных в этом разделе считают секцией разделенных данных исходного изображения.

Обзор процесса декодирования задан следующим образом:

– Декодирование блоков NAL определено в п. 8.1.

– Процессы в п. 8.2 определяют процессы декодирования, используя элементы синтаксиса в этом слое секции и выше.
  – Переменные и функции, связанные с порядком счета изображений находят в п. 8.2.1. (необходимые только для активизации одной секции изображения).
  – Переменные и функции, связанные с макроблоком при отображении группы секции, находят в п. 8.2.2. (необходимые только для активизации одной секции изображения).

- Метод объединения различных разделенных частей в тех случаях, когда секцию разделенных данных используют, как описано в п. 8.2.3.
- Перед декодированием каждой секции для внешнего предсказания необходимо отыскать списки контрольных изображений, как описано в п. 8.2.4.
- Если текущее изображение является контрольным, то после того, как все секции текущего изображения декодированы, процесс разметки декодированного контрольного изображения (по п. 8.2.5) определит, как использовано в процессе декодирования текущее изображение для внешнего предсказания последующих декодированных изображений.

– Процессы в пп. 8.3, 8.4, 8.5, 8.6 и 8.7 определяют процессы декодирования, используя элементы синтаксиса в слое этого макроблока и выше.
  - Процесс внутреннего предсказания для макроблоков I и SI, за исключением макроблоков I_PCM, как определено в п. 8.3, содержит на выходе образцы intra предсказания (внутреннего предсказания). Для макроблоков I_PCM в п. 8.3 непосредственно определен процесс конструирования изображения. Выходом являются образцы, созданные до процесса фильтрового разделения блоков.
  - Процесс inter предсказания (внешнего предсказания) для макроблоков P и B определен в п. 8.4 с образцами внешнего предсказания в качестве выхода.
  - Процесс декодирования коэффициентов преобразования и процесс конструирования изображения до процесса фильтрового разделения блоков (разблокирования) определены в п. 8.5. С помощью этих процессов находят образцы для макроблоков I и B и для макроблоков P и секций P. Выходом являются образцы, созданные до процесса фильтрового разделения блоков.
  - Процесс декодирования для макроблоков P в секциях SP или для макроблоков SI определен в п. 8.6. С помощью этих процессов находят образцы для макроблоков P в секциях SP и для макроблоков SI. Выходом являются образцы, созданные до процесса фильтрового разделения блоков.
  - Образцы, созданные до процесса фильтрового разделения блоков, который является следующим для краев блоков и макроблоков, обрабатывают фильтром разделения блоков, как определено в п. 8.7, с декодированными образцами на выходе.

8.1 Процесс декодирования блока NAL

Входами в этот процесс являются блоки NAL.

Выходами этого процесса являются структуры синтаксиса RBSP, инкапсулированные в блоки NAL.

Процесс декодирования каждого блока NAL извлекает структуру синтаксиса RBSP из блока NAL и затем выполняет процесс декодирования, определенный для структуры синтаксиса RBSP в блоке NAL следующим образом.

Пункт 8.2 описывает процесс декодирования для блоков NAL с nal_unit_type, равным от 1 до 5.

Пункт 8.3 описывает процесс декодирования для макроблока или части макроблока, кодированных в блоках NAL с nal_unit_type, равным 1, 2 и 5.

Пункт 8.4 описывает процесс декодирования для макроблока или части макроблока, кодированных в блоках NAL с nal_unit_type, равным 1 и 2.

Пункт 8.5 описывает процесс декодирования для макроблока или части макроблока, кодированных в блоках NAL с nal_unit_type, равным 1 и от 3 до 5.

Пункт 8.6 описывает процесс декодирования для макроблока или части макроблока, кодированных в блоках NAL с nal_unit_type, равным 1 и от 3 до 5.

Пункт 8.7 описывает процесс декодирования для макроблока или части макроблока, кодированных в блоках NAL с nal_unit_type, равным от 1 до 5.

Блоки NAL с nal_unit_type, равным 7 и 8, содержат установки параметров последовательности и установки параметров изображения соответственно. Установки параметров изображения используют в процессах декодирования других блоков NAL, как это определено контролем установки параметров изображения в заголовках секций каждого изображения. Установки параметров последовательности используют в процессах декодирования других блоков NAL, как это определено контролем установки параметров последовательности в установках параметров изображения каждой последовательности.

Не определен нормативный процесс декодирования для блоков NAL с nal_unit_type, равным 6, 9, 10, 11 и 12.

8.2 Процесс декодирования секции

8.2.1 Процесс декодирования для вычисления порядка изображения

Выходами этого процесса являются TopFieldOrderCnt (если применяют) и BottomFieldOrderCnt (если применяют).

Вычисления порядков изображений используют, чтобы определить для контрольных изображений порядки первоначальных изображений при декодировании B секций (см. пп. 8.2.4.2.3 и 8.2.4.2.4), представить разности порядков изображений между кадрами и полями для отыскания вектора движения во временном прямом режиме (см. п. 8.4.1.2.3), для предполагаемого режима взвешивания предсказания в B секциях (см. п. 8.4.2.3.2) и для контроля согласованности декодера (см. п. C.4).

Информацию о вычислении порядка изображения находят для каждого кадра, поля (либо декодированного из кодированного поля, либо как часть декодированного кадра) или дополнительной пары полей следующим образом:

– Каждый кодированный кадр связан с двумя вычислениями порядков изображения, которые называют TopFieldOrderCnt и BottomFieldOrderCnt для верхнего и нижнего поля соответственно.

– Каждое кодированное поле связано с вычислением порядков изображения, которые называют TopFieldOrderCnt для верхнего кодированного поля и BottomFieldOrderCnt для нижнего поля.

– Каждая дополнительная пара полей связана с двумя вычислениями порядков изображения, которые называют TopFieldOrderCnt для верхнего кодированного поля и BottomFieldOrderCnt для нижнего поля соответственно.

TopFieldOrderCnt и BottomFieldOrderCnt указывают на порядок изображения соответствующего верхнего или нижнего поля относительно первого поля на выходе от предыдущего IDR изображения или от предыдущего контрольного изображения, включая значение memory_management_control_operation, равное 5, в порядке декодирования.

TopFieldOrderCnt и BottomFieldOrderCnt находят, активируя процессы декодирования для вычисления типа порядка изображения 0, 1 и 2 согласно пп. 8.2.1.1, 8.2.1.2 и 8.2.1.3, соответственно. Если текущее изображение включает операцию управления памятью, равную 5, то после декодирования текущего изображения tempPicOrderCnt устанавливают равным PicOrderCnt(CurrPic), TopFieldOrderCnt текущего изображения (если имеется) устанавливают равным TopFieldOrderCnt – tempPicOrderCnt, а BottomFieldOrderCnt текущего изображения (если имеется) устанавливают равным BottomFieldOrderCnt – tempPicOrderCnt.

Поток битов не должен содержать данных, которые возникают в Min(TopFieldOrderCnt, BottomFieldOrderCnt), не равном 0 для кодированного IDR кадра, TopFieldOrderCnt – не равном 0 для кодированного верхнего IDR поля или BottomFieldOrderCnt – не равном 0 для кодированного нижнего IDR поля. Следовательно, по крайней мере одно из значений TopFieldOrderCnt и BottomFieldOrderCnt должно быть равным 0 для полей кодированного IDR кадра.

Если текущее изображение – не IDR изображение, используют следующие условия:

– Рассмотрим список переменных listD, содержащий значения TopFieldOrderCnt и BottomFieldOrderCnt как элементы, связанные со списком изображений, включая все из следующего:
  – Первое изображение в списке – это предыдущее изображение любого из следующих типов:
    – IDR изображение,
    – изображение, содержащее значение memory_management_control_operation, равное 5.
  – Следующие дополнительные изображения:
    – Если pic_order_cnt_type равно 0, то все прочие изображения, которые следуют в порядке декодирования за первым изображением в списке и не являются "несуществующими" кадрами, которые процесс декодирования посчитал промежутками в значении frame_num, определенном в п. 8.2.5.2, либо предшествуют текущему изображению в порядке декодирования, либо являются текущими изображениями. Если pic_order_cnt_type равно 0, все прочие изображения, которые следуют в порядке декодирования за первым изображением в списке и не являются "несуществующими" кадрами, которые процесс декодирования посчитал промежутками в значении frame_num, определенном в п. 8.2.5.2, то текущее изображение включают в listD до того, как начнет действовать процесс разметки декодированного контрольного изображения.
    – Иначе (pic_order_cnt_type не равно 0), все прочие изображения, которые следуют в порядке декодирования за первым изображением в списке и либо предшествуют текущему изображению в порядке декодирования, либо являются текущими изображениями. Если pic_order_cnt_type не равно 0, текущее изображение включают в listD до того, как начнет действовать процесс разметки декодированного контрольного изображения.
– Рассмотрим список переменной listO, который содержит элементы listD, отсортированные в возрастающем порядке. listO не должен содержать ничего из следующего:
  – Пару TopFieldOrderCnt и BottomFieldOrderCnt для кадра или дополнительной пары полей, которые не занимают последовательных позиций в listO.
  – TopFieldOrderCnt, который принимает значение, равное другому TopFieldOrderCnt.
  – BottomFieldOrderCnt, который принимает значение, равное другому BottomFieldOrderCnt.

– BottomFieldOrderCnt, который принимает значение, равное TopFieldOrderCnt, до тех пор, пока BottomFieldOrderCnt и TopFieldOrderCnt принадлежат тому же кодированному кадру или дополнительной паре полей.

Поток битов не должен содержать данные, полученные в значениях TopFieldOrderCnt, BottomFieldOrderCnt, PicOrderCntMsb или FrameNumOffset, использованные в процессе декодирования, как определено в пп. 8.2.1.1–8.2.1.3, и которые выходят за диапазон значений от -2^{31} до $2^{31} - 1$ включительно.

Функция PicOrderCnt(picX) определена следующим образом:

```
if ( picX – это кадр или дополнительная пара полей )
    PicOrderCnt( picX ) = Min( TopFieldOrderCnt, BottomFieldOrderCnt ) кадра или дополнительной пары
полей picX
else if( picX – это верхнее поле )
    PicOrderCnt( picX ) = TopFieldOrderCnt поля picX                                    (8-1)
else if( picX – это нижнее поле )
    PicOrderCnt( picX ) = BottomFieldOrderCnt поля picX.
```

Далее DiffPicOrderCnt(picA, picB) определено следующим образом:

```
DiffPicOrderCnt( picA, picB ) = PicOrderCnt( picA ) – PicOrderCnt( picB ).          (8-2)
```

Поток битов должен содержать данные, полученные в значениях DiffPicOrderCnt(picA, picB) и использованные в процессе декодирования, которые находятся в диапазоне от -2^{15} до $2^{15} - 1$ включительно.

ПРИМЕЧАНИЕ. – Примем, что X – текущее изображение, а Y и Z – два другие изображения в той же последовательности. Y и Z рассматриваем как имеющие тот же порядок на выходе от X, если оба значения, DiffPicOrderCnt(X, Y) и DiffPicOrderCnt (X, Z) – положительные или отрицательные.

ПРИМЕЧАНИЕ. – Многие приложения присваивают PicOrderCnt(X) значение, пропорциональное времени выборки изображения X относительно времени выборки IDR изображения.

Если текущее изображение включает memory_management_control_operation, равное 5, то PicOrderCnt(CurrPic) должно быть больше, чем PicOrderCnt (любого другого изображения в listD).

8.2.1.1 Процесс декодирования для вычисления порядка изображения типа 0

Этот процесс запускают, если pic_order_cnt_type равно 0.

Входом в этот процесс является значение PicOrderCntMsb предыдущего контрольного изображения в порядке декодирования, как определено в этом пункте.

Выходом этого процесса является одно или оба значения TopFieldOrderCnt или BottomFieldOrderCnt.

Переменные prevPicOrderCntMsb и prevPicOrderCntLsb находят следующим образом:

– Если текущее изображение – IDR изображение, prevPicOrderCntMsb устанавливают равным 0 и prevPicOrderCntLsb устанавливают равным 0.

– Иначе (текущее изображение – не IDR изображение), используют следующие условия:

  – Если предыдущее контрольное изображение в порядке декодирования включало memory_management_control_operation, равное 5, используют следующие условия:

    – Если предыдущее контрольное изображение в порядке декодирования – не нижнее поле, то prevPicOrderCntMsb устанавливают равным 0, а prevPicOrderCntLsb устанавливают равным значению TopFieldOrderCnt для предыдущего контрольного изображения в порядке декодирования.

    – Иначе (предыдущее контрольное изображение в порядке декодирования – нижнее поле), prevPicOrderCntMsb устанавливают равным 0, и prevPicOrderCntLsb устанавливают равным 0.

  – Иначе (предыдущее контрольное изображение в порядке декодирования не включало memory_management_control_operation, равное 5), prevPicOrderCntMsb устанавливают равным PicOrderCntMsb предыдущего контрольного изображения в порядке декодирования, а prevPicOrderCntLsb устанавливают равным значению pic_order_cnt_lsb предыдущего контрольного изображения в порядке декодирования.

PicOrderCntMsb текущего изображения находят следующим образом:

```
if( ( pic_order_cnt_lsb < prevPicOrderCntLsb ) &&
  ( ( prevPicOrderCntLsb – pic_order_cnt_lsb ) >= ( MaxPicOrderCntLsb / 2 ) ) )
    PicOrderCntMsb = prevPicOrderCntMsb + MaxPicOrderCntLsb                         (8-3)
else if( ( pic_order_cnt_lsb > prevPicOrderCntLsb ) &&
    ( ( pic_order_cnt_lsb – prevPicOrderCntLsb ) > ( MaxPicOrderCntLsb / 2 ) ) )
```

```
        PicOrderCntMsb = prevPicOrderCntMsb – MaxPicOrderCntLsb
    else
        PicOrderCntMsb = prevPicOrderCntMsb.
```

Если текущее изображение – это не нижнее поле, то TopFieldOrderCnt находят следующим образом:

```
    if( !field_pic_flag | | !bottom_field_flag )
        TopFieldOrderCnt = PicOrderCntMsb + pic_order_cnt_lsb.                    (8-4)
```

Если текущее изображение – не верхнее поле, то BottomFieldOrderCnt находят следующим образом:

```
    if( !field_pic_flag )
        BottomFieldOrderCnt = TopFieldOrderCnt + delta_pic_order_cnt_bottom
    else if( bottom_field_flag )                                                  (8-5)
        BottomFieldOrderCnt = PicOrderCntMsb + pic_order_cnt_lsb.
```

8.2.1.2 Процесс декодирования для вычисления порядка изображения типа 1

Этот процесс запускают, если pic_order_cnt_type равно 1.

Входом в этот процесс является значение FrameNumOffset предыдущего контрольного изображения в порядке декодирования, как определено в этом пункте.

Выходами этого процесса является одно или оба значения TopFieldOrderCnt или BottomFieldOrderCnt.

Значения TopFieldOrderCnt и BottomFieldOrderCnt находят, как определено в этом пункте. Положим что prevFrameNum равно frame_num предыдущего изображения в порядке декодирования.

Если текущее изображение – не IDR изображение, то переменную prevFrameNumOffset находят следующим образом:

– Если предыдущее изображение в порядке декодирования включало memory_management_control_operation, равное 5, то prevFrameNumOffset устанавливают равным 0.

– Иначе (предыдущее изображение в порядке декодирования не включало memory_management_control_operation, равное 5), prevFrameNumOffset устанавливают равным значению FrameNumOffset предыдущего изображения в порядке декодирования.

  ПРИМЕЧАНИЕ. – Если gaps_in_frame_num_value_allowed_flag равно 1, то предыдущее изображение в порядке декодирования может быть "несуществующим" кадром, найденным процессом декодирования для промежутков в frame_num и определенным в п. 8.2.5.2.

Отысканию предшествуют следующие обязательные шаги:

1. Переменную FrameNumOffset находят следующим образом:

```
    if( nal_unit_type = = 5 )
        FrameNumOffset = 0
    else if( prevFrameNum > frame_num )                                           (8-6)
        FrameNumOffset = prevFrameNumOffset + MaxFrameNum
    else
        FrameNumOffset = prevFrameNumOffset.
```

2. Переменную absFrameNum находят следующим образом:

```
    if( num_ref_frames_in_pic_order_cnt_cycle != 0 )
        absFrameNum = FrameNumOffset + frame_num
    else                                                                          (8-7)
        absFrameNum = 0
    if( nal_ref_idc = = 0 && absFrameNum > 0 )
        absFrameNum = absFrameNum – 1.
```

3. Если absFrameNum > 0, picOrderCntCycleCnt и frameNumInPicOrderCntCycle находят следующим образом:

```
    if( absFrameNum > 0 ) {
        picOrderCntCycleCnt = ( absFrameNum – 1 ) / num_ref_frames_in_pic_order_cnt_cycle
        frameNumInPicOrderCntCycle = ( absFrameNum – 1 ) % num_ref_frames_in_pic_order_cnt_cycle   (8-8)
    }
```

4. Переменную expectedDeltaPerPicOrderCntCycle находят следующим образом:

```
expectedDeltaPerPicOrderCntCycle = 0
for( i = 0; i < num_ref_frames_in_pic_order_cnt_cycle; i++ )
    expectedDeltaPerPicOrderCntCycle += offset_for_ref_frame[ i ].                    (8-9)
```

5. Переменную expectedPicOrderCnt находят следующим образом:

```
if( absFrameNum > 0 ){
    expectedPicOrderCnt = picOrderCntCycleCnt * expectedDeltaPerPicOrderCntCycle
    for( i = 0; i <= frameNumInPicOrderCntCycle; i++ )
        expectedPicOrderCnt = expectedPicOrderCnt + offset_for_ref_frame[ i ]
} else
    expectedPicOrderCnt = 0
if( nal_ref_idc = = 0 )                                                           (8-10)
    expectedPicOrderCnt = expectedPicOrderCnt + offset_for_non_ref_pic.
```

6. Переменные TopFieldOrderCnt или BottomFieldOrderCnt находят следующим образом:

```
if( !field_pic_flag ) {
    TopFieldOrderCnt = expectedPicOrderCnt + delta_pic_order_cnt[ 0 ]
    BottomFieldOrderCnt = TopFieldOrderCnt +
        offset_for_top_to_bottom_field + delta_pic_order_cnt[ 1 ]                 (8-11)
} else if( !bottom_field_flag )
    TopFieldOrderCnt = expectedPicOrderCnt + delta_pic_order_cnt[ 0 ]
else
    BottomFieldOrderCnt = expectedPicOrderCnt + offset_for_top_to_bottom_field + delta_pic_order_cnt[ 0 ].
```

8.2.1.3 Процесс декодирования для вычисления порядка изображения типа 2

Этот процесс запускают, если pic_order_cnt_type равно 2.

Выходами этого процесса является одно или оба значения TopFieldOrderCnt или BottomFieldOrderCnt.

Положим prevFrameNum равным frame_num из предыдущего изображения в порядке декодирования.

Если текущее изображение – не IDR изображение, то переменную prevFrameNumOffset находят следующим образом:

– Если предыдущее изображение в порядке декодирования включало memory_management_control_operation, равное 5, то prevFrameNumOffset устанавливают равным 0.

– Иначе (предыдущее изображение в порядке декодирования не включало memory_management_control_operation, равное 5), prevFrameNumOffset устанавливают равным значению FrameNumOffset из предыдущего изображения в порядке декодирования.

  ПРИМЕЧАНИЕ. – Если gaps_in_frame_num_value_allowed_flag равно 1, то предыдущее изображение в порядке декодирования может быть "несуществующим" кадром, найденным процессом декодирования для промежутков в frame_num, определенным в п. 8.2.5.2.

Переменную FrameNumOffset находят следующим образом:

```
if( nal_unit_type = = 5 )
    FrameNumOffset = 0
else if( prevFrameNum > frame_num )                                               (8-12)
    FrameNumOffset = prevFrameNumOffset + MaxFrameNum
else
    FrameNumOffset = prevFrameNumOffset.
```

Переменную tempPicOrderCnt находят следующим образом:

```
if( nal_unit_type = = 5 )
    tempPicOrderCnt = 0
else if( nal_ref_idc = = 0 )                                                      (8-13)
    tempPicOrderCnt = 2 * ( FrameNumOffset + frame_num ) – 1
else
    tempPicOrderCnt = 2 * ( FrameNumOffset + frame_num ).
```

Переменные TopFieldOrderCnt или BottomFieldOrderCnt находят следующим образом:

```
if( !field_pic_flag ) {
      TopFieldOrderCnt = tempPicOrderCnt
      BottomFieldOrderCnt = tempPicOrderCnt                                    (8-14)
} else if( bottom_field_flag )
      BottomFieldOrderCnt = tempPicOrderCnt
else
      TopFieldOrderCnt = tempPicOrderCnt.
```

ПРИМЕЧАНИЕ. – Вычисление порядка изображения типа 2 нельзя использовать в кодированной видеопоследовательности, содержащей непрерывно следующие неконтрольные изображения, которые могли появиться более чем в одном из этих изображений, имеющих то же значение TopFieldOrderCnt, или более чем в одном из этих изображений, имеющих то же значение BottomFieldOrderCnt.

ПРИМЕЧАНИЕ. – На выходе результатами вычисления порядка изображения типа 2 является порядок такой же, как порядок декодирования.

8.2.2 Процесс декодирования для отображения макроблока в группу секции

Входами в этот процесс являются установка параметров действующего изображения и заголовок той секции, из которой должна быть декодирована заданная секция.

Выходом этого процесса является отображение макроблока в группу секции MbToSliceGroupMap.

Этот процесс активируют в начале каждой секции.

ПРИМЕЧАНИЕ. – Выход этого процесса равен всем секциям изображения.

Если num_slice_groups_minus1 равно 1 и slice_group_map_type равно 3, 4 или 5, то группы секций 0 и 1 имеют форму и размеры, которые определяются значением slice_group_change_direction_flag, как показано в таблице 8-1 и описано в пп. 8.2.2.4–8.2.2.6.

Таблица 8-1 – Детальный тип отображения группы секции

slice_group_map_type	slice_group_change_direction_flag	Детальный тип отображения группы секции
3	0	Вне блока по часовой стрелке
3	1	Вне блока против часовой стрелки
4	0	Растровое сканирование
4	1	Обратное растровое сканирование
5	0	Вытеснение направо
5	1	Вытеснение налево

В таком случае блоки отображения группы секции MapUnitsInSliceGroup0 в определенном возрастающем порядке располагают в группе секции 0, а оставшиеся блоки отображения группы секции изображения PicSizeInMapUnits – MapUnitsInSliceGroup0 располагают в группе секции 1.

Если num_slice_groups_minus1 равно 1, а slice_group_map_type равно 4 или 5, переменную sizeOfUpperLeftGroup определяют следующим образом:

```
sizeOfUpperLeftGroup = ( slice_group_change_direction_flag ?
                    ( PicSizeInMapUnits – MapUnitsInSliceGroup0 ) : MapUnitsInSliceGroup0 ).   (8-15)
```

Переменную mapUnitToSliceGroupMap находят следующим образом:

– Если num_slice_groups_minus1 равно 0, отображение блока в отображенную группу секции производят для всех i в диапазоне от 0 до PicSizeInMapUnits – 1 включительно, как определено равенством:

```
mapUnitToSliceGroupMap[ i ] = 0.                                                (8-16)
```

– Иначе (num_slice_groups_minus1 не равно 0), mapUnitToSliceGroupMap находят следующим образом:

  – Если slice_group_map_type равно 0, mapUnitToSliceGroupMap находят так, как это определено в п. 8.2.2.1.

  – Иначе, если slice_group_map_type равно 1, mapUnitToSliceGroupMap находят так, как это определено в п. 8.2.2.2.

  – Иначе, если slice_group_map_type равно 2, mapUnitToSliceGroupMap находят так, как это определено в п. 8.2.2.3.

– Иначе, если slice_group_map_type равно 3, mapUnitToSliceGroupMap находят так, как это определено в п. 8.2.2.4.
– Иначе, если slice_group_map_type равно 4, mapUnitToSliceGroupMap находят так, как это определено в п. 8.2.2.5.
– Иначе, если slice_group_map_type равно 5, mapUnitToSliceGroupMap находят так, как это определено в п. 8.2.2.6.
– Иначе (slice_group_map_type равно 6) mapUnitToSliceGroupMap находят так, как это определено в п. 8.2.2.7.

После отыскания mapUnitToSliceGroupMap активируют процесс, определенный в п. 8.2.2.8 для преобразования отображенного блока в отображенную группу секции mapUnitToSliceGroupMap, а макроблока – в отображенную группу секции MbToSliceGroupMap. После того как отображение макроблока в группу секции найдено, как это определено в п. 8.2.2.8, функцию NextMbAddress(n) определяют как значение переменной nextMbAddress по алгоритму:

```
i = n + 1
while( i < PicSizeInMbs && MbToSliceGroupMap[ i ] != MbToSliceGroupMap[ n ] )
    i++;
nextMbAddress = i.                                                    (8-17)
```

8.2.2.1 Спецификация типа чередующейся группы секции

Спецификацию этого пункта применяют, если slice_group_map_type равно 0.

Отображенный блок в отображенной группе секции формируют, как определено ниже:

```
i = 0
do
    for( iGroup = 0; iGroup <= num_slice_groups_minus1 && i < PicSizeInMapUnits;
   i += run_length_minus1[ iGroup++ ] + 1 )
    for( j = 0; j <= run_length_minus1[ iGroup ] && i + j < PicSizeInMapUnits; j++ )
        mapUnitToSliceGroupMap[ i + j ] = iGroup                      (8-18)
while( i < PicSizeInMapUnits ).
```

8.2.2.2 Спецификация типа отображения распределенной группы секции

Спецификацию этого пункта применяют, если slice_group_map_type равно 1.

Отображенный блок в отображенной группе секции формируют, как определено ниже:

```
for( i = 0; i < PicSizeInMapUnits; i++ )
    mapUnitToSliceGroupMap[ i ] = ( ( i % PicWidthInMbs ) +
                                    ( ( ( i / PicWidthInMbs ) * ( num_slice_groups_minus1 + 1 ) ) / 2 ) )
                                    % ( num_slice_groups_minus1 + 1 ).  (8-19)
```

8.2.2.3 Спецификация переднего плана с типом отображения сдвига группы секции налево вверх

Спецификацию этого пункта применяют, если slice_group_map_type равно 2.

Отображенный блок в отображенной группе секции формируют, как определено ниже:

```
for( i = 0; i < PicSizeInMapUnits; i++ )
    mapUnitToSliceGroupMap[ i ] = num_slice_groups_minus1
for( iGroup = num_slice_groups_minus1 – 1; iGroup >= 0; iGroup-- ) {
    yTopLeft = top_left[ iGroup ] / PicWidthInMbs
    xTopLeft = top_left[ iGroup ] % PicWidthInMbs
    yBottomRight = bottom_right[ iGroup ] / PicWidthInMbs
    xBottomRight = bottom_right[ iGroup ] % PicWidthInMbs
    for( y = yTopLeft; y <= yBottomRight; y++ )
        for( x = xTopLeft; x <= xBottomRight; x++ )
            mapUnitToSliceGroupMap[ y * PicWidthInMbs + x ] = iGroup  (8-20)
}
```

После использования процесса, определенного равенством 8-20, должно быть, по крайней мере, одно значение i от 0 до PicSizeInMapUnits – 1 включительно, для которого mapUnitToSliceGroupMap[i] равно iGroup для каждого значения iGroup от 0 до num_slice_groups_minus1 включительно (т. е. каждая группа секции должна содержать, по крайней мере, один блок отображения группы секции).

ПРИМЕЧАНИЕ. – Прямоугольники могут перекрываться. Группа секции 0 содержит макроблоки, которые находятся внутри прямоугольника, определенного верхней левой [0] и нижней правой [0] координатой. Группа секции, в которой ID группы секции больше 0 и меньше, чем num_slice_groups_minus1, содержит макроблоки, которые находятся внутри определенного прямоугольника для этой группы секции, и не находятся внутри прямоугольника, определенного для любой группы секции, имеющей меньшее значение ID группы секции. Группа секции с ID группы секции, равным num_slice_groups_minus1, содержит макроблоки, которых нет в других группах секций.

8.2.2.4 Спецификация типов отображения групп секций вне блока

Спецификацию этого пункта применяют, если slice_group_map_type равно 3.

Отображенный блок в отображенной группе секции формируют, как определено ниже:

```
for( i = 0; i < PicSizeInMapUnits; i++ )
    mapUnitToSliceGroupMap[ i ] = 1
x = ( PicWidthInMbs – slice_group_change_direction_flag ) / 2
y = ( PicHeightInMapUnits – slice_group_change_direction_flag ) / 2
( leftBound, topBound ) = ( x, y )
( rightBound, bottomBound ) = ( x, y )
( xDir, yDir ) = ( slice_group_change_direction_flag – 1, slice_group_change_direction_flag )
for( k = 0; k < MapUnitsInSliceGroup0; k += mapUnitVacant ) {
    mapUnitVacant = ( mapUnitToSliceGroupMap[ y * PicWidthInMbs + x ] = = 1 )
    if( mapUnitVacant )
        mapUnitToSliceGroupMap[ y * PicWidthInMbs + x ] = 0                    (8-21)
    if( xDir = = –1 && x = = leftBound ) {
        leftBound = Max( leftBound – 1, 0 )
        x = leftBound
        ( xDir, yDir ) = ( 0, 2 * slice_group_change_direction_flag – 1 )
    } else if( xDir = = 1 && x = = rightBound ) {
        rightBound = Min( rightBound + 1, PicWidthInMbs – 1 )
        x = rightBound
        ( xDir, yDir ) = ( 0, 1 – 2 * slice_group_change_direction_flag )
    } else if( yDir = = –1 && y = = topBound ) {
        topBound = Max( topBound – 1, 0 )
        y = topBound
        ( xDir, yDir ) = ( 1 – 2 * slice_group_change_direction_flag, 0 )
    } else if( yDir = = 1 && y = = bottomBound ) {
        bottomBound = Min( bottomBound + 1, PicHeightInMapUnits – 1 )
        y = bottomBound
        ( xDir, yDir ) = ( 2 * slice_group_change_direction_flag – 1, 0 )
    } else
        ( x, y ) = ( x + xDir, y + yDir )
}
```

8.2.2.5 Спецификация типов отображения групп секций растрового сканирования

Спецификацию этого пункта применяют, если slice_group_map_type равно 4.

Отображенный блок в отображенной группе секции формируют, как определено ниже:

```
for( i = 0; i < PicSizeInMapUnits; i++ )
    if( i < sizeOfUpperLeftGroup )
        mapUnitToSliceGroupMap[ i ] = slice_group_change_direction_flag
    else                                                                       (8-22)
        mapUnitToSliceGroupMap[ i ] = 1 – slice_group_change_direction_flag.
```

8.2.2.6 Спецификация типов отображения вытесненных групп секций

Спецификацию этого пункта применяют, если slice_group_map_type равно 5.

Отображенный блок в отображенной группе секции формируют, как определено ниже:

```
k = 0;
for( j = 0; j < PicWidthInMbs; j++ )
    for( i = 0; i < PicHeightInMapUnits; i++ )
        if( k++ < sizeOfUpperLeftGroup )
            mapUnitToSliceGroupMap[ i * PicWidthInMbs + j ] = slice_group_change_direction_flag
```

else (8-23)
mapUnitToSliceGroupMap[i * PicWidthInMbs + j] = 1 – slice_group_change_direction_flag.

8.2.2.7 Спецификация типа отображения группы секции, заданной в явной форме

Спецификацию этого пункта применяют, если slice_group_map_type равно 6.

Отображенный блок в отображенной группе секции формируют, как определено ниже:

mapUnitToSliceGroupMap[i] = slice_group_id[i]. (8-24)

для всех i в диапазоне от 0 до PicSizeInMapUnits – 1 включительно.

8.2.2.8 Спецификация для преобразования отображенного блока в отображенную группу секции и макроблока в отображенную группу секции

Для каждого значения i в диапазоне от 0 до PicSizeInMbs – 1 включительно отображение макроблока в группу секции определено следующим образом:

– Если frame_mbs_only_flag равно 1 или field_pic_flag равно 1, отображение макроблока в группу секции определено как:

MbToSliceGroupMap[i] = mapUnitToSliceGroupMap[i]. (8-25)

– Иначе, если MbaffFrameFlag равно 1, отображение макроблока в группу секции определено как:

MbToSliceGroupMap[i] = mapUnitToSliceGroupMap[i / 2]. (8-26)

– Иначе (frame_mbs_only_flag равно 0, mb_adaptive_frame_field_flag равно 0 и field_pic_flag равно 0), отображение макроблока в группу секции определено как:

MbToSliceGroupMap[i] = mapUnitToSliceGroupMap[(i / (2 * PicWidthInMbs)) * PicWidthInMbs
+ (i % PicWidthInMbs)]. (8-27)

8.2.3 Процесс декодирования для разделения данных секции

Входами в этот процесс являются:

– RBSP слой разделенной части A данных секции,

– RBSP слой разделенной части B данных секции, имеющей тот же идентификатор slice_id, что и в RBSP слое разделенной части A данных секции, если в данных секции присутствуют элементы синтаксиса категории 3, и

– RBSP слой разделенной части C данных секции, имеющей тот же идентификатор slice_id, что и в RBSP слое разделенной части A данных секции, если в данных секции присутствуют элементы синтаксиса категории 4.

ПРИМЕЧАНИЕ. – Не обязательно должны быть представлены RBSP слой разделенной части B данных секции и RBSP слой разделенной части C данных секции.

Выход этого процесса – кодированная секция.

Если не используют разделение данных секции, то кодированные секции представляют слоем секции без RBSP разделения, который содержит заголовок секции и далее следует структура синтаксиса данных секции, содержащая все элементы синтаксиса категорий 2, 3 и 4 (см. столбец категорий в п. 7.3) данных макроблока для макроблоков этой секции.

Если используют разделение данных секции, то данные макроблока секции разделяют на части, от одной до трех, каждая в отдельном блоке NAL. Разделенная часть A содержит заголовок части A данных секции и все элементы синтаксиса категории 2. Разделенная часть B, если она присутствует, содержит заголовок части B данных секции и все элементы синтаксиса категории 3. Разделенная часть C, если она присутствует, содержит заголовок части C данных секции и все элементы синтаксиса категории 4.

При использовании разделения данных секции элементы синтаксиса каждой категории анализируют из отдельного блока NAL, который может не присутствовать, если отсутствуют символы соответствующей категории. В кодированной секции процесс декодирования должен обрабатывать разделенные части данных секции способом, эквивалентным обработке соответствующего слоя секция без разделения RBSP, с помощью извлечения каждого элемента синтаксиса из разделенной части данных секции, в которой элемент синтаксиса появляется в зависимости от назначения разделенной части в таблицах синтаксиса в п. 7.3.

ПРИМЕЧАНИЕ. – Элементы синтаксиса категории 3 существенны для декодирования остатков данных макроблоков типов I и SI. Элементы синтаксиса категории 4 существенны для декодирования остатков данных макроблоков типов P и B. Категория 2 включает все прочие элементы синтаксиса, связанные с декодированием макроблоков, и информацию о них часто обозначают как информационный заголовок. Заголовок части A данных секции содержит все элементы синтаксиса заголовка секции и дополнительно идентификатор slice_id секции, который используют для объединения разделенных частей данных секции B и C с разделенной частью A. Заголовки разделенных частей данных секции B и C содержат элемент синтаксиса slice_id, который устанавливает их объединение с разделенной частью A данных секции.

8.2.4 Процесс декодирования для создания списков контрольных изображений

Этот процесс активируют в начале декодирования каждой секции P, SP или B.

Выходами этого процесса являются список контрольных изображений RefPicList0 и, если декодируют B секцию, второй список контрольных изображений RefPicList1.

Декодированные контрольные изображения помечены как "использованные для краткосрочного контроля" или "использованные для долгосрочного контроля", как определено потоком битов в п. 8.2.5. Краткосрочные декодированные контрольные изображения идентифицированы значением frame_num. Долгосрочным декодированным контрольным изображениям присвоен долгосрочный индекс кадра, как это определено потоком битов и в п. 8.2.5.

Пункт 8.2.4.1 активируют, чтобы определить:

– присвоение переменным FrameNum, FrameNumWrap и PicNum каждого из краткосрочных контрольных изображений, и

– присвоение переменной LongTermPicNum каждого из долгосрочных контрольных изображений.

Контрольные изображения адресуют по их индексам контроля, как определено в п. 8.4.2.1. Индекс контроля – это индекс в списках переменных PicNum и LongTermPicNum, которые называются списком контрольного изображения. Если декодирована секция P или SP, существует единственный список контрольного изображения RefPicList0. Если декодирована секция B, существует второй независимый список контрольных изображений RefPicList1 дополнительно к RefPicList0.

В начале декодирования каждой секции список контрольного изображения RefPicList0, а для секций B RefPicList1, находят следующим образом:

– Первоначальный список контрольного изображения RefPicList0, а для секций B RefPicList1, находят, как определено в п. 8.2.4.2.

– Первоначальный список контрольного изображения RefPicList0, а для секций B RefPicList1, модифицируют, как определено в п. 8.2.4.3.

Число входов в модифицированном списке контрольного изображения RefPicList0 – это num_ref_idx_l0_active_minus1 + 1, а для секций B число входов в модифицированном списке контрольного изображения RefPicList1 – это num_ref_idx_l1_active_minus1 + 1. В модифицированных списках контрольных изображений RefPicList0 или RefPicList1 может появиться более одного индекса.

8.2.4.1 Процесс декодирования номеров изображений

Этот процесс активируют, если активируют процесс декодирования для создания списков контрольного изображения, определенный в п. 8.2.4, или процесс разметки декодированного контрольного изображения, определенный в п. 8.2.5.

Переменные FrameNum, FrameNumWrap, PicNum, LongTermFrameIdx и LongTermPicNum используют для процесса инициализации списков контрольного изображения (по п. 8.2.4.2), модификации процесса списков контрольного изображения (по п. 8.2.4.3) и для процесса разметки декодированного контрольного изображения (по п. 8.2.5).

Каждому краткосрочному контрольному изображению переменные FrameNum и FrameNumWrap присваивают следующим образом. Во-первых, FrameNum устанавливают равным элементу синтаксиса frame_num, который декодирован в заголовке(ках) секции соответствующего краткосрочного контрольного изображения. Затем переменную FrameNumWrap находят как

```
if( FrameNum > frame_num )
    FrameNumWrap = FrameNum – MaxFrameNum                    (8-28)
else
    FrameNumWrap = FrameNum,
```

где значение frame_num, использованное в равенстве 8-28, – это frame_num в заголовке(ках) секции текущего изображения.

Каждому долгосрочному контрольному изображению переменную LongTermFrameIdx присваивают, как определено в п. 8.2.5.

Каждому краткосрочному контрольному изображению присваивают переменную PicNum, а каждому долгосрочному контрольному изображению присваивают переменную LongTermPicNum. Значения этих переменных зависят от значений field_pic_flag и bottom_field_flag текущего изображения, и их устанавливают следующим образом:

– Если field_pic_flag равно 0, используют следующие условия:

  – Для каждого краткосрочного контрольного кадра или дополнительной контрольной пары полей:

    PicNum = FrameNumWrap. (8-29)

  – Для каждого долгосрочного контрольного кадра или долгосрочной дополнительной контрольной пары полей:

    LongTermPicNum = LongTermFrameIdx. (8-30)

ПРИМЕЧАНИЕ. – При декодировании кадра значение MbaffFrameFlag не оказывает влияния на выводы формул в пп. 8.2.4.2, 8.2.4.3 и 8.2.5.

– Иначе (field_pic_flag равно 1), используют следующие условия:

  – Для каждого краткосрочного контрольного поля используют следующие условия:

    – Если контрольное поле имеет ту же четность, что и текущее поле, то

    $$PicNum = 2 * FrameNumWrap + 1. \quad (8\text{-}31)$$

    – Иначе (контрольное поле имеет обратную четность текущего поля),

    $$PicNum = 2 * FrameNumWrap. \quad (8\text{-}32)$$

  – Для каждого долгосрочного контрольного поля используют следующие условия:

    – Если контрольное поле имеет ту же четность, что и текущее поле, то

    $$LongTermPicNum = 2 * LongTermFrameIdx + 1. \quad (8\text{-}33)$$

    – Иначе (контрольное поле имеет обратную четность текущего поля),

    $$LongTermPicNum = 2 * LongTermFrameIdx. \quad (8\text{-}34)$$

8.2.4.2 Процесс инициализации списков контрольного изображения

Процесс инициализации активируют, если декодирован заголовок секции P, SP или B.

Выходами этого процесса являются исходный список контрольного изображения RefPicList0 и, если декодирована секция B, исходный список контрольного изображения RefPicList1.

Для RefPicList0 и RefPicList1 начальными входами служат переменные PicNum и LongTermPicNum, как определено в пп. 8.2.4.2.1–8.2.4.2.5.

Если число входов в исходных значениях RefPicList0 или RefPicList1, созданных, как это определено в пп. 8.2.4.2.1–8.2.4.2.5, будет больше чем num_ref_idx_l0_active_minus1 + 1 или num_ref_idx_l1_active_minus1 + 1, соответственно, то позиции предыдущих дополнительных входов num_ref_idx_l0_active_minus1 или num_ref_idx_l1_active_minus1 аннулируют в исходном списке контрольного изображения.

Если число входов в исходных значениях RefPicList0 or RefPicList1, созданных, как это определено в пп. 8.2.4.2.1–8.2.4.2.5, будет меньше чем num_ref_idx_l0_active_minus1 + 1 или num_ref_idx_l1_active_minus1 + 1, соответственно, то оставшиеся входы в исходном списке контрольного изображения устанавливают равными "нет контрольного изображения".

8.2.4.2.1 Процесс инициализации списка контрольного изображения в кадрах секций P и SP

Процесс инициализации активируют при декодировании секции P или SP в кодированном кадре.

Выходом этого процесса является исходный список контрольных изображений RefPicList0.

Если этот процесс активирован, то должен быть, по крайней мере, один контрольный кадр или дополнительная контрольная пара полей, которые помечены как "использованные для краткосрочного контроля" или "использованные для долгосрочного контроля".

Список контрольного изображения RefPicList0 составлен так, что краткосрочные контрольные кадры и краткосрочная дополнительная контрольная пара полей имеют меньшие индексы, чем долгосрочные контрольные кадры и долгосрочная дополнительная контрольная пара полей.

Краткосрочные контрольные кадры и дополнительная контрольная пара полей расположены так, что начинают с кадра или дополнительной пары полей с наивысшим значением PicNum и далее следуют в нисходящем порядке до кадра или дополнительной пары полей с самым низким значением PicNum.

Долгосрочные контрольные кадры и дополнительная контрольная пара полей расположены так, что начинают с кадра или дополнительной пары полей с самым низким значением LongTermPicNum и далее следуют в возрастающем порядке до кадра или дополнительной пары полей с наивысшим значением LongTermPicNum.

ПРИМЕЧАНИЕ. – Непарное контрольное поле не используют для внешнего предсказания при декодировании кадра, независимо от значения MbaffFrameFlag.

Например, если три контрольных кадра помечены как "использованные для краткосрочного контроля" с PicNum, равным 300, 302 и 303, а два контрольных кадра помечены как "использованные для долгосрочного контроля" с LongTermPicNum, равным 0 и 3, то исходный порядок индексов будет следующим:

– RefPicList0[0] устанавливают равным краткосрочному контрольному изображению с PicNum = 303,

– RefPicList0[1] устанавливают равным краткосрочному контрольному изображению с PicNum = 302,

– RefPicList0[2] устанавливают равным краткосрочному контрольному изображению с PicNum = 300,

– RefPicList0[3] устанавливают равным долгосрочному контрольному изображению с LongTermPicNum = 0 и

– RefPicList0[4] устанавливают равным долгосрочному контрольному изображению с LongTermPicNum = 3.

8.2.4.2.2 Процесс инициализации списка контрольного изображения в полях секций P и SP

Процесс инициализации активируют при декодировании секции P или SP в кодированном поле.

Выходом этого процесса является исходный список контрольных изображений RefPicList0.

При декодировании поля каждое поле, включенное в список контрольного изображения, имеет в списке отличный индекс.

ПРИМЕЧАНИЕ. – При декодировании поля имеется, по крайней мере, вдвое больше действующих изображений, доступных для контроля точно так же, как при декодировании кадра в том же положении порядка декодирования.

Два упорядоченных списка контрольных кадров, refFrameList0ShortTerm и refFrameList0LongTerm, находят следующим образом. С целью формирования этого списка кадры, декодированные кадры, дополнительные контрольные пары полей, непарные контрольные поля и контрольные кадры, в которых единственное поле отмечено как "использованное для краткосрочного контроля" или "использованное для долгосрочного контроля", – все рассматривают как контрольные кадры.

– Значения FrameNumWrap всех кадров, имеющих одно или более полей, помеченных как "использованные для краткосрочного контроля", включают в список краткосрочных контрольных кадров refFrameList0ShortTerm. Если текущее поле – второе поле (в порядке декодирования) дополнительной контрольной пары полей, а первое поле помечено как "использованное для краткосрочного контроля", то значение FrameNumWrap первого поля включают в список refFrameList0ShortTerm. В списке refFrameList0ShortTerm порядок расположения начинается с кадра с наивысшим значением FrameNumWrap и далее следует в нисходящем порядке до кадра с самым низким значением FrameNumWrap.

– Значения LongTermFrameIdx всех кадров, имеющих одно или более полей, помеченных как "использованные для долгосрочного контроля", включают в список долгосрочных контрольных кадров refFrameList0LongTerm. Если текущее поле – второе поле (в порядке декодирования) дополнительной контрольной пары полей, а первое поле помечено как "использованное для долгосрочного контроля", то значение LongTermFrameIdx первого поля включено в список refFrameList0LongTerm. В списке refFrameList0LongTerm порядок расположения начинается с кадра с самым низким значением LongTermFrameIdx и далее следует в возрастающем порядке до кадра с самым высоким значением LongTermFrameIdx.

Процесс, определенный в п. 8.2.4.2.5, активируют со значениями refFrameList0ShortTerm и refFrameList0LongTerm, заданными как входные, а выходные присвоены значению RefPicList0.

8.2.4.2.3 Процесс инициализации списка контрольного изображения в кадрах секций B

Процесс инициализации активируют при декодировании секции B в кодированном кадре.

Выходами этого процесса являются исходные списки контрольного изображения RefPicList0 и RefPicList1.

Если этот процесс активирован, то должен быть, по крайней мере, один контрольный кадр или дополнительная контрольная пара полей, которые помечены как "использованные для краткосрочного контроля" или "использованные для долгосрочного контроля".

Для секций B порядок краткосрочного контрольного изображения в списках контрольных изображений RefPicList0 и RefPicList1 зависит от порядка на выходе, который задан PicOrderCnt( ). Если pic_order_cnt_type равно 0, контрольные изображения, которые помечены как "несуществующие", как это определено в п. 8.2.5.2, не включают ни в RefPicList0, ни в RefPicList1.

ПРИМЕЧАНИЕ. – Если gaps_in_frame_num_value_allowed_flag равно 1, кодеры должны изменить порядок в списке контрольного изображения, чтобы обеспечить правильное выполнение процесса декодирования (в частности, если pic_order_cnt_type равно 0. В этом случае значение PicOrderCnt( ) не считают "несуществующими" кадрами).

Список контрольного изображения RefPicList0 составлен так, что краткосрочные контрольные кадры и краткосрочная дополнительная контрольная пара полей имеют меньшие индексы, чем долгосрочные контрольные кадры и долгосрочная дополнительная контрольная пара полей. Список составлен следующим образом:

– Краткосрочные контрольные кадры и краткосрочные дополнительные контрольные пары полей начинаются с краткосрочного контрольного кадра или дополнительной контрольной пары полей frm0 с наибольшим значением PicOrderCnt(frm0), но меньшим значения PicOrderCnt(CurrPic). Эта последовательность продолжается в убывающем порядке до краткосрочного контрольного кадра или дополнительной контрольной пары полей frm1, которые имеют наименьшее значение PicOrderCnt(frm1), а затем переходит к краткосрочному контрольному кадру или дополнительной контрольной паре полей frm2 с наименьшим значением PicOrderCnt(frm2), которое

больше, чем значение PicOrderCnt(CurrPic) текущего кадра, а далее последовательность продолжается в возрастающем порядке до краткосрочного контрольного кадра или дополнительной контрольной пары полей frm3, которая имеет наибольшее значение PicOrderCnt(frm3).

– Долгосрочные контрольные кадры и долгосрочные дополнительные контрольные пары полей начинаются с долгосрочного контрольного кадра или дополнительной контрольной пары полей, которые имеют наименьшее значение LongTermPicNum. Эта последовательность продолжается в возрастающем порядке до долгосрочного контрольного кадра или дополнительной контрольной пары полей с наивысшим значением LongTermPicNum.

Список контрольного изображения RefPicList1 составлен так, что краткосрочные контрольные кадры и краткосрочные дополнительные контрольные пары полей имеют меньшие индексы, чем долгосрочные контрольные кадры и долгосрочные дополнительные контрольные пары полей. Список составлен следующим образом:

– Краткосрочные контрольные кадры и краткосрочные дополнительные контрольные пары полей начинаются с краткосрочного контрольного кадра или дополнительной контрольной пары полей frm4 с наименьшим значением PicOrderCnt(frm4), но большим значения PicOrderCnt(CurrPic) текущего кадра. Эта последовательность продолжается в возрастающем порядке до краткосрочного контрольного кадра или дополнительной контрольной пары полей frm5, которые имеют наибольшее значение PicOrderCnt(frm5), и продолжается до краткосрочного контрольного кадра или дополнительной контрольной пары полей frm6 с наибольшим значением PicOrderCnt(frm6), но меньшим чем PicOrderCnt(CurrPic) текущего кадра, и далее продолжается в убывающем порядке до краткосрочного контрольного кадра или дополнительной контрольной пары полей frm7 с наименьшим значением PicOrderCnt(frm7).

– Долгосрочные контрольные кадры и долгосрочные дополнительные контрольные пары полей начинаются с долгосрочного контрольного кадра или дополнительной контрольной пары полей, которые имеют наименьшее значение LongTermPicNum, и продолжаются в возрастающем порядке до долгосрочного контрольного кадра или дополнительной контрольной пары полей с наибольшим значением LongTermPicNum.

– Если список контрольного изображения RefPicList1 имеет более одного входа, а RefPicList1 идентичен списку контрольного изображения RefPicList0, то первые два входа RefPicList1[0] и RefPicList1[1] переключают.

ПРИМЕЧАНИЕ. – Непарное контрольное поле не используют для внешнего предсказания кадров, независимо от значения MbaffFrameFlag.

8.2.4.2.4 Процесс инициализации списка контрольного изображения в полях секций B

Процесс инициализации активируют при декодировании в кодированном поле секции B.

Выходами этого процесса являются исходные списки контрольного изображения RefPicList0 и RefPicList1.

При декодировании поля каждое поле сохраненного контрольного кадра идентифицируют как отдельное контрольное изображение с особым индексом. Порядок краткосрочных контрольных изображений в списках контрольного изображения RefPicList0 и RefPicList1 зависит от порядка на выходе, заданного PicOrderCnt( ). Если pic_order_cnt_type равно 0, то контрольные изображения, помеченные как "несуществующие", как это определено в п. 8.2.5.2, не включают ни в RefPicList0, ни в RefPicList1.

ПРИМЕЧАНИЕ. – Если gaps_in_frame_num_value_allowed_flag равно 1, кодеры должны изменить порядок в списке контрольного изображения, чтобы обеспечить правильное выполнение процесса декодирования (в частности, если pic_order_cnt_type равно 0. В этом случае значения PicOrderCnt( ) не считают "несуществующими" кадрами).

ПРИМЕЧАНИЕ. – При декодировании поля имеется, по крайней мере, вдвое больше действующих изображений, доступных для контроля точно так же, как при декодировании кадра в том же положении порядка декодирования.

Три упорядоченных списка контрольных кадров, refFrameList0ShortTerm, refFrameList1ShortTerm и refFrameListLongTerm, находят следующим образом. С целью формирования этого списка кадров термин контрольный вход относят к декодированным контрольным кадрам, дополнительным контрольным парам полей или к непарным контрольным полям. Если pic_order_cnt_type равно 0, термин контрольный вход не относят к кадрам, которые помечены как "несуществующие", как это определено в п. 8.2.5.2:

– refFrameList0ShortTerm по порядку начинают с контрольного входа f0 с наибольшим значением PicOrderCnt(f0), но меньшим или равным значению PicOrderCnt(CurrPic) текущего поля, и далее продолжают в нисходящем порядке до краткосрочного контрольного входа f1, который имеет наименьшее значение PicOrderCnt(f1), затем продолжают до контрольного входа f2 с наименьшим значением PicOrderCnt(f2), но большим значения PicOrderCnt(CurrPic) текущего поля, а далее в возрастающем порядке до краткосрочного контрольного входа f3, который имеет наибольшее значение PicOrderCnt(f3).

ПРИМЕЧАНИЕ. – Если в порядке декодирования за текущем полем следует кодированное поле fldPrev, с которым эти поля формируют дополнительную контрольную пару полей, fldPrev включают в список refFrameList0ShortTerm с помощью PicOrderCnt(fldPrev) и используют метод упорядочения, описанный в предыдущем предложении.

– refFrameList1ShortTerm по порядку начинают с контрольного входа f4 с наименьшим значением PicOrderCnt(f4), но большим значения PicOrderCnt(CurrPic) текущего поля, и далее продолжают в возрастающем порядке до краткосрочного контрольного входа f5, который имеет наибольшее значение PicOrderCnt(f5), затем продолжают до контрольного входа f6 с наибольшим значением PicOrderCnt(f6), но меньшим или равным

значению PicOrderCnt(CurrPic) текущего поля, а далее в убывающем порядке до краткосрочного контрольного входа f7, который имеет наименьшее значение PicOrderCnt(f7).

ПРИМЕЧАНИЕ. – Если в порядке декодирования за текущем полем следует кодированное поле fldPrev, с которым эти поля формируют дополнительную контрольную пару полей, fldPrev включают в список refFrameList1ShortTerm с помощью PicOrderCnt(fldPrev) и используют метод упорядочения, описанный в предыдущем предложении.

– refFrameListLongTerm по порядку начинают с контрольного входа, имеющего наименьшее значение LongTermFrameIdx, и продолжают в возрастающем порядке до контрольного входа, имеющего наибольшее значение LongTermFrameIdx.

ПРИМЕЧАНИЕ. – Если дополнительное поле текущего изображения помечено как "использованное для долгосрочного контроля", это поле включают в список refFrameListLongTerm. Контрольный вход, в котором только одно поле помечено как "использованное для долгосрочного контроля", включают в список refFrameListLongTerm.

Процесс, определенный в п. 8.2.4.2.5, активируют с refFrameList0ShortTerm и refFrameListLongTerm, заданными как вход и выход, которые присвоены значению RefPicList0.

Процесс, определенный в п. 8.2.4.2.5, активируют с refFrameList1ShortTerm и refFrameListLongTerm, заданными как вход и выход, которые присвоены значению RefPicList1.

Если список контрольного изображения RefPicList1 имеет более одного входа, а RefPicList1 идентичен списку контрольного изображения RefPicList0, то первые два входа RefPicList1[0] и RefPicList1[1] переключают.

8.2.4.2.5 Процесс инициализации списков контрольных изображений в полях

Входами этого процесса являются списки контрольных кадров refFrameListXShortTerm (с X равным 0 или 1) и refFrameListLongTerm.

Выход этого процесса – список контрольных изображений RefPicListX (это может быть RefPicList0 или RefPicList1).

Список контрольных изображений RefPicListX – это список в таком порядке, при котором краткосрочные контрольные поля имеют более низкие индексы, чем долгосрочные контрольные поля. Заданные списки контрольных кадров refFrameListXShortTerm и refFrameListLongTerm находят следующим образом:

– Краткосрочные контрольные поля располагают, выбирая контрольные поля из упорядоченного списка кадров refFrameListXShortTerm переменой полей различной четности, начиная с поля, которое имеет ту же четность, что и текущее поле (если это поле присутствует). Если какое-нибудь поле контрольного кадра не было декодировано или отмечено как "использованное для краткосрочного контроля", такое исчезающее поле игнорируют и вместо него из упорядоченного списка кадров refFrameListXShortTerm выбирают следующее доступное сохраненное контрольное поле той же четности и вводят его в RefPicListX. Если в упорядоченном списке кадров refFrameListXShortTerm отсутствует краткосрочное контрольное поле альтернативной четности, в RefPicListX вводят еще не индексированные поля соответствующей четности в том порядке, в котором они расположены в упорядоченном списке кадров refFrameListXShortTerm.
– Долгосрочные контрольные поля упорядочивают, выбирая контрольные поля из упорядоченного списка кадров refFrameListLongTerm переменой полей различной четности, начиная с поля, которое имеет ту же четность, что и текущее поле (если это поле присутствует). Если какое-нибудь поле контрольного кадра не было декодировано или отмечено как "использованное для долгосрочного контроля", такое исчезающее поле игнорируют и вместо него из упорядоченного списка кадров refFrameListLongTerm выбирают следующее доступное сохраненное контрольное поле той же четности и вводят его в RefPicListX. Если в упорядоченном списке кадров refFrameListLongTerm отсутствует долгосрочное контрольное поле альтернативной четности, в refFrameListLongTerm вводят еще не индексированные поля соответствующей четности в том порядке, в котором они расположены в упорядоченном списке кадров refFrameListLongTerm.

8.2.4.3 Процесс изменения порядка в списках контрольных изображений

Вход этого процесса – список контрольных изображений RefPicList0 и, при декодировании секции B, также список контрольных изображений RefPicList1.

Выходы этого процесса – возможный модифицированный список контрольных изображений RefPicList0 и, при декодировании секции B, также возможный модифицированный список контрольных изображений RefPicList1.

Если ref_pic_list_reordering_flag_l0 равно 1, используют следующие условия:

– Допустим, что refIdxL0 – индекс в списке контрольных изображений RefPicList0. Первоначально он установлен равным 0.
– Соответствующие элементы синтаксиса reordering_of_pic_nums_idc обработаны в порядке их поступления в поток битов. Для каждого из этих элементов синтаксиса используют следующие условия:
  – Если reordering_of_pic_nums_idc равно 0 или 1, активируют процесс, определенный в п. 8.2.4.3.1, с RefPicList0 и refIdxL0, заданными как вход, и выходом, присвоенным значениям RefPicList0 и refIdxL0.
  – Иначе, если reordering_of_pic_nums_idc равно 2, активируют процесс, определенный в п. 8.2.4.3.2, с RefPicList0 и refIdxL0, заданными как вход, и выходом, присвоенным значениям RefPicList0 и refIdxL0.

– Иначе (reordering_of_pic_nums_idc равно 3), процесс изменения порядка в списке контрольных изображений RefPicList0 закончен.

Если ref_pic_list_reordering_flag_l1 равно 1, используют следующие условия:

– Допустим, что refIdxL1 – индекс в списке контрольных изображений RefPicList1. Первоначально он установлен равным 0.

– Соответствующие элементы синтаксиса reordering_of_pic_nums_idc обработаны в порядке их поступления в поток битов. Для каждого из этих элементов синтаксиса используют следующие условия:

  – Если reordering_of_pic_nums_idc равно 0 или 1, активируют процесс, определенный в п. 8.2.4.3.1, с RefPicList1 и refIdxL1 заданными как вход, и выходом, присвоенным значениям RefPicList1 и refIdxL1.

  – Иначе, если reordering_of_pic_nums_idc равно 2, активируют процесс, определенный в п. 8.2.4.3.2, с RefPicList1 и refIdxL1 заданными как вход, и выходом, присвоенным значениям RefPicList1 и refIdxL1.

  – Иначе (reordering_of_pic_nums_idc равно 3), процесс изменения порядка в списке контрольных изображений RefPicList1 закончен.

8.2.4.3.1 Процесс изменения порядка в списках контрольных изображений для краткосрочных изображений

Входы в этот процесс – список контрольных изображений RefPicListX (с X равным 0 или 1) и индекс refIdxLX в этом списке.

Выходы этого процесса – возможный модифицированный список контрольных изображений RefPicListX и возрастающий индекс refIdxLX.

Переменную picNumLXNoWrap находят следующим образом:

– Если reordering_of_pic_nums_idc равно 0

```
if( picNumLXPred – ( abs_diff_pic_num_minus1 + 1 ) < 0 )
    picNumLXNoWrap = picNumLXPred – ( abs_diff_pic_num_minus1 + 1 ) + MaxPicNum      (8-35)
else
    picNumLXNoWrap = picNumLXPred – ( abs_diff_pic_num_minus1 + 1 ).
```

– Иначе (reordering_of_pic_nums_idc равно 1),

```
if( picNumLXPred + ( abs_diff_pic_num_minus1 + 1 ) >= MaxPicNum )
    picNumLXNoWrap = picNumLXPred + ( abs_diff_pic_num_minus1 + 1 ) – MaxPicNum      (8-36)
else
    picNumLXNoWrap = picNumLXPred + ( abs_diff_pic_num_minus1 + 1 ).
```

Значение picNumLXPred – это предсказанное значение переменной picNumLXNoWrap. Если процесс, определенный в этом пункте, активируют первый раз для секции (при первом появлении которой reordering_of_pic_nums_idc равно 0 или 1 в синтаксисе ref_pic_list_reordering( )), то picNumL0Pred и picNumL1Pred первоначально устанавливают равным CurrPicNum. После каждого присвоения picNumLXNoWrap значение picNumLXNoWrap присваивают значению picNumLXPred.

Переменную picNumLX находят следующим образом:

```
if( picNumLXNoWrap > CurrPicNum )
    picNumLX = picNumLXNoWrap – MaxPicNum                                          (8-37)
else
    picNumLX = picNumLXNoWrap.
```

Значение picNumLX должно быть равным PicNum контрольного изображения, которое помечено как "использованное для краткосрочного контроля" и не должно быть равным значению PicNum краткосрочного контрольного изображения, которое помечено как "несуществующее".

Следующая процедура должна быть выполнена, чтобы заменить изображение с номером краткосрочного изображения picNumLX на индекс позиции refIdxLX, сдвинуть позиции всех других оставшихся изображений дальше в списке и прибавить значение индекса refIdxLX:

```
for( cIdx = num_ref_idx_lX_active_minus1 + 1; cIdx > refIdxLX; cIdx-- )
    RefPicListX[ cIdx ] = RefPicListX[ cIdx – 1]
RefPicListX[ refIdxLX++ ] = значению краткосрочного контрольного изображения с PicNum, равным
picNumLX
nIdx = refIdxLX
for( cIdx = refIdxLX; cIdx <= num_ref_idx_lX_active_minus1 + 1; cIdx++ )           (8-38)
    if( PicNumF( RefPicListX[ cIdx ] ) != picNumLX )
        RefPicListX[ nIdx++ ] = RefPicListX[ cIdx ],
```

где функцию PicNumF(RefPicListX[cIdx]) находят следующим образом:

– Если изображение RefPicListX[cIdx] помечено как "использованное для краткосрочного контроля", то PicNumF(RefPicListX[cIdx]) – это значение PicNum изображения RefPicListX[cIdx].

– Иначе (изображение RefPicListX[cIdx] не помечено как "использованное для краткосрочного контроля"), PicNumF(RefPicListX[cIdx]) равно MaxPicNum.

ПРИМЕЧАНИЕ. – Значение MaxPicNum никогда не может быть равным picNumLX.

ПРИМЕЧАНИЕ. – В этой процедуре псевдокодирования длина списка RefPicListX временно сделана на один элемент длиннее, чем необходимая длина для окончательного списка. После выполнения этой процедуры следует сохранить только элементы этого списка от 0 до num_ref_idx_lX_active_minus1.

8.2.4.3.2 Процесс изменения порядка в списках контрольных изображений для долгосрочных изображений

Входы в этот процесс – список контрольных изображений RefPicListX (с X равным 0 или 1) и индекс refIdxLX в этом списке.

Выходы этого процесса – возможный модифицированный список контрольных изображений RefPicListX и возрастающий индекс refIdxLX.

Следующая процедура должна быть выполнена, чтобы заменить изображение с номером долгосрочного изображения long_term_pic_num на индекс позиции refIdxLX, сдвинуть позиции всех других оставшихся изображений дальше в списке и прибавить значение индекса refIdxLX:

```
for( cIdx = num_ref_idx_lX_active_minus1 + 1; cIdx > refIdxLX; cIdx-- )
    RefPicListX[ cIdx ] = RefPicListX[ cIdx – 1]
RefPicListX[ refIdxLX++ ] = значению долгосрочного контрольного изображения с LongTermPicNum,
равным long_term_pic_num
nIdx = refIdxLX
for( cIdx = refIdxLX; cIdx <= num_ref_idx_lX_active_minus1 + 1; cIdx++ )                    (8-39)
    if( LongTermPicNumF( RefPicListX[ cIdx ] ) != long_term_pic_num )
        RefPicListX[ nIdx++ ] = RefPicListX[ cIdx ],
```

где функцию LongTermPicNumF(RefPicListX[cIdx]) находят следующим образом:

– Если изображение RefPicListX[cIdx] помечено как "использованное для долгосрочного контроля", то LongTermPicNumF(RefPicListX[cIdx]) – это значение LongTermPicNum изображения RefPicListX[cIdx].

– Иначе (изображение RefPicListX[cIdx] не помечено как "использованное для долгосрочного контроля"), LongTermPicNumF(RefPicListX[cIdx]) равно 2 * (MaxLongTermFrameIdx + 1).

ПРИМЕЧАНИЕ. – Значение 2 * (MaxLongTermFrameIdx + 1) никогда не может быть равным long_term_pic_num.

ПРИМЕЧАНИЕ. – В этой процедуре псевдокодирования длина списка RefPicListX временно сделана на один элемент длиннее, чем необходимая длина для окончательного списка. После выполнения этой процедуры следует сохранить только элементы этого списка от 0 до num_ref_idx_lX_active_minus1.

8.2.5 Процесс разметки декодированного контрольного изображения

Этот процесс активируют для декодирования изображений, если nal_ref_idc не равно 0.

ПРИМЕЧАНИЕ. – Один описанный в этом пункте процесс (процесс декодирования промежутков в frame_num, определенный в п. 8.2.5.2) может быть также активирован, если nal_ref_idc равно 0, как определено в разделе 8.

Декодированное изображение с nal_ref_idc, не равным 0, отнесенное к контрольному изображению, помечено как "использованное для краткосрочного контроля" или "использованное для долгосрочного контроля". Оба поля декодированного контрольного кадра помечены так же, как кадр. Пара дополнительных контрольных полей помечена так же, как эти поля. Изображение, которое помечено как "использованное для краткосрочного контроля", идентифицировано его номером FrameNum и, если это – поле, то его четностью. Изображение, которое помечено как "использованное для долгосрочного контроля", идентифицировано его индексом LongTermFrameIdx и, если это – поле, то его четностью.

Кадры или дополнительная пара полей, помеченные как "использованные для краткосрочного контроля" или как "использованные для долгосрочного контроля", можно использовать при декодировании кадра как контрольные для внешнего предсказания до появления кадра, дополнительной пары полей или одной из составляющих поля, помеченных как "не использовано для контроля". Поле, помеченное как "использованное для краткосрочного контроля" или как "использованное для долгосрочного контроля", можно использовать при декодировании поля как контрольное для внешнего предсказания до появления поля, помеченного как "не использовано для контроля".

Изображение может быть помечено как "не использовано для контроля" с помощью скользящего окна в процессе разметки контрольного изображения, механизмом "первым пришел – первым обслужен", описанным в п. 8.2.5.3 или процессом разметки контрольного изображения с адаптивным управлением памятью, приспособленной к адаптивной операции разметки (п. 8.2.5.4).

Краткосрочное контрольное изображение идентифицировано для использования в процессе декодирования переменными FrameNum и FrameNumWrap и изображения PicNum, а долгосрочное контрольное изображение идентифицировано для использования в процессе декодирования номером долгосрочного изображения LongTermPicNum. Если текущее изображение – это не IDR изображение, то активируют процедуру п. 8.2.4.1, чтобы определить присвоение переменных FrameNum, FrameNumWrap, PicNum и LongTermPicNum.

8.2.5.1 Последовательность операций для процесса разметки декодированного контрольного изображения

Разметку декодированного контрольного изображения производят следующими шагами:

1. Если frame_num текущего изображения не равно PrevRefFrameNum и не равно (PrevRefFrameNum + 1) % MaxFrameNum, то процесс декодирования для промежутков в frame_num выполняют согласно п. 8.2.5.2.
2. Все секции текущего изображения декодированы.
3. В зависимости от того, является ли текущее изображение IDR изображением, используют следующие условия:
   - Если текущее изображение – это IDR изображение, используют следующие условия:
     - Все контрольные изображения должны быть помечены как "не использовано для контроля".
     - В зависимости от значения long_term_reference_flag, используют следующие условия:
       - Если long_term_reference_flag равно 0, IDR изображение должно быть помечено "использовано для краткосрочного контроля", а MaxLongTermFrameIdx должно быть установлено равным "нет индексов долгосрочных кадров".
       - Иначе (long_term_reference_flag равно 1), IDR изображение должно быть помечено как "использовано для долгосрочного контроля", LongTermFrameIdx для IDR изображения должно быть установлено равным 0 и MaxLongTermFrameIdx должно быть установлено равным 0.
   - Иначе (текущее изображение – не IDR изображение), используют следующие условия:
     - Если adaptive_ref_pic_marking_mode_flag равно 0, активируют процесс, определенный в п. 8.2.5.3.
     - Иначе (adaptive_ref_pic_marking_mode_flag равно 1), активируют процесс, определенный в п. 8.2.5.4.
4. Если текущее изображение – не IDR изображение и не было помечено как "использованное для долгосрочного контроля" значением memory_management_control_operation, равным 6, его отмечают как "использовано для краткосрочного контроля".

После разметки текущего декодированного контрольного изображения общее число кадров, по крайней мере, с одним полем, помеченным как "использовано для контроля", плюс число дополнительных пар полей, по крайней мере, с одним полем, помеченным как "использовано для контроля", плюс число непарных полей, помеченных как "использовано для контроля", не должно быть больше, чем Max(num_ref_frames, 1).

8.2.5.2 Процесс декодирования для промежутков в frame_num

Этот процесс активируют, если frame_num не равно PrevRefFrameNum и не равно (PrevRefFrameNum + 1) % MaxFrameNum.

ПРИМЕЧАНИЕ. – Хотя этот процесс определен в виде пункта внутри п. 8.2.5 (в котором описан процесс, который активируют, если только nal_ref_idc не равно 0), данный процесс можно также активировать, если nal_ref_idc равно 0 (как определено в разделе 8). Причины, по которым этот пункт помещен в структуру данной Рекомендации | Международного стандарта, – чисто исторические.

ПРИМЕЧАНИЕ. – Этот процесс можно активировать только для соответствующего потока битов, если gaps_in_frame_num_value_allowed_flag равно 1. Если gaps_in_frame_num_value_allowed_flag равно 0, а frame_num не равно PrevRefFrameNum и не равно (PrevRefFrameNum + 1) % MaxFrameNum, процесс декодирования следует считать непреднамеренной потерей изображений.

Если этот процесс активируют, набор значений frame_num, принадлежащих к "несуществующим" изображениям, находят как все полученные в равенстве 7-10 значения UnusedShortTermFrameNum, за исключением frame_num для текущего изображения.

Процесс декодирования должен создать и разметить кадр для каждого из его значений frame_num, принадлежащих к "несуществующим" изображениям в порядке, в котором значения UnusedShortTermFrameNum создаются равенством 7-10, используя "скользящее окно" изображения процесса разметки, как определено в п. 8.2.5.3. Созданные кадры должны быть также помечены как "несуществующие" и "использованные для краткосрочного контроля". Образец значений созданных кадров можно установить на любое значение. Эти созданные кадры, которые помечены как "несуществующие", не должны учитываться в процессе внешнего предсказания, не должны считаться командами изменения порядка в списках контрольных изображений для краткосрочных изображений (п. 8.2.4.3.1) и не должны учитываться в процессе присвоения LongTermFrameIdx краткосрочному изображению (п. 8.2.5.4.3).

Если pic_order_cnt_type не равно 0, TopFieldOrderCnt и BottomFieldOrderCnt находят для каждого "несуществующего" кадра, активируя процесс декодирования для вычисления порядка изображений в п. 8.2.1. Если активируют процесс в п. 8.2.1 для конкретного "несуществующего" кадра, текущее изображение считают изображением, имеющим значение frame_num, которое считают равным UnusedShortTermFrameNum, nal_ref_idc – равным 0, nal_unit_type – равным 5, field_pic_flag – равным 0, adaptive_ref_pic_marking_mode_flag – равным 0, delta_pic_order_cnt[0] (если необходимо) – равным 0 и delta_pic_order_cnt[1] (если необходимо) – равным 0.

ПРИМЕЧАНИЕ. – Процесс декодирования следует считать непреднамеренной потерей изображения, если любое из значений frame_num, относящееся к "несуществующим" изображениям, считают процессом внешнего предсказания, изменением порядка команд в списках контрольных изображений для краткосрочных изображений (п. 8.2.4.3.1) или полагают

принадлежащим к процессу присвоения LongTermFrameIdx краткосрочному изображению (п. 8.2.5.4.3). Процесс декодирования не следует считать непреднамеренной потерей изображения, если операцию управления памятью, не равную 3, применяют к кадру, помеченному как "несуществующий".

8.2.5.3 Скользящее окно процесса разметки декодированного контрольного изображения

Этот процесс активируют, если adaptive_ref_pic_marking_mode_flag равно 0.

В зависимости от свойств текущего изображения, как будет описано ниже, используют следующие условия:

– Если текущее изображение – кодированное поле, то есть второе поле в порядке декодирования дополнительной контрольной пары полей, а первое поле помечено как "использованное для краткосрочного контроля", то текущее изображение также помечают как "использованное для краткосрочного контроля".

  Иначе используют следующие условия:

  – Допустим, что numShortTerm – общее число контрольных кадров, дополнительной контрольной пары полей и непарных контрольных полей, для которых, по крайней мере, одно поле помечено как "использованное для краткосрочного контроля". Допустим, что numLongTerm – общее число контрольных кадров, дополнительной контрольной пары полей и непарных контрольных полей, для которых, по крайней мере, одно поле помечено как "использованное для долгосрочного контроля".

  – Если numShortTerm + numLongTerm равно Max(num_ref_frames, 1), условие, что numShortTerm больше 0, должно быть выполнено, а краткосрочный контрольный кадр, дополнительная контрольная пара полей или непарное контрольное поле, которое имеет наименьшее значение FrameNumWrap, – помечают как "не использовано для контроля". Если это – кадр или дополнительная пара полей, то оба эти поля также помечают как "не использовано для контроля".

8.2.5.4 Процесс разметки декодированного контрольного изображения с адаптивным управлением памятью

Этот процесс активируют, если adaptive_ref_pic_marking_mode_flag равно 1.

Команды memory_management_control_operation со значениями от 1 до 6 обрабатывают в порядке их поступления в потоке битов после того, как декодировано текущее изображение. Для каждой из команд memory_management_control_operation активируют один из процессов, определенных в пп. 8.2.5.4.1–8.2.5.4.6, в зависимости от значения memory_management_control_operation. Команда memory_management_control_operation со значением 0 определяет конец команд memory_management_control_operation.

Операции управления памятью применяют к изображениям следующим образом:

– Если field_pic_flag равно 0, команды memory_management_control_operation применяют к определенным кадрам или дополнительным контрольным парам полей.

– Иначе (field_pic_flag равно 1), команды memory_management_control_operation применяют к отдельным определенным контрольным полям.

8.2.5.4.1 Процесс разметки краткосрочного изображения как "не использованного для контроля"

Этот процесс активируют, если memory_management_control_operation равно 1.

Допустим, что picNumX определено как

$$picNumX = CurrPicNum - (difference\_of\_pic\_nums\_minus1 + 1). \quad (8\text{-}40)$$

В зависимости от field_pic_flag значение picNumX используют для разметки краткосрочных изображений как "не использованных для контроля" следующим образом:

– Если field_pic_flag равно 0, краткосрочный контрольный кадр или краткосрочную дополнительную контрольную пару полей, определенных picNumX, и оба этих поля помечают как "не использованные для контроля".

– Иначе (field_pic_flag равно 1), краткосрочное контрольное поле, определенное picNumX, помечают как "не использованное для контроля". Если это контрольное поле – часть контрольного кадра или дополнительной контрольной пары полей, то кадр или дополнительную пару полей также помечают как "не использованные для контроля", но разметку другого поля не меняют.

8.2.5.4.2 Процесс разметки долгосрочного изображения как "не использованного для контроля"

Этот процесс активируют, если memory_management_control_operation равно 2.

В зависимости от field_pic_flag используют значение LongTermPicNum для разметки долгосрочного изображения как "не использованного для контроля" следующим образом:

– Если field_pic_flag равно 0, долгосрочный контрольный кадр или долгосрочную дополнительную контрольную пару полей, имеющих LongTermPicNum, равное long_term_pic_num, и оба этих поля помечают как "не использованные для контроля".

– Иначе (field_pic_flag равно 1), долгосрочное контрольное поле, определенное LongTermPicNum, равным long_term_pic_num, помечают как "не использовано для контроля". Если это контрольное поле – часть контрольного кадра или дополнительной контрольной пары полей, то кадр или дополнительную пару полей также помечают как "не использованные для контроля", но разметку другого поля не меняют.

8.2.5.4.3 Процесс присвоения значения LongTermFrameIdx краткосрочному контрольному изображению

Этот процесс активируют, если memory_management_control_operation равно 3.

Для заданного элемента синтаксиса difference_of_pic_nums_minus1 переменную picNumX находят, как определено в п. 8.2.5.4.1. Значение picNumX должно относиться к кадру или к дополнительной контрольной паре полей или непарных контрольных полей, помеченных как "использованные для краткосрочного контроля" и не помеченных как "несуществующие".

Если LongTermFrameIdx, равное long_term_frame_idx, уже присвоено долгосрочному контрольному кадру или долгосрочной дополнительной контрольной паре полей, то этот кадр или дополнительная пара полей и оба поля помечают как "не использованные для контроля". Если LongTermFrameIdx уже присвоено непарному контрольному полю, и это поле не является дополнительным полем изображения, определенного picNumX, то это поле помечают как "не использованное для контроля".

В зависимости от field_pic_flag значение LongTermFrameIdx используют для разметки изображения от "использованные для краткосрочного контроля" до "использованные для долгосрочного контроля" следующим образом:

– Если field_pic_flag равно 0, разметку краткосрочного контрольного кадра или краткосрочной дополнительной контрольной пары полей, определенных picNumX, а также обоих этих полей меняют с "использованные для краткосрочного контроля" на "использованные для долгосрочного контроля" и присваивают LongTermFrameIdx значение long_term_frame_idx.
– Иначе (field_pic_flag равно 1), разметку краткосрочного контрольного поля, определенного picNumX, меняют с "использованные для краткосрочного контроля" на "использованные для долгосрочного контроля" и присваивают LongTermFrameIdx значение long_term_frame_idx.

8.2.5.4.4 Процесс декодирования MaxLongTermFrameIdx

Этот процесс активируют, если memory_management_control_operation равно 4.

Все изображения, для которых LongTermFrameIdx больше, чем max_long_term_frame_idx_plus1 – 1, и которые помечены как "использованные для долгосрочного контроля", должны быть помечены как "не использованные для контроля".

Переменную MaxLongTermFrameIdx находят следующим образом:

– Если max_long_term_frame_idx_plus1 равно 0, MaxLongTermFrameIdx должно быть установлено равным "нет индексов долгосрочных кадров".
– Иначе (max_long_term_frame_idx_plus1 больше, чем 0), MaxLongTermFrameIdx должно быть установлено равным max_long_term_frame_idx_plus1 – 1.

ПРИМЕЧАНИЕ. – Команда memory_management_control_operation, равная 4, может быть использована для разметки долгосрочных контрольных изображений как "не использованных для контроля". Частота передачи значения max_long_term_frame_idx_plus1 не определена данной Рекомендацией | Международным стандартом. Однако кодеру следует отправлять команду memory_management_control_operation, равную 4, до получения сообщения ошибки, как, например, сообщения о запросе на внутреннее обновление.

8.2.5.4.5 Процесс разметки всех контрольных изображений как "не использованных для контроля" и установка MaxLongTermFrameIdx на значение "нет индексов долгосрочных кадров"

Этот процесс активируют, если memory_management_control_operation равно 5.

Все контрольные изображения помечены как "не использованные для контроля", а переменную MaxLongTermFrameIdx устанавливают равной "нет индексов долгосрочных кадров".

8.2.5.4.6 Процесс присвоения текущему изображению индекса долгосрочного кадра

Этот процесс активируют, если memory_management_control_operation равно 6.

Если переменная LongTermFrameIdx, равная long_term_frame_idx, уже присвоена долгосрочному контрольному кадру или долгосрочной дополнительной контрольной паре полей, то этот кадр или дополнительная пара полей и оба этих поля помечают как "не использованные для контроля". Если LongTermFrameIdx уже присвоено непарному контрольному полю, а это поле не является дополнительным для текущего изображения, то это поле помечают как "не использованное для контроля".

Текущее изображение помечено как "использованное для долгосрочного контроля", и ему присваивают индекс LongTermFrameIdx, равный long_term_frame_idx.

Если field_pic_flag равно 0, оба поля помечены как "использованные для долгосрочного контроля" и им присваивают индекс LongTermFrameIdx, равный long_term_frame_idx.

Если field_pic_flag равно 1, а текущее изображение – второе (в порядке декодирования) поле дополнительной контрольной пары полей, эта пара также помечена как "использованная для долгосрочного контроля" и ей присваивают индекс LongTermFrameIdx, равный long_term_frame_idx.

После разметки текущего декодированного контрольного изображения общее число кадров, по крайней мере, с одним полем, помеченным как "использованное для контроля", плюс число дополнительных пар полей, по крайней мере, с одним полем, помеченным как "использованное для контроля", плюс число непарных полей, помеченных как "использованные для контроля", не должно превышать Max(num_ref_frames, 1).

ПРИМЕЧАНИЕ. – При некоторых обстоятельствах вышеуказанное утверждение может накладывать ограничение на порядок, в котором элемент синтаксиса memory_management_control_operation, равный 6, может появиться в декодированном контрольном изображении, помечая синтаксис относительно элемента синтаксиса со значением memory_management_control_operation, равным 1, 2 или 4.

8.3 Процесс внутреннего предсказания (Intra предсказания)

Этот процесс активируют для типов макроблоков I и SI.

Входы в этот процесс создают образцы перед процессом фильтрового разделения на блоки смежных макроблоков и предсказания для режима Intra_4x4, связанного со значениями Intra4x4PredMode от смежных макроблоков.

Выходы этого процесса определены следующим образом:

– Если mb_type не равно I_PCM, то это – образцы Intra предсказания компонентов макроблока, или в случае процесса предсказания образцов яркости Intra_4x4 выходами будут массивы образцов яркости 4x4 как части массива яркости 16x16 предсказания образцов макроблока.

– Иначе (mb_type равно I_PCM) – созданные образцы до процесса фильтрового разделения на блоки смежных макроблоков.

Переменную MvCnt устанавливают равной 0.

В зависимости от значения mb_type используют следующие условия:

– Если mb_type равно I_PCM, активируют процесс, определенный в п. 8.3.4.

– Иначе (mb_type не равно I_PCM), используют следующие условия:

  – Процесс декодирования в режимах Intra предсказаний описывают для компонентов яркости следующим образом:

    – Если режим предсказания макроблока равен Intra_4x4, применяют спецификацию п. 8.3.1.

    – Иначе (режим предсказания макроблока равен Intra_16x16), применяют спецификацию п. 8.3.2.

  – Процесс декодирования в режимах Intra предсказаний для компонентов цветности описан в п. 8.3.3.

Образцы, использованные в процессе Intra предсказания, должны иметь значения образцов до их изменения любыми операциями фильтрового разделения на блоки.

8.3.1 Процесс предсказания образцов яркости Intra_4x4

Этот процесс активируют, если режим предсказания макроблока равен Intra_4x4.

Входы в этот процесс – созданные образцы яркости до процесса фильтрового разделения смежных макроблоков и связанных с ними значений Intra4x4PredMode от других смежных макроблоков или пар макроблоков.

Выходы этого процесса – массивы образцов яркости 4x4 как части массивов яркости 16x16 предсказанных образцов макроблока $pred_L$.

Компонент яркости макроблока состоит из 16 блоков 4x4 образцов яркости. Эти блоки – инверсно сканированные, с использованием процесса инверсного сканирования блока яркости 4x4, как определено в п. 6.4.3.

Для всех блоков яркости 4x4 с компонентом яркости макроблока luma4x4BlkIdx = 0..15 переменную Intra4x4PredMode[luma4x4BlkIdx] находят, как определено в п. 8.3.1.1.

Образцы каждого блока яркости 4x4 проиндексированы с использованием luma4x4BlkIdx = 0..15,

1. Активируют процесс предсказания образца Intra_4x4 по п. 8.3.1.2 с помощью luma4x4BlkIdx и создают образцы до (в порядке декодирования) процесса фильтрового разделения на блоки смежных блоков яркости в качестве входа и образцами предсказания яркости Intra_4x4 $pred4x4_L[x, y]$ с x, y = 0..3 на выходе.

2. Положение верхнего левого образца блока яркости 4x4 с индексом luma4x4BlkIdx внутри текущего макроблока находят, активируя процесс инверсного сканирования блока яркости 4x4 в п. 6.4.3 с luma4x4BlkIdx в качестве входа и присвоением (xO, yO) с x, y = 0..3 на выходе.

$$pred_L[xO + x, yO + y] = pred4x4_L[x, y] \quad (8\text{-}41)$$

3. Процесс декодирования коэффициентов преобразования и процесс создания изображения по п. 8.5 активируют до процесса фильтрового разделения на блоки с $pred_L$ и luma4x4BlkIdx в качестве входа и созданными образцами текущего блока яркости 4x4 luma block S'_L в качестве выхода.

8.3.1.1 Процесс отыскания Intra4x4PredMode

Входы в этот процесс – это индекс блока яркости 4x4luma4x4BlkIdx и массивы переменных Intra4x4PredMode, которые предварительно (в порядке декодирования) находят для смежных макроблоков.

Выход этого процесса – переменная Intra4x4PredMode[luma4x4BlkIdx].

В таблице 8-2 показаны значения Intra4x4PredMode[luma4x4BlkIdx] и связанные с ними имена.

Таблица 8-2 – Спецификация Intra4x4PredMode[luma4x4BlkIdx] и связанные с этим имена

Intra4x4PredMode[luma4x4BlkIdx]	Имя Intra4x4PredMode[luma4x4BlkIdx]
0	Intra_4x4_Vertical (режим предсказания)
1	Intra_4x4_Horizontal (режим предсказания)
2	Intra_4x4_DC (режим предсказания)
3	Intra_4x4_Diagonal_Down_Left (режим предсказания)
4	Intra_4x4_Diagonal_Down_Right (режим предсказания)
5	Intra_4x4_Vertical_Right (режим предсказания)
6	Intra_4x4_Horizontal_Down (режим предсказания)
7	Intra_4x4_Vertical_Left (режим предсказания)
8	Intra_4x4_Horizontal_Up (режим предсказания)

Intra4x4PredMode[luma4x4BlkIdx], обозначенные как 0, 1, 3, 4, 5, 6, 7 и 8, представляют направления предсказаний, как показано на рисунке 8-1.

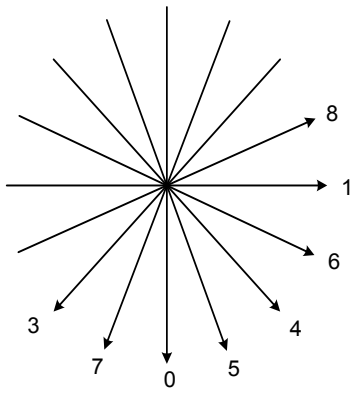

Рисунок 8-1 – Направления режимов предсказаний Intra_4x4 (информативное)

Допустим, что intra4x4PredModeA и intra4x4PredModeB – переменные, которые определяют режим внутреннего предсказания смежных блоков яркости 4x4.

Intra4x4PredMode[luma4x4BlkIdx] находят следующим образом:

– Процесс, определенный в п. 6.4.7.3, активируют с luma4x4BlkIdx, заданным как вход, а выход присвоен значениям mbAddrA, luma4x4BlkIdxA, mbAddrB и luma4x4BlkIdxB.

– Переменную dcOnlyPredictionFlag находят следующим образом:

  – Если одно из следующих условий – истина, то устанавливают dcOnlyPredictionFlag равным 1:

– макроблок с адресом mbAddrA не доступен;

– макроблок с адресом mbAddrB не доступен;

– макроблок с адресом mbAddrA доступен и кодирован в режиме Inter предсказания с constrained_intra_pred_flag, равным 1;

– макроблок с адресом mbAddrB доступен и кодирован в режиме Inter предсказания с constrained_intra_pred_flag, равным 1.

– Иначе dcOnlyPredictionFlag установлено равным 0.

– Для значения N, которое заменяют любым значением A или B, переменные intra4x4PredModeN находят следующим образом:

– Если dcOnlyPredictionFlag равно 1 или макроблок с адресом mbAddrN не кодирован в режиме предсказания макроблока Intra_4x4, то intra4x4PredModeN устанавливают равным 2 (режим предсказания Intra_4x4_DC).

– Иначе (dcOnlyPredictionFlag равно 0 и макроблок с адресом mbAddrN кодирован в режиме предсказания макроблока Intra_4x4), intra4x4PredModeN устанавливают равным Intra4x4PredMode[luma4x4BlkIdxN], где Intra4x4PredMode – массив переменных, присвоенных макроблоку mbAddrN.

– Intra4x4PredMode[luma4x4BlkIdx] находят, применяя следующую процедуру:

```
predIntra4x4PredMode = Min( intra4x4PredModeA, intra4x4PredModeB )
if( prev_intra4x4_pred_mode_flag[ luma4x4BlkIdx ] )
    Intra4x4PredMode[ luma4x4BlkIdx ] = predIntra4x4PredMode
else                                                                    (8-42)
    if( rem_intra4x4_pred_mode[ luma4x4BlkIdx ] < predIntra4x4PredMode )
      Intra4x4PredMode[ luma4x4BlkIdx ] = rem_intra4x4_pred_mode[ luma4x4BlkIdx ]
    else
      Intra4x4PredMode[ luma4x4BlkIdx ] = rem_intra4x4_pred_mode[ luma4x4BlkIdx ] + 1.
```

8.3.1.2 Предсказание образца Intra_4x4

Этот процесс активируют для каждого блока яркости 4x4 макроблока с режимом предсказания, равным Intra_4x4, за которым следует преобразование процесса декодирования и процесс создания изображения перед разделением на блоки яркости 4x4.

Входы в этот процесс – индекс блока яркости 4x4 с индексом luma4x4BlkIdx и созданные образцы (в порядке декодирования) перед процессом фильтрового разделения на блоки смежных блоков яркости.

Выход этого процесса – предсказанные образцы $pred4x4_L[x, y]$, с x, y = 0..3 для блока яркости 4x4 с индексом luma4x4BlkIdx.

Положение верхнего левого образца блока яркости 4x4 с индексом luma4x4BlkIdx внутри текущего макроблока находят, активируя процесс инверсного сканирования блока яркости 4x4 по п. 6.4.3 с luma4x4BlkIdx в качестве входа и присвоением (xO, yO) в качестве выхода.

13 смежных образцов p[x, y], которые являются созданными образцами яркости перед процессом фильтрового разделения на блоки с x = –1, y = –1..3 и x = 0..7, y = –1, находят следующим образом:

– Положение яркости (xN, yN) определено равенствами:

$$xN = xO + x \tag{8-43}$$

$$yN = yO + y. \tag{8-44}$$

– Процесс отыскания смежных положений по п. 6.4.8 активируют для расположений яркости с (xN, yN) как вход, и mbAddrN и (xW, yW) как выход.

– Каждый образец p[x, y] с x = –1, y = –1..3 и x = 0..7, y = –1 находят следующим образом:

– Если любое из следующих условий – истина, то образец p[x, y] помечен как "не доступно для предсказания Intra_4x4":

– mbAddrN не доступно;

– макроблок mbAddrN кодирован в режиме Inter предсказания, а constrained_intra_pred_flag равно 1;

– тип mb_type макроблока mbAddrN равен SI с constrained_intra_pred_flag, равным 1, а текущий макроблок не имеет mb_type, равного SI;

– x больше 3, а luma4x4BlkIdx равно 3 или 11.

– Иначе образец p[x, y] помечен "доступно для предсказания Intra_4x4", а образцу яркости в положении яркости (xW, yW) внутри макроблока mbAddrN присвоено значение p[x, y].

Если образцы p[x, –1] с x = 4..7 помечены как "не доступно для предсказания Intra_4x4", а образец p[3, –1] помечен как "доступно для предсказания Intra_4x4," то образец со значением p[3, –1] заменяют на образец со значением p[x, –1] с x = 4..7, а образцы p[x, –1] с x = 4..7 помечены как "доступно для предсказания Intra_4x4".

ПРИМЕЧАНИЕ. – Считают, что каждый блок создан в кадре до декодирования следующего блока.

В зависимости от Intra4x4PredMode[luma4x4BlkIdx] должен быть использован один из режимов предсказаний Intra_4x4, определенный в пп. 8.3.1.2.1–8.3.1.2.9.

8.3.1.2.1 Спецификация режима предсказания Intra_4x4_Vertical

Режим предсказания Intra_4x4 должен быть использован, если Intra4x4PredMode[luma4x4BlkIdx] равно 0.

Этот режим должен быть использован, если только образцы p[x, –1] с x = 0..3 помечены как "доступно для предсказания Intra_4x4".

Значения образцов предсказания $pred4x4_L$[x, y] с x, y = 0..3 находят следующим образом:

$$pred4x4_L[x, y] = p[x, -1] \text{ с } x, y = 0..3. \quad (8\text{-}45)$$

8.3.1.2.2 Спецификация режима предсказания Intra_4x4_Horizontal

Режим предсказания Intra_4x4 должен быть использован, если Intra4x4PredMode[luma4x4BlkIdx] равно 1.

Этот режим должен быть использован, если только образцы p[–1, y] с y = 0..3 помечены как "доступно для предсказания Intra_4x4".

Значения образцов предсказания $pred4x4_L$[x, y] с x, y = 0..3 находят следующим образом:

$$pred4x4_L[x, y] = p[-1, y] \text{ с } x,y = 0..3. \quad (8\text{-}46)$$

8.3.1.2.3 Спецификация режима предсказания Intra_4x4_DC

Режим предсказания Intra_4x4 должен быть использован, если Intra4x4PredMode[luma4x4BlkIdx] равно 2.

Значения предсказания образцов $pred4x4_L$[x, y] с x, y = 0..3 находят следующим образом:

– Если все образцы p[x, –1] с x = 0..3 и p[–1, y] с y = 0..3 помечены как "доступно для предсказания Intra_4x4", то значения предсказания образцов $pred4x4_L$[x, y] с x, y = 0..3 находят следующим образом:

$$pred4x4_L[x, y] = (p[0, -1] + p[1, -1] + p[2, -1] + p[3, -1] + p[-1, 0] + p[-1, 1] + p[-1, 2] + p[-1, 3] + 4) >> 3. \quad (8\text{-}47)$$

– Иначе, если имеются образцы p[x, -1] с x = 0..3, помеченные как "не доступно для предсказания Intra_4x4", и все образцы p[–1, y] с y = 0..3, помеченные как "доступно для предсказания Intra_4x4", то значения предсказания образцов $pred4x4_L$[x, y] с x, y = 0..3 находят следующим образом:

$$pred4x4_L[x, y] = (p[-1, 0] + p[-1, 1] + p[-1, 2] + p[-1, 3] + 2) >> 2. \quad (8\text{-}48)$$

– Иначе, если имеются образцы p[-1, y] с y = 0..3, помеченные как "не доступно для предсказания Intra_4x4", и все образцы p[x, –1] с x = 0 .. 3, помеченные как "доступно для предсказания Intra_4x4", то значения образцов предсказания $pred4x4_L$[x, y] с x, y = 0 .. 3 находят следующим образом:

$$pred4x4_L[x, y] = (p[0, -1] + p[1, -1] + p[2, -1] + p[3, -1] + 2) >> 2. \quad (8\text{-}49)$$

– Иначе (некоторые образцы p[x, –1] с x = 0..3 и некоторые образцы p[–1, y] с y = 0..3, помеченные как "не доступно для предсказания Intra_4x4"), значения образцов предсказания $pred4x4_L$[x, y] с x, y = 0..3 находят следующим образом:

$$pred4x4_L[x, y] = 128. \quad (8\text{-}50)$$

ПРИМЕЧАНИЕ. – Блок яркости 4x4 можно всегда предсказать, используя этот режим.

8.3.1.2.4 Спецификация режима предсказания Intra_4x4_Diagonal_Down_Left

Режим предсказания Intra_4x4 должен быть использован, если Intra4x4PredMode[luma4x4BlkIdx] равно 3.

Этот режим должен быть использован, если только образцы p[x, –1] с x = 0..7 помечены как "доступно для предсказания Intra_4x4".

Значения образцов предсказания $pred4x4_L$[x, y] с x, y = 0..3 находят следующим образом:

– Если x равно 3 и y равно 3,

$$pred4x4_L[x, y] = (p[6, –1] + 3 * p[7, –1] + 2) >> 2. \quad (8\text{-}51)$$

– Иначе (x не равно 3 или y не равно 3),

$$pred4x4_L[x, y] = (p[x + y, –1] + 2 * p[x + y + 1, –1] + p[x + y + 2, –1] + 2) >> 2. \quad (8\text{-}52)$$

8.3.1.2.5 Спецификация режима предсказания Intra_4x4_Diagonal_Down_Right

Режим предсказания Intra_4x4 должен быть использован, если Intra4x4PredMode[luma4x4BlkIdx] равно 4.

Этот режим должен быть использован, если только образцы p[x, –1] с x = 0..3 и p[–1, y] с y = –1..3 помечены как "доступно для предсказания Intra_4x4".

Значения образцов предсказания $pred4x4_L$[x, y] с x, y = 0..3 находят следующим образом:

– Если x больше, чем y,

$$pred4x4_L[x, y] = (p[x – y – 2, –1] + 2 * p[x – y – 1, –1] + p[x – y, –1] + 2) >> 2. \quad (8\text{-}53)$$

– Иначе, если x меньше, чем y,

$$pred4x4_L[x, y] = (p[–1, y – x – 2] + 2 * p[–1, y – x – 1] + p[–1, y – x] + 2) >> 2. \quad (8\text{-}54)$$

– Иначе (x равно y)

$$pred4x4_L[x, y] = (p[0, –1] + 2 * p[–1, –1] + p[–1, 0] + 2) >> 2. \quad (8\text{-}55)$$

8.3.1.2.6 Спецификация режима предсказания Intra_4x4_Vertical_Right

Режим предсказания Intra_4x4 должен быть использован, если Intra4x4PredMode[luma4x4BlkIdx] равно 5.

Этот режим должен быть использован, если только образцы p[x, –1] с x = 0..3 и p[–1, y] с y = –1..3 помечены как "доступно для предсказания Intra_4x4".

Допустим, что переменная zVR установлена равной 2 * x – y.

Значения образцов предсказания $pred4x4_L$[x, y] с x, y = 0..3 находят следующим образом:

– Если zVR равно 0, 2, 4 или 6,

$$pred4x4_L[x, y] = (p[x – (y >> 1) – 1, -1] + p[x – (y >> 1), –1] + 1) >> 1. \quad (8\text{-}56)$$

– Иначе, если zVR равно 1, 3 или 5,

$$pred4x4_L[x, y] = (p[x – (y >> 1) – 2, -1] + 2 * p[x – (y >> 1) – 1, –1] + p[x – (y >> 1), –1] + 2) >> 2. \quad (8\text{-}57)$$

– Иначе, если zVR равно –1,

$$pred4x4_L[x, y] = (p[–1, 0] + 2 * p[–1, –1] + p[0, –1] + 2) >> 2. \quad (8\text{-}58)$$

– Иначе (zVR равно –2 или –3),

$$pred4x4_L[x, y] = (p[–1, y – 1] + 2 * p[-1, y – 2] + p[–1, y – 3] + 2) >> 2. \quad (8\text{-}59)$$

8.3.1.2.7 Спецификация режима предсказания Intra_4x4_Horizontal_Down

Режим предсказания Intra_4x4 должен быть использован, если Intra4x4PredMode[luma4x4BlkIdx] равно 6.

Этот режим должен быть использован, если только образцы p[x, –1] с x = 0..3 и p[–1, y] с y = –1..3 помечены как "доступно для предсказания Intra_4x4".

Допустим, что переменная zHD установлена равной 2 * y – x.

Значения образцов предсказания $pred4x4_L[x, y]$ с x, y = 0..3 находят следующим образом:

– Если zHD равно 0, 2, 4 или 6,

$$pred4x4_L[x, y] = (p[-1, y -(x >> 1) - 1] + p[-1, y - (x >> 1)] + 1) >> 1. \quad (8\text{-}60)$$

– Иначе, если zHD равно 1, 3 или 5,

$$pred4x4_L[x, y] = (p[-1, y - (x >> 1) - 2] + 2 * p[-1, y - (x >> 1) - 1] + p[-1, y - (x >> 1)] + 2) >> 2. \quad (8\text{-}61)$$

– Иначе, если zHD равно –1

$$pred4x4_L[x, y] = (p[-1, 0] + 2 * p[-1, -1] + p[0, -1] + 2) >> 2. \quad (8\text{-}62)$$

– Иначе (zHD равно –2 или –3),

$$pred4x4_L[x, y] = (p[x - 1, -1] + 2 * p[x - 2, -1] + p[x - 3, -1] + 2) >> 2. \quad (8\text{-}63)$$

8.3.1.2.8 Спецификация режима предсказания Intra_4x4_Vertical_Left

Режим предсказания Intra_4x4 должен быть использован, если Intra4x4PredMode[luma4x4BlkIdx] равно 7.

Этот режим должен быть использован, если только образцы p[x, –1] с x = 0..7 помечены как "доступно для предсказания Intra_4x4".

Значения образцов предсказания $pred4x4_L[x, y]$ с x, y = 0..3 находят следующим образом:

– Если y равно 0 или 2,

$$pred4x4_L[x, y] = (p[x + (y >> 1), -1] + p[x + (y >> 1) + 1, -1] + 1) >> 1. \quad (8\text{-}64)$$

– Иначе (y равно 1 или 3),

$$pred4x4_L[x, y] = (p[x + (y >> 1), -1] + 2 * p[x + (y >> 1) + 1, -1] + p[x + (y >> 1) + 2, -1] + 2) >> 2. \quad (8\text{-}65)$$

8.3.1.2.9 Спецификация режима предсказания Intra_4x4_Horizontal_Up

Режим предсказания Intra_4x4 должен быть использован, если Intra4x4PredMode[luma4x4BlkIdx] равно 8.

Этот режим должен быть использован, если только образцы p[–1, y] с y = 0..3 помечены как "доступно для предсказания Intra_4x4".

Допустим, что переменная zHU установлена равной x + 2 * y.

Значения образцов предсказания $pred4x4_L[x, y]$ с x, y = 0..3 находят следующим образом:

– Если zHU равно 0, 2 или 4,

$$pred4x4_L[x, y] = (p[-1, y + (x >> 1)] + p[-1, y + (x >> 1) + 1] + 1) >> 1. \quad (8\text{-}66)$$

– Иначе, если zHU равно 1 или 3,

$$pred4x4_L[x, y] = (p[-1, y + (x >> 1)] + 2 * p[-1, y + (x >> 1) + 1] + p[-1, y + (x >> 1) + 2] + 2) >> 2. \quad (8\text{-}67)$$

– Иначе, если zHU равно 5,

$$pred4x4_L[x, y] = (p[-1, 2] + 3 * p[-1, 3] + 2) >> 2. \quad (8\text{-}68)$$

– Иначе (zHU больше 5),

$$pred4x4_L[x, y] = p[-1, 3]. \quad (8\text{-}69)$$

8.3.2 Процесс предсказания для образцов яркости Intra_16x16

Процесс активируют, если режим предсказания макроблока равен Intra_16x16. Этот режим определяет, как находят образцы яркости текущего макроблока при Intra предсказании.

Входами в этот процесс являются образцы, созданные до процесса разделения на блоки – отделения блоков от смежных блоков яркости (если такие доступны).

Выходы этого процесса – Intra предсказания образцов яркости для текущего макроблока $pred_L[x, y]$.

33 смежных образца p[x, y], которые создают образцы яркости до процесса фильтрового разделения на блоки с x = –1, y = –1..15 и с x = 0..15, y = –1, находят следующим образом:

– Активируют процесс отыскания смежных положений по п. 6.4.8 для положений яркости с (x, y), присвоенным значению (xN, yN) как входу, а mbAddrN и (xW, yW) – как выходу.

– Каждый образец p[x, y] с x = –1, y = –1..15 с x = 0..15, y = –1 находят следующим образом:

  – Если любое из следующих условий – истина, образец p[x, y] помечен как "не доступен для предсказания Intra_16x16":

    – mbAddrN не доступен;

    – макроблок mbAddrN кодирован в режиме Inter предсказания и ограничен constrained_intra_pred_flag, равным 1;

    – макроблок mbAddrN имеет mb_type, равное SI, и ограничен constrained_intra_pred_flag, равным 1.

  – Иначе образец p[x, y] помечен как "доступен для предсказания Intra_16x16", а образцу яркости в положении яркости (xW, yW) внутри макроблока mbAddrN присвоено значение p[x, y].

Предположим, что $pred_L[x, y]$ с x, y = 0..15 указывает на образцы предсказания для блока образцов яркости 16x16.

Режим предсказаний Intra_16x16 определен в таблице 8-3.

Таблица 8-3 – Спецификация Intra16x16PredMode и связанные с этим имена

Intra16x16PredMode	Имя Intra16x16PredMode
0	Intra_16x16_Vertical (режим предсказания)
1	Intra_16x16_Horizontal (режим предсказания)
2	Intra_16x16_DC (режим предсказания)
3	Intra_16x16_Plane (режим предсказания)

В зависимости от Intra16x16PredMode должен быть использован один из режимов предсказаний Intra_16x16, определенный в пп. 8.3.2.1–8.3.2.4.

8.3.2.1 Спецификация режима предсказания Intra_16x16_Vertical

Этот режим предсказания Intra_16x16 должен быть использован, если образцы p[x, –1] с x = 0..15 помечены как "доступно для предсказания Intra_16x16".

$$pred_L[x, y] = p[x, -1] \text{ с } x, y = 0..15 \tag{8-70}$$

8.3.2.2 Спецификация режима предсказания Intra_16x16_Horizontal

Этот режим предсказания Intra_16x16 должен быть использован, если только образцы p[–1, y] с y = 0..15 помечены как "доступно для предсказания Intra_16x16".

$$pred_L[x, y] = p[-1, y] \text{ с } x, y = 0..15 \tag{8-71}$$

8.3.2.3 Спецификация режима предсказания Intra_16x16_DC

Этот режим предсказания Intra_16x16 должен быть использован в зависимости от того, помечены ли смежные образцы как " доступно для предсказания Intra_16x16", следующим образом:

– Если все смежные образцы p[x, –1] с x = 0..15 и p[–1, y] с y = 0..15 помечены как " доступно для предсказания Intra_16x16", предсказания для всех образцов яркости в макроблоке заданы равенством:

$$pred_L[x, y] = \left(\sum_{x'=0}^{15} p[x', -1] + \sum_{y'=0}^{15} p[-1, y'] + 16\right) >> 5 \text{ , с } x, y = 0..15. \tag{8-72}$$

– Иначе, если имеются смежные образцы p[x, –1] с x = 0..15, помеченные как "не доступно для предсказания Intra_16x16 ", и все смежные образцы p[–1, y] с y = 0..15, помеченные как "доступно для предсказания Intra_16x16", то предсказания для всех образцов яркости в макроблоке заданы равенством:

$$pred_L[\ x, y\ ] = (\sum_{y'=0}^{15} p[-1, y'] + 8) >> 4 \text{ , с } x, y = 0..15. \quad (8\text{-}73)$$

– Иначе, если имеются смежные образцы p[–1, y] с y = 0..15, помеченные как "не доступно для предсказания Intra_16x16", и все смежные образцы p[x, –1] с x = 0..15, помеченные как "доступно для предсказания Intra_16x16", предсказания для всех образцов яркости в макроблоке заданы равенством:

$$pred_L[\ x, y\ ] = (\sum_{x'=0}^{15} p[x', -1] + 8) >> 4 \text{ , с } x, y = 0..15. \quad (8\text{-}74)$$

– Иначе (некоторые из смежных образцов p[x, –1] с x = 0..15 и некоторые из смежных образцов p[–1, y] с y = 0..15 помечены как "не доступно для предсказания Intra_16x16"), предсказания для всех образцов яркости в макроблоке заданы равенством:

$$pred_L[\ x, y\ ] = 128, \text{ с } x, y = 0..15. \quad (8\text{-}75)$$

8.3.2.4 Спецификация режима предсказания Intra_16x16_Plane

Этот режим предсказания Intra_16x16 должен быть использован только в том случае, если образцы p[x, –1] с x = –1..15 и p[–1, y] с y = 0..15 помечены как "доступно для предсказания Intra_16x16".

$$pred_L[\ x, y\ ] = Clip1(\ (\ a + b * (\ x - 7\ ) + c * (\ y - 7\ ) + 16\ ) >> 5\ ) \text{ с } x, y = 0..15, \quad (8\text{-}76)$$

где:

$$a = 16 * (\ p[\ -1, 15\ ] + p[\ 15, -1\ ]\ ) \quad (8\text{-}77)$$

$$b = (\ 5 * H + 32\ ) >> 6 \quad (8\text{-}78)$$

$$c = (\ 5 * V + 32\ ) >> 6, \quad (8\text{-}79)$$

а H и V определены в равенствах 8-80 и 8-81:

$$H = \sum_{x'=0}^{7} (\ x'+1\ ) * (\ p[\ 8+x', -1\ ] - p[\ 6-x', -1\ ]\ ) \quad (8\text{-}80)$$

$$V = \sum_{y'=0}^{7} (y'+1) * (p[-1, 4+y'] - p[-1, 2-y']). \quad (8\text{-}81)$$

8.3.3 Процесс Intra предсказания для образцов цветности

Этот процесс активируют для типов макроблоков I и SI. Процесс определяет, как находят Intra предсказания для образцов цветности текущего макроблока.

Входы в этот процесс – образцы, созданные до процесса отделения блоков от смежных блоков цветности (если таковые имеются).

Выходы этого процесса – образцы цветности Intra предсказания для текущего макроблока $pred_{Cb}[\ x, y\ ]$ и $pred_{Cr}[\ x, y\ ]$.

Оба блока цветности (Cb и Cr) макроблока должны использовать один и тот же режим предсказания. Режим предсказания применяют к каждому из блоков цветности по отдельности. Процесс, определенный в этом пункте, активируют для каждого блока цветности. В оставшейся части этого пункт за блок цветности принимают один из двух блоков цветности, а индекс C используют как замену индексов Cb или Cr.

17 смежных образцов p[x, y], которые создают образцы цветности до процесса фильтрового разделения на блоки с x = –1, y = –1..7 и с x = 0..7, y = –1, находят следующим образом:

– Процесс отыскания для смежных положений по п. 6.4.8 активируют для расположений цветности со значением (x, y), присвоенным (xN, yN) как входу, а mbAddrN и (xW, yW) – как выходу.

– Каждый образец p[x, y] находят следующим образом:

– Если любое из следующих условий – истина, то образец p[x, y] помечен как "не доступно для Intra предсказания цветности"

  – mbAddrN не доступно;

  – макроблок mbAddrN кодирован в режиме Inter предсказания с флагом constrained_intra_pred_flag, равным 1;

  – макроблок mbAddrN имеет mb_type, равное SI, с флагом constrained_intra_pred_flag, равным 1, а текущий макроблок не имеет mb_type, равное SI.

– Иначе образец p[x, y] помечен как "доступно для Intra предсказания цветности", а образцу цветности компонента C в положении цветности (xW, yW) внутри макроблока mbAddrN присвоено значение p[x, y].

Положим, что $pred_C$[x, y] с x, y = 0..7 обозначает образцы предсказания для блока образцов цветности.

Режимы Intra предсказаний цветности приведены в таблице 8-4.

Таблица 8-4 – Спецификация режима Intra предсказаний цветности и связанные с этим имена

intra_chroma_pred_mode	Имя intra_chroma_pred_mode
0	Intra_Chroma_DC (режим предсказания)
1	Intra_Chroma_Horizontal (режим предсказания)
2	Intra_Chroma_Vertical (режим предсказания)
3	Intra_Chroma_Plane (режим предсказания)

В зависимости от intra_chroma_pred_mode должен использоваться один из режимов Intra предсказания цветности, определенный в пп. 8.3.3.1–8.3.3.4.

8.3.3.1 Спецификация режима предсказания Intra_Chroma_DC

Значения образцов предсказания $pred_C$[x, y] с x = 0..3 и y = 0..3 находят следующим образом:

– Если все образцы p[x, –1] с x = 0..3 и все образцы p[–1, y] с y = 0..3 помечены как "доступно для Intra предсказания цветности",

$$pred_C[x, y] = \left(\sum_{x'=0}^{3} p[x', -1] + \sum_{y'=0}^{3} p[-1, y'] + 4 \right) >> 3 \text{ , с } x = 0..3 \text{ и } y = 0..3. \quad (8\text{-}82)$$

– Иначе, если все образцы p[x, –1] с x = 0..3 помечены как "доступно для Intra предсказания цветности", а любые образцы p[–1, y] с y = 0..3 помечены как "не доступно для Intra предсказания цветности",

$$pred_C[x, y] = \left(\sum_{x'=0}^{3} p[x', -1] + 2 \right) >> 2 \text{ с } x = 0..3 \text{ и } y = 0..3. \quad (8\text{-}83)$$

– Иначе, если имеются образцы p[x, –1] с x = 0..3, помеченные как "не доступно для Intra предсказания цветности", а все образцы p[-1, y] с y = 0..3 помечены как "доступно для Intra предсказания цветности",

$$pred_C[x, y] = \left(\sum_{y'=0}^{3} p[-1, y'] + 2 \right) >> 2 \text{ с } x = 0..3 \text{ и } y = 0..3. \quad (8\text{-}84)$$

– Иначе (некоторые образцы p[x, –1] с x = 0..3 и некоторые образцы p[–1, y] с y = 0..3, помеченные как "не доступно для Intra предсказания цветности")

$$pred_C[x, y] = 128, \text{ с } x = 0..3 \text{ и } y = 0..3. \quad (8\text{-}85)$$

Значения образцов предсказания $pred_C$[x, y] с x = 4..7 и y = 0..3 находят следующим образом:

– Если все образцы p[x, –1] с x = 4..7 помечены как "доступно для Intra предсказания цветности",

$$pred_C[x, y] = \left(\sum_{x'=4}^{7} p[x', -1] + 2 \right) >> 2 \text{ с } x = 4..7 \text{ и } y = 0..3. \quad (8\text{-}86)$$

– Иначе, если все образцы p[–1, y] с y = 0..3 помечены как "доступно для Intra предсказания цветности",

$$\text{pred}_C[\,x, y\,] = \left(\sum_{y'=0}^{3} p[-1, y'] + 2 \right) >> 2 \quad \text{с } x = 4..7 \text{ и } y = 0..3. \tag{8-87}$$

– Иначе (некоторые образцы p[x, –1] с x = 4..7 и некоторые образцы p[–1, y] с y = 0..3, помеченные как "не доступно для Intra предсказания цветности"),

$$\text{pred}_C[\,x, y\,] = 128 \text{ с } x = 4..7 \text{ и } y = 0..3. \tag{8-88}$$

Значения образцов предсказания $\text{pred}_C[\,x, y\,]$ с x = 0..3 и y = 4..7 находят следующим образом:

– Если все образцы p[–1, y] с y = 4..7 помечены как "доступно для Intra предсказания цветности",

$$\text{pred}_C[\,x, y\,] = \left(\sum_{y'=4}^{7} p[-1, y'] + 2 \right) >> 2 \quad \text{с } x = 0..3 \text{ и } y = 4..7. \tag{8-89}$$

– Иначе, если все образцы p[x, –1] с x = 0..3 помечены как "доступно для Intra предсказания цветности",

$$\text{pred}_C[\,x, y\,] = \left(\sum_{x'=0}^{3} p[x', -1] + 2 \right) >> 2 \quad \text{с } x = 0..3 \text{ и } y = 4..7. \tag{8-90}$$

– Иначе (некоторые образцы p[x, –1] с x = 0..3 и некоторые образцы p[–1, y] с y = 4..7, помеченные как "не доступно для Intra предсказания цветности"),

$$\text{pred}_C[\,x, y\,] = 128 \text{ с } x = 0..3 \text{ и } y = 4..7. \tag{8-91}$$

Значения образцов предсказания $\text{pred}_C[\,x, y\,]$ с x = 4..7 и y = 4..7 находят следующим образом:

– Если все образцы p[x, –1] с x = 4..7 и все образцы p[–1, y] с y = 4..7 помечены как "доступно для Intra предсказания цветности",

$$\text{pred}_C[\,x, y\,] = \left(\sum_{x'=4}^{7} p[x', -1] + \sum_{y'=4}^{7} p[-1, y'] + 4 \right) >> 3 \quad \text{с } x = 4..7 \text{ и } y = 4..7. \tag{8-92}$$

– Иначе, если все образцы p[x, –1] с x = 4..7 помечены как "доступно для Intra предсказания цветности", а любые образцы p[–1, y] с y = 4..7 помечены как "не доступно для Intra предсказания цветности",

$$\text{pred}_C[\,x, y\,] = \left(\sum_{x'=4}^{7} p[x', -1] + 2 \right) >> 2 \quad \text{с } x = 4..7 \text{ и } y = 4..7. \tag{8-93}$$

– Иначе, если имеются образцы p[x, –1] с x = 4..7, помеченные как "не доступно для Intra предсказания цветности", а все образцы p[-1, y] с y = 4..7 помечены как "доступно для Intra предсказания цветности",

$$\text{pred}_C[\,x, y\,] = \left(\sum_{y'=4}^{7} p[-1, y'] + 2 \right) >> 2 \quad \text{с } x = 4..7 \text{ и } y = 4..7. \tag{8-94}$$

– Иначе (некоторые образцы p[x, –1] с x = 4..7 и некоторые образцы p[–1, y] с y = 4..7 помечены как "не доступно для Intra предсказания цветности"),

$$\text{pred}_C[\,x, y\,] = 128 \text{ с } x = 4..7 \text{ и } y = 4..7. \tag{8-95}$$

8.3.3.2 Спецификация режима предсказания Intra_Chroma_Horizontal

Этот режим должен быть использован, если только образцы p[–1, y] с y = 0..7 помечены как "доступно для Intra предсказания цветности".

Значения образцов предсказания $\text{pred}_C[\,x, y\,]$ находят следующим образом:

$$\text{pred}_C[\,x, y\,] = p[\,-1, y\,] \text{ с } x, y = 0..7. \tag{8-96}$$

8.3.3.3 Спецификация режима предсказания Intra_Chroma_Vertical

Этот режим должен быть использован, если только образцы p[x, –1] с x = 0..7 помечены как "доступно для Intra предсказания цветности".

Значения образцов предсказания $pred_C$[x, y] находят следующим образом:

$$pred_C[x, y] = p[x, –1] \text{ с } x, y = 0..7. \quad (8\text{-}97)$$

8.3.3.4 Спецификация режима предсказания Intra_Chroma_Plane

Это режим должен быть использован, если только образцы p[x, –1] с x = 0..7 и p[–1, y], с y = –1..7 помечены как "доступно для Intra предсказания цветности".

Значения образцов предсказания $pred_C$[x, y] находят следующим образом:

$$pred_C[x, y] = Clip1((a + b * (x – 3) + c * (y – 3) + 16) >> 5), \text{ с } x, y = 0..7, \quad (8\text{-}98)$$

где:

$$a = 16 * (p[–1, 7] + p[7, –1]) \quad (8\text{-}99)$$

$$b = (17 * H + 16) >> 5 \quad (8\text{-}100)$$

$$c = (17 * V + 16) >> 5, \quad (8\text{-}101)$$

а H и V определены следующим образом:

$$H = \sum_{x'=0}^{3} (x'+1) * (p[4 + x', -1] - p[2 - x', -1]) \quad (8\text{-}102)$$

$$V = \sum_{y'=0}^{3} (y'+1) * (p[-1, 4 + y'] - p[-1, 2 - y']) \quad (8\text{-}103)$$

8.3.4 Процесс создания образцов для макроблоков I_PCM

Этот процесс активируют, если mb_type равно I_PCM.

Выходы этого процесса – созданные образцы макроблока S'_L, S'_{Cb} и S'_{Cr} перед процессом фильтрового разделения на блоки.

Переменную dy находят следующим образом:

– Если MbaffFrameFlag равно 1, а текущий макроблок – это макроблок поля, dy устанавливают равным 2.

– Иначе (MbaffFrameFlag равно 0 или текущий макроблок – это макроблок кадра), dy устанавливают равным 1.

Положение верхнего левого образца яркости текущего макроблока находят, активируя процесс инверсного сканирования макроблока по п. 6.4.1 с CurrMbAddr как вход, и выходом, присвоенным значению (xP, yP).

Созданные образцы перед процессом разделения на блоки создают, как определено ниже:

```
for( i = 0; i < 256; i++ )
    S'L[ xP + ( i % 16 ), yP + dy * ( i / 16 ) ) ] = pcm_byte[ i ]                    (8-104)

for( i = 0; i < 64; i++ ) {
    S'Cb[ ( xP >> 1 ) + ( i % 8 ), ( ( yP + 1 ) >> 1 ) + dy * ( i / 8 ) ] = pcm_byte[ i + 256 ]   (8-105)
    S'Cr[ ( xP >> 1 ) + ( i % 8 ), ( ( yP + 1 ) >> 1 ) + dy * ( i / 8 ) ] = pcm_byte[ i + 320 ]
}
```

8.4 Процесс Inter предсказания

Этот процесс активируют при декодировании макроблоков типов P и B.

Выходы этого процесса – образцы Inter предсказания для текущего макроблока, которые представляют массив 16x16 $pred_L$ образцов яркости и два массива 8x8 $pred_{Cr}$ и $pred_{Cb}$ образцов цветности, с одним из каждых компонентов цветности Cb и Cr.

Разделение макроблока определено значением mb_type. Каждое разделение макроблока обозначают mbPartIdx. Если разделение макроблока состоит из разделенных частей, которые равны субмакроблокам, каждый субмакроблок может быть далее разделен на части субмакроблока, как определено значением sub_mb_type. Каждое разделение

субмакроблока обозначают значением subMbPartIdx. Если разделение макроблока не состоит из субмакроблоков, subMbPartIdx устанавливают равным 0.

Для каждой части разделенного макроблока или субмакроблока определены следующие шаги.

Функции MbPartWidth( ), MbPartHeight( ), SubMbPartWidth( ) и SubMbPartHeight( ), описывающие ширину и высоту частей макроблока и субмакроблока, определены в таблицах 7-10, 7-11, 7-14 и 7-15.

Переменные partWidth и partHeight находят следующим образом:

– Если mb_type не равно P_8x8, или P_8x8ref0, или B_8x8, используют следующие условия:

$$partWidth = MbPartWidth(mb\_type) \quad (8\text{-}106)$$

$$partHeight = MbPartHeight(mb\_type). \quad (8\text{-}107)$$

– Иначе (mb_type равно P_8x8 или P_8x8ref0, или B_8x8),

$$partWidth = SubMbPartWidth(sub\_mb\_type[mbPartIdx]) \quad (8\text{-}108)$$

$$partHeight = SubMbPartHeight(sub\_mb\_type[mbPartIdx]). \quad (8\text{-}109)$$

Если mb_type равно B_Skip или B_Direct_16x16, или sub_mb_type[mbPartIdx] равно B_Direct_8x8, процесс Inter предсказания определен для

$$partWidth = 4 \quad (8\text{-}110)$$

$$partHeight = 4 \quad (8\text{-}111)$$

со значениями mbPartIdx 0..3. Для каждого субмакроблока, индексированного значением mbPartIdx, subMbPartIdx принимает значения 0..3.

Допустим, что переменная MvCnt первоначально была установлена на 0 до всякой активации этого макроблока по п. 8.4.1.

Процесс Inter предсказания для части макроблока mbPartIdx и субмакроблока subMbPartIdx состоит из следующих шагов.

1. Процесс отыскания компонентов вектора движения и индексов контроля, как определено в п. 8.4.1.

   Входы этого процесса:

   – часть макроблока mbPartIdx;

   – часть субмакроблока subMbPartIdx.

   Выходы этого процесса:

   – векторы движения яркости mvL0 и mvL1, а также цветности mvCL0 и mvCL1;

   – индексы контроля refIdxL0 и refIdxL1;

   – флаги использования списка предсказания predFlagL0 и predFlagL1;

   – вычисление вектора движения части субмакроблока subMvCnt.

2. Приращение subMvCnt переменной MvCnt.

3. Процесс декодирования образцов Inter предсказания, как определено в п. 8.4.2.

   Входы этого процесса:

   – часть макроблока mbPartIdx;

   – часть субмакроблока subMbPartIdx;

   – переменные, определяющие ширину и высоту части: partWidth и partHeight;

   – векторы движения яркости mvL0 и mvL1, а также цветности mvCL0 и mvCL1;

   – индексы контроля refIdxL0 и refIdxL1;

   – флаги использования списка предсказания predFlagL0 и predFlagL1.

Выходы этого процесса:

– образцы inter предсказания (pred), которые представляют массив (partWidth)x(partHeight) $predPart_L$ образцов предсказания яркости и два массива (partWidth/2)x(partHeight/2) $predPart_{Cr}$ и $predPart_{Cb}$ образцов предсказания цветности, по одному для каждого из компонентов цветности Cb и Cr.

Для использования в процессе отыскания переменных, активированных позже в процессе декодирования, сделаны следующие присвоения:

MvL0[mbPartIdx][subMbPartIdx] = mvL0 (8-112)

MvL1[mbPartIdx][subMbPartIdx] = mvL1 (8-113)

RefIdxL0[mbPartIdx] = refIdxL0 (8-114)

RefIdxL1[mbPartIdx] = refIdxL1 (8-115)

PredFlagL0[mbPartIdx] = predFlagL0 (8-116)

PredFlagL1[mbPartIdx] = predFlagL1. (8-117)

Положение верхнего левого образца части от разделения относительно верхнего левого образца макроблока находят, активируя процесс инверсного сканирования части макроблока, как описано в п. 6.4.2.1, с mbPartIdx в качестве входа и (xP, yP) в качестве выхода.

Положение верхнего левого образца вторично разделенной части макроблока относительно верхнего левого образца части макроблока находят, активируя процесс инверсного сканирования части субмакроблока, как описано в п. 6.4.2.2, с subMbPartIdx в качестве входа и (xS, yS) в качестве выхода.

Макроблок предсказания формируют, помещая образцы предсказания частей макроблока или субмакроблока на их корректные позиции в макроблоке следующим образом:

Переменную $pred_L$[xP + xS + x, yP + yS + y] с x = 0 .. partWidth – 1, y = 0 .. partHeight – 1 находят как

$pred_L$[xP + xS + x, yP + yS + y] = $predPart_L$[x, y]. (8-118)

Переменную $pred_C$[xP / 2 + xS / 2 + x, yP / 2 + yS / 2 + y] с x = 0 .. partWidth/2 – 1, y = 0 .. partHeight/2 – 1, с заменой C на Cb или Cr, находят как

$pred_C$[xP / 2 + xS / 2 + x, yP / 2 + yS / 2 + y] = $predPart_C$[x, y]. (8-119)

8.4.1 Процесс отыскания компонентов вектора движения и индексов контроля

Входы в этот процесс:

– разделенная часть макроблока mbPartIdx;

– разделенная часть субмакроблока subMbPartIdx.

Выходы этого процесса:

– векторы движения яркости mvL0 и mvL1, а также векторы движения цветности mvCL0 и mvCL1;

– индексы контроля refIdxL0 и refIdxL1;

– флаги использования списка предсказания predFlagL0 и predFlagL1;

– переменная вычисления вектора движения части субмакроблока subMvCnt.

Для отыскания переменных mvL0 и mvL1, а также refIdxL0 и refIdxL1 используют следующие условия:

– Если mb_type равно P_Skip, активируют процесс отыскания векторов движения яркости для пропущенных макроблоков в секциях P и SP по п. 8.4.1.1, с выходом векторов движения яркости mvL0 и индексов контроля refIdxL0, а predFlagL0 устанавливают равным 1. Значения mvL1 и refIdxL1 помечают как не доступные, а predFlagL1 устанавливают равным 0. Переменную вычисленного вектора движения части субмакроблока subMvCnt устанавливают равной 1.

– Иначе, если mb_type равно B_Skip или B_Direct_16x16, или sub_mb_type[mbPartIdx] равно B_Direct_8x8, активируют процесс отыскания векторов движения яркости B_Skip, B_Direct_16x16 и B_Direct_8x8 в секциях B п. 8.4.1.2 с mbPartIdx и subMbPartIdx в качестве входа, а в качестве выхода – векторы движения яркости mvL0, mvL1, индексы контроля refIdxL0, refIdxL1, вычисления вектора движения части субмакроблока subMvCnt и флаги использования списка предсказания predFlagL0 и predFlagL1.

– Иначе с заменой X на 0 или 1 в переменных predFlagLX, mvLX, refIdxLX, а также Pred_LX и в элементах синтаксиса ref_idx_lX и mvd_lX, используют следующие условия:

– Переменные refIdxLX и predFlagLX находят следующим образом:

  – Если MbPartPredMode(mb_type, mbPartIdx) или SubMbPredMode(sub_mb_type[mbPartIdx]) равно Pred_LX или BiPred, то

    refIdxLX = ref_idx_lX[mbPartIdx] (8-120)

    predFlagLX = 1. (8-121)

  – Иначе переменные refIdxLX и predFlagLX опредяют как

    refIdxLX = –1 (8-122)

    predFlagLX = 0. (8-123)

– Переменную subMvCnt при вычислении вектора движения части субмакроблока устанавливают равной predFlagL0 + predFlagL1.

– Если predFlagLX равно 1, активируют процесс отыскания предсказания вектора движения по п. 8.4.1.3 с mbPartIdx subMbPartIdx, refIdxLX и суффиксом списка LX в качестве входа, а в качестве выхода – mvpLX. Векторы движения яркости находят как

  mvLX[0] = mvpLX[0] + mvd_lX[mbPartIdx][subMbPartIdx][0] (8-124)

  mvLX[1] = mvpLX[1] + mvd_lX[mbPartIdx][subMbPartIdx][1] . (8-125)

Для отыскания переменных векторов движения цветности используют следующие условия. Если predFlagLX (с X либо 0, либо 1) равно 1, активируют процесс отыскания векторов движения цветности по п. 8.4.1.4 с mvLX и refIdxLX в качестве входа, а в качестве выхода – mvCLX.

8.4.1.1 Процесс отыскания векторов движения яркости пропущенных макроблоков в секциях P и SP

Процесс активируют, если mb_type равно P_Skip.

Выходы этого процесса – вектор движения mvL0 и индекс контроля refIdxL0.

Индекс контроля refIdxL0 для пропущенного макроблока находят следующим образом:

refIdxL0 = 0 (8-126)

Для отыскания вектора движения mvL0 макроблока типа P_Skip используют следующие условия:

– Положим, что currSubMbType установлен равным sub_mb_type[0]. Активируют процесс, определенный в п. 8.4.1.3.2, с mbPartIdx, установленным на 0, subMbPartIdx, установленным на 0, currSubMbType и суффиксом списка L0 в качестве входа, а в качестве выхода – присвоенные значения mbAddrA, mbAddrB, mvL0A, mvL0B, refIdxL0A и refIdxL0B.

– Переменная mvL0 определена следующим образом:

  – Если любое из следующих условий – истина, то оба компонента вектора движения mvL0 устанавливают на 0:

    – mbAddrA не доступно;

    – mbAddrB не доступно;

    – refIdxL0A равно 0, и оба компонента mvL0A равны 0;

    – refIdxL0B равно 0, и оба компонента mvL0B равны 0.

  – Иначе активируют процесс для отыскания предсказания вектора движения яркости, как определено в п. 8.4.1.3, с mbPartIdx = 0, subMbPartIdx = 0, refIdxL0 и суффиксом списка L0 в качестве входа, а в качестве выхода – присвоенное значение mvL0.

  ПРИМЕЧАНИЕ. – Выходу прямо присвоено значение mvL0, поскольку предсказанный параметр равен действительному вектору движения.

8.4.1.2 Процесс отыскания векторов движения яркости B_Skip, B_Direct_16x16 и B_Direct_8x8

Процесс активируют, если mb_type равно B_Skip или B_Direct_16x16, или sub_mb_type[mbPartIdx] равно B_Direct_8x8.

Входы этого процесса – mbPartIdx и subMbPartIdx.

Выходы этого процесса – это индексы контроля refIdxL0, refIdxL1, векторы движений mvL0 и mvL1, вычисленная часть вектора движения subMvCnt и флаги использованных списков предсказаний predFlagL0 и predFlagL1.

Процесс отыскания зависит от значения direct_spatial_mv_pred_flag, которое представлено в потоке битов в синтаксисе заголовка секции, как определено в п. 7.3.3, и описано следующим образом:

– Если direct_spatial_mv_pred_flag равно 1, то режим, в котором этот процесс находится на выходе, считают пространственным прямым режимом предсказания.

– Иначе (direct_spatial_mv_pred_flag равно 0), режим, в котором этот процесс находится на выходе, считают временным прямым режимом предсказания.

Оба эти прямые режимы предсказания, пространственный и временный, используют близко расположенные векторы движений и индексы контроля, как это определено в п. 8.4.1.2.1.

Векторы движений и индексы контроля находят следующим образом:

– Если используют пространственный прямой режим предсказания, то применяют прямой вектор движения и режим предсказания индекса контроля, определенный в п. 8.4.1.2.2, с subMvCnt в качестве выхода.

– Иначе (используют временный прямой режим предсказания), используют прямой вектор движения и режим предсказания индекса контроля, определенный в п. 8.4.1.2.3, а переменную subMvCnt находят следующим образом:

  – Если subMbPartIdx равно 0, subMvCnt устанавливают равным 2.

  – Иначе (subMbPartIdx не равно 0), subMvCnt устанавливают равным 0.

8.4.1.2.1 Процесс вывода совпадающих разделенных частей субмакроблоков 4x4

Входы в этот процесс – mbPartIdx и subMbPartIdx.

Выходы этого процесса – изображение colPic, совпадающий (по месту) макроблок mbAddrCol, вектор движения mvCol, индекс контроля refIdxCol и переменная vertMvScale (которыми могут быть One_To_One, Frm_To_Fld или Fld_To_Frm).

Положим, что firstRefPicL1 – это контрольное изображение, отнесенное к RefPicList1[0].

Если firstRefPicL1 – это кадр или дополнительная пара полей, то допустим, что firstRefPicL1Top и firstRefPicL1Bottom – это верхнее и нижнее поля firstRefPicL1, и положим, что следующие переменные определены как

topAbsDeclifPOC = Abs(DeclifPicOrderCnt(firstRefPicL1Top, CurrPic)) (8-127)

bottomAbsDeclifPOC = Abs(DeclifPicOrderCnt(firstRefPicL1Bottom, CurrPic)). (8-128)

Переменная colPic определяет изображение, которое содержит совпадающий по месту макроблок, как определено в таблице 8-5.

Таблица 8-5 – Спецификация переменной colPic

field_pic_flag	Первый вход в RefPicList1 это …	mb_field_decoding_flag	Дополнительное условие	colPic
1	поле декодированного кадра			кадр, содержащий firstRefPicL1
	декодированное поле			firstRefPicL1
0	декодированный кадр			firstRefPicL1
	дополнительная пара полей	0	topAbsDeclifPOC < bottomAbsDeclifPOC	верхнее поле firstRefPicL1
			topAbsDeclifPOC >= bottomAbsDeclifPOC	нижнее поле firstRefPicL1
		1	(CurrMbAddr & 1) = = 0	верхнее поле firstRefPicL1
			(CurrMbAddr & 1) != 0	нижнее поле firstRefPicL1

Если direct_8x8_inference_flag равно 1, subMbPartIdx устанавливают следующим образом:

subMbPartIdx = mbPartIdx. (8-129)

Положим, что PicCodingStruct(X) – функция с аргументом X либо CurrPic, либо colPic. Это показано в таблице 8-6.

Таблица 8-6 – Спецификация PicCodingStruct(X)

X кодирован с field_pic_flag, равным …	mb_adaptive_frame_field_flag	PicCodingStruct(X)
1		FLD
0	0	FRM
0	1	AFRM

При luma4x4BlkIdx = mbPartIdx * 4 + subMbPartIdx, активируют процесс инверсного сканирования блока яркости 4x4, как определено в п. 6.4.3, с luma4x4BlkIdx в качестве входа и (x, y), присвоенного значению (xCol, yCol), в качестве выхода.

Таблица 8-7 определяет совпадающий адрес макроблока mbAddrCol, yM, и переменную vertMvScale в два этапа:

1. Спецификация адреса макроблока mbAddrX в зависимости от PicCodingStruct(CurrPic) и PicCoding Struct(colPic).

   ПРИМЕЧАНИЕ. – Для CurrPic и colPic невозможны типы кодирующих изображений ни (FRM, AFRM), ни (AFRM, FRM), т. к. эти типы кодирующих изображений должны быть разделены IDR изображением.

2. Спецификацию mbAddrCol, yM и vertMvScale, зависящую от mb_field_decoding_flag и переменной field DecodingFlagX, находят следующим образом:

   – Если макроблок mbAddrX в изображении colPic – это макроблок поля, то fieldDecodingFlagX устанавливают равным 1.

   – Иначе (макроблок mbAddrX в изображении colPic – это макроблок кадра), fieldDecodingFlagX устанавливают равным 0.

Неопределенные значения в таблице 8-7 указывают, что значение соответствующей переменной не отвечает текущему ряду таблицы.

Значение mbAddrCol устанавливают равным CurrMbAddr или одному из следующих значений:

mbAddrCol1 = 2 * PicWidthInMbs * (CurrMbAddr / PicWidthInMbs) +
(CurrMbAddr % PicWidthInMbs) + PicWidthInMbs * (yCol / 8) (8-130)

mbAddrCol2 = 2 * CurrMbAddr + (yCol / 8) (8-131)

mbAddrCol3 = 2 * CurrMbAddr + bottom_field_flag (8-132)

mbAddrCol4 = PicWidthInMbs * (CurrMbAddr / (2 * PicWidthInMbs)) +
(CurrMbAddr % PicWidthInMbs) (8-133)

mbAddrCol5 = CurrMbAddr / 2 (8-134)

mbAddrCol6 = 2 * (CurrMbAddr / 2) + ((topAbsDeслиfPOC < bottomAbsDeслиfPOC) ? 0 : 1) (8-135)

mbAddrCol7 = 2 * (CurrMbAddr / 2) + (yCol / 8). (8-136)

Таблица 8-7 – Спецификация mbAddrCol, yM и vertMvScale

PicCodingStruct(CurrPic)	PicCodingStruct(colPic)	mbAddrX	mb_field_decoded_flag	fieldDecodingFlagX	mbAddrCol	yM	vertMvScale
FLD	FLD				CurrMbAddr	yCol	One_To_One
	FRM				mbAddrCol1	(2 * yCol) % 16	Frm_To_Fld
	AFRM	2*CurrMbAddr		0	mbAddrCol2	(2 * yCol) % 16	Frm_To_Fld
				1	mbAddrCol3	yCol	One_To_One
FRM	FLD				mbAddrCol4	8 * ((CurrMbAddr / PicWidthInMbs) % 2) + 4 * (yCol / 8)	Fld_To_Frm
	FRM				CurrMbAddr	yCol	One_To_One
AFRM	FLD		0		mbAddrCol5	8 * (CurrMbAddr % 2) +4 * (yCol / 8)	Fld_To_Frm
			1		mbAddrCol5	yCol	One_To_One
	AFRM	CurrMbAddr	0	0	CurrMbAddr	yCol	One_To_One
				1	mbAddrCol6	8 * (CurrMbAddr % 2) + 4 * (yCol / 8)	Fld_To_Frm
		CurrMbAddr	1	0	mbAddrCol7	(2 * yCol) % 16	Frm_To_Fld
				1	CurrMbAddr	yCol	One_To_One

Положим, что mbPartIdxCol – индекс разделенной части макроблока, расположенной вместе с макроблоком, а subMbPartIdxCol – субиндекс разделенной части макроблока, расположенной вместе с субмакроблоком. Часть в макроблоке mbAddrCol внутри изображения colPic, покрывающего образец (xCol, yM), должна быть присвоена mbPartIdxCol, а часть субмакроблока внутри разделенной части mbPartIdxCol, покрывающей образец (xCol, yM) в макроблоке mbAddrCol внутри изображения colPic, должна быть присвоена subMbPartIdxCol.

Флаги использованных предсказаний predFlagL0Col и predFlagL1Col устанавливают равными PredFlagL0[mbPartIdxCol] и PredFlagL1[mbPartIdxCol] соответственно. Это флаги использованных предсказаний, которые присвоены частям макроблока mbAddrCol\mbPartIdxCol внутри изображения colPic.

Вектор движения mvCol и индекс контроля refIdxCol находят следующим образом:

– Если макроблок mbAddrCol кодирован в режиме Intra предсказания макроблока или если оба флага использованных предсказаний, predFlagL0Col и predFlagL1Col, равны 0, то оба компонента mvCol устанавливают равными 0, а refIdxCol устанавливают равным –1.

– Иначе используют следующие условия:

  – Если predFlagL0Col равно 1, вектор движения mvCol и индекс контроля refIdxCol устанавливают равными MvL0[mbPartIdxCol][subMbPartIdxCol] и RefIdxL0[mbPartIdxCol] соответственно. Эти значения – вектор движения mvL0 и индекс контроля refIdxL0, которые присвоены частям (суб)макроблока mbAddrCol\mbPartIdxCol\subMbPartIdxCol внутри изображения colPic.

  – Иначе (predFlagL0Col равно 0 и predFlagL1Col равно 1), вектор движения mvCol и индекс контроля refIdxCol устанавливают равными MvL1[mbPartIdxCol][subMbPartIdxCol] и RefIdxL1[mbPartIdxCol] соответственно. Эти значения – вектор движения mvL1 и индекс контроля refIdxL1, которые присвоены частям (суб)макроблока mbAddrCol\mbPartIdxCol\subMbPartIdxCol внутри изображения colPic.

8.4.1.2.2 Процесс отыскания режима предсказания пространственного прямого вектора движения яркости и индекса контроля

Процесс активируют, если direct_spatial_mv_pred_flag равно 1 и любое из следующих условий является истиной:

- mb_type равно B_Skip;
- mb_type равно B_Direct_16x16;
- sub_mb_type[mbPartIdx] равно B_Direct_8x8.

Входы в этот процесс – mbPartIdx, subMbPartIdx.

Выходы из этого процесса – индексы контроля refIdxL0, refIdxL1, векторы движений mvL0 и mvL1, вычисленный вектор движения субчасти subMvCnt и флаги использованных списков предсказаний, predFlagL0 и predFlagL1.

Индексы контроля refIdxL0 и refIdxL1, а также переменную directZeroPredictionFlag находят, применяя следующие шаги по порядку.

1. Положим, что переменная currSubMbType установлена равной sub_mb_type[mbPartIdx].
2. Активируют процесс, определенный в п. 8.4.1.3.2, с mbPartIdx = 0, subMbPartIdx = 0, currSubMbType и суффиксом списка L0 в качестве входа, а в качестве выхода – присвоенные значения векторам движений mvL0N и индексам контроля refIdxL0N с заменой N на A, B или C.
3. Активируют процесс, определенный в п. 8.4.1.3.2, с mbPartIdx = 0, subMbPartIdx = 0, currSubMbType и суффиксом списка L1 в качестве входа, а в качестве выхода – присвоенные значения векторам движений mvL1N и индексам контроля refIdxL1N с заменой N на A, B или C.

   ПРИМЕЧАНИЕ. – Векторы движений mvL0N, mvL1N и индексы контроля refIdxL0N, refIdxL1N идентичны для всех разделенных частей субмакроблока 4x4 любого макроблока.
4. Индексы контроля refIdxL0, refIdxL1 и directZeroPredictionFlag находят как

$$refIdxL0 = MinPositive(refIdxL0A, MinPositive(refIdxL0B, refIdxL0C)) \quad (8\text{-}137)$$
$$refIdxL1 = MinPositive(refIdxL1A, MinPositive(refIdxL1B, refIdxL1C)) \quad (8\text{-}138)$$
$$directZeroPredictionFlag = 0, \quad (8\text{-}139)$$

   где

$$MinPositive(x,y) = \begin{cases} Min(x,y) & \text{если } x >= 0 \text{ и } y >= 0 \\ Max(x,y) & \text{иначе.} \end{cases} \quad (8\text{-}140)$$

5. Если оба индекса контроля refIdxL0 и refIdxL1 меньше 0,

$$refIdxL0 = 0 \quad (8\text{-}141)$$
$$refIdxL1 = 0 \quad (8\text{-}142)$$
$$directZeroPredictionFlag = 1. \quad (8\text{-}143)$$

Активируют процесс, определенный в п. 8.4.1.2.1, с mbPartIdx, subMbPartIdx, заданными в качестве входа, а в качестве выхода – присвоенные значения refIdxCol и mvCol.

Переменную colZeroFlag находят следующим образом:

– Если все из следующих условий истинны, то colZeroFlag устанавливают равным 1:
  - контрольное изображение, заданное RefPicList1[0], считают краткосрочным контрольным изображением;
  - refIdxCol равно 0;
  - оба вектора движения компонентов mvCol[0] и mvCol[1] находятся в диапазоне от –1 до 1 в единицах, определенных следующим образом:
    - Если находящийся в том же месте макроблок – это макроблок кадра, единицы mvCol[0] и mvCol[1] являются единицами четверти яркости образцов кадра.
    - Иначе (находящийся в том же месте макроблок – это макроблок поля), единицы mvCol[0] и mvCol[1] являются единицами четверти яркости образцов поля.

  ПРИМЕЧАНИЕ. – Для определенности установленного выше условия значение mvCol[1] не масштабируют, чтобы использовать единицы вектора движения для текущего макроблока в случаях, когда текущий макроблок является макроблоком кадра, а расположенный в этом месте макроблок является макроблоком поля или если текущий макроблок является макроблоком поля, а расположенный в этом месте макроблок является макроблоком кадра. Этот аспект отличается от использования mvCol[1] во временном прямом режиме, как определено в п. 8.4.1.2.3, в котором применяют вектор движения

расположенного в этом месте макроблока, чтобы использовать те же самые единицы, что и для вектора движения текущего макроблока, используя в этих случаях равенство 8-146 или равенство 8-147.

– Иначе colZeroFlag устанавливают равным 0.

Векторы движений mvLX (с X равным 0 или 1) находят следующим образом:

– Если любое из следующих условий – истина, оба компонента вектора движения mvLX устанавливают равным 0:
  – directZeroPredictionFlag равно 1;
  – refIdxLX меньше 0;
  – refIdxLX равно 0 и colZeroFlag равно 1.
– Иначе активируют процесс, определенный в п. 8.4.1.3, с mbPartIdx = 0, subMbPartIdx = 0, refIdxLX и суффиксом списка LX в качестве входа, а в качестве выхода – значение, присвоенное mvLX.

ПРИМЕЧАНИЕ. – Именно в вышеприведенном случае возвратный вектор движения mvLX идентичен для всех частей 4x4 субмакроблока любого макроблока.

Флаги использованных предсказаний predFlagL0 и predFlagL1 должны быть найдены, как определено в таблице 8-8.

Таблица 8-8 – Назначения флагов использованных предсказаний

refIdxL0	refIdxL1	predFlagL0	predFlagL1
>= 0	>= 0	1	1
>= 0	< 0	1	0
< 0	>= 0	0	1

Переменную subMvCnt находят следующим образом:

– Если subMbPartIdx не равно 0 или direct_8x8_inference_flag равно 0, то subMvCnt устанавливают равным 0.
– Иначе (subMbPartIdx равно 0 и direct_8x8_inference_flag равно 1), subMvCnt устанавливают равным predFlagL0 + predFLagL1.

8.4.1.2.3 Процесс вывода режимов предсказания временного прямого вектора движения яркости и индекса контроля

Этот процесс активируют, если direct_spatial_mv_pred_flag равно 0, а любое из следующих условий – истина:

– mb_type равно B_Skip
– mb_type равно B_Direct_16x16
– sub_mb_type[mbPartIdx] равно B_Direct_8x8.

Входы в этот процесс – mbPartIdx и subMbPartIdx.

Выходы из этого процесса – векторы движений mvL0 и mvL1, индексы контроля refIdxL0 и refIdxL1 и флаги использованных списков предсказаний, predFlagL0 и predFlagL1.

Процесс, определенный в п. 8.4.1.2.1, активируют с mbPartIdx, subMbPartIdx, заданными в качестве входа, а в качестве выхода – присвоенные значения colPic, mbAddrCol, mvCol, refIdxCol и vertMvScale.

Индексы контроля refIdxL0 и refIdxL1 находят следующим образом:

refIdxL0 = ((refIdxCol < 0) ? 0 : MapColToList0(refIdxCol)) (8-144)

refIdxL1 = 0. (8-145)

ПРИМЕЧАНИЕ. – Если текущий макроблок – это макроблок поля, то refIdxL0 и refIdxL1 – индексы списка поля; иначе (текущий макроблок – макроблок кадра) refIdxL0 и refIdxL1 – индексы списка кадра или дополнительной контрольной пары полей.

Положим, что refPicCol – это кадр, поле или дополнительная пара полей, которые направляет индекс контроля refIdxCol при декодировании расположенного в том же месте макроблока mbAddrCol внутри изображения colPic. Функция MapColToList0(refIdxCol) определена следующим образом:

– Если vertMvScale равно One_To_One, используют следующие условия:
  – Если field_pic_flag равно 0, а текущий макроблок – макроблок поля, используют следующие условия:

- Положим, что refIdxL0Frm – это наименьший обозначенный индекс контроля в текущем списке контрольного изображения RefPicList0, которое сравнивает кадр или дополнительную пару полей, содержащих поле refPicCol. RefPicList0 должен содержать переменную PicNum или LongTermPicNum, которая сравнивает кадр или дополнительную пару полей, содержащих refPicCol. Возвратное значение MapColToList0( ) определено следующим образом:
  - Если поле, на которое ссылается refIdxCol, имеет ту же четность, что и текущий макроблок, то MapColToList0(refIdxCol) возвращает индекс контроля (refIdxL0Frm << 1).
  - Иначе (поле, на которое ссылается refIdxCol, имеет противоположную четность текущего макроблока), MapColToList0(refIdxCol) возвращает индекс контроля ((refIdxL0Frm << 1) + 1).
- Иначе (field_pic_flag равно 1 или текущий макроблок – это макроблок кадра), MapColToList0(refIdxCol) возвращает наименьший обозначенный индекс контроля refIdxL0 в текущем списке контрольных изображений RefPicList0, который контролирует refPicCol. Значение RefPicList0 должно содержать переменные PicNum или LongTermPicNum, которые контролируют refPicCol.

– Иначе, если vertMvScale равно Frm_To_Fld, используют следующие условия:
  - Если field_pic_flag равно 0, положим, что refIdxL0Frm – наименьшее значение индекса контроля в текущем списке индексов контроля RefPicList0, который контролирует refPicCol. Значение MapColToList0(refIdxCol) возвращает индекс контроля (refIdxL0Frm << 1). Значение RefPicList0 должно содержать переменные PicNum или LongTermPicNum, которые контролируют refPicCol.
  - Иначе (field_pic_flag равно 1), MapColToList0(refIdxCol) возвращает наименьшее значение индекса контроля refIdxL0 в списке текущего контрольного изображения RefPicList0, которое сравнивает поле refPicCol с той же четностью, что и текущего изображения CurrPic. Значение RefPicList0 должно содержать переменные PicNum или LongTermPicNum, которые контролируют поле refPicCol с той же четностью, что и текущего изображения CurrPic.
- Иначе (vertMvScale равно Fld_To_Frm), MapColToList0(refIdxCol) возвращает наименьшее значение индекса контроля refIdxL0 в списке текущего контрольного изображения RefPicList0, которое контролирует кадр или дополнительную пару полей, содержащих refPicCol. Значение RefPicList0 должно содержать переменные PicNum или LongTermPicNum, которые контролируют кадр или дополнительную пару полей, содержащих refPicCol.

ПРИМЕЧАНИЕ. – Декодированное контрольное изображение, которое было помечено как "использованное для краткосрочного контроля" в то время, когда его использовали как эталонное (контрольное) в процессе декодирования изображения, содержащего в том же месте макроблок, можно было бы модифицировать и пометить как "использованное для долгосрочного контроля" прежде, чем использовать его в качестве эталонного для inter предсказания с режимом прямого предсказания текущего макроблока.

В зависимости от значения vertMvScale вертикальный компонент mvCol модифицируют следующим образом:

– Если vertMvScale равно Frm_To_Fld,

$$mvCol[1] = mvCol[1] / 2. \quad (8\text{-}146)$$

– Иначе, если vertMvScale равно Fld_To_Frm,

$$mvCol[1] = mvCol[1] * 2. \quad (8\text{-}147)$$

– Иначе (vertMvScale равно One_To_One), mvCol[1] остается неизменным.

Переменные currPicOrField, pic0 и pic1 находят следующим образом:

– Если field_pic_flag равно 0, а текущий макроблок – это макроблок поля, используют следующие условия:
  - currPicOrField – это поле текущего изображения CurrPic, которое имеет ту же четность, что и текущий макроблок.
  - pic1 – это поле RefPicList1[0], которое имеет ту же четность, что и текущий макроблок.
  - Положим, что frame0 – это кадр или дополнительная пара полей, на которые ссылаются с помощью RefPicList0[refIdxL0 /2].
  - Переменную pic0 находят следующим образом:
    - Если refIdxL0 % 2 равно 0, pic0 – это поле frame0, которое имеет ту же четность, что и текущий макроблок.
    - Иначе (refIdxL0 % 2 не равно 0), pic0 – это поле frame0, которое имеет противоположную текущему макроблоку четность.
– Иначе (field_pic_flag равно 1 или текущий макроблок – это макроблок кадра), currPicOrField – текущее изображение CurrPic, pic1 – декодированное контрольное изображение, на которое ссылаются с помощью RefPicList1[0], а pic0 – декодированное контрольное изображение, на которое ссылаются с помощью RefPicList0[refIdxL0].

Два вектора движений mvL0 и mvL1 каждой части 4x4 субмакроблока текущего макроблока находят следующим образом:

ПРИМЕЧАНИЕ. – Часто встречается ситуации, когда многие части субмакроблока 4x4 имеют общие векторы движений и контрольные изображения. В этих случаях временный режим прямой компенсации движения может

вычислить значение образца Inter предсказания в единицах блоков образцов яркости больших размеров, чем 4x4. Например, если direct_8x8_inference_flag равно 1, по крайней мере, каждый квадрат образца яркости 8x8 макроблока совместно использует те же векторы движений и контрольные изображения.

– Если индекс контроля refIdxL0 ссылается на долгосрочное изображение или DiffPicOrderCnt(pic1, pic0) равно 0, векторы движений mvL0, mvL1 режима прямого разделения на части находят как

$$mvL0 = mvCol \tag{8-148}$$

$$mvL1 = 0. \tag{8-149}$$

– Иначе векторы движений mvL0, mvL1 находят как масштабированные версии вектора движения mvCol, расположенного в том же месте части субмакроблока, как определено ниже (см. рисунок 8-2)

$$tx = (16\,384 + Abs(td / 2)) / td \tag{8-150}$$

$$DistScaleFactor = Clip3(-1024, 1023, (tb * tx + 32) >> 6) \tag{8-151}$$

$$mvL0 = (DistScaleFactor * mvCol + 128) >> 8 \tag{8-152}$$

$$mvL1 = mvL0 – mvCol, \tag{8-153}$$

где tb и td находят следующим образом:

$$tb = Clip3(-128, 127, DiffPicOrderCnt(currPicOrField, pic0)) \tag{8-154}$$

$$td = Clip3(-128, 127, DiffPicOrderCnt(pic1, pic0)). \tag{8-155}$$

ПРИМЕЧАНИЕ. – Значения mvL0 и mvL1 не могут выходить за диапазоны, определенные в Приложении A.

Оба флага использованных предсказаний predFlagL0 и predFlagL1 устанавливают на 1.

Рисунок 8-2 иллюстрирует временный прямой режим предполагаемого вектора движения, если текущее изображение временно находится между списками 0 и 1 контрольных изображений.

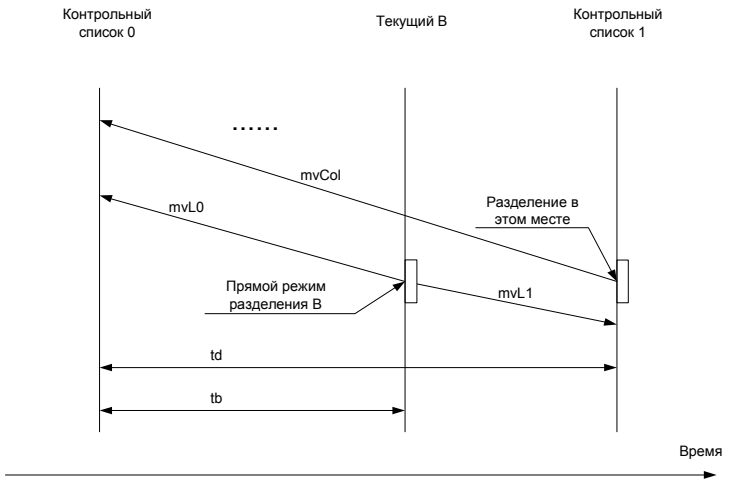

Рисунок 8-2 – Пример временного прямого режима предполагаемого вектора движения (информативное)

8.4.1.3 Процесс отыскания предсказания вектора движения яркости

Входы в этот процесс:

– индекс разделенной части макроблока mbPartIdx,

– субиндекс разделенной части макроблока subMbPartIdx,

– суффикс списка LX,

– индекс контроля текущей части refIdxLX.

Выход этого процесса – предсказания mvpLX вектора движения mvLX.

Положим, что currSubMbType установлено равным sub_mb_type[mbPartIdx].

Процесс отыскания смежных блоков для данных движения по п. 8.4.1.3.2 активируют с mbPartIdx, subMbPartIdx, currSubMbType и суффиксом списка LX в качестве входа и с mbAddrN\mbPartIdxN\subMbPartIdxN, индексами контроля refIdxLXN и векторами движений mvLXN с заменой N на A, B или C – в качестве выхода.

Процесс отыскания для предсказания медианы вектора движения яркости согласно п. 8.4.1.3.1 активируют с mbAddrN\mbPartIdxN\subMbPartIdxN, mvLXN, refIdxLXN с заменой N на A, B или C, и refIdxLX в качестве входа и mvpLX – в качестве выхода, пока одно из следующих условий не будет истиной:

– MbPartWidth(mb_type) равно 16, MbPartHeight(mb_type) равно 8, mbPartIdx равно 0 и refIdxLXB равно refIdxLX,

mvpLX = mvLXB. (8-156)

– MbPartWidth(mb_type) равно 16, MbPartHeight(mb_type) равно 8, mbPartIdx равно 1 и refIdxLXA равно refIdxLX,

mvpLX = mvLXA. (8-157)

– MbPartWidth(mb_type) равно 8, MbPartHeight(mb_type) равно 16, mbPartIdx равно 0 и refIdxLXA равно refIdxLX,

mvpLX = mvLXA. (8-158)

– MbPartWidth(mb_type) равно 8, MbPartHeight(mb_type) равно 16, mbPartIdx равно 1 и refIdxLXC равно refIdxLX,

mvpLX = mvLXC. (8-159)

Рисунок 8-3 иллюстрирует не медианные предсказания, как описано выше.

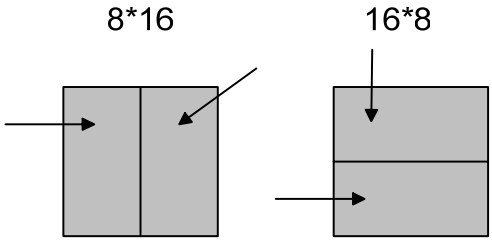

Рисунок 8-3 – Предсказания ориентированной сегментации (информативное)

8.4.1.3.1 Процесс отыскания предсказания медианы вектора движения яркости

Входы в этот процесс:

– смежные части mbAddrN\mbPartIdxN\subMbPartIdxN (с заменой N на A, B или C),

– векторы движений mvLXN (с заменой N на A, B или C) смежных частей,

– индексы контроля refIdxLXN (с заменой N на A, B или C) смежных частей, и

– индекс контроля refIdxLX текущей части.

Выход этого процесса – предсказание вектора движения mvpLX.

Переменную mvpLX находят следующим образом:

– Если обе части mbAddrB\mbPartIdxB\subMbPartIdxB и mbAddrC\mbPartIdxC\subMbPartIdxC не доступны, а mbAddrA\mbPartIdxA\subMbPartIdxA доступно, то

mvLXB = mvLXA (8-160)

mvLXC = mvLXA (8-161)

refIdxLXB = refIdxLXA (8-162)

refIdxLXC = refIdxLXA. (8-163)

– В зависимости от индексов контроля refIdxLXA, refIdxLXB или refIdxLXC используют следующие условия:

  – Если один и только один из индексов контроля refIdxLXA, refIdxLXB или refIdxLXC равен индексу контроля refIdxLX текущей части, используют следующие условия. Положим, что refIdxLXN – индекс контроля, который равен refIdxLX, а вектор движения mvLXN присвоен предсказанию вектора движения mvpLX:

    mvpLX = mvLXN. (8-164)

  – Иначе каждый компонент предсказания вектора движения mvpLX задан медианой соответствующих векторных компонентов вектора движения mvLXA, mvLXB и mvLXC:

    mvpLX[0] = Median(mvLXA[0], mvLXB[0], mvLXC[0]) (8-165)

    mvpLX[1] = Median(mvLXA[1], mvLXB[1], mvLXC[1]). (8-166)

8.4.1.3.2 Процесс отыскания данных движения смежных частей

Входы в этот процесс:

– индекс разделенной части макроблока mbPartIdx,

– субиндекс разделенной части макроблока subMbPartIdx,

– тип текущего субмакроблока currSubMbType,

– суффикс списка LX.

Выходы из этого процесса (с заменой N на A, B или C):

– mbAddrN\mbPartIdxN\subMbPartIdxN, определяющие смежные части,

– векторы движений mvLXN смежных частей, и

– индексы контроля refIdxLXN смежных частей.

Части mbAddrN\mbPartIdxN\subMbPartIdxN с заменой N на A, B или C находят в следующем порядке.

1. Положим, что mbAddrD\mbPartIdxD\subMbPartIdxD – переменные, определяющие дополнительную смежную часть.

2. Процесс по п. 6.4.7.5 активируют с mbPartIdx, currSubMbType и subMbPartIdx в качестве входа, а в качестве выхода – присвоенные значения mbAddrN\mbPartIdxN\subMbPartIdxN с заменой N на A, B, C или D.

3. Если часть mbAddrC\mbPartIdxC\subMbPartIdxC не доступна, используют следующие условия:

   mbAddrC = mbAddrD (8-167)

   mbPartIdxC = mbPartIdxD (8-168)

   subMbPartIdxC = subMbPartIdxD. (8-169)

Векторы движений mvLXN и индексы контроля refIdxLXN (с заменой N на A, B или C) находят следующим образом:

– Если часть макроблока или часть субмакроблока mbAddrN\mbPartIdxN\subMbPartIdxN не доступны или mbAddrN кодировано в режиме Intra предсказания, или если predFlagLX значения mbAddrN\mbPartIdxN\subMbPartIdxN равно 0, оба компонента mvLXN устанавливают равными 0, а refIdxLXN устанавливают равным –1.

– Иначе используют следующие условия:

  – Вектор движения mvLXN и индекс контроля refIdxLXN устанавливают равными MvLX[mbPartIdxN][subMbPartIdxN] и RefIdxLX[mbPartIdxN], которые, соответственно, являются вектором движения mvLX и индексом контроля refIdxLX, присвоенными части (суб)макроблока mbAddrN\mbPartIdxN\subMbPartIdxN.

  – Далее переменные mvLXN[1] и refIdxLXN обрабатывают следующим образом:

– Если текущий макроблок – макроблок поля, а макроблок mbAddrN – макроблок кадра, то

$$mvLXN[\ 1\ ] = mvLXN[\ 1\ ] / 2 \quad (8\text{-}170)$$

$$refIdxLXN = refIdxLXN * 2. \quad (8\text{-}171)$$

– Иначе, если текущий макроблок – макроблок кадра, а макроблок mbAddrN – макроблок поля, то

$$mvLXN[\ 1\ ] = mvLXN[\ 1\ ] * 2 \quad (8\text{-}172)$$

$$refIdxLXN = refIdxLXN / 2. \quad (8\text{-}173)$$

– Иначе вертикальный компонент вектора движения mvLXN[1] и индекс контроля refIdxLXN остаются неизменными.

8.4.1.4 Процесс отыскания векторов движения цветности

Входы в этот процесс – вектор движения яркости mvLX и индекс контроля refIdxLX.

Выходы из этого процесса – вектор движения цветности mvCLX.

Вектор движения цветности находят из соответствующего вектора движения яркости. Поскольку точность векторов движения яркости составляет одну четвертую образца, а цветность имеет половину разрешения по сравнению с яркостью, точность векторов движения цветности составляет одну восьмую от образца, т. е. значение 1 для вектора движения цветности относится к одной восьмой размещения образца.

ПРИМЕЧАНИЕ. – Например, если вектор яркости применяют к образцам яркости 8x16, то соответствующий вектор цветности применяют к образцам цветности 4x8, а если вектор яркости применяют к образцам яркости 4x4, то соответствующий вектор цветности применяют к образцам цветности 2x2.

При отыскании вектора движения mvCLX используют следующие условия:

– Если текущий макроблок – макроблок кадра, горизонтальную и вертикальную компоненты векторов движения цветности mvCLX находят умножением соответствующих компонентов векторов движения яркости mvLX на 2, отображая одну четвертую единицы образца mvLX в одну восьмую единицы образца mvCLX

$$mvCLX[\ 0\ ] = mvLX[\ 0\ ] \quad (8\text{-}174)$$

$$mvCLX[\ 1\ ] = mvLX[\ 1\ ]. \quad (8\text{-}175)$$

– Иначе (текущий макроблок – макроблок поля), находят только горизонтальную компоненту вектора движения цветности mvCLX[0], используя равенство 8-174. Вертикальная компонента вектора движения цветности mvCLX[1] зависит от четности текущего поля или текущего макроблока и контрольного изображения, которые учитывают индексом контроля refIdxLX. Значение mvCLX[1] находят из mvLX[1] согласно таблице 8-9.

Таблица 8-9 – Отыскание вертикальной компоненты вектора цветности в режиме кодирования поля

Условия четности		**mvCLX[1]**
Контрольное изображение (refIdxLX)	Текущее поле (изображение/макроблок)	
Верхнее поле	Нижнее поле	mvLX[1] + 2
Нижнее поле	Верхнее поле	mvLX[1] – 2
Иначе		mvLX[1]

8.4.2 Процесс декодирования образцов Inter предсказания

Входы в этот процесс:

– часть макроблока mbPartIdx;

– часть субмакроблока subMbPartIdx;

– переменные, определяющие ширину и высоту частей, partWidth и partHeight;

– векторы движения яркости mvL0 и mvL1 и векторы движения цветности mvCL0 и mvCL1;

– индексы контроля refIdxL0 и refIdxL1;

– флаги использованных списков предсказаний predFlagL0 и predFlagL1.

Выходы из этого процесса:

– Образцы Inter предсказания predPart, которые являются массивами (partWidth)x(partHeight) значения $predPart_L$ образцов предсказания яркости и двух массивов (partWidth/2)x(partHeight/2) значений $predPart_{Cb}$, $predPart_{Cr}$ образцов предсказания цветности, по одному для каждого компонента цветности Cb и Cr.

Положим, что $predPartL0_L$ и $predPartL1_L$ – массивы (partWidth)x(partHeight) предсказанных значений образца яркости, а $predPartL0_{Cb}$, $predPartL1_{Cb}$, $predPartL0_{Cr}$ и $predPartL1_{Cr}$ – массивы (partWidth/2)x(partHeight/2) предсказанных значений образца цветности.

При замене LX на L0 или L1 в переменных predFlagLX, RefPicListX, refIdxLX, refPicLX, predPartLX определяют следующее.

Если predFlagLX равно 1, то используют следующие условия:

– Контрольный кадр, состоящий из упорядоченного двухразмерного массива $refPicLX_L$ образцов яркости и двух упорядоченных двухразмерных массивов, $refPicLX_{Cb}$ и $refPicLX_{Cr}$, образцов цветности, находят, активируя процесс, определенный в п. 8.4.2.1, с refIdxLX и RefPicListX, заданными как вход.

– Массивы $predPartLX_L$, $predPartLX_{Cb}$ и $predPartLX_{Cr}$ находят, активируя процесс, определенный в п. 8.4.2.2, с текущей частью, определенной значениями mbPartIdx\subMbPartIdx, векторами движений mvLX, mvCLX и контрольными массивами с $refPicLX_L$, $refPicLX_{Cb}$ и $refPicLX_{Cr}$, заданными как вход.

При замене C на L, Cb или Cr массив $predPart_C$ для образцов предсказания компоненты C находят, активируя процесс, определенный в п. 8.4.2.3, с текущей частью, определенной mbPartIdx и subMbPartIdx и массивами $predPartL0_C$ и $predPartL1_C$, а также predFlagL0 и predFlagL1, заданными как вход.

8.4.2.1 Процесс выбора контрольного изображения

Вход в этот процесс – индекс контроля refIdxLX.

Выход этого процесса – контрольное изображение, состоящее из двухразмерного массива образцов яркости $refPicLX_L$ и двух двухразмерных массивов образцов цветности $refPicLX_{Cb}$ и $refPicLX_{Cr}$.

Список контрольного изображения RefPicListX – это список переменных PicNum (для краткосрочных контрольных изображений) и LongTermPicNum (для долгосрочных контрольных изображений) ранее декодированных контрольных кадров, дополнительной контрольной пары полей или непарного контрольного поля, которые помечены как "использованные для контроля", как определено в п. 8.2.5.

В зависимости от field_pic_flag значения PicNum и LongTermPicNum определяют следующим образом:

– Если field_pic_flag равно 1, все входы RefPicListX – это переменные PicNum и LongTermPicNum декодированного контрольного поля или поля декодированных контрольных кадров.

– Иначе (field_pic_flag равно 0), все входы RefPicListX – это переменные PicNum и LongTermPicNum декодированных контрольных кадров или дополнительной контрольной пары полей.

Список контрольного изображения RefPicListX находят, как определено в п. 8.2.4.

При отыскании контрольного изображения используют следующие условия:

– Если field_pic_flag равно 1, контрольное поле или поле контрольного кадра, на которые ссылаются с помощью PicNum = RefPicListX[refIdxLX] или LongTermPicNum = RefPicListX[refIdxLX], должно быть выходом. Выход контрольного поля или поле контрольного кадра состоит из массива ($PicWidthInSamples_L$)x($PicHeightInSamples_L$) образцов яркости $refPicLX_L$ и двух массивов ($PicWidthInSamples_C$)x($PicHeightInSamples_C$) образцов цветности $refPicLX_{Cb}$ и $refPicLX_{Cr}$.

– Иначе (field_pic_flag равно 0), используют следующие условия:

  – Если текущий макроблок – макроблок кадра, то контрольный кадр или дополнительная контрольная пара полей, задаваемых PicNum = RefPicListX[refIdxLX] или LongTermPicNum = RefPicListX[refIdxLX], должны быть выходом. Выход контрольного кадра или дополнительной контрольной пары полей состоит из массива ($PicWidthInSamples_L$)x($PicHeightInSamples_L$) образцов яркости $refPicLX_L$ и двух массивов ($PicWidthInSamples_C$)x($PicHeightInSamples_C$) образцов цветности $refPicLX_{Cb}$ и $refPicLX_{Cr}$.

  – Иначе (текущий макроблок – макроблок поля), используют следующие условия:

    – Положим, что refFrame – контрольный кадр или дополнительная контрольная пара полей, задаваемых PicNum = RefPicListX[refIdxLX / 2] или LongTermPicNum = RefPicListX[refIdxLX / 2].

    – Поле refFrame выбирают следующим образом:

– Если refIdxLX % 2 равно 0, поле reframe, которое имеет ту же четность, что и у текущего макроблока, должно быть выходом.

– Иначе (refIdxLX % 2 равно 1), поле reframe, которое имеет четность, противоположную четности текущего макроблока, должно быть выходом.

– Выход контрольного поля или поля контрольного кадра состоит из массива (PicWidthInSamples$_L$)x(PicHeightInSamples$_L$/2) образцов яркости refPicLX$_L$ и двух массивов (PicWidthInSamples$_C$)x(PicHeightInSamples$_C$/2) образцов цветности refPicLX$_{Cb}$ и refPicLX$_{Cr}$.

Массивы образцов контрольных изображений refPicLX$_L$, refPicLX$_{Cb}$, refPicLX$_{Cr}$ соответствуют массивам декодированных образцов S_L, S_{Cb}, S_{Cr}, полученных в п. 8.7 для предварительно декодированных изображений.

8.4.2.2 Процесс интерполяции фрагментарного образца

Входы в этот процесс:

– текущее разделение на части, заданное индексом разделения mbPartIdx и субиндексом разделенной части макроблока subMbPartIdx;

– ширина и высота – partWidth, partHeight – этой части в единицах от образца яркости,

– вектор движения яркости mvLX, заданный в одной четвертой единицы от образца яркости,

– вектор движения цветности mvCLX, заданный в одной восьмой единицы от образца цветности, и

– выбранные массивы образцов контрольных изображений refPicLX$_L$, refPicLX$_{Cb}$ и refPicLX$_{Cb}$.

Выходы из этого процесса:

– массив (partWidth)x(partHeight), predPartLX$_L$, значения предсказания образца яркости; и

– два массива (partWidth/2)x(partHeight/2) – predPartLX$_{Cb}$ и predPartLX$_{Cr}$ – значений предсказания образца цветности.

Положим, что (xA_L, yA_L) – местоположение, заданное в полных единицах образца для верхнего левого образца яркости текущего разделения, которое определено значением mbPartIdx\subMbPartIdx относительно верхнего левого положения образца яркости заданного двухразмерного массива образца яркости.

Положим, что ($xInt_L$, $yInt_L$) – местоположение, заданное в полных единицах образца яркости, а ($xFrac_L$, $yFrac_L$) – сдвиг, заданный в единицах четверти образца. Эти переменные используют только в этом пункте для определения общего расположения фракций образца внутри массивов контрольного образца refPicLX$_L$, refPicLX$_{Cb}$ и refPicLX$_{Cr}$.

Для каждого расположения образца яркости ($0 <= x_L < partWidth$, $0 <= y_L < partHeight$) внутри массива предсказанного образца яркости predLX$_L$ значение соответствующего предсказанного образца яркости predLX$_L$[x_L, y_L] находят следующим образом:

$$xInt_L = xA_L + (mvLX[0] >> 2) + x_L \quad (8\text{-}176)$$
$$yInt_L = yA_L + (mvLX[1] >> 2) + y_L \quad (8\text{-}177)$$

$$xFrac_L = mvLX[0] \,\&\, 3 \quad (8\text{-}178)$$
$$yFrac_L = mvLX[1] \,\&\, 3. \quad (8\text{-}179)$$

– Значение предсказанного образца predLX$_L$[x_L, y_L] находят, активируя процесс, определенный в п. 8.4.2.2.1, с ($xInt_L$, $yInt_L$), ($xFrac_L$, $yFrac_L$) и refPicLX$_L$, заданными как вход.

Положим, что ($xInt_C$, $yInt_C$) – местоположение, заданное в полных единицах образца цветности, а ($xFrac_C$, $yFrac_C$) – сдвиг, заданный в одной восьмой единицы образца. Эти переменные используют только в этом пункте для определения общего расположения фракций образца внутри массивов контрольного образца refPicLX$_{Cb}$ и refPicLX$_{Cr}$.

Для каждого местоположения образца цветности ($0 <= x_C < partWidth/2$, $0 <= y_C < partHeight/2$) внутри массивов предсказанного образца цветности predPartLX$_{Cb}$ и predPartLX$_{Cr}$ соответствующие значения предсказанных образцов цветности predPartLX$_{Cb}$[x_C, y_C] и predPartLX$_{Cr}$[x_C, y_C] находят следующим образом:

$$xInt_C = (xA_L >> 1) + (mvCLX[0] >> 3) + x_C \quad (8\text{-}180)$$
$$yInt_C = (yA_L >> 1) + (mvCLX[1] >> 3) + y_C \quad (8\text{-}181)$$

$$xFrac_C = mvCLX[0] \,\&\, 7 \quad (8\text{-}182)$$
$$yFrac_C = mvCLX[1] \,\&\, 7. \quad (8\text{-}183)$$

– Предсказанное значение образца predPartLX$_{Cb}$[x_C, y_C] находят, активируя процесс, определенный в п. 8.4.2.2.2, с ($xInt_C$, $yInt_C$), ($xFrac_C$, $yFrac_C$) и refPicLX$_{Cb}$, заданными как вход.

– Предсказанное значение образца predPartLX$_{Cr}$[x_C, y_C] находят, активируя процесс, определенный в п. 8.4.2.2.2, с (xInt$_C$, yInt$_C$), (xFrac$_C$, yFrac$_C$) и refPicLX$_{Cr}$, заданными как вход.

8.4.2.2.1 Процесс интерполяции образца яркости

Входы в этот процесс:

– местоположение яркости в полных единицах образца (xInt$_L$, yInt$_L$),

– местоположение сдвига яркости в единицах фракций образца (xFrac$_L$, yFrac$_L$), и

– массив образца яркости выбранного контрольного изображения refPicLX$_L$.

Выход этого процесса – предсказанное значение образца яркости predPartLX$_L$[x_L, y_L].

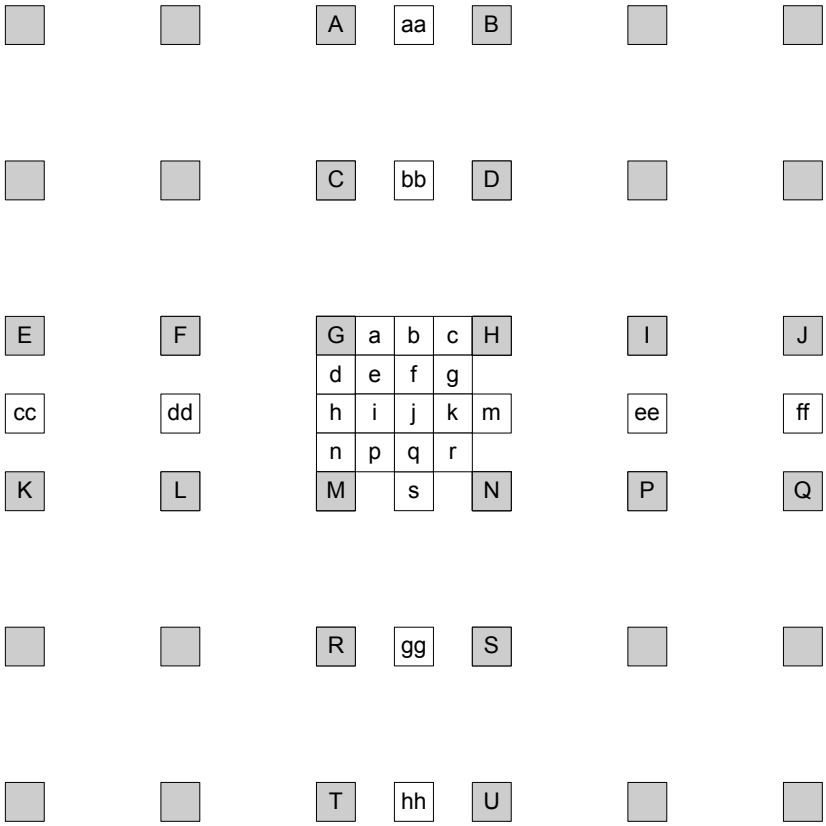

Рисунок 8-4 – Расположение целых образцов (затененные блоки с заглавными буквами) и фракций образцов (незатененные блоки со строчными буквами) для интерполяции четверти образца яркости

Переменную refPicHeightEffective$_L$, которая представляет высоту действующего массива контрольного изображения яркости, находят следующим образом:

– Если MbaffFrameFlag равно 0 или mb_field_decoding_flag равно 0, refPicHeightEffective$_L$ устанавливают равным PicHeightInSamples$_L$.

– Иначе (MbaffFrameFlag равно 1 и mb_field_decoding_flag равно 1), refPicHeightEffective$_L$ устанавливают равным PicHeightInSamples$_L$ / 2.

На рисунке 8-4 положения, обозначенные заглавными буквами в затененных блоках, представляют положения образцов яркости в полном образце внутри заданного двухразмерного массива refPicLX$_L$ образцов яркости. Эти образцы могут быть использованы для создания предсказаний значения образца яркости predPartLX$_L$[x_L, y_L]. Положения (xZ$_L$, yZ$_L$) для каждого из соответствующих образцов яркости Z (где Z может быть A, B, C, D, E, F, G, H, I, J, K, L, M, N, P, Q, R, S, T или U внутри заданного массива refPicLX$_L$ образцов яркости) находят следующим образом:

$$xZ_L = \text{Clip3}(0, \text{PicWidthInSamples}_L - 1, \text{xInt}_L + \text{xDZ}_L)$$
$$yZ_L = \text{Clip3}(0, \text{refPicHeightEffective}_L - 1, \text{yInt}_L + \text{yDZ}_L). \quad (8\text{-}184)$$

Таблица 8-10 определяет значения (xDZ_L, yDZ_L) для различных расположений Z.

Таблица 8-10 – Различное расположение полного образца яркости

Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	P	Q	R	S	T	U
xDZ_L	0	1	0	1	–2	–1	0	1	2	3	–2	–1	0	1	2	3	0	1	0	1
yDZ_L	–2	–2	–1	–1	0	0	0	0	0	0	1	1	1	1	1	1	2	2	3	3

При заданных образцах яркости от "A" до "U" в положениях полных образцов от (xA_L, yA_L) до (xU_L, yU_L) образцы яркости от "a" до "s" в положениях фракций образцов положения находят по следующим правилам. Значения предсказания яркости для половины положений образцов должны быть получены с применением фильтра с 6-ю отводами со значениями отводов (1, –5, 20, 20, –5, 1). Значения предсказания яркости для положения четверти образца должны быть получены усреднением положений полных образцов и половин образцов. Процесс для каждого положения фракции описан ниже.

– Образцы в положениях половины образца, обозначенные как b, должны быть найдены при первом вычислении промежуточного значения, обозначенного как b_1, присоединением 6-отводного фильтра к ближайшему целочисленному положению образцов в горизонтальном направлении. Образцы в положениях половины образца, обозначенные как h, должны быть найдены при первом вычислении промежуточного значения, обозначенного как h_1, присоединением 6-отводного фильтра к ближайшему целочисленному положению образцов в вертикальном направлении:

$$b_1 = (E - 5 * F + 20 * G + 20 * H - 5 * I + J) \quad (8\text{-}185)$$
$$h_1 = (A - 5 * C + 20 * G + 20 * M - 5 * R + T). \quad (8\text{-}186)$$

Окончательное значение предсказания b и h должно быть получено, используя равенства:

$$b = Clip1((b_1 + 16) >> 5) \quad (8\text{-}187)$$
$$h = Clip1((h_1 + 16) >> 5). \quad (8\text{-}188)$$

– Образцы в положениях половины образца, обозначенные как j, должны быть найдены при первом вычислении промежуточного значения, обозначенного как j_1, присоединением 6-отводного фильтра к ближайшему целочисленному положению образцов либо в горизонтальном, либо в вертикальном направлении, поскольку это даст одинаковый результат.

$$j_1 = cc - 5 * dd + 20 * h_1 + 20 * m_1 - 5 * ee + ff, \text{ или} \quad (8\text{-}189)$$
$$j_1 = aa - 5 * bb + 20 * b_1 + 20 * s_1 - 5 * gg + hh, \quad (8\text{-}190)$$

где предварительные значения, обозначенные как aa, bb, gg, s_1 и hh, должны быть получены с применением фильтра с 6-ю отводами горизонтально тем же способом, что и нахождение b_1. Предварительные значения, обозначенные как cc, dd, ee, m_1 и ff, должны быть получены с применением фильтра с 6-ю отводами вертикально тем же способом, что и нахождение h_1.Окончательное значение предсказания j должно быть получено, используя уравнение:

$$j = Clip1((j_1 + 512) >> 10). \quad (8\text{-}191)$$

– Окончательные значения предсказания s и m должны быть получены из s_1 и m_1 тем же способом, что и нахождение b и h, как:

$$s = Clip1((s_1 + 16) >> 5) \quad (8\text{-}192)$$
$$m = Clip1((m_1 + 16) >> 5). \quad (8\text{-}193)$$

– Образцы в положениях четверти образца, обозначенные как a, c, d, n, f, i, k и q, должны быть получены усреднением с округлением в большую сторону двух ближайших образцов в положениях целых и половин образцов, используя уравнения:

$$a = (G + b + 1) >> 1 \quad (8\text{-}194)$$
$$c = (H + b + 1) >> 1 \quad (8\text{-}195)$$
$$d = (G + h + 1) >> 1 \quad (8\text{-}196)$$
$$n = (M + h + 1) >> 1 \quad (8\text{-}197)$$
$$f = (b + j + 1) >> 1 \quad (8\text{-}198)$$
$$i = (h + j + 1) >> 1 \quad (8\text{-}199)$$
$$k = (j + m + 1) >> 1 \quad (8\text{-}200)$$
$$q = (j + s + 1) >> 1. \quad (8\text{-}201)$$

– Образцы в положениях четверти образца, обозначенные как e, g, p и r, должны быть получены усреднением с округлением в большую сторону двух ближайших образцов в положениях половин образцов в диагональном направлении, используя уравнения:

$$e = (b + h + 1) >> 1 \quad (8\text{-}202)$$
$$g = (b + m + 1) >> 1 \quad (8\text{-}203)$$
$$p = (h + s + 1) >> 1 \quad (8\text{-}204)$$
$$r = (m + s + 1) >> 1. \quad (8\text{-}205)$$

Сдвиг расположения яркости фракции в единицах образца ($xFrac_L$, $yFrac_L$) определяет, какой из созданных образцов яркости в положениях полного образца и фракции образца присвоен значению предсказанного образца яркости $predPartLX_L[x_L, y_L]$. Это присвоение выполняют согласно таблице 8-11. Значение $predPartLX_L[x_L, y_L]$ должно быть выходом.

Таблица 8-11 – Присвоение предсказанного образца яркости $predPartLX_L[x_L, y_L]$

$xFrac_L$	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
$yFrac_L$	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
$predPartLX_L[x_L, y_L]$	G	D	h	n	a	e	i	p	b	f	j	q	c	g	k	r

8.4.2.2.2 Процесс интерполяции образца цветности

Входы в этот процесс:

– расположение цветности в единицах полного образца ($xInt_C$, $yInt_C$),

– сдвиг расположения цветности в единицах фракции образца ($xFrac_C$, $yFrac_C$), и

– компоненты образцов цветности из выбранного контрольного изображения $refPicLX_C$.

Выход этого процесса – значение предсказанного образца цветности $predPartLX_C[x_C, y_C]$. На рисунке 8-5 положения, обозначенные как A, B, C и D, представляют образцы цветности в расположении полного образца внутри заданного двухразмерного массива $refPicLX_C$ образцов цветности.

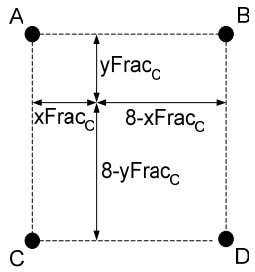

Рисунок 8-5 – Расположение фракций образца в зависимости от переменных в интерполяции цветности и окружающих целочисленных положений образцов A, B, C и D

Переменную $refPicHeightEffective_C$, которая выражает высоту действующего массива контрольного изображения цветности, находят следующим образом:

– Если MbaffFrameFlag равно 0 или mb_field_decoding_flag равно 0, то $refPicHeightEffective_C$ устанавливают равным $PicHeightInSamples_C$.

– Иначе (MbaffFrameFlag равно 1 и mb_field_dicoding_flag равно 1), $refPicHeightEffective_C$ устанавливают равным $PicHeightInSamples_C$ / 2.

Координаты образца, определенные в равенствах 8-206–8-213, используют для создания предсказанного значения образца цветности $predPartLX_C[x_C, y_C]$.

$$xA_C = Clip3(0, PicWidthInSamples_C – 1, xInt_C) \quad (8\text{-}206)$$
$$xB_C = Clip3(0, PicWidthInSamples_C – 1, xInt_C + 1) \quad (8\text{-}207)$$

$$xC_C = Clip3(0, PicWidthInSamples_C – 1, xInt_C) \quad (8\text{-}208)$$
$$xD_C = Clip3(0, PicWidthInSamples_C – 1, xInt_C + 1) \quad (8\text{-}209)$$

$$yA_C = Clip3(0, refPicHeightEffective_C – 1, yInt_C) \quad (8\text{-}210)$$
$$yB_C = Clip3(0, refPicHeightEffective_C – 1, yInt_C) \quad (8\text{-}211)$$
$$yC_C = Clip3(0, refPicHeightEffective_C – 1, yInt_C + 1) \quad (8\text{-}212)$$
$$yD_C = Clip3(0, refPicHeightEffective_C – 1, yInt_C + 1). \quad (8\text{-}213)$$

При заданных образцах цветности A, B, C и D в положениях полных образцов, определенных равенствами от 8-206 до 8-213, значение предсказанного образца цветности $predPartLX_C[x_C, y_C]$ находят следующим образом:

$$predPartLX_C[x_C, y_C] = ((8 – xFrac_C) * (8 – yFrac_C) * A + xFrac_C * (8 – yFrac_C) * B + (8 – xFrac_C) * yFrac_C * C + xFrac_C * yFrac_C * D + 32) >> 6. \quad (8\text{-}214)$$

8.4.2.3 Процесс взвешенного предсказания образца

Входы в этот процесс:

– mbPartIdx: текущее разделение, заданное индексом разделения,

– subMbPartIdx: субиндекс разделенной части макроблока,

– predFlagL0 и predFlagL1: флаги использованных списков предсказаний,

– $predPartLX_L$: массив (partWidth)x(partHeight) предсказанных образцов яркости (с заменой LX на L0 или L1 в зависимости от predFlagL0 и predFlagL1),

– $predPartLX_{Cb}$ и $predPartLX_{Cr}$: массивы (partWidth/2)x(partHeight/2) предсказанных образцов цветности, по одному для каждого из компонентов цветности Cb и Cr (с заменой LX на L0 или L1 в зависимости от predFlagL0 и predFlagL1).

Выходы из этого процесса:

– $predPart_L$: массив (partWidth)x(partHeight) предсказанных образцов яркости и

– $predPart_{Cb}$ и $predPart_{Cr}$: массивы (partWidth/2)x(partHeight/2) предсказанных образцов цветности, по одному для каждого из компонентов цветности Cb и Cr.

Для макроблоков или разделенных частей с predFlagL0, равным 1, в секциях P и SP используют следующие условия:

– Если weighted_pred_flag равно 0, процесс взвешенного по умолчанию предсказания образца, как описано в п. 8.4.2.3.1, активируют с теми же входами и выходами, что и в описанном в этом пункте процессе.

– Иначе (weighted_pred_flag равно 1), подробный взвешенный процесс предсказания, как описано в п. 8.4.2.3.2, активируют с теми же входами и выходами, что и в описанном в этом пункте процессе.

Для макроблоков или разделенных частей с predFlagL0 или predFlagL1, равными 1, в секциях B используют следующие условия:

– Если weighted_bipred_idc равно 0, процесс взвешенного по умолчанию предсказания образца, как описано в п. 8.4.2.3.1, активируют с теми же входами и выходами, что и в описанном в этом пункте процессе.

– Иначе, если weighted_bipred_idc равно 1, подробный взвешенный процесс предсказания, как описано в п. 8.4.2.3.2, для макроблоков или разделенных частей с predFlagL0 или predFlagL1, равными 1, активируют с теми же входами и выходами, что и в описанном в этом пункте процессе.

– Иначе (weighted_bipred_idc равно 2), используют следующие условия:

  – Если predFlagL0 равно 1 и predFlagL1 равно 1, неявный взвешенный образец предсказания, как описано в п. 8.4.2.3.2, активируют с теми же входами и выходами, что и в описанном в этом пункте процессе.

  – Иначе (predFlagL0 или predFlagL1 равны 1, но не оба вместе), процесс взвешенного по умолчанию предсказания образца, как описано в п. 8.4.2.3.1, активируют с теми же входами и выходами, что и в описанном в этом пункте процессе.

8.4.2.3.1 Процесс взвешенного по умолчанию предсказания образца

Входы в этот процесс – те же, что и в п. 8.4.2.3.

Выходы этого процесса – те же, что и в п. 8.4.2.3.

В зависимости от компонента, для которого находят блок предсказания, используют следующие условия:

– Если предсказанные значения образца яркости $predPart_L[x, y]$ найдены, используют следующие условия: значение C установлено равным L, x – установлено равным 0 .. partWidth – 1, а y – установлено равным 0 .. partHeight – 1.

– Иначе, если предсказанные значения компонента образца цветности Cb $predPart_{Cb}[x, y]$ найдены, используют следующие условия: C установлено равным Cb, x установлено равным 0 .. partWidth / 2 – 1 и y установлено равным 0 .. partHeight / 2 – 1.

– Иначе (предсказанные значения компонента образца цветности Cr $predPart_{Cr}[x, y]$ найдены), используют следующие условия: C установлено равным Cr, x установлено равным 0 .. partWidth / 2 – 1 и y установлено равным 0 .. partHeight / 2 – 1.

Предсказанное значение образца находят следующим образом:

– Если predFlagL0 равно 1 и predFlagL1 равно 0, то для текущего разделения

$$predPart_C[x, y] = predPartL0_C[x, y]. \quad (8\text{-}215)$$

– Иначе, если predFlagL0 равно 0 и predFlagL1 равно 1, то для текущего разделения,

$$predPart_C[x, y] = predPartL1_C[x, y]. \quad (8\text{-}216)$$

– Иначе (predFlagL0 и predFlagL1 равны 1 для текущего разделения),

$$predPart_C[x, y] = (predPartL0_C[x, y] + predPartL1_C[x, y] + 1) >> 1. \quad (8\text{-}217)$$

8.4.2.3.2 Процесс взвешенного предсказания образца

Входы в этот процесс – те же, что и в п. 8.4.2.3.

Выходы этого процесса – те же, что и в п. 8.4.2.3.

В зависимости от компонента, для которого находят блок предсказания, используют следующие условия:

– Если предсказанные значения образца яркости $predPart_L[x, y]$ найдены, используют следующие условия: C установлено равным L, x установлено равным 0 .. partWidth – 1 и y установлено равным 0 .. partHeight – 1.

– Иначе, если предсказанные значения компонента образца цветности Cb $predPart_{Cb}[x, y]$ найдены, используют следующие условия: C установлено равным Cb, x установлено равным 0 .. partWidth / 2 – 1, а y установлено равным 0 .. partHeight / 2 – 1.

– Иначе (предсказанные значения компонента образца цветност Cr $predPart_{Cr}[x, y]$ найдены), используют следующие условия: C установлено равным Cr, x установлено равным 0 .. partWidth / 2 – 1, а y установлено равным 0 .. partHeight / 2 – 1.

Предсказанные значения образца находят следующим образом:

– Если разделение mbPartIdx\subMbPartIdx имеет predFlagL0, равное 1, а predFlagL1 равно 0, то окончательные предсказанные значения образца $predPart_C[x, y]$ находят как

$$\begin{aligned} &\text{if(logWD >= 1)} \\ &\quad predPart_C[x, y] = Clip1(((predPartL0_C[x, y] * w_0 + 2^{logWD - 1}) >> logWD) + o_0) \\ &\text{else} \\ &\quad predPart_C[x, y] = Clip1(predPartL0_C[x, y] * w_0 + o_0). \end{aligned} \quad (8\text{-}218)$$

– Иначе, если разделение mbPartIdx\subMbPartIdx имеет predFlagL0, равное 0, а predFlagL1 равно 1, то окончательные предсказанные значения образца $predPart_C[x, y]$ находят как

$$\begin{aligned} &\text{if(logWD >= 1)} \\ &\quad predPart_C[x, y] = Clip1(((predPartL1_C[x, y] * w_1 + 2^{logWD - 1}) >> logWD) + o_1) \\ &\text{else} \\ &\quad predPart_C[x, y] = Clip1(predPartL1_C[x, y] * w_1 + o_1). \end{aligned} \quad (8\text{-}219)$$

– Иначе (разделение mbPartIdx\subMbPartIdx имеет оба значения, predFlagL0 и predFlagL1, равными 1), окончательные предсказанные значения образца $predPart_C[x, y]$ находят как

$$\begin{aligned} predPart_C[x, y] = Clip1(((predPartL0_C[x, y] * w_0 + predPartL1_C[x, y] * w_1 + 2^{logWD}) >> \\ (logWD + 1)) + ((o_0 + o_1 + 1) >> 1)). \end{aligned} \quad (8\text{-}220)$$

Переменные в вышеприведенных соотношениях для предсказанных образцов находят следующим образом:

– Если weighted_bipred_idc равно 2, а slice_type равно B, то неявный режим взвешенного предсказания используют следующим образом:

$$logWD = 5 \quad (8\text{-}221)$$

$o_0 = 0$ (8-222)

$o_1 = 0,$ (8-223)

а w_0 и w_1 находят следующим образом:

– Переменные currPicOrField, pic0 и pic1 находят следующим образом:
  – Если field_pic_flag равно 0, а текущий макроблок – макроблок поля, используют следующие условия:
    – currPicOrField – это поле текущего изображения CurrPic, которое имеет ту же четность, что и у текущего макроблока.
    – Положим, что frame0 – кадр или дополнительная пара полей, на которую ссылаются с помощью RefPicList0[refIdxL0 / 2].
    – Переменную pic0 находят следующим образом:
      – Если refIdxL0 % 2 равно 0, pic0 – поле frame0, которое имеет ту же четность, что и у текущего макроблока.
      – Иначе (refIdxL0 % 2 не равно 0), pic0 – поле frame0, которое имеет четность, противоположную четности текущего макроблока.
    – Положим, что frame1 – кадр или дополнительная пара полей, на которую ссылаются с помощью RefPicList1[refIdxL1 / 2].
    – Переменную pic1 находят следующим образом:
      – Если refIdxL1 % 2 равно 0, pic1 – поле frame1, которое имеет ту же четность, что и у текущего макроблока.
      – Иначе (refIdxL1 % 2 не равно 0), pic1 – поле frame1, которое имеет четность, противоположную четности текущего макроблока.
  – Иначе (field_pic_flag равно 1 или текущий макроблок – макроблок кадра), currPicOrField – текущее изображение CurrPic, pic1 – декодированное контрольное изображение, на которое ссылаются с помощью RefPicList1[refIdxL1], а pic0 – декодированное контрольное изображение, на которое ссылаются с помощью RefPicList0[refIdxL0].
– Переменные tb, td, tx и DistScaleFactor находят из значений currPicOrField, pic0, pic1, используя равенства 8-154, 8-155, 8-150 и 8-151 соответственно.
– Если DiffPicOrderCnt(pic1, pic0) равно 0 или если одно или оба значения pic1 и pic0 – долгосрочные контрольные изображения или (DistScaleFactor >> 2) < -64 или (DistScaleFactor >> 2) > 128, то w_0 и w_0 находят как

  $w_0 = 32$ (8-224)

  $w_1 = 32.$ (8-225)

– Иначе

  $w_0 = 64 – (DistScaleFactor >> 2)$ (8-226)

  $w_1 = DistScaleFactor >> 2.$ (8-227)

– Иначе (weighted_pred_flag равно 1 в секциях P или SP или weighted_bipred_idc равно 1 в секциях B), явный режим взвешенного предсказания используют следующим образом:
  – Переменные refIdxL0WP и refIdxL1WP находят следующим образом:
    – Если MbaffFrameFlag равно 1, а текущий макроблок – макроблок поля, то

      refIdxL0WP = refIdxL0 >> 1 (8-228)

      refIdxL1WP = refIdxL1 >> 1. (8-229)

    – Иначе (MbaffFrameFlag равно 0 или текущий макроблок – макроблок кадра),

      refIdxL0WP = refIdxL0 (8-230)

      refIdxL1WP = refIdxL1. (8-231)

– Переменные logWD, w_0, w_1, o_0 и o_1 находят следующим образом:

  – Если C в $predPart_C[x, y]$ заменяют на L для образцов яркости, то

    logWD = luma_log2_weight_denom (8-232)

    w_0 = luma_weight_l0[refIdxL0WP] (8-233)

    w_1 = luma_weight_l1[refIdxL1WP] (8-234)

    o_0 = luma_offset_l0[refIdxL0WP] (8-235)

    o_1 = luma_offset_l1[refIdxL1WP]. (8-236)

  – Иначе (C в $predPart_C[x, y]$ заменяют на Cb или Cr для образцов цветности, с iCbCr = 0 для Cb, iCbCr = 1 для Cr),

    logWD = chroma_log2_weight_denom (8-237)

    w_0 = chroma_weight_l0[refIdxL0WP][iCbCr] (8-238)

    w_1 = chroma_weight_l1[refIdxL1WP][iCbCr] (8-239)

    o_0 = chroma_offset_l0[refIdxL0WP][iCbCr] (8-240)

    o_1 = chroma_offset_l1[refIdxL1WP][iCbCr]. (8-241)

Если используют явный режим взвешенного предсказания, а разделение mbPartIdx\subMbPartIdx имеет оба значения, predFlagL0 и predFlagL1, равные 1, то должно действовать следующее ограничение

$$–128 <= w_0 + w_1 <= ((logWD = = 7) ? 127 : 128). \quad (8\text{-}242)$$

ПРИМЕЧАНИЕ. – Для явного режима взвешенного предсказания каждый вес, w_0 и w_1, гарантировано находится в диапазоне –64..128, а ограничение, выраженное в равенстве 8-242, хотя это явно и не указано, всегда будет выполняться. Для явного режима взвешенного предсказания с logWD, равным 7, если один из двух весов, w_0 или w_1, считают равным 128 (поскольку последовательность luma_weight_l0_flag, luma_weight_l1_flag, chroma_weight_l0_flag, или chroma_weight_l1_flag, равна 0), другой вес (w_1 или w_0) должен иметь отрицательное значение в порядке ограничения, наложенного на равенство 8-242 (и поэтому другое значение флага luma_weight_l0_flag, luma_weight_l1_flag, chroma_weight_l0_flag или chroma_weight_l1_flag должно быть равно 1).

8.5 Процесс декодирования коэффициента преобразования и процесс построения изображения перед фильтровым разделением на блоки

Входы в этот процесс – Intra16x16DCLevel (если доступно), Intra16x16ACLevel (если доступно), LumaLevel (если доступно), ChromaDCLevel, ChromaACLevel и доступные массивы образцов Inter или Intra предсказаний текущего макроблока с доступными компонентами $pred_L$, $pred_{Cb}$ или $pred_{Cr}$.

ПРИМЕЧАНИЕ. – При декодировании макроблока в режиме предсказания Intra_4x4 массив предсказания компонентов яркости макроблока может быть неполным, поскольку для каждого блока яркости 4x4 процесс предсказания Intra_4x4 для образцов яркости, как это определено в п. 8.3.1, и процесс, определенный в этом пункте, являются итеративными.

Выходы из этого процесса – построенные массивы образцов перед процессом фильтрового разделения на блоки соответствующих компонентов S'_L, S'_{Cb} или S'_{Cr}.

ПРИМЕЧАНИЕ. – При декодировании макроблока в режиме предсказания Intra_4x4 компонент яркости массивов образцов макроблоков, построенных перед процессом фильтрового разделения на блоки, может быть неполным, поскольку для каждого блока яркости 4x4 процесс предсказания Intra_4x4 для образцов яркости, как это определено в п. 8.3.1, и процесс, определенный в этом пункте, являются итеративными.

В этом пункте определяют декодирование коэффициента преобразования и построение изображения перед фильтровым разделением на блоки.

Если текущий макроблок кодирован как P_Skip или B_Skip, то для текущего макроблока все значения LumaLevel, ChromaDCLevel, ChromaACLevel устанавливают равными 0.

8.5.1 Спецификация процесса декодирования преобразования остальных блоков

Если режим предсказания текущего макроблока не равен Intra_16x16, переменная LumaLevel содержит уровни для коэффициентов преобразования яркости. Для блока яркости 4x4, индексированного luma4x4BlkIdx = 0..15, определены следующие шаги:

1. Активируют процесс инверсного сканирования коэффициентов преобразования, как описано в п. 8.5.4, с LumaLevel[luma4x4BlkIdx] в качестве входа и двухразмерным массивом в качестве выхода.

2. Активируют процесс масштабирования и преобразования оставшихся блоков 4x4, как это определено в п. 8.5.8, со значениями 'c' в качестве входа и 'r' в качестве выхода.

3. Положение верхнего левого образца блока яркости 4x4 с индексом luma4x4BlkIdx внутри макроблока находят, активируя процесс инверсного сканирования блока яркости 4x4, согласно п. 6.4.3 с luma4x4BlkIdx в качестве входа и с выходом, присвоенным значению (xO, yO).

4. Массив u 4x4 с элементами u_{ij} для i, j = 0..3 находят как

   $u_{ij} = Clip1(pred_L[xO + j, yO + i] + r_{ij})$. (8-243)

5. Активируют процесс построения изображения перед процессом фильтрового разделения на блоки, как указано в п. 8.5.9, с luma4x4BlkIdx и 'u' в качестве входа и S' в качестве выхода.

8.5.2 Спецификация процесса декодирования преобразования для образцов яркости режима предсказания макроблока Intra_16x16

Если режим предсказания текущего макроблока равен Intra_16x16, переменные Intra16x16DCLevel и Intra16x16ACLevel содержат уровни коэффициентов преобразования яркости. Декодирование коэффициентов преобразования производят следующими ступенями:

1. Декодируют DC коэффициенты преобразования яркости 4x4 всех блоков яркости 4x4 макроблока.

   a. Активируют процесс инверсного сканирования коэффициентов преобразования, как описано в п. 8.5.4, с Intra16x16DCLevel в качестве входа и двухразмерным массивом с в качестве выхода.

   b. Активируют процесс масштабирования и преобразования DC коэффициентов преобразования яркости для типа макроблока Intra_16x16, как определено в п. 8.5.6, со значением 'c' в качестве входа и 'dcY' в качестве выхода.

2. Для блока яркости 4x4, индексированного luma4x4BlkIdx = 0..15, определена такая последовательность действий.

   a. Находят переменную lumaList, которая является списком из 16 входов. Первый вход lumaList представляет соответствующее значение из массива dcY. На рисунке 8-6 показано присвоение индексов массива dcY значению luma4x4BlkIdx. Две цифры в малых квадратах относятся к индексам i и j в dcY_{ij}, а цифры в больших квадратах относятся к luma4x4BlkIdx.

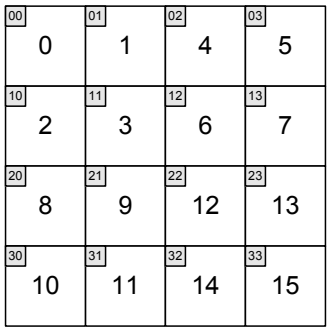

Рисунок 8-6 – Присвоение индексов массива dcY значению luma4x4BlkIdx

   Элементы в lumaList с индексами k = 1..15 определены как

   lumaList[k] = Intra16x16ACLevel[luma4x4BlkIdx][k – 1]. (8-244)

   b. Процесс сканирования инверсного преобразования коэффициентов, как описано в п. 8.5.4, активируют с lumaList в качестве входа и двухразмерным массивом 'c' в качестве выхода.

   c. Процесс масштабирования и преобразования для остальных блоков 4x4, как определено в п. 8.5.8, активируют со значением 'c' в качестве входа и 'r' в качестве выхода.

d. Положение верхнего левого образца блока яркости 4x4 с индексом luma4x4BlkIdx внутри макроблока находят, активируя процесс инверсного сканирования блока яркости 4x4 в п. 6.4.3, с luma4x4BlkIdx в качестве входа и с выходом, присвоенным значению (xO, yO).

e. Массив u 4x4 с элементами u_{ij} для i, j = 0..3 находят как

$$u_{ij} = \text{Clip1}(\text{pred}_L[xO + j, yO + i] + r_{ij}). \quad (8\text{-}245)$$

f. Активируют процесс построения изображения перед процессом фильтрового разделения на блоки, как указано в п. 8.5.9, с luma4x4BlkIdx, 'u' в качестве входа и S' в качестве выхода.

8.5.3 Спецификация преобразования процесса декодирования образцов цветности

Для каждого компонента цветности переменные ChromaDCLevel[iCbCr] и ChromaACLevel[iCbCr] с iCbCr, установленным равным 0 для Cb, и iCbCr, установленным равным 1 для Cr, содержат уровни для обоих компонентов коэффициентов преобразования цветности. Для каждого компонента цветности декодирование преобразования производят раздельно следующими шагами:

1. Декодируют DC коэффициенты преобразования цветности 2x2 блоков цветности 4x4 с компонентом индексированного iCbCr макроблока:

   a. Массив c 2x2 находят, используя процесс инверсного растрового сканирования, применяемый к ChromaDCLevel следующим образом:

$$c = \begin{bmatrix} \text{ChromaDCLevel}[\text{iCbCr}][0] & \text{ChromaDCLevel}[\text{iCbCr}][1] \\ \text{ChromaDCLevel}[\text{iCbCr}][2] & \text{ChromaDCLevel}[\text{iCbCr}][3] \end{bmatrix}. \quad (8\text{-}246)$$

   b. Процесс масштабирования и преобразования для DC коэффициентов преобразования цветности, как определено в п. 8.5.7, активируют с 'c' в качестве входа и dcC в качестве выхода.

2. Для каждого блока цветности 4x4 индексированного chroma4x4BlkIdx = 0..3 компонента индексированного iCbCr определены следующие ступени:

   a. Находят переменную chromaList, которая является списком из 16 входов. Первый вход chromaList – это соответствующее значение из массива dcC. На рисунке 8-7 показано присвоение индексов массива dcC значению chroma4x4BlkIdx. Две цифры в малых квадратах относятся к индексам i и j в dcC_{ij}, а цифры в больших квадратах – к chroma4x4BlkIdx.

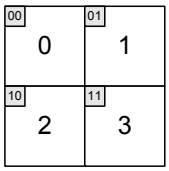

Рисунок 8-7 – Присвоение индексов массива dcC значению chroma4x4BlkIdx

Элементы в chromaList с индексами k = 1..15 определены как

$$\text{chromaList}[k] = \text{ChromaACLevel}[\text{chroma4x4BlkIdx}][k - 1]. \quad (8\text{-}247)$$

   b. Активируют процесс инверсного сканирования коэффициентов преобразования, как описано в п. 8.5.4, с chromaList в качестве входа и двухразмерным массивом c в качестве выхода.

   c. Активируют процессы масштабирования и преобразования для остальных блоков 4x4, как определено в п. 8.5.8, с 'c' в качестве входа и 'r' в качестве выхода.

   d. Положение верхнего левого образца блока цветности 4x4 с индексом chroma4x4BlkIdx внутри макроблока находят следующим образом:

$$xO = \text{InverseRasterScan}(\text{chroma4x4BlkIdx}, 4, 4, 8, 0) \quad (8\text{-}248)$$

$$yO = \text{InverseRasterScan}(\text{chroma4x4BlkIdx}, 4, 4, 8, 1). \quad (8\text{-}249)$$

   e. Массив u 4x4 с элементами u_{ij} для i, j = 0..3 находят как

$$u_{ij} = \text{Clip1}(\text{pred}_C[xO + j, yO + i] + r_{ij}). \quad (8\text{-}250)$$

f. Процесс построения изображения перед процессом фильтрового разделения на блоки согласно п. 8.5.9 активируют с chroma4x4BlkIdx, 'u' в качестве входа и S' в качестве выхода.

8.5.4 Процесс инверсного сканирования коэффициентов преобразования

Вход в этот процесс – это список из 16 значений.

Выход этого процесса – переменная 'c', содержащая двухразмерный массив 4x4 значений уровней 'c', заданных положениям в блоке преобразования.

Процесс декодирования отображает последовательность уровней коэффициентов преобразования в положения уровней коэффициентов преобразования. Для этого отображения используют два инверсно сканированных образца, показанных на рисунке 8-8.

Инверсное сканирование зигзагом должно использоваться для макроблоков кадра, а инверсное сканирование полем должно использоваться для макроблоков поля.

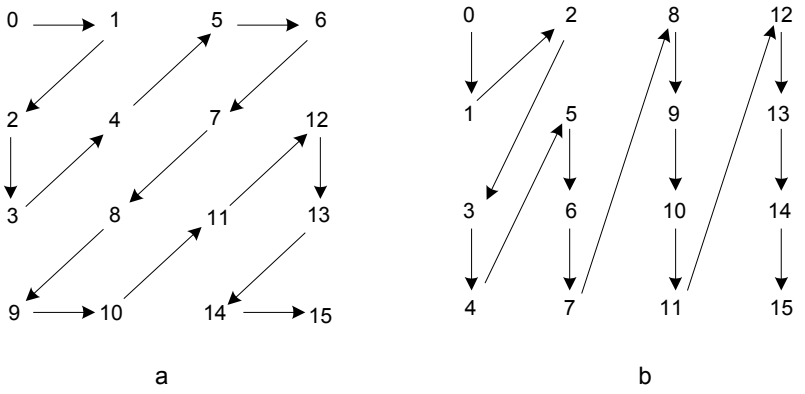

Рисунок 8-8 – a) Сканирование зигзагом. b) Сканирование полем

В таблице 8-12 показано отображение индексов idx списка входа из 16 элементов в индексы i и j двухразмерного массива 'c'.

Таблица 8-12 – Спецификация отображения idx в c_{ij} для сканирования зигзагом и полем

idx	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
зигзаг	c_{00}	c_{01}	c_{10}	c_{20}	c_{11}	c_{02}	c_{03}	c_{12}	c_{21}	c_{30}	c_{31}	c_{22}	c_{13}	c_{23}	c_{32}	c_{33}
поле	c_{00}	c_{10}	c_{01}	c_{20}	c_{30}	c_{11}	c_{21}	c_{31}	c_{02}	c_{12}	c_{22}	c_{32}	c_{03}	c_{13}	c_{23}	c_{33}

8.5.5 Процесс отыскания параметров квантования цветности и функции масштабирования

Выходы из этого процесса:

– QP_C: параметр квантования цветности,

– QS_C: дополнительный параметр квантования цветности, который требуется для декодирования секций SP и SI (если применяют).

ПРИМЕЧАНИЕ. – Значения QP параметров квантования QP_Y, QP_C, QS_Y и QS_C всегда находятся в диапазоне от 0 до 51 включительно.

Значение QP_C для цветности определяют из текущего значения QP_Y и значения chroma_qp_index_offset.

ПРИМЕЧАНИЕ. – Уравнения для масштабирования определены таким образом, что эквивалентный множитель масштабирования уровня коэффициента преобразования удваивается при каждом приращении на 6 в QP_Y. Следовательно, возрастание множителя масштабирования происходит примерно на 12% при каждом возрастании на 1 значения QP_Y.

Значение QP_C следует определять, как это показано в таблице 8-13, основанной на обозначении индекса как qP_I. Значения qP_I должны быть получены следующим образом:

$$qP_I = \text{Clip3}(0, 51, QP_Y + \text{chroma\_qp\_index\_offset}). \quad (8\text{-}251)$$

Таблица 8-13 – Спецификация QP_C как функции от qP_I

qP_I	<30	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
QP_C	$=qP_I$	29	30	31	32	32	33	34	34	35	35	36	36	37	37	37	38	38	38	39	39	39	39

Если текущая секция – это секция SP или SI, то QS_C находят, используя вышеуказанный процесс, заменой QP_Y на QS_Y и QP_C на QS_C.

Функция LevelScale(m, i, j) определена следующим образом:

$$\text{LevelScale}(m,i,j) = \begin{cases} v_{m0} \text{ для } (i,j) \in \{(0,0),(0,2),(2,0),(2,2)\}, \\ v_{m1} \text{ для } (i,j) \in \{(1,1),(1,3),(3,1),(3,3)\}, \\ v_{m2} \text{ иначе}, \end{cases} \tag{8-252}$$

где первая и вторая надписи v – это, соответственно, ряд и столбец индексов матрицы, определенной как:

$$v = \begin{bmatrix} 10 & 16 & 13 \\ 11 & 18 & 14 \\ 13 & 20 & 16 \\ 14 & 23 & 18 \\ 16 & 25 & 20 \\ 18 & 29 & 23 \end{bmatrix}. \tag{8-253}$$

8.5.6 Процесс масштабирования и преобразования DC коэффициентов преобразования яркости макроблоков типа Intra_16x16

Входы в этот процесс – значения уровней коэффициентов преобразования для DC коэффициентов преобразования яркости макроблоков Intra_16x16 как массива 'c' 4x4 с элементами c_{ij}, где i и j формируют двухразмерный индекс частоты.

Выходы из этого процесса – 16-уровневые DC значения блоков яркости 4x4 макроблоков Intra_16x16 как массива 4x4 dcY с элементами dcY_{ij}.

Инверсное преобразование DC коэффициентов преобразования яркости 4x4 определено как:

$$f = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} c_{00} & c_{01} & c_{02} & c_{03} \\ c_{10} & c_{11} & c_{12} & c_{13} \\ c_{20} & c_{21} & c_{22} & c_{23} \\ c_{30} & c_{31} & c_{32} & c_{33} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}. \tag{8-254}$$

Поток битов, соответствующий данной Рекомендации | Международному стандарту, не должен содержать данные, которые появляются в любом элементе f_{ij} из f, выходящими за диапазон целых значений от -2^{15} до $2^{15}-1$ включительно.

После инверсного преобразования масштабирование выполняют следующим образом:

– Если QP_Y больше или равно 12, масштабированный результат должен быть получен как

$$dcY_{ij} = (f_{ij} * \text{LevelScale}(QP_Y \% 6, 0, 0)) << (QP_Y / 6 - 2), \text{ c } i,j = 0..3 . \tag{8-255}$$

– Иначе (QP_Y меньше, чем 12), масштабированный результат должен быть получен как

$$dcY_{ij} = (f_{ij} * \text{LevelScale}(QP_Y \% 6, 0, 0) + 2^{1-QP_Y/6}) >> (2 - QP_Y / 6), \text{ c } i,j = 0..3 . \tag{8-256}$$

Поток битов, соответствующий данной Рекомендации | Международному стандарту, не должен содержать данных, которые появляются в любом элементе dcY_{ij} из dcY, выходящими за диапазон целых значений от -2^{15} до $2^{15}-1$ включительно.

ПРИМЕЧАНИЕ. – Если entropy_coding_mode_flag равно 0, а QP_Y меньше 10, диапазон значений, который может быть предоставлен для элементов c_{ij} из 'c' будет недостаточным для полного диапазона значений элементов dcY_{ij} из dcY, что может быть необходимым для формирования хорошей аппроксимации содержания любого возможного источника изображения при использовании макроблока типа Intra_16x16.

ПРИМЕЧАНИЕ. – Поскольку этот диапазон накладывает ограничение на элементы dcY_{ij} из dcY после сдвига вправо в равенстве 8-256, в декодере должен поддерживаться более широкий диапазон значений перед сдвигом вправо.

8.5.7 Процесс масштабирования и преобразования DC коэффициентов преобразования цветности

Входы в этот процесс – значения уровней коэффициентов преобразования DC коэффициентов преобразования цветности одного из компонентов цветности макроблока как массива 'c' 2x2 с элементами c_{ij}, где i и j формируют двухразмерный индекс частоты.

Выходы из этого процесса – 4-уровневые DC значения как массив dcC 2x2 с элементами dcC_{ij}.

Инверсное преобразование DC коэффициентов преобразования цветности 2x2 определено как:

$$f = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{8-257}$$

Поток битов, соответствующий данной Рекомендации | Международному стандарту, не должен содержать данных, которые появляются в любом элементе f_{ij} из f, выходящими за диапазон целых значений от -2^{15} до $2^{15} - 1$ включительно.

После инверсного преобразования масштабирование выполняют следующим образом:

– Если QP_C больше или равно 6, масштабированный результат должен быть получен как

$$dcC_{ij} = (f_{ij} * \text{LevelScale}(QP_C \,\%\, 6, 0, 0)) << (QP_C / 6 - 1), \quad c \quad i, j = 0, 1 . \tag{8-258}$$

– Иначе (QP_C меньше 6), масштабированный результат должен быть получен как

$$dcC_{ij} = (f_{ij} * \text{LevelScale}(QP_C \,\%\, 6, 0, 0)) >> 1, \quad c \quad i, j = 0, 1 . \tag{8-259}$$

Поток битов, соответствующий данной Рекомендации | Международному стандарту, не должен содержать данных, которые появляются в любом элементе dcC_{ij} из dcC, выходящими за диапазон целых значений от -2^{15} до $2^{15} - 1$ включительно.

ПРИМЕЧАНИЕ. – Если entropy_coding_mode_flag равно 0, а QP_Y меньше 4, диапазон значений, который может быть предоставлен для элементов c_{ij} из 'c', будет недостаточным для полного диапазона значений элементов dcC_{ij} из dcC, что может быть необходимым для формирования хорошей аппроксимации содержания любого возможного источника изображения.

ПРИМЕЧАНИЕ. – Поскольку этот диапазон накладывает ограничение на элементы dcC_{ij} из dcC после сдвига вправо в равенстве 8-259, в декодере должен поддерживаться более широкий диапазон значений до сдвига вправо.

8.5.8 Процесс масштабирования и преобразования остальных блоков 4x4

Вход в этот процесс – массив 'c' 4x4 с элементами c_{ij}, которые либо являются массивом, связанным с оставшимся блоком компонента яркости, либо массивом, связанным с оставшимся блоком компонента цветности.

Выходы из этого процесса – значения оставшихся образцов как массива 'r' 4x4 с элементами r_{ij}.

Переменную sMbFlag находят следующим образом:

– Если mb_type равно SI или режим предсказания макроблока в секции SP равен Inter, sMbFlag устанавливают равным 1,

– Иначе (mb_type не равно SI и режим предсказания макроблока в секции SP не равен Inter), sMbFlag устанавливают равным 0.

Переменную qP находят следующим образом:

– Если входной массив c относится к оставшемуся блоку яркости, а sMbFlag равно 0,

$$qP = QP_Y. \tag{8-260}$$

– Иначе, если входной массив c относится к оставшемуся блоку яркости, а sMbFlag равно 1,

$$qP = QS_Y. \tag{8-261}$$

– Иначе, если входной массив c относится к оставшемуся блоку цветности, а sMbFlag равно 0,

$$qP = QP_C. \tag{8-262}$$

– Иначе (входной массив c относится к оставшемуся блоку цветности, а sMbFlag равно 1),

$$qP = QS_C. \tag{8-263}$$

Масштабирование уровней c_{ij} коэффициентов преобразования блока 4x4 происходит следующим образом:

– Если все из следующих условий истинны:

  – i равно 0;

  – j равно 0;

  – c относится к оставшемуся блоку яркости, кодированному с использованием режима предсказания Intra_16x16, или относится к оставшемуся блоку цветности, то переменную d_{00} находят как

$$d_{00} = c_{00}, \tag{8-264}$$

– Иначе

$$d_{ij} = (c_{ij} * \mathrm{LevelScale}(qP \% 6, i, j)) << (qP / 6), \quad c \;\; i, j = 0..3, \text{ кроме случаев, упомянутых выше.} \tag{8-265}$$

Поток битов не должен содержать данных, которые появляются в любом элементе d_{ij} из d c i, j = 0..3 и превышают диапазон целых значений от -2^{15} до $2^{15} – 1$ включительно.

Процесс преобразования должен превращать блок масштабированных коэффициентов преобразования в блок образцов на выходе способом, математически эквивалентным следующему.

Во-первых, каждый (горизонтальный) ряд масштабированных коэффициентов преобразуют, используя одноразмерное инверсное преобразование следующим образом.

Вычисляют множество предварительных значений:

$$e_{i0} = d_{i0} + d_{i2}, \;\; c \;\; i = 0..3 \tag{8-266}$$

$$e_{i1} = d_{i0} – d_{i2}, \;\; c \;\; i = 0..3 \tag{8-267}$$

$$e_{i2} = (d_{i1} >> 1) – d_{i3}, \;\; c \;\; i = 0..3 \tag{8-268}$$

$$e_{i3} = d_{i1} + (d_{i3} >> 1), \;\; c \;\; i = 0..3. \tag{8-269}$$

Поток битов не должен содержать данных, которые появляются в любом элементе e_{ij} из e c i, j = 0..3 и превышают диапазон целых значений от -2^{15} до $2^{15} – 1$ включительно.

Затем из этих предварительных значений вычисляют преобразованный результат:

$$f_{i0} = e_{i0} + e_{i3}, \;\; c \;\; i = 0..3 \tag{8-270}$$

$$f_{i1} = e_{i1} + e_{i2}, \;\; c \;\; i = 0..3 \tag{8-271}$$

$$f_{i2} = e_{i1} – e_{i2}, \;\; c \;\; i = 0..3 \tag{8-272}$$

$$f_{i3} = e_{i0} – e_{i3}, \;\; c \;\; i = 0..3. \tag{8-273}$$

Поток битов не должен содержать данных, которые появляются в любом элементе f_{ij} из f c i, j = 0..3 и превышают диапазон целых значений от -2^{15} до $2^{15} – 1$ включительно.

Далее каждый (вертикальный) столбец результирующей матрицы преобразуют, используя то же самое одноразмерное инверсное преобразование следующим образом.

Вычисляют множество предварительных значений:

$$g_{0j} = f_{0j} + f_{2j}, \;\; c \;\; j = 0..3 \tag{8-274}$$

$$g_{1j} = f_{0j} – f_{2j}, \;\; c \;\; j = 0..3 \tag{8-275}$$

$$g_{2j} = (f_{1j} >> 1) – f_{3j}, \;\; c \;\; j = 0..3 \tag{8-276}$$

$$g_{3j} = f_{1j} + (f_{3j} >> 1), \;\; c \;\; j = 0..3. \tag{8-277}$$

Поток битов не должен содержать данных, которые появляются в любом элементе g_{ij} из g c i, j = 0..3 и превышают диапазон целых значений от -2^{15} до $2^{15} – 1$ включительно.

Затем из этих предварительных значений вычисляют преобразованный результат:

$$h_{0j} = g_{0j} + g_{3j}, \text{ c } j = 0..3 \quad (8\text{-}278)$$

$$h_{1j} = g_{1j} + g_{2j}, \text{ c } j = 0..3 \quad (8\text{-}279)$$

$$h_{2j} = g_{1j} - g_{2j}, \text{ c } j = 0..3 \quad (8\text{-}280)$$

$$h_{3j} = g_{0j} - g_{3j}, \text{ c } j = 0..3. \quad (8\text{-}281)$$

Поток битов не должен содержать данных, которые появляются в любом элементе h_{ij} из h с i, j = 0..3 и превышают диапазон целых значений от -2^{15} до $2^{15} - 33$ включительно.

После выполнения обоих инверсных одноразмерных преобразований (горизонтального и вертикального) для получения массива преобразованных образцов окончательные значения оставшегося созданного образца должны быть получены как

$$r_{ij} = (h_{ij} + 2^5) >> 6 \text{ c } i, j = 0..3. \quad (8\text{-}282)$$

8.5.9 Процесс конструирования изображения перед процессом фильтрового разделения на блоки

Входы в этот процесс:

– luma4x4BlkIdx или chroma4x4BlkIdx

– массив образцов 'u' 4x4 с элементами u_{ij}, которые являются либо блоком яркости, либо цветности.

Выходы из этого процесса – сконструированные блоки образцов s' перед процессом фильтрового разделения на блоки.

Положение верхнего левого образца яркости текущего макроблока находят, активируя процесс инверсного сканирования макроблока по п. 6.4.1 с CurrMbAddr в качестве входа и присвоением значения (xP, yP) в качестве выхода.

Если 'u' – это блок яркости, то для каждого образца u_{ij} блока яркости 4x4 используют следующие условия:

– Положение верхнего левого образца блока яркости 4x4 с индексом luma4x4BlkIdx внутри макроблока находят, активируя процесс инверсного сканирования блока яркости 4x4 по п. 6.4.3 с luma4x4BlkIdx в качестве входа и присвоением (xO, yO) в качестве выхода.

– В зависимости от переменной MbaffFrameFlag, используют следующие условия:

  – Если MbaffFrameFlag равно 1, а текущий макроблок – макроблок поля, то

    $$S'_L[xP + xO + j, yP + 2 * (yO + i)] = u_{ij} \text{ c } i, j = 0..3. \quad (8\text{-}283)$$

  – Иначе (MbaffFrameFlag равно 0 или текущий макроблок – макроблок кадра),

    $$S'_L[xP + xO + j, yP + yO + i] = u_{ij} \text{ c } i, j = 0..3. \quad (8\text{-}284)$$

Если 'u' – блок цветности, то для каждого образца u_{ij} блока цветности 4x4 используют следующие условия:

– Надпись C в переменных S'_C и $pred_C$ заменяют на Cb для компоненты цветности Cb и на Cr для компоненты цветности Cr.

– Положение верхнего левого образца блока цветности 4x4 с индексом chroma4x4BlkIdx внутри макроблока находят следующим образом:

  $$xO = InverseRasterScan(chroma4x4BlkIdx, 4, 4, 8, 0) \quad (8\text{-}285)$$

  $$yO = InverseRasterScan(chroma4x4BlkIdx, 4, 4, 8, 1). \quad (8\text{-}286)$$

– В зависимости от переменной MbaffFrameFlag используют следующие условия:

  – Если MbaffFrameFlag равно 1, а текущий макроблок – макроблок поля, то

    $$S'_C[(xP >> 1) + xO + j, ((yP + 1) >> 1) + 2 * (yO + i)] = u_{ij} \text{ c } i, j = 0..3. \quad (8\text{-}287)$$

  – Иначе (MbaffFrameFlag равно 0 или текущий макроблок – макроблок кадра),

    $$S'_C[(xP >> 1) + xO + j, ((yP + 1) >> 1) + yO + i] = u_{ij} \text{ c } i, j = 0..3. \quad (8\text{-}288)$$

8.6 Процесс декодирования макроблоков P в секциях SP или макроблоков SI

Процесс активируют при декодировании макроблоков типа P в секциях типа SP или макроблоков типа SI в секциях SI.

Входы в этот процесс – предсказанные остаточные уровни коэффициентов преобразования и предсказанные образцы текущего макроблока.

Выходы из этого процесса – декодированные образцы текущего макроблока перед процессом фильтрового разделения на блоки.

Этот пункт определяет процесс декодирования коэффициентов преобразования и конструирования изображения для макроблоков типов P в секциях SP и макроблоков типов SI в секциях SI.

ПРИМЕЧАНИЕ. – Секции SP используют при кодировании с Inter предсказанием, чтобы воспользоваться временной избыточностью в последовательности, аналогично кодированию секции P. Однако в отличие от кодирования секции P кодирование секции SP допускает идентичную реконструкцию секции, если даже используют различные контрольные изображения. SI секции используют пространственные предсказания аналогично I секциям. Кодирование SI секции допускает идентичную реконструкцию соответствующей SP секции. Свойства секций SP и SI помогают обеспечить функциональность потока битов: коммутацию, стыковку, прямой доступ, устойчивость к ошибкам с прямой и обратной коррекцией.

Секция SP состоит из макроблоков, кодированных либо как тип I, либо как тип P.

Секция SI состоит из макроблоков, кодированных либо как тип I, либо как тип SI.

Процессы декодирования коэффициентов преобразования и конструирование изображения перед процессом фильтрового разделения на блоки для макроблоков типа I в секциях SI должны активировать, как определено в п. 8.5. Макроблок типа SI должен быть декодирован, как описано ниже.

Если текущий макроблок кодирован как P_Skip, то для текущего макроблока все значения LumaLevel, ChromaDCLevel, ChromaACLevel устанавливают равными 0.

8.6.1 Процесс декодирования SP некоммутируемых изображений

Процесс активируют при декодировании макроблоков типа P в SP секциях, в которых sp_for_switch_flag равно 0.

Входы в этот процесс – образцы Inter предсказания для текущего макроблока из п. 8.4 и предсказания уровней остатка коэффициентов преобразования.

Выходы из этого процесса – декодированные образцы текущего макроблока перед процессом фильтрового разделения на блоки.

Этот пункт применим ко всем макроблокам в SP секциях, в которых sp_for_switch_flag равно 0, за исключением тех макроблоков, режим предсказания которых равен Intra_4x4 или Intra_16x16. Этот пункт не применим к SI секциям.

8.6.1.1 Процесс декодирования коэффициентов преобразования яркости

Входы в этот процесс – образцы Inter предсказания яркости для текущего макроблока $pred_L$ из п. 8.4 и предсказания уровней остатка коэффициентов преобразования, LumaLevel и индекс блока яркости 4x4luma4x4BlkIdx.

Выходы из этого процесса – декодированные образцы яркости текущего макроблока перед процессом фильтрового разделения на блоки.

Положение верхнего левого образца блока яркости 4x4 с индексом luma4x4BlkIdx внутри текущего макроблока находят, активируя процесс инверсного сканирования блока яркости 4x4 по п. 6.4.3 с luma4x4BlkIdx в качестве входа и выходом, присвоенным (x, y).

Положим, что переменная p – это массив 4x4 образцов предсказания с элементами p_{ij}, которые находят следующим образом:

$$p_{ij} = pred_L[\ x + j, y + i\ ] \quad c\ i, j = 0..3. \tag{8-289}$$

Переменную p преобразуют, формируя коэффициент преобразования c^p согласно 8-290:

$$c^p = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} p_{00} & p_{01} & p_{02} & p_{03} \\ p_{10} & p_{11} & p_{12} & p_{13} \\ p_{20} & p_{21} & p_{22} & p_{23} \\ p_{30} & p_{31} & p_{32} & p_{33} \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix}. \tag{8-290}$$

Процесс инверсного сканирования коэффициентов преобразования, как описано в п. 8.5.4, активируют с LumaLevel[luma4x4BlkIdx] в качестве входа и двухразмерным массивом c^r с элементами c_{ij}^r в качестве выхода.

Предсказанные остаточные коэффициенты преобразования c^r масштабируют, используя параметр квантования QP_Y и добавляя к коэффициентам преобразования предсказанных блоков значения c^p c i, j = 0..3, следующим образом:

$$c_{ij}^s = c_{ij}^p + (\ (\ (\ c_{ij}^r * LevelScale(\ QP_Y \% 6, i, j\ ) * A_{ij}\ ) << (\ QP_Y / 6\ )\ )\ >> 6\ ), \tag{8-291}$$

где LevelScale(m, i, j) определено из равенства 8-252, а A_{ij} определено как:

$$A_{ij} = \begin{cases} 16 \text{ для } (i,j) \in \{(0,0),(0,2),(2,0),(2,2)\}, \\ 25 \text{ для } (i,j) \in \{(1,1),(1,3),(3,1),(3,3)\}, \\ 20 \text{ иначе;} \end{cases} \quad (8\text{-}292)$$

функция LevelScale2(m, i, j), используемая в формулах ниже, определяется как:

$$\text{LevelScale2}(m,i,j) = \begin{cases} w_{m0} & \text{для } (i,j) \in \{(0,0),(0,2),(2,0),(2,2)\}, \\ w_{m1} & \text{для } (i,j) \in \{(1,1),(1,3),(3,1),(3,3)\}, \\ w_{m2} & \text{иначе;} \end{cases} \quad (8\text{-}293)$$

где первая и вторая надписи w обозначают, соответственно, ряд и столбец индексов матрицы, определенной как:

$$w = \begin{bmatrix} 13107 & 5243 & 8066 \\ 11916 & 4660 & 7490 \\ 10082 & 4194 & 6554 \\ 9362 & 3647 & 5825 \\ 8192 & 3355 & 5243 \\ 7282 & 2893 & 4559 \end{bmatrix}. \quad (8\text{-}294)$$

Результирующую сумму, c^s, квантуют с параметром квантования QS_Y и с i, j = 0..3 следующим образом:

$$c_{ij} = (\text{Sign}(c_{ij}^{\,s}) * (\text{Abs}(c_{ij}^{\,s}) * \text{LevelScale2}(QS_Y \% 6, i, j) + (1 << (14 + QS_Y / 6)))) >> (15 + QS_Y / 6). \quad (8\text{-}295)$$

Процесс масштабирования и преобразования для остатка блоков 4x4, как определено в п. 8.5.8, активируют с 'c' в качестве входа и 'r' в качестве выхода.

Массив u 4x4 с элементами u_{ij} находят следующим образом:

$$u_{ij} = \text{Clip1}(r_{ij}) \text{ с } i, j = 0..3. \quad (8\text{-}296)$$

Процесс конструирования изображения перед процессом фильтрового разделения на блоки согласно п. 8.5.9 активируют с luma4x4BlkIdx, 'u' в качестве входа и с S' в качестве выхода.

8.6.1.2 Процесс декодирования коэффициентов преобразования цветности

Входы в этот процесс – образцы Inter предсказания цветности для текущего макроблока из п. 8.4 и предсказания уровней остаточных коэффициентов преобразования ChromaDCLevel и ChromaACLevel.

Выходы этого процесса – декодированные образцы цветности текущего макроблока перед процессом фильтрового разделения на блоки.

Процесс активируют дважды: один раз для компонента Cb и другой – для компонента Cr. Компонент находят заменой C на Cb для компонента Cb и C на Cr для компонента Cr. Положим, что iCbCr выбирает текущий компонент цветности.

Для каждого блока 4x4 текущий компонент цветности индексируют, используя chroma4x4BlkIdx; при значениях chroma4x4BlkIdx, равных 0..3. Действуют следующие условия:

– Положение верхнего левого образца блока цветности 4x4 с индексом chroma4x4BlkIdx внутри макроблока находят следующим образом:

$$x = \text{InverseRasterScan}(\text{chroma4x4BlkIdx}, 4, 4, 8, 0) \quad (8\text{-}297)$$

$$y = \text{InverseRasterScan}(\text{chroma4x4BlkIdx}, 4, 4, 8, 1). \quad (8\text{-}298)$$

– Положим, что p – массив 4x4 образцов предсказания с элементами p_{ij}, полученными следующим образом:

$$p_{ij} = \text{pred}_C[x + j, y + i] \text{ с } i, j = 0..3. \quad (8\text{-}299)$$

– Массив 'p' 4x4 преобразуют, формируя коэффициенты преобразования c^p(chroma4x4BlkIdx) с помощью равенства 8-290.

– Находят переменную chromaList, которая представляет список из 16 входов, chromaList[0] устанавливают равным 0, chromaList[k] с индексом k = 1..15 определяют следующим образом:

chromaList[k] = ChromaACLevel[iCbCr][chroma4x4BlkIdx][k – 1]. (8-300)

– Процесс инверсного сканирования коэффициентов преобразования активируют, как описано в п. 8.5.4, с chromaList в качестве входа и массивом c^r 4x4 в качестве выхода.

– Предсказанные остаточные коэффициенты преобразования c^r масштабируют, используя параметр квантования QP_C и добавляя к коэффициентам преобразования предсказанных блоков значения c^p с i, j = 0..3, за исключением комбинации i = 0, j = 0, следующим образом:

$$c_{ij}^{s} = c_{ij}^{p}(\text{chroma4x4BlkIdx}) + (((c_{ij}^{r} * \text{LevelScale}(QP_C \% 6, i, j) * A_{ij}) << (QP_C / 6)) >> 6). \quad (8\text{-}301)$$

– Результирующую сумму, c^s, квантуют с параметром квантования QS_C и с i, j = 0..3, за исключением комбинации I = 0, j = 0, следующим образом: Нахождение c_{00}(chroma4x4BlkIdx) описано ниже в этом пункте:

$$c_{ij}(\text{chroma4x4BlkIdx}) = (\text{Sign}(c_{ij}^{s}) * (\text{Abs}(c_{ij}^{s}) * \text{LevelScale2}(QS_C \% 6, i, j) + (1 << (14 + QS_C / 6)))) >> (15 + QS_C / 6). \quad (8\text{-}302)$$

– Процесс масштабирования и преобразования для остаточных блоков 4x4, как определено в п. 8.5.8, активируют с 'c'(chroma4x4BlkIdx) в качестве входа и 'r' в качестве выхода.

– Массив 'u' 4x4 с элементами u_{ij} находят следующим образом:

$$u_{ij} = \text{Clip1}(r_{ij}) \text{ с } i, j = 0..3. \quad (8\text{-}303)$$

– Процесс конструирования изображения перед процессом фильтрового разделения на блоки по п. 8.5.9 активируют с chroma4x4BlkIdx и 'u' в качестве входа и с S’ в качестве выхода.

Отыскание уровней DC коэффициентов преобразования c_{00}(chroma4x4BlkIdx) определено следующим образом: DC коэффициенты преобразования 4-х предсказанных блоков цветности 4x4 текущего компонента макроблока объединяют в матрицу 2x2 с элементами c_{00}^{p}(chroma4x4BlkIdx) и выполняют преобразование 2x2 над DC коэффициентами преобразования следующим образом:

$$dc^{p} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} c_{00}^{p}(0) & c_{00}^{p}(1) \\ c_{00}^{p}(2) & c_{00}^{p}(3) \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \quad (8\text{-}304)$$

Предсказанные остаточные уровни DC коэффициента преобразования цветности, ChromaDCLevel[iCbCr][k] с k = 0..3, масштабируют, используя параметр квантования QP и добавляя к DC коэффициентам преобразования следующее:

$$dc_{ij}^{s} = dc_{ij}^{p} + (((\text{ChromaDCLevel}[iCbCr][j * 2 + i] * \text{LevelScale}(QP_C \% 6, 0, 0) * A_{00}) << (QP_C / 6)) >> 5) \text{ с } i, j = 0, 1. \quad (8\text{-}305)$$

Массив dc^s 2x2 квантуют, используя параметр квантования QS_C:

$$dc_{ij}^{r} = (\text{Sign}(dc_{ij}^{s}) * (\text{Abs}(dc_{ij}^{s}) * \text{LevelScale2}(QS_C \% 6, 0, 0) + (1 << (15 + QS_C / 6)))) >> (16 + QS_C / 6) \text{ с } i, j = 0, 1. \quad (8\text{-}306)$$

Находят массив 'f' 2x2 с элементами f_{ij} и i, j = 0..1:

$$f = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} dc_{00}^{r} & dc_{01}^{r} \\ dc_{10}^{r} & dc_{11}^{r} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \quad (8\text{-}307)$$

Элементы f_{ij} из f масштабируют:

– Если QS_C больше или равно 6, c_{00}( ) находят как

$$c_{00}(j * 2 + i) = (f_{ij} * \text{LevelScale}(QS_C \% 6, 0, 0)) << (QS_C / 6 – 1) \text{ с } i, j = 0, 1. \quad (8\text{-}308)$$

– Иначе (QS_C меньше 6), c_{00}( ) находят как

$$c_{00}(j * 2 + i) = (f_{ij} * \text{LevelScale}(QS_C \% 6, 0, 0)) >> 1 \text{ с } i, j = 0, 1. \quad (8\text{-}309)$$

8.6.2 Процесс декодирования секций SP и SI для коммутации изображений

Процесс активируют при декодировании макроблоков типа P в секциях SP, в которых sp_for_switch_flag равно 1, и при декодировании макроблоков типа SI в секциях SI.

Входы в этот процесс – предсказанные уровни остатков коэффициентов преобразования и предсказанные массивы образцов $pred_L$, $pred_{Cb}$, $pred_{Cr}$ для текущего макроблока.

Выходы из этого процесса – декодированные образцы текущего макроблока перед процессом фильтрового разделения на блоки.

8.6.2.1 Процесс декодирования коэффициентов преобразования яркости

Входы в этот процесс – предсказанные образцы яркости $pred_L$ и предсказанные уровни остатков коэффициентов преобразования яркости, LumaLevel.

Выходы из этого процесса – декодированные образцы яркости текущего макроблока перед процессом фильтрового разделения на блоки.

Массив 'p' 4x4 с элементами p_{ij} с i, j = 0..3 находят, как указано в п. 8.6.1.1, преобразованием согласно равенству 8-290, для отыскания коэффициента преобразования c^p. Затем эти коэффициенты преобразования квантуют с параметром квантования QS_Y:

$$c_{ij}^s = (Sign(c_{ij}^p) * (Abs(c_{ij}^p) * LevelScale2(QS_Y \% 6, i, j) + (1 << (14 + QS_Y / 6)))) >> (15 + QS_Y / 6) \quad \text{с i, j = 0..3.} \qquad (8\text{-}310)$$

Процесс инверсного сканирования коэффициентов преобразования, как описано в п. 8.5.4, активируют с LumaLevel[luma4x4BlkIdx] в качестве входа и двухразмерным массивом c^r с элементами c_{ij}^r в качестве выхода.

Массив 'c' 4x4 с элементами c_{ij} с i, j = 0..3 находят следующим образом:

$$c_{ij} = c_{ij}^r + c_{ij}^s \text{ с i, j = 0..3.} \qquad (8\text{-}311)$$

Процесс масштабирования и преобразования остаточных блоков 4x4, как определено в п. 8.5.8, активируют с 'c' в качестве входа и 'r' в качестве выхода.

Массив 'u' 4x4 с элементами u_{ij} находят следующим образом:

$$u_{ij} = Clip1(r_{ij}) \text{ с i, j = 0..3.} \qquad (8\text{-}312)$$

Процесс конструирования изображения перед процессом фильтрового разделения на блоки по п. 8.5.9 активируют с luma4x4BlkIdx, 'u' в качестве входа и с S' в качестве выхода.

8.6.2.2 Процесс декодирования коэффициентов преобразования цветности

Входы в этот процесс – предсказанные образцы цветности для текущего макроблока из п. 8.4 и предсказанные уровни остатка коэффициентов преобразования, ChromaDCLevel и ChromaACLevel.

Выходы из этого процесса – декодированные образцы цветности текущего макроблока перед процессом фильтрового разделения на блоки.

Процесс активируют дважды: первый раз для компонента Cb, а второй – для компонента Cr. Компонент находят заменой C на Cb для компонента Cb и C на Cr для компонента Cr. Положим, что iCbCr выбирает текущий компонент цветности.

Каждый блок 4x4 текущего компонента цветности индексируют, используя chroma4x4BlkIdx, со значениями chroma4x4BlkIdx, равными 0..3. Действуют следующие условия:

1. Массив 'p' 4x4 с элементами p_{ij} с i, j = 0..3 находят, как указано в п. 8.6.1.2, преобразованием, согласно равенству 8-290, для отыскания коэффициентов преобразования c^p(chroma4x4BlkIdx). Затем эти коэффициенты преобразования квантуют, за исключением комбинации i = 0, j = 0, с параметром квантования QS_C с i, j = 0..3. Обработка c_{00}^p(chroma4x4BlkIdx) описана ниже в этом пункте:

$$c_{ij}^s = (Sign(c_{ij}^p(chroma4x4BlkIdx)) * (Abs(c_{ij}^p(chroma4x4BlkIdx)) * LevelScale2(QS_C \% 6, i, j) + (1 << (14 + QS_C / 6)))) >> (15 + QS_C / 6). \qquad (8\text{-}313)$$

– Находят переменную chromaList, которая имеется в списке из 16 входов. Значение chromaList[0] устанавливают равным 0. Значение chromaList[k] с индексами k = 1..15 определяют следующим образом:

$$chromaList[k] = ChromaACLevel[iCbCr][chroma4x4BlkIdx][k – 1]. \qquad (8\text{-}314)$$

– Процесс инверсного сканирования коэффициентов преобразования, как это описано в п. 8.5.4, активируют с chromaList в качестве входа и двухразмерным массивом c^r(chroma4x4BlkIdx) с элементами c_{ij}^r(chroma4x4BlkIdx) в качестве выхода.

– Массив 'c'(chroma4x4BlkIdx) 4x4 с элементами c_{ij}(chroma4x4BlkIdx) с i, j = 0..3, за исключением комбинации i = 0, j = 0, находят следующим образом: Отыскание c_{00}(chroma4x4BlkIdx) описано ниже:

$$c_{ij}(\text{chroma4x4BlkIdx}) = c_{ij}^{r}(\text{chroma4x4BlkIdx}) + c_{ij}^{s}. \quad (8\text{-}315)$$

– Процесс масштабирования и преобразования остатка блоков 4x4, как это определено в п. 8.5.8, активируют с 'c'(chroma4x4BlkIdx) в качестве входа и 'r' в качестве выхода.

– Массив 'u' 4x4 с элементами u_{ij} находят следующим образом:

$$u_{ij} = \text{Clip1}(r_{ij}) \text{ с } i, j = 0..3. \quad (8\text{-}316)$$

– Процесс конструирования изображения перед процессом фильтрового разделения на блоки по п. 8.5.9 активируют с chroma4x4BlkIdx, 'u' в качестве входа и с S’ в качестве выхода.

Нахождение уровней DC коэффициентов преобразования c_{00}(chroma4x4BlkIdx) производят следующим образом: DC коэффициенты преобразования 4-х предсказанных блоков цветности 4x4 текущих компонентов макроблока, c_{00}^{p}(chroma4x4BlkIdx), объединяют в матрицу 2x2, и преобразование 2x2 добавляют к DC коэффициентам преобразования этих блоков согласно равенству 8-304, получая в результате DC коэффициенты преобразования dc_{ij}^{p}.

Затем эти DC коэффициенты преобразования квантуют с параметром квантования QS_C, заданным как:

$$dc_{ij}^{s} = (\text{Sign}(dc_{ij}^{p}) * (\text{Abs}(dc_{ij}^{p}) * \text{LevelScale2}(QS_C \% 6, 0, 0) + (1 << (15 + QS_C / 6)))) >> (16 + QS_C / 6) \quad \text{с } i, j = 0, 1. \quad (8\text{-}317)$$

Проанализированные предсказанные остаточные DC коэффициенты преобразования цветности, ChromaDCLevel[iCbCr][k] с k = 0..3 добавляют к этим квантованным DC коэффициентам преобразования предсказанного блока, как показано ниже:

$$dc_{ij}^{r} = dc_{ij}^{s} + \text{ChromaDCLevel}[\text{iCbCr}][j * 2 + i] \text{ с } i, j = 0, 1. \quad (8\text{-}318)$$

Массив 'f' 2x2 с элементами f_{ij} и i, j = 0..1 находят, используя равенство 8-307.

Массив 'f' 2x2 с элементами f_{ij} и i, j = 0..1 копируют следующим образом:

$$c_{00}(j * 2 + i) = f_{ij} \text{ с } i, j = 0, 1. \quad (8\text{-}319)$$

8.7 Процесс фильтрового разделения на блоки

Обусловленная фильтрация должна быть приложена ко всем блокам 4x4 краев изображения, за исключением краев на границе изображения и любых краев, для которых процесс фильтрового разделения на блоки аннулирован с помощью команды disable_deblocking_filter_idc, как определено ниже. Процесс фильтрации выполняют на основе макроблока после завершения процесса конструирования изображения и перед процессом фильтрового разделения на блоки (как определено в пп. 8.5 и 8.6) полностью декодированного изображения. Все макроблоки изображения обрабатывают в порядке возрастания адресов макроблоков.

ПРИМЕЧАНИЕ. – Перед операцией фильтрового разделения на блоки каждого макроблока разделенные на блоки образцы макроблоков или пары макроблоков выше (если такие имеются) и макроблоки или пары макроблоков слева (если имеются) от текущего макроблока всегда доступны, поскольку процесс фильтрового разделения на блоки выполняют после завершения процесса конструирования изображения перед процессом фильтрового разделения на блоки для всего декодированного изображения.

Процесс фильтрового разделения на блоки для компонентов яркости и цветности активируют раздельно. В каждом макроблоке вертикальные края фильтруют первыми, слева направо, затем фильтруют горизонтальные края сверху вниз. Процесс фильтрового разделения на блоки яркости выполняют на 4-х краях 16-ти образцов, а процесс фильтрового разделения на блоки для каждого компонента цветности выполняют на 2-х краях 8-ми образцов. В горизонтальном направлении, как показано на левой стороне рисунка 8-9, и в вертикальном направлении, как показано на правой стороне рисунка 8-9. Значения образцов выше и слева от текущего макроблока, которые могут быть модифицированы операцией фильтрового разделения на блоки предварительных макроблоков, должны быть использованы как вход в процесс фильтрового разделения на блоки текущего макроблока, а в дальнейшем могут быть модифицированы во время фильтрации текущего макроблока. Значение образца, модифицированного во время фильтрации вертикальных краев, используют как вход для фильтрации горизонтальных краев этого же макроблока.

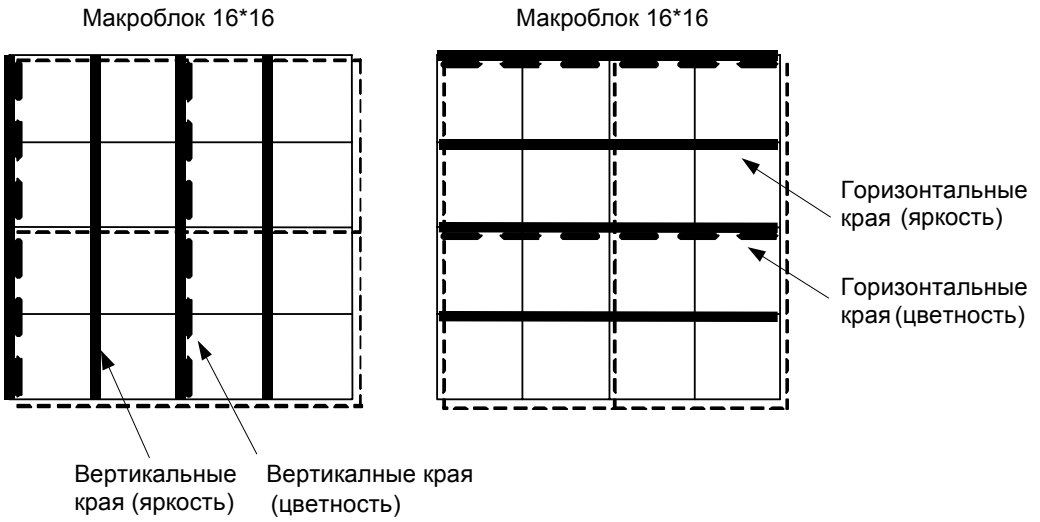

Рисунок 8-9 – Границы в макроблоке, которые следует фильтровать (границы яркости показаны сплошными линиями, а цветности – пунктирными линиями)

Для каждого макроблока в возрастающем порядке mbAddr используют следующие условия:

1. Находят переменные fieldModeMbFlag, filterInternalEdgesFlag, filterLeftMbEdgeFlag и filterTopMbEdgeFlag.
   - Переменную fieldModeMbFlag находят следующим образом:
     - Если любое из следующих условий – истина, то fieldModeMbFlag устанавливают равным 1.
       - field_pic_flag равно 1;
       - MbaffFrameFlag равно 1, а макроблок mbAddr – макроблок поля.
     - Иначе fieldModeMbFlag устанавливают равным 0.
   - Переменную filterInternalEdgesFlag находят следующим образом:
     - Если disable_deblocking_filter_idc для секции, которая содержит макроблок mbAddr, равно 1, то переменную filterInternalEdgesFlag устанавливают на 0.
     - Иначе (disable_deblocking_filter_idc для секции, которая содержит макроблок mbAddr, не равно 1), переменную filterInternalEdgesFlag устанавливают на 1.
   - Переменную filterLeftMbEdgeFlag находят следующим образом:
     - Если любое из следующих условий – истина, то переменную filterLeftMbEdgeFlag устанавливают на 0:
       - левый вертикальный край макроблока mbAddr представляет границу изображения;
       - disable_deblocking_filter_idc для секции, которая содержит макроблок mbAddr, равно 1;
       - disable_deblocking_filter_idc для секции, которая содержит макроблок mbAddr, равно 2, а левый вертикальный край макроблока mbAddr представляет границу секции.
     - Иначе переменную filterLeftMbEdgeFlag устанавливают равной 1.
   - Переменную filterTopMbEdgeFlag находят следующим образом:
     - Если любое из следующих условий – истина, то переменную filterTopMbEdgeFlag устанавливают на 0:
       - верхний горизонтальный край макроблока mbAddr представляет границу изображения;
       - disable_deblocking_filter_idc для секции, которая содержит макроблок mbAddr, равно 1;
       - disable_deblocking_filter_idc для секции, которая содержит макроблок mbAddr, равно 2, а верхний горизонтальный край макроблока mbAddr представляет границу секции,

– иначе переменную filterTopMbEdgeFlag устанавливают равной 1.

2. Заданными переменными fieldModeMbFlag, filterInternalEdgesFlag, filterLeftMbEdgeFlag и filterTopMbEdgeFlag для фильтрации с разделением на блоки управляют следующим образом:

– Если filterLeftMbEdgeFlag равно 1, то определяют фильтрацию левых вертикальных краев яркости:

– Активируют процесс, определенный в п. 8.7.1, с mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag и (xE_k, yE_k) = (0, k) с k = 0..15 как вход и с S'_L как выход.

– Если filterInternalEdgesFlag равно 1, то определяют фильтрацию внутренних вертикальных краев яркости:

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag и (xE_k, yE_k) = (4, k) с k = 0..15 как вход и с S'_L как выход.

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag и (xE_k, yE_k) = (8, k) с k = 0..15 как вход и с S'_L как выход.

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag и (xE_k, yE_k) = (12, k) с k = 0..15 как вход и с S'_L как выход.

– Если filterTopMbEdgeFlag равно 1, то определяют фильтрацию верхних горизонтальных краев яркости:

– Если MbaffFrameFlag равно 1, (mbAddr % 2) равно 0, mbAddr больше или равно 2 * PicWidthInMbs, то макроблок mbAddr – это макроблок кадра, а макроблок (mbAddr – 2 * PicWidthInMbs + 1) – это макроблок поля, то используют следующие условия:

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1 и (xE_k, yE_k) = (k, 0) с k = 0..15 как вход, и с S'_L как выход.

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1 и (xE_k, yE_k) = (k, 1) с k = 0..15 как вход, и с S'_L как выход.

– Иначе процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag и (xE_k, yE_k) = (k, 0) с k = 0..15 как вход и с S'_L как выход.

– Если filterInternalEdgesFlag равно 1, фильтрацию внутренних горизонтальных краев яркости определяют следующим образом:

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag и (xE_k, yE_k) = (k, 4) с k = 0..15 как вход и с S'_L как выход.

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag и (xE_k, yE_k) = (k, 8) с k = 0..15 как вход и с S'_L как выход.

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 0, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag и (xE_k, yE_k) = (k, 12) с k = 0..15 как вход и с S'_L как выход.

– Для обоих компонентов цветности iCbCr = 0 и 1 используют следующие условия:

– Если filterLeftMbEdgeFlag равно 1, то определяют фильтрацию левых вертикальных краев цветности:

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 1, fieldModeFilteringFlag = 1 и (xE_k, yE_k) = (0, k) с k = 0..7 как вход и с S'_C с заменой C на Cb для iCbCr = 0 и на Cr для iCbCr = 1 в качестве выхода.

– Если filterInternalEdgesFlag равно 1, то фильтрацию внутренних вертикальных краев цветности определяют следующим образом:

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 1, fieldModeFilteringFlag = fieldModeMbFlag и (xE_k, yE_k) = (4, k) с k = 0..7 как вход и с S'_C с заменой C на Cb для iCbCr = 0 и на Cr для iCbCr = 1 в качестве выхода.

– Если filterTopMbEdgeFlag равно 1, то определяют фильтрацию верхних горизонтальных краев цветности:

– Если MbaffFrameFlag равно 1, (mbAddr % 2) равно 0, mbAddr больше или равно 2 * PicWidthInMbs, а макроблок mbAddr – макроблок кадра и макроблок (mbAddr – 2 * PicWidthInMbs + 1) – макроблок поля, то используют следующие условия:

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1 и (xE_k, yE_k) = (k, 0) с k = 0..7 как вход и с S'_C с заменой C на Cb для iCbCr = 0 и на Cr для iCbCr = 1 в качестве выхода.

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = 1 и (xE_k, yE_k) = (k, 1) с k = 0..7 как вход и с S'_C с заменой C на Cb для iCbCr = 0 и на Cr для iCbCr = 1 в качестве выхода.

– Иначе процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag и (xE_k, yE_k) = (k, 0) с k = 0..7 как вход и с S'_C с заменой C на Cb для iCbCr = 0 и на Cr для iCbCr = 1 в качестве выхода.

– Если filterInternalEdgesFlag равно 1, то фильтрацию внутренних горизонтальных краев цветности определяют следующим образом:

– Процесс, определенный в п. 8.7.1, активируют с mbAddr, chromaEdgeFlag = 1, iCbCr, verticalEdgeFlag = 0, fieldModeFilteringFlag = fieldModeMbFlag и (xE_k, yE_k) = (k, 4) с k = 0..7 как вход и с S'_C с заменой C на Cb для iCbCr = 0 и на Cr для iCbCr = 1 в качестве выхода.

ПРИМЕЧАНИЕ. – Если применяют режим фильтрации поля (fieldModeFilteringFlag равно 1) для верхних горизонтальных краев макроблока кадра, то для вертикальной фильтрации верхних и нижних границ макроблока могут потребоваться некоторые образцы, которые расширят внутренние края блока, чтобы выполнить внутреннюю фильтрацию в режим кадра.

ПРИМЕЧАНИЕ. – Во всех случаях фильтруют 3 горизонтальных края яркости, 1 горизонтальный край цветности для Cb и 1 горизонтальный край цветности для Cr. Эти края являются внутренними для макроблока. Если применяют фильтрацию в режиме поля (fieldModeFilteringFlag равно 1) к верхним краям макроблока кадра, 2 горизонтальным краям яркости, 2 горизонтальным краям цветности для Cb и 2 – для Cr между макроблоками кадра, а верхнюю пару макроблоков фильтруют в режиме поля, то в целом фильтруют до 5 горизонтальных краев яркости, 3 горизонтальных краев цветности для Cb и 3 – для Cr. Считают, что такой операцией управляет макроблок кадра. Во всех других случаях фильтруют не более 4 горизонтальных краев яркости, 2 горизонтальных краев цветности для Cb и 2 – для Cr. Считают, что такой операцией управляет особый макроблок.

Наконец, значения массивов S'_L, S'_{Cb}, S'_{Cr} присваивают массивам S_L, S_{Cb}, S_{Cr} (которые представляют декодированное изображение) соответственно.

8.7.1 Процесс фильтрации краев блоков

Вход в этот процесс – mbAddr, chromaEdgeFlag, индекс компонента цветности iCbCr (если chromaEdgeFlag равно 1), verticalEdgeFlag, fieldModeFilteringFlag, и множество из 16 положений образцов яркости (если chromaEdgeFlag равно 0) или 8 (если chromaEdgeFlag равно 1) положений образцов цветности (xE_k, yE_k) с k = 0 .. nE – 1, выраженными относительно верхнего левого угла макроблока mbAddr. Множество положений образцов (xE_k, yE_k) представляют положения образцов непосредственно справа от вертикального края (если verticalEdgeFlag равно 1) или непосредственно ниже вертикального края (если verticalEdgeFlag равно 0).

Переменную nE находят следующим образом:

– Если chromaEdgeFlag равно 0, то nE равно 16.

– Иначе (chromaEdgeFlag равно 1), nE равно 8.

Положим, что s' – переменная, определяющая массив образцов яркости или цветности, которую находят следующим образом:

– Если chromaEdgeFlag равно 0, то s' представляет массив S'_L образцов яркости текущего изображения.

– Иначе, если chromaEdgeFlag равно 1, а iCbCr равно 0, то s' представляет массив S'_{Cb} образцов цветности компонента цветности Cb текущего изображения.

– Иначе (chromaEdgeFlag равно 1 и iCbCr равно 1), s' представляет массив S'_{Cr} образцов цветности компонента цветности Cr текущего изображения.

Переменную dy находят следующим образом:

– Если fieldModeFilteringFlag равно 1 и MbaffFrameFlag равно 1, dy устанавливают равным 2.

– Иначе (fieldModeFilteringFlag равно 0 или MbaffFrameFlag равно 0), dy устанавливают равным 1.

Положение верхнего левого образца яркости макроблока mbAddr находят, активируя процесс инверсного сканирования макроблока по п. 6.4.1 с mbAddr в качестве входа, а в качестве выхода – присвоенные значения (xP, yP).

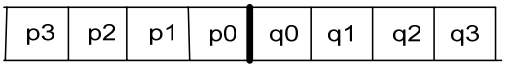

Рисунок 8-10 – Условные обозначения для описания образцов блоков 4x4 вдоль горизонтальной или вертикальной границы

Для каждого положения образца (xE_k, yE_k), k = 0 .. nE – 1 используют следующие условия:

– Процесс фильтрации применяют к множеству из восьми образцов блоков 4x4 вдоль горизонтального и вертикального краев, обозначенных как p_i и q_i с i = 0..3, как показано на рисунке 8-10, с краем, расположенным между p_0 и q_0. Значения p_i и q_i с i = 0..3 определены следующим образом:

  – Если verticalEdgeFlag равно 1,

$$q_i = s'[\ xP + xE_k + i,\ yP + yE_k\ ] \tag{8-320}$$

$$p_i = s'[\ xP + xE_k - i - 1,\ yP + yE_k\ ]. \tag{8-321}$$

  – Иначе (verticalEdgeFlag равно 0),

$$q_i = s'[\ xP + xE_k,\ yP + dy * (\ yE_k + i\ ) - (yE_k \% 2\ )\ ] \tag{8-322}$$

$$p_i = s'[\ xP + xE_k,\ yP + dy * (\ yE_k - i - 1\ ) - (yE_k \% 2\ )\ ]. \tag{8-323}$$

– Процесс, определенный в п. 8.7.2, активируют со значениями образца p_i и q_i (i = 0..3), chromaEdgeFlag, verticalEdgeFlag и fieldModeFilteringFlag как вход и с выходом, присвоенным отфильтрованным результирующим значениям образца p'_i и q'_i с i = 0..2.

– Вход значений образца p_i и q_i с i = 0..2 заменяют соответствующими отфильтрованными результирующими значениями образца p'_i и q'_i с i = 0..2 внутри массива s' образцов следующим образом:

  – Если verticalEdgeFlag равно 1,

$$s'[\ xP + xE_k + i,\ yP + yE_k\ ] = q'_i \tag{8-324}$$

$$s'[\ xP + xE_k - i - 1,\ yP + yE_k\ ] = p'_i. \tag{8-325}$$

  – Иначе (verticalEdgeFlag равно 0),

$$s'[\ xP + xE_k,\ yP + dy * (\ yE_k + i\ ) - (\ yE_k \% 2\ )\ ] = q'_i \tag{8-326}$$

$$s'[\ xP + xE_k,\ yP + dy * (\ yE_k - i - 1\ ) - (\ yE_k \% 2\ )\ ] = p'_i. \tag{8-327}$$

8.7.2 Процесс фильтрации множества образцов вдоль горизонтального и вертикального краев блока

Входы в этот процесс – вход значений образца p_i и q_i с i в диапазоне 0..3 для одного множества образцов вдоль края, который следует фильтровать, chromaEdgeFlag, verticalEdgeFlag и fieldModeFilteringFlag.

Выходы этого процесса – отфильтрованные результирующие значения образца p'_i и q'_i с i в диапазоне 0..2.

Переменную bS, которую определяют интенсивностью фильтрации границы в зависимости от степени яркости, находят следующим образом:

– Если chromaEdgeFlag равно 0, процесс отыскания интенсивности фильтрации границы, определенный в п. 8.7.2.1, активируют с p_0, q_0 и verticalEdgeFlag как вход и с выходом, присвоенным значению bS.

– Иначе (chromaEdgeFlag равно 1), используют следующие условия:

  – Если fieldModeFilteringFlag равно 0, то значение bS, которое используют для фильтрации множества образцов горизонтального или вертикального края цветности, должно быть установлено равным значению bS от фильтрации множества образцов горизонтального или вертикального краев яркости соответственно, которое содержит положение (2 * x, 2 * y) образца яркости внутри массива яркости кадра, где (x, y) – это положение образца цветности q_0 внутри массива цветности этого кадра.

  – Иначе (fieldModeFilteringFlag равно 1), значение bS, которое используют для фильтрации множества образцов горизонтального или вертикального краев цветности, должно быть установлено равным значению bS от фильтрации множества образцов горизонтального или вертикального края яркости соответственно, которое содержит положение (2 * x, 2 * y) образца яркости внутри массива яркости того же поля, где (x, y) – это положение образца цветности q_0 внутри массива цветности этого поля.

Процесс, определенный в п. 8.7.2.2, активируют с p_0, q_0, p_1, q_1, chromaEdgeFlag и bS как вход, а выход – присвоенные значения filterSamplesFlag, indexA, α и β.

В зависимости от переменной filterSamplesFlag используют следующие условия:

– Если filterSamplesFlag равно 1, используют следующие условия:

  – Если bS меньше 4, активируют процесс, определенный в п. 8.7.2.3, с p_i и q_i (i = 0..3), chromaEdgeFlag, bS, β и indexA, заданными как вход, а выход – присвоенные значения p'_i и q'_i (i = 0..2).

– Иначе (bS равно 4), процесс, определенный в п. 8.7.2.4, активируют с p_i и q_i (i = 0..3), chromaEdgeFlag, α и β, заданными как вход, а выход – присвоенные значения p'_i и q'_i (i = 0..2).

– Иначе (filterSamplesFlag равно 0), результирующие отфильтрованные образцы p'_i и q'_i (i = 0..2) заменяют соответствующими образцами входа p_i и q_i:

$$\text{для } I = 0..2, \qquad p'_I = p_i \tag{8-328}$$

$$\text{для } I = 0..2, \qquad q'_I = q_i. \tag{8-329}$$

8.7.2.1 Процесс отыскания интенсивности фильтрации границы в зависимости от степени яркости

Входы в этот процесс – вход значений образца p_0 и q_0 единственного множества образцов вдоль края, который следует фильтровать, а также verticalEdgeFlag.

Выход этого процесса – переменная bS.

Положим, что переменную mixedModeEdgeFlag можно найти следующим образом:

– Если MbaffFrameFlag равно 1 и образцы p_0 и q_0 – разные пары макроблоков, одна из которых – пара макроблоков поля, а другая – пара макроблоков, то mixedModeEdgeFlag устанавливают равным 1.

– Иначе mixedModeEdgeFlag устанавливают равным 0.

Переменную bS находят следующим образом:

– Если край блока является также и краем макроблока, а любое из следующих условий – истина, то значение bS, равное 4, должно быть выходом:

  – образцы p_0 и q_0 находятся оба в макроблоках кадра, и любой из них или оба, p_0 или $q_{0,}$ расположены в макроблоке, кодированном с использованием режима Intra предсказания макроблока;

  – образцы p_0 и q_0 находятся оба в макроблоках кадра, и любой из них или оба, p_0 или $q_{0,}$ расположены в макроблоке, который входит в секцию со значением slice_type, равным SP или SI;

  – MbaffFrameFlag равно 1 или field_pic_flag равно 1, verticalEdgeFlag равно 1, и любой из образцов или оба, p_0 или $q_{0,}$ расположены в макроблоке, кодированном с использованием режима Intra предсказания макроблока;

  – MbaffFrameFlag равно 1 или field_pic_flag равно 1, verticalEdgeFlag равно 1, и любой из образцов или оба, p_0 или $q_{0,}$ расположены в макроблоке, который входит в секцию со значением slice_type, равным SP или SI.

– Иначе, если любое из следующих условий – истина, то значение bS, равное 3, должно быть выходом:

  – mixedModeEdgeFlag равно 0, и любой из образцов или оба, p_0 или $q_{0,}$ расположены в макроблоке, кодированном с использованием режима Intra предсказания макроблока;

  – mixedModeEdgeFlag равно 0, и любой из образцов или оба, p_0 или $q_{0,}$ расположены в макроблоке, который входит в секцию со значением slice_type, равным SP или SI;

  – mixedModeEdgeFlag равно 1, verticalEdgeFlag равно 0, и любой из образцов или оба, p_0 или $q_{0,}$ расположены в макроблоке, кодированном с использованием режима Intra предсказания макроблока;

  – mixedModeEdgeFlag равно 1, verticalEdgeFlag равно 0, и любой из образцов или оба, p_0 или $q_{0,}$ расположены в макроблоке, который входит в секцию со значением slice_type, равным SP или SI.

– Иначе, если следующее условие – истина, то значение bS, равное 2, должно быть выходом:

  – блок яркости 4x4, содержащий образец p_0, или блок яркости 4x4, содержащий образец q_0, с ненулевыми уровнями коэффициентов преобразования.

– Иначе, если следующее условие – истина, то значение bS, равное 1, должно быть выходом:

  – mixedModeEdgeFlag равно 1;

– mixedModeEdgeFlag равное 0, и предсказание разделения макроблока/субмакроблока, содержащее образец p_0 с контрольными изображениями или числом векторов движений, отличных от тех, которые используют для предсказания разделения макроблока/субмакроблока, содержащих образец q_0.

ПРИМЕЧАНИЕ. – Определение того, используются ли те же самые или различные контрольные изображения для разделения двух макроблоков/субмакроблоков, основано только на том, какие изображения сравнивают. При этом не учитывают, было ли сформировано предсказание с использованием индекса в списке 0 или в списке 1, а также не учитывают, было или не было различным положение индекса в списке контрольного изображения.

– mixedModeEdgeFlag равно 0, и один вектор движения используют, чтобы предсказать разделение макроблока/субмакроблока, содержащего образец p_0, а другой вектор движения – чтобы предсказать разделение макроблока/субмакроблока, содержащего образец q_0. При этом учитывают абсолютную разность между горизонтальными или вертикальными компонентами векторов движения, большую или равную 4 в единицах четверти яркости образцов кадра;

– mixedModeEdgeFlag равно 0, и два вектора движения и два различных контрольных изображения используют, чтобы предсказать разделение макроблока/субмакроблока, содержащего образец p_0, а два вектора движения тех же контрольных изображений – чтобы предсказать разделение макроблока/субмакроблока, содержащего образец q_0. При этом учитывают абсолютную разность между горизонтальными или вертикальными компонентами двух векторов движения для предсказания разделения двух макроблоков/субмакроблоков для этого же контрольного изображения. Эта разность должна быть больше или равна 4 в единицах четверти яркости образцов кадра;

– mixedModeEdgeFlag равно 0, и два вектора движения для того же контрольного изображения используют, чтобы предсказать разделение макроблока/субмакроблока, содержащего образец p_0, а два вектора движения тех же контрольных изображений, которые уже были использованы, чтобы предсказать разделение макроблока/субмакроблока, содержащего образец p_0, используют для предсказания разделения макроблока/субмакроблока, содержащего образец q_0, а оба из следующих условий являются истиной:

  – абсолютная разность между горизонтальными или вертикальными компонентами векторов движения списка 0, которую используют для предсказания разделения двух макроблоков/субмакроблоков, больше или равна 4 в единицах четверти яркости образцов кадра, или абсолютная разность между горизонтальными или вертикальными компонентами векторов движений списка 1, которую используют для предсказания разделения двух макроблоков/субмакроблоков, больше или равна 4 в единицах четверти яркости образцов кадра;

  – абсолютная разность между горизонтальными или вертикальными компонентами векторов движения списка 0, которую используют для предсказания разделения двух макроблоков/субмакроблоков, содержащих образец p_0, и такая же разность для предсказания разделения двух макроблоков/субмакроблоков, содержащих образец q_0 списка 1, больше или равна 4 в единицах четверти яркости образцов кадра. Или абсолютная разность между горизонтальными или вертикальными компонентами векторов движения списка 1, которую используют для предсказания разделения двух макроблоков/субмакроблоков, содержащих образец p_0, и такая же разность для предсказания разделения двух макроблоков/субмакроблоков, содержащих образец q_0 списка 0, больше или равна 4 в единицах четверти яркости образцов кадра.

    ПРИМЕЧАНИЕ. – Вертикальная разность в 4 единицы четверти яркости образцов кадра соответствует разности в 2 единицы четверти яркости образцов поля.

– Иначе выходом должно быть значение bS, равное 0.

8.7.2.2 Процесс отыскания порогов для краев каждого блока

Входы в этот процесс – вход значений образцов p_0, q_0, p_1 и q_1 одного множества образцов вдоль края, предназначенного для фильтрации, chromaEdgeFlag и bS для множества входов образцов, как это определено в п. 8.7.2.

Выходы из этого процесса – переменная filterSamplesFlag, которая указывает, были ли фильтрованы входы образцов, значение indexA и значения порогов переменных α и β.

Положим, что qP_p и qP_q – переменные, определяющие значения параметров квантования для макроблоков, содержащих образцы p_0 и q_0 соответственно. Переменные qP_z (с заменой z на p или q) находят следующим образом:

– Если chromaEdgeFlag равно 0, используют следующие условия:

  – Если макроблок, содержащий образец z_0, это макроблок I_PCM, то qP_z устанавливают на 0.

  – Иначе (макроблок, содержащий образец z_0, это не макроблок I_PCM), qP_z устанавливают на значение QP_Y макроблока, содержащего образец z_0.

– Иначе (chromaEdgeFlag равно 1), используют следующие условия:

  – Если макроблок, содержащий образец z_0, это макроблок I_PCM, то qP_z устанавливают на значение QP_C, которое соответствует значению 0 для QP_Y, как это определено в п. 8.5.5.

  – Иначе (макроблок, содержащий образец z_0, это не макроблок I_PCM), qP_z устанавливают на значение QP_C, которое соответствует значению QP_Y макроблока, содержащего образец z_0, как это определено в п. 8.5.5.

Положим, что qPav – переменная, определяющая средний параметр квантования. Ее находят следующим образом:

$$qPav = (qP_p + qP_q + 1) >> 1. \quad (8\text{-}330)$$

ПРИМЕЧАНИЕ. – В секциях SP и SI значение qP_{av} находят таким же образом, как и в других типах секций. При фильтровом разделении на блоки значение QS_Y из равенства 7-17 не используют.

Положим, что indexA – переменная, которую используют для доступа к таблице α (таблица 8-14), а также к таблице t_{C0} (таблица 8-15), которые применяют для фильтрации краев с помощью значения bS меньше 4, как определено в п. 8.7.2.3. Положим, что indexB – переменная, которую используют для доступа к таблице β (таблица 8-14). Переменные indexA и indexB находят следующим образом: Значения FilterOffsetA и FilterOffsetB – это значения переменных, определенных в п. 7.4.3 для секции, которая содержит макроблок с образцом q_0:

$$indexA = Clip3(0, 51, qP_{av} + FilterOffsetA) \quad (8\text{-}331)$$

$$indexB = Clip3(0, 51, qP_{av} + FilterOffsetB). \quad (8\text{-}332)$$

Пороги переменных α и β определены в таблице 8-14 в зависимости от значений indexA и indexB.

Переменную filterSamplesFlag находят как

$$filterSamplesFlag = (bS != 0 \;\&\&\; Abs(p_0 - q_0) < \alpha \;\&\&\; Abs(p_1 - p_0) < \beta \;\&\&\; Abs(q_1 - q_0) < \beta). \quad (8\text{-}333)$$

Таблица 8-14 – Отыскание значений indexA и indexB в зависимости от переменных α и β смещения порогов

	indexA (для α) или indexB (для β)																									
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
α	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	5	6	7	8	9	10	12	13
β	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	2	3	3	3	3	4	4	4

Таблица 8-14 (конец) – Отыскание значений indexA и indexB в зависимости от переменных α и β смещения порогов

	indexA (для α) или indexB (для β)																									
	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
α	15	17	20	22	25	28	32	36	40	45	50	56	63	71	80	90	101	113	127	144	162	182	203	226	255	255
β	6	6	7	7	8	8	9	9	10	10	11	11	12	12	13	13	14	14	15	15	16	16	17	17	18	18

8.7.2.3 Процесс фильтрации краев со значением bS меньше 4

Входы в этот процесс – вход значений образцов p_i и q_i (i = 0..2) одного множества образцов вдоль края, предназначенного для фильтрации, переменная chromaEdgeFlag, bS, β и indexA для множества входов образцов, как это определено в 8.7.2.

Выходы этого процесса – результирующие значения фильтрации образцов p'_i и q'_i (i = 0..2) для множества значений входов образцов.

Результирующие значения фильтрации образцов p'_0 и q'_0 находят как

$$\Delta = Clip3(-t_C, t_C, ((((q_0 - p_0) << 2) + (p_1 - q_1) + 4) >> 3)) \quad (8\text{-}334)$$

$$p'_0 = Clip1(p_0 + \Delta) \quad (8\text{-}335)$$

$$q'_0 = Clip1(q_0 - \Delta), \quad (8\text{-}336)$$

где порог t_C определяют следующим образом:

– Если chromaEdgeFlag равно 0,

$$t_C = t_{C0} + ((a_p < \beta) ? 1 : 0) + ((a_q < \beta) ? 1 : 0). \quad (8\text{-}337)$$

– Иначе (chromaEdgeFlag равно 1),

$$t_C = t_{C0} + 1. \quad (8\text{-}338)$$

Порог t_{C0} определен в таблице 8-15 в зависимости от значений indexA и bS.

Положим, что a_p и a_q – два порога переменных, определенных как

$$a_p = Abs(p_2 – p_0) \quad (8\text{-}339)$$

$$a_q = Abs(q_2 – q_0). \quad (8\text{-}340)$$

Результирующие значения фильтрации образца p'_1 находят следующим образом:

– Если chromaEdgeFlag равно 0, а a_p меньше чем β, то

$$p'_1 = p_1 + Clip3(–t_{C0}, t_{C0}, (p_2 + ((p_0 + q_0 + 1) >> 1) – (p_1 << 1)) >> 1). \quad (8\text{-}341)$$

– Иначе (chromaEdgeFlag равно 1 или a_p больше или равно β),

$$p'_1 = p_1. \quad (8\text{-}342)$$

Результирующие значения фильтрации образца q'_1 находят следующим образом:

– Если chromaEdgeFlag равно 0, а a_q меньше чем β, то

$$q'_1 = q_1 + Clip3(–t_{C0}, t_{C0}, (q_2 + ((p_0 + q_0 + 1) >> 1) – (q_1 << 1)) >> 1). \quad (8\text{-}343)$$

– Иначе (chromaEdgeFlag равно 1 или a_q больше или равно β),

$$q'_1 = q_1. \quad (8\text{-}344)$$

Результирующие значения фильтрации образцов p'_2 и q'_2 всегда устанавливают равными входным образцам p_2 и q_2:

$$p'_2 = p_2 \quad (8\text{-}345)$$

$$q'_2 = q_2. \quad (8\text{-}346)$$

Таблица 8-15 – Значение переменной усечения фильтра t_{C0} как функции от indexA и bS

	IndexA																									
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
bS = 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
bS = 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
bS = 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Таблица 8-15 (конец) – Значение переменной усечения фильтра t_{C0} как функции от indexA и bS

	IndexA																									
	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
bS = 1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	4	5	6	6	7	8	9	10	11	13
bS = 2	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5	5	6	7	8	8	10	11	12	13	15	17
bS = 3	1	2	2	2	2	3	3	3	4	4	4	5	6	6	7	8	9	10	11	13	14	16	18	20	23	25

8.7.2.4 Процесс фильтрации краев при значении bS, равном 4

Входы в этот процесс – значения входов образцов p_i и q_i (i = 0..3) одного множества образцов вдоль края, предназначенного для фильтрации, переменная chromaEdgeFlag и значения порогов переменных α и β для этого множества образцов, как это определено в п. 8.7.2.

Выходы этого процесса – результирующие значения фильтрации образцов p'_i и q'_i (i = 0..2) для этого множества значений входов образцов.

Положим, что a_p и a_q – два порога переменных, определенных в п. 8.7.2.3, в равенствах 8-339 и 8-340, соответственно.

Результирующие значения фильтрации образца p'_i (i = 0..2) находят следующим образом:

– Если chromaEdgeFlag равно 0 и выполняются следующие условия:

$$a_p < \beta \;\&\&\; Abs(p_0 - q_0) < ((\alpha >> 2) + 2), \tag{8-347}$$

то переменные p'_0, p'_1, и p'_2 находят как

$$p'_0 = (p_2 + 2*p_1 + 2*p_0 + 2*q_0 + q_1 + 4) >> 3 \tag{8-348}$$

$$p'_1 = (p_2 + p_1 + p_0 + q_0 + 2) >> 2 \tag{8-349}$$

$$p'_2 = (2*p_3 + 3*p_2 + p_1 + p_0 + q_0 + 4) >> 3. \tag{8-350}$$

– Иначе (chromaEdgeFlag равно 1 или условия в 8-347 не выполняются), переменные p'_0, p'_1 и p'_2 находят как

$$p'_0 = (2*p_1 + p_0 + q_1 + 2) >> 2 \tag{8-351}$$

$$p'_1 = p_1 \tag{8-352}$$

$$p'_2 = p_2. \tag{8-353}$$

Результирующие значения фильтрации образцов q'_i (i = 0..2) находят следующим образом:

– Если chromaEdgeFlag равно 0 и выполняются следующие условия:

$$a_q < \beta \;\&\&\; Abs(p_0 - q_0) < ((\alpha >> 2) + 2), \tag{8-354}$$

то переменные q'_0, q'_1 и q'_2 находят как

$$q'_0 = (p_1 + 2*p_0 + 2*q_0 + 2*q_1 + q_2 + 4) >> 3 \tag{8-355}$$

$$q'_1 = (p_0 + q_0 + q_1 + q_2 + 2) >> 2 \tag{8-356}$$

$$q'_2 = (2*q_3 + 3*q_2 + q_1 + q_0 + p_0 + 4) >> 3. \tag{8-357}$$

– Иначе (chromaEdgeFlag равно 1 или условия в 8-354 не выполняются), переменные q'_0, q'_1 и q'_2 находят как

$$q'_0 = (2*q_1 + q_0 + p_1 + 2) >> 2 \tag{8-358}$$

$$q'_1 = q_1 \tag{8-359}$$

$$q'_2 = q_2. \tag{8-360}$$

9 Процесс анализа

Входы в этот процесс – биты из RBSP.

Выходы этого процесса – значения элементов синтаксиса.

Процесс активируют, если дескрипторы элемента синтаксиса в таблицах синтаксиса в п. 7.3 равны ue(v), me(v), se(v), te(v) (см. п. 9.1), ce(v) (см. п. 9.2) или ae(v) (см. п. 9.3).

9.1 Процесс анализа кодов Exp-Golomb

Процесс активируют, если дескрипторы элементов синтаксиса в таблицах синтаксиса в п. 7.3 равны ue(v), me(v), se(v), или te(v). Для элементов синтаксиса в пп. 7.3.4 и 7.3.5 этот процесс активируют, если только entropy_coding_mode_flag равно 0.

Входы в этот процесс – биты из RBSP.

Выходы этого процесса – значения элементов синтаксиса.

Элементы синтаксиса, кодированные как ue(v), me(v) или se(v), – это Exp-Golomb-coded. Элементы синтаксиса, кодированные как te(v) – это усеченные элементы Exp-Golomb-coded. Процесс анализа этих элементов синтаксиса начинают с чтения стартовых битов в текущем положении в потоке битов до и включая появление первого ненулевого бита, и подсчитывая число продвинутых битов, которые равны 0. Этот процесс должен быть эквивалентен следующему:

```
leadingZeroBits = –1;
for( b = 0; !b; leadingZeroBits++ )
    b = read_bits( 1 ).
```

Переменную codeNum определяют как:

$$codeNum = 2^{leadingZeroBits} - 1 + read\_bits(leadingZeroBits),$$

где возвратное значение от read_bits(leadingZeroBits) интерпретируют как бинарное представление целого без знака с наиболее значащим битом, записанным первым.

Таблица 9-1 показывает структуру кода Exp-Golomb с разделением строки битов на биты "префикса" и "суффикса". Биты "префикса" – это те биты, которые проанализированы в вышеуказанном псевдокоде для вычисления leadingZeroBits и показаны в виде 0 или 1 в строке битов столбца таблицы 9-1. Биты "суффикса" – это те биты, которые проанализированы при вычислении codeNum и показаны в виде x_i в таблице 9-1, с i в диапазоне от 0 до leadingZeroBits – 1 включительно. Каждый x_i может принимать значения 0 или 1.

Таблица 9-1 – Строки битов с битами "префикса" и "суффикса" и с присвоением диапазону кода codeNum (информативное)

Форма строки битов	Диапазон codeNum
1	0
0 1 x_0	1–2
0 0 1 x_1 x_0	3–6
0 0 0 1 x_2 x_1 x_0	7–14
0 0 0 0 1 x_3 x_2 x_1 x_0	15–30
0 0 0 0 0 1 x_4 x_3 x_2 x_1 x_0	31–62
…	…

Таблица 9-2 иллюстрирует однозначное присвоение строк битов значениям codeNum.

Таблица 9-2 – Строки битов Exp-Golomb и codeNum в явной форме, использованные как ue(v) (информативное)

Строка битов	codeNum
1	0
0 1 0	1
0 1 1	2
0 0 1 0 0	3
0 0 1 0 1	4
0 0 1 1 0	5
0 0 1 1 1	6
0 0 0 1 0 0 0	7
0 0 0 1 0 0 1	8
0 0 0 1 0 1 0	9
…	…

В зависимости от дескриптора, значение элемента синтаксиса находят следующим образом:

– Если элемент синтаксиса кодирован как ue(v), значение этого элемента синтаксиса равно codeNum.

– Иначе, если элемент синтаксиса кодирован как se(v), значение этого элемента синтаксиса находят, активируя процесс отображения для имеющих знак кодов Exp-Golomb, как это определено в п. 9.1.1 с codeNum в качестве входа.

– Иначе, если элемент синтаксиса кодирован как me(v), значение элемента синтаксиса находят, активируя процесс отображения для кодированного образца блока, как это определено в п. 9.1.2 с codeNum в качестве входа.

– Иначе (элемент синтаксиса кодирован как te(v)), диапазон элемента синтаксиса должен быть определен в первую очередь. Диапазон этого элемента синтаксиса может лежать между 0 и x, с x больше или равным 1, и его используют для нахождения значения элемента синтаксиса следующим образом:

  – Если x больше чем 1, codeNum и значение элемента синтаксиса должно быть найдено тем же способом, что и для элементов синтаксиса, кодированных как ue(v).

  – Иначе (x равно 1), процесс анализа codeNum, который равен значению элемента синтаксиса, задан процессом, эквивалентным следующему:

```
b = read_bits( 1 )
codeNum = !b.
```

9.1.1 Процесс отображения кодов Exp-Golomb со знаком

Вход в этот процесс – codeNum, как это определено в п. 9.1.

Выход этого процесса – значение элемента синтаксиса, кодированного как se(v).

Элемент синтаксиса присваивают значению codeNum упорядочением элементов синтаксиса по их абсолютным значениям в возрастающем порядке и представляя положительную величину для заданного абсолютного значения наименьшим codeNum. В таблице 9-3 приведены правила присвоения.

Таблица 9-3 – Присвоение элементов синтаксиса значениям codeNum для элементов синтаксиса se(v), кодированных кодом Exp-Golomb со знаком

codeNum	Значение элемента синтаксиса
0	0
1	1
2	–1
3	2
4	–2
5	3
6	–3
k	$(-1)^{k+1}$ Ceil(k÷2)

9.1.2 Процесс отображения кодированного блока образца

Вход в этот процесс – codeNum, как это определено в п. 9.1.

Выход этого процесса – значение элементов синтаксиса coded_block_pattern, кодированных как me(v).

В таблице 9-4 показано присвоение coded_block_pattern значению codeNum в зависимости от того, является ли режим предсказания макроблока Intra_4x4 или Inter.

Таблица 9-4 – Присвоение codeNum значению coded_block_pattern для режимов предсказания макроблока

codeNum	coded_block_pattern	
	Intra_4x4	Inter
0	47	0
1	31	16
2	15	1
3	0	2
4	23	4
5	27	8
6	29	32
7	30	3
8	7	5
9	11	10
10	13	12
11	14	15
12	39	47
13	43	7
14	45	11
15	46	13
16	16	14
17	3	6
18	5	9
19	10	31
20	12	35
21	19	37
22	21	42
23	26	44
24	28	33
25	35	34
26	37	36
27	42	40
28	44	39
29	1	43
30	2	45
31	4	46
32	8	17

33	17	18
34	18	20
35	20	24
36	24	19
37	6	21
38	9	26
39	22	28
40	25	23
41	32	27
42	33	29
43	34	30
44	36	22
45	40	25
46	38	38
47	41	41

9.2 Процесс анализа CAVLC для уровней коэффициентов преобразования

Процесс активируют, если элементы синтаксического анализа с дескриптором равны ce(v) (согласно п. 7.3.5.3.1) и если entropy_coding_mode_flag равно 0.

Входы в этот процесс – биты из данных секции, максимальное число ненулевых уровней коэффициентов преобразования maxNumCoeff, индекс блока яркости luma4x4BlkIdx или индекс блока цветности chroma4x4BlkIdx текущего блока уровней коэффициентов преобразования.

Выход этого процесса – список coeffLevel, содержащий уровни коэффициентов преобразования блока яркости с индексом блока luma4x4BlkIdx или блока цветности с индексом блока chroma4x4BlkIdx.

Процесс описывают в следующем порядке:

1. Все уровни коэффициентов преобразования, с индексами от 0 до maxNumCoeff –1, в списке coeffLevel устанавливают равными 0.
2. Общее число ненулевых уровней коэффициентов преобразования TotalCoeff(coeff_token) и число концевых уровней коэффициентов преобразования TrailingOnes(coeff_token) находят из анализа coeff_token (см. п. 9.2.1) следующим образом:
   - Если число ненулевых уровней коэффициентов преобразования TotalCoeff(coeff_token) равно 0, список coeffLevel, содержащий значения 0, возвращают и далее не производят никаких действий.
   - Иначе выполняют следующие шаги:
     a. Ненулевые уровни коэффициентов преобразования находят анализом trailing_ones_sign_flag, level_prefix, и level_suffix (см. п. 9.2.2).
     b. Однократные проходы нулевых уровней коэффициентов преобразования перед каждым ненулевым уровнем коэффициентов преобразования находят анализом total_zeros и run_before (см. п. 9.2.3).
     c. Информацию об уровнях и проходах собирают в список coeffLevel (см. п. 9.2.4).

9.2.1 Процесс анализа общего числа уровней коэффициентов преобразования и концевых уровней коэффициентов преобразования

Входы в этот процесс – биты из данных секции, максимальное число ненулевых уровней коэффициентов преобразования maxNumCoeff, индекс блока яркости luma4x4BlkIdx или блока цветности chroma4x4BlkIdx текущего блока преобразования.

Выходы этого процесса – TotalCoeff(coeff_token) и TrailingOnes(coeff_token).

Элемент синтаксиса coeff_token декодируют, используя одно из пяти значений VLC, определенных в пятом крайнем правом столбце таблицы 9-5. Каждое значение VLC определяет TotalCoeff(coeff_token) и TrailingOnes(coeff_token) для заданного кодового слова coeff_token. Выбор VLC зависит от переменной nC, которую находят следующим образом:

– Если процесс анализа CAVLC активируют для ChromaDCLevel, то nC устанавливают равным –1.

– Иначе применяют следующее:

  – Если процесс анализа CAVLC активируют для Intra16x16DCLevel, то luma4x4BlkIdx устанавливают равным 0.

  – Переменные blkA и blkB находят следующим образом:

    – Если процесс анализа CAVLC активируют для Intra16x16DCLevel, Intra16x16ACLevel или LumaLevel, то активируют процесс, определенный в п. 6.4.7.3, с luma4x4BlkIdx в качестве входа, и присваивают выходу значения mbAddrA, mbAddrB, luma4x4BlkIdxA и luma4x4BlkIdxB. Блоку яркости 4x4, определенному mbAddrA\luma4x4BlkIdxA, присваивают blkA, а блоку яркости 4x4, определенному mbAddrB\luma4x4BlkIdxB, присваивают blkB.

    – Иначе (процесс анализа CAVLC активируют для ChromaACLevel), процесс, определенный в п. 6.4.7.4, активируют с chroma4x4BlkIdx в качестве входа, а выходу присваивают значения mbAddrA, mbAddrB, chroma4x4BlkIdxA и chroma4x4BlkIdxB. Блоку цветности 4x4, определенному mbAddrA\iCbCr\chroma4x4BlkIdxA, присваивают blkA, а блоку цветности 4x4, определенному mbAddrB\iCbCr\chroma4x4BlkIdxB, присваивают blkB.

  – Допустим, что nA и nB – число ненулевых уровней коэффициентов преобразования (заданных TotalCoeff(coeff_token)) в блоке уровней коэффициентов преобразования blkA, расположенном слева от текущего блока, и в блоке уровней коэффициентов преобразования blkB, расположенном выше текущего блока, соответственно.

  – С заменой N на A и B в mbAddrN, blkN и nN применяют следующее:

    – Если любое из следующих условий истина, то nN устанавливают равным 0:

      – mbAddrN недоступно;

      – текущий макроблок кодирован с использованием режима Intra предсказания, constrained_intra_pred_flag равно 1, а mbAddrN кодирован с использованием режима Inter предсказания, а также используют данные разделения секции (nal_unit_type находится в диапазоне от 2 до 4 включительно);

      – макроблок mbAddrN имеет значения mb_type, равные P_Skip или B_Skip;

      – все остаточные AC уровни коэффициентов преобразования смежного блока blkN равны 0, благодаря соответствующим битам CodedBlockPatternLuma или CodedBlockPatternChroma, равным 0.

    – Иначе, если mbAddrN – макроблок I_PCM, то nN устанавливают равным 16.

    – Иначе nN устанавливают равным значению TotalCoeff(coeff_token) смежного блока blkN.

      ПРИМЕЧАНИЕ. – Значения nA и nB, которые находят, используя TotalCoeff(coeff_token), не включают DC уровни коэффициентов преобразования в макроблоках Intra_16x16 или DC уровни коэффициентов преобразования в блоках цветности, т. к. эти уровни коэффициентов преобразования декодируют раздельно. Если блоки выше или слева принадлежат макроблоку Intra_16x16 или являются блоками цветности, то nA и nB – это число декодированных ненулевых AC уровней коэффициентов преобразования.

      ПРИМЕЧАНИЕ. – При анализе Intra16x16DCLevel значения nA и nB основаны на числе ненулевых уровней коэффициентов преобразования в смежных блоках 4x4 и на числе ненулевых DC уровней коэффициентов преобразования в смежных блоках 16x16.

  – При заданных значениях nA и nB, переменную nC находят следующим образом:

    – Если доступны оба значения mbAddrA и mbAddrB, переменную nC устанавливают равной (nA + nB + 1) >> 1.

    – Иначе (mbAddrA недоступно или mbAddrB недоступно), переменную nC устанавливают равной nA + nB.

Результирующее значение TotalCoeff(coeff_token) от декодирования coeff_token должно лежать в диапазоне от 0 до maxNumCoeff включительно.

Таблица 9-5 – Отображение coeff_token в TotalCoeff(coeff_token) и в TrailingOnes(coeff_token)

TrailingOnes (coeff_token)	TotalCoeff (coeff_token)	0 <= nC < 2	2 <= nC < 4	4 <= nC < 8	8 <= nC	nC = = -1
0	0	1	11	1111	0000 11	01
0	1	0001 01	0010 11	0011 11	0000 00	0001 11
1	1	01	10	1110	0000 01	1
0	2	0000 0111	0001 11	0010 11	0001 00	0001 00
1	2	0001 00	0011 1	0111 1	0001 01	0001 10
2	2	001	011	1101	0001 10	001
0	3	0000 0011 1	0000 111	0010 00	0010 00	0000 11
1	3	0000 0110	0010 10	0110 0	0010 01	0000 011
2	3	0000 101	0010 01	0111 0	0010 10	0000 010
3	3	0001 1	0101	1100	0010 11	0001 01
0	4	0000 0001 11	0000 0111	0001 111	0011 00	0000 10
1	4	0000 0011 0	0001 10	0101 0	0011 01	0000 0011
2	4	0000 0101	0001 01	0101 1	0011 10	0000 0010
3	4	0000 11	0100	1011	0011 11	0000 000
0	5	0000 0000 111	0000 0100	0001 011	0100 00	–
1	5	0000 0001 10	0000 110	0100 0	0100 01	–
2	5	0000 0010 1	0000 101	0100 1	0100 10	–
3	5	0000 100	0011 0	1010	0100 11	–
0	6	0000 0000 0111 1	0000 0011 1	0001 001	0101 00	–
1	6	0000 0000 110	0000 0110	0011 10	0101 01	–
2	6	0000 0001 01	0000 0101	0011 01	0101 10	–
3	6	0000 0100	0010 00	1001	0101 11	–
0	7	0000 0000 0101 1	0000 0001 111	0001 000	0110 00	–
1	7	0000 0000 0111 0	0000 0011 0	0010 10	0110 01	–
2	7	0000 0000 101	0000 0010 1	0010 01	0110 10	–
3	7	0000 0010 0	0001 00	1000	0110 11	–
0	8	0000 0000 0100 0	0000 0001 011	0000 1111	0111 00	–
1	8	0000 0000 0101 0	0000 0001 110	0001 110	0111 01	–
2	8	0000 0000 0110 1	0000 0001 101	0001 101	0111 10	–
3	8	0000 0001 00	0000 100	0110 1	0111 11	–
0	9	0000 0000 0011 11	0000 0000 1111	0000 1011	1000 00	–
1	9	0000 0000 0011 10	0000 0001 010	0000 1110	1000 01	–
2	9	0000 0000 0100 1	0000 0001 001	0001 010	1000 10	–

3	9	0000 0000 100	0000 0010 0	0011 00	1000 11	–
0	10	0000 0000 0010 11	0000 0000 1011	0000 0111 1	1001 00	–
1	10	0000 0000 0010 10	0000 0000 1110	0000 1010	1001 01	–
2	10	0000 0000 0011 01	0000 0000 1101	0000 1101	1001 10	–
3	10	0000 0000 0110 0	0000 0001 100	0001 100	1001 11	–
0	11	0000 0000 0001 111	0000 0000 1000	0000 0101 1	1010 00	–
1	11	0000 0000 0001 110	0000 0000 1010	0000 0111 0	1010 01	–
2	11	0000 0000 0010 01	0000 0000 1001	0000 1001	1010 10	–
3	11	0000 0000 0011 00	0000 0001 000	0000 1100	1010 11	–
0	12	0000 0000 0001 011	0000 0000 0111 1	0000 0100 0	1011 00	–
1	12	0000 0000 0001 010	0000 0000 0111 0	0000 0101 0	1011 01	–
2	12	0000 0000 0001 101	0000 0000 0110 1	0000 0110 1	1011 10	–
3	12	0000 0000 0010 00	0000 0000 1100	0000 1000	1011 11	–
0	13	0000 0000 0000 1111	0000 0000 0101 1	0000 0011 01	1100 00	–
1	13	0000 0000 0000 001	0000 0000 0101 0	0000 0011 1	1100 01	–
2	13	0000 0000 0001 001	0000 0000 0100 1	0000 0100 1	1100 10	–
3	13	0000 0000 0001 100	0000 0000 0110 0	0000 0110 0	1100 11	–
0	14	0000 0000 0000 1011	0000 0000 0011 1	0000 0010 01	1101 00	–
1	14	0000 0000 0000 1110	0000 0000 0010 11	0000 0011 00	1101 01	–
2	14	0000 0000 0000 1101	0000 0000 0011 0	0000 0010 11	1101 10	–
3	14	0000 0000 0001 000	0000 0000 0100 0	0000 0010 10	1101 11	–
0	15	0000 0000 0000 0111	0000 0000 0010 01	0000 0001 01	1110 00	–
1	15	0000 0000 0000 1010	0000 0000 0010 00	0000 0010 00	1110 01	–
2	15	0000 0000 0000 1001	0000 0000 0010 10	0000 0001 11	1110 10	–
3	15	0000 0000 0000 1100	0000 0000 0000 1	0000 0001 10	1110 11	–
0	16	0000 0000 0000 0100	0000 0000 0001 11	0000 0000 01	1111 00	–
1	16	0000 0000 0000 0110	0000 0000 0001 10	0000 0001 00	1111 01	–
2	16	0000 0000 0000 0101	0000 0000 0001 01	0000 0000 11	1111 10	–
3	16	0000 0000 0000 1000	0000 0000 0001 00	0000 0000 10	1111 11	–

9.2.2 Процесс анализа информационного уровня

Входы в этот процесс – биты из данных секции, число ненулевых уровней коэффициентов преобразования TotalCoeff(coeff_token) и число концевых ненулевых уровней коэффициентов преобразования TrailingOnes(coeff_token).

Выход этого процесса – список с именами уровней, в котором содержатся уровни коэффициентов преобразования.

Первоначально индекс i устанавливают равным 0. Затем следующую процедуру итеративно применяют TrailingOnes(coeff_token) раз, чтобы декодировать концевые уровни коэффициентов преобразования (если таковые есть):

– 1-битовый элемент синтаксиса trailing_ones_sign_flag декодируют и оценивают следующим образом:

  - Если trailing_ones_sign_flag равно 0, то значение +1 присваивают level[i].
  - Иначе (trailing_ones_sign_flag равно 1), значение –1 присваивают level[i].
- Индекс i возрастает на 1.

После декодирования концевого уровня коэффициентов преобразования переменную suffixLength инициализируют следующим образом:

- Если TotalCoeff(coeff_token) больше 10, а TrailingOnes(coeff_token) меньше 3, suffixLength устанавливают равным 1.
- Иначе (TotalCoeff(coeff_token) меньше или равно 10 или TrailingOnes(coeff_token) равно 3), suffixLength устанавливают равным 0.

Следующую процедуру применяют итеративно (TotalCoeff(coeff_token) – TrailingOnes(coeff_token)) раз, чтобы декодировать остаток уровней (если таковые есть):

- Элемент синтаксиса level_prefix декодируют, используя значения VLC, определенные в таблице 9-6.
- Переменную levelSuffixSize устанавливают равной переменной suffixLength, за исключением следующих двух случаев.
- Если level_prefix равно 14 и suffixLength равно 0, levelSuffixSize устанавливают равным 4.
- Если level_prefix равно 15, levelSuffixSize устанавливают равным 12.
- Элемент синтаксиса level_suffix декодируют следующим образом:
  - Если levelSuffixSize больше 0, элемент синтаксиса level_suffix декодируют как целое без знака представление u(v) с levelSuffixSize битов.
  - Иначе (levelSuffixSize равно 0), элемент синтаксиса level_suffix должен быть принят равным 0.
- Переменную levelCode устанавливают равной (level_prefix << suffixLength) + level_suffix.
- Если level_prefix равно 15, а suffixLength равно 0, levelCode возрастает на 15.
- Если индекс i равен TrailingOnes(coeff_token), а TrailingOnes(coeff_token) меньше 3, levelCode возрастает на 2.
- Переменную level[i] находят следующим образом:
  - Если levelCode – четное число, то значение (levelCode + 2) >> 1 присваивают level[i].
  - Иначе значение (–levelCode – 1) >> 1 присваивают level[i].
- Если suffixLength равно 0, suffixLength устанавливают равным 1.
- Если абсолютное значение level[i] больше (3 << (suffixLength – 1)), а suffixLength меньше 6, suffixLength возрастает на 1.
- Индекс i возрастает на 1.

Таблица 9-6 – Таблица кодовых слов для level_prefix

level_prefix	Строка битов
0	1
1	01
2	001
3	0001
4	0000 1
5	0000 01
6	0000 001
7	0000 0001
8	0000 0000 1
9	0000 0000 01
10	0000 0000 001
11	0000 0000 0001
12	0000 0000 0000 1
13	0000 0000 0000 01
14	0000 0000 0000 001
15	0000 0000 0000 0001

9.2.3 Процесс анализа информации проходов

Входы в этот процесс – биты из данных секции, число ненулевых уровней коэффициентов преобразования TotalCoeff(coeff_token) и максимальное число ненулевых уровней коэффициентов преобразования maxNumCoeff.

Выход этого процесса – список проходов нулевых уровней коэффициентов преобразования, которые предшествовали ненулевым уровням коэффициентов преобразования в результате проходов.

Первоначально индекс i устанавливают равным 0.

Переменную zerosLeft находят следующим образом:

– Если число ненулевых уровней коэффициентов преобразования TotalCoeff(coeff_token) равно максимальному числу ненулевых уровней коэффициентов преобразования maxNumCoeff, то переменную zerosLeft устанавливают равной 0.

– Иначе (число ненулевых уровней коэффициентов преобразования TotalCoeff(coeff_token) меньше максимального числа ненулевых уровней коэффициентов преобразования maxNumCoeff), total_zeros декодируют, а zerosLeft устанавливают равным его значению.

Значение VLC, которое используют для декодирования total_zeros, находят следующим образом:

– Если maxNumCoeff равно 4, используют одно из значений VLC, определенных в таблице 9-9.

– Иначе (maxNumCoeff не равно 4), используют VLC из таблицы 9-7 и таблицы 9-8.

Следующую процедуру далее применяют итеративно (TotalCoeff(coeff_token) – 1) раз:

– Переменную run[i] находят следующим образом:

  – Если zerosLeft больше нуля, то значение run_before декодируют, основываясь на таблице 9-10 и zerosLeft. Значение run[i] устанавливают равным run_before.

  – Иначе (zerosLeft равно 0), run[i] устанавливают равным 0.

– Значение run[i] вычитают из zerosLeft, а результат присваивают значению zerosLeft. Результат вычитания должен быть больше или равен 0.

– Индекс i возрастает на 1.

Наконец, значение zerosLeft присваивают run[i].

Таблица 9-7 – Таблицы total_zeros для блоков 4x4 с TotalCoeff(coeff_token) от 1 до 7

total_zeros	TotalCoeff(coeff_token)						
	1	2	3	4	5	6	7
0	1	111	0101	0001 1	0101	0000 01	0000 01
1	011	110	111	111	0100	0000 1	0000 1
2	010	101	110	0101	0011	111	101
3	0011	100	101	0100	111	110	100
4	0010	011	0100	110	110	101	011
5	0001 1	0101	0011	101	101	100	11
6	0001 0	0100	100	100	100	011	010
7	0000 11	0011	011	0011	011	010	0001
8	0000 10	0010	0010	011	0010	0001	001
9	0000 011	0001 1	0001 1	0010	0000 1	001	0000 00
10	0000 010	0001 0	0001 0	0001 0	0001	0000 00	
11	0000 0011	0000 11	0000 01	0000 1	0000 0		
12	0000 0010	0000 10	0000 1	0000 0			
13	0000 0001 1	0000 01	0000 00				
14	0000 0001 0	0000 00					
15	0000 0000 1						

Таблица 9-8 – Таблицы total_zeros для блоков 4x4 с TotalCoeff(coeff_token) от 8 до 15

total_zeros	TotalCoeff(coeff_token)							
	8	9	10	11	12	13	14	15
0	0000 01	0000 01	0000 1	0000	0000	000	00	0
1	0001	0000 00	0000 0	0001	0001	001	01	1
2	0000 1	0001	001	001	01	1	1	
3	011	11	11	010	1	01		
4	11	10	10	1	001			
5	10	001	01	011				
6	010	01	0001					
7	001	0000 1						
8	0000 00							

Таблица 9-9 – Таблицы total_zeros для DC блоков цветности 2x2

total_zeros	TotalCoeff(coeff_token)		
	1	2	3
0	1	1	1
1	01	01	0
2	001	00	
3	000		

Таблица 9-10 – Таблицы для run_before

run_before	zerosLeft						
	1	2	3	4	5	6	>6
0	1	1	11	11	11	11	111
1	0	01	10	10	10	000	110
2	–	00	01	01	011	001	101
3	–	–	00	001	010	011	100
4	–	–	–	000	001	010	011
5	–	–	–	–	000	101	010
6	–	–	–	–	–	100	001
7	–	–	–	–	–	–	0001
8		–	–	–	–	–	00001
9	–	–	–	–	–	–	000001
10	–	–	–	–	–	–	0000001
11	–	–	–	–	–	–	00000001
12	–	–	–	–	–	–	000000001
13	–	–	–	–	–	–	0000000001
14	–	–	–	–	–	–	00000000001

9.2.4 Объединение информации об уровне и проходе

Вход в этот процесс – список вызванных уровней коэффициентов преобразования, список вызванных проходов и число ненулевых уровней коэффициентов преобразования TotalCoeff(coeff_token).

Выход этого процесса – список coeffLevel уровней коэффициентов преобразования.

Переменную coeffNum устанавливают равной –1, а индекс i устанавливают равным (TotalCoeff(coeff_token) – 1). Следующую процедуру итеративно применяют TotalCoeff(coeff_token) раз:

– coeffNum возрастает на run[i] + 1.

– coeffLevel[coeffNum] устанавливают равным level[i].

– Индекс i уменьшается на 1.

9.3 Процесс анализа CABAC данных секции

Процесс активируют при анализе элементов синтаксиса с дескриптором ae(v) (согласно пп. 7.3.4 и 7.3.5) и при entropy_coding_mode_flag равном 1.

Входы в этот процесс – запрос значения элемента синтаксиса и значений ранее анализированных элементов синтаксиса.

Выход этого процесса – значение элемента синтаксиса.

При начале анализа данных секции по п. 7.3.4 активируют процесс инициализации процесса анализа CABAC, как это определено в п. 9.3.1.

Анализ элементов синтаксиса производят следующим образом:

Для каждого запрошенного значения элемента синтаксиса находят бинаризацию, как описано в п. 9.3.2.

Бинаризация элемента синтаксиса и последовательность анализированных бинов определяет протекание процесса декодирования, как это описано в п. 9.3.3.

Для каждого бина в процессе бинаризации элемента синтаксиса, проиндексированного переменной binIdx, индекс контекста ctxIdx находят, как это определено в п. 9.3.3.1.

Для каждого ctxIdx активируют процесс арифметического декодирования, как это определено в п. 9.3.3.2.

Результирующую последовательность (b_0 .. b_{binIdx}) проанализированных бинов сравнивают со множеством строк бинов, заданных процессом бинаризации после декодирования каждого бина. Если последовательность соответствует строке бинов в заданном множестве, то элементу синтаксиса должно быть присвоено соответствующее значение.

В случае если запрошенное значение элемента синтаксиса обработано для типа элемента синтаксиса mb_type, а декодированное значение типа mb_type – это I_PCM, устройство декодирования должно быть инициализировано после декодирования pcm_alignment_zero_bit и всех данных pcm_byte, как это определено в п. 9.3.1.2.

Весь процесс анализа CABAC проиллюстрирован на блок-схеме рисунка 9-1 с аббревиатурой SE для элемента синтаксиса.

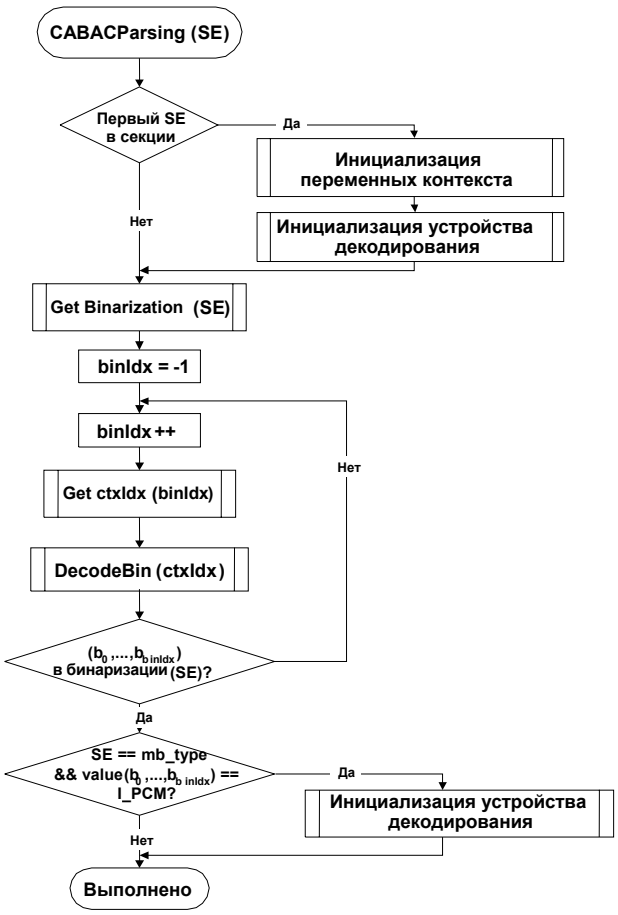

Рисунок 9-1 – Иллюстрация процесса анализа CABAC элемента синтаксиса SE (информативное)

9.3.1 Процесс инициализации

Выходы этого процесса – инициализированные внутренние переменные CABAC.

Процессы в пп. 9.3.1.1 и 9.3.1.2 активируют при начале анализа данных секции типа секций из п. 7.3.4.

Процесс в п. 9.3.1.2 также активируют после декодирования pcm_alignment_zero_bit и всех данных pcm_byte макроблока типа I_PCM.

9.3.1.1 Процесс инициализации контекстных переменных

Выходы этого процесса – инициализированные контекстные переменные CABAC, индексированные значением ctxIdx.

В таблицах 9-12–9-23 содержатся значения переменных n и m, использованных при инициализации контекстных переменных, которые присвоены всем элементам синтаксиса в пп. 7.3.4 и 7.3.5, за исключением флага конца секции (end-of-slice flag).

Для каждой контекстной переменной инициализируют две переменные – pStateIdx и valMPS.

ПРИМЕЧАНИЕ. – Переменная pStateIdx соответствует индексу состояния вероятности, а переменная valMPS – значению наиболее вероятного символа, как далее описано в п. 9.3.3.2.

Два присвоенных для инициализации значения, pStateIdx и valMPS, находят из $SliceQP_Y$, которое в свою очередь находят из равенства 7-16. Заданы две таблицы входов (m, n),

1. preCtxState = Clip3(1, 126, ((m * $SliceQP_Y$) >> 4) + n)
2. if(preCtxState <= 63) {

   pStateIdx = 63 – preCtxState

```
        valMPS = 0
    } else {
        pStateIdx = preCtxState – 64
        valMPS = 1
    }
```

В таблице 9-11 перечислены значения ctxIdx, которые требуют инициализации для каждого из типов секций. В этой таблице также перечислены числа, которые включают значения m и n, необходимые для инициализации. Для типов секций P, SP и B инициализация также зависит от значения элемента синтаксиса cabac_init_idc. Заметим, что имя этого элемента синтаксиса не влияет на процесс инициализации.

Таблица 9-11 – Объединение ctxIdx и элементов синтаксиса для каждого типа секций в процессе инициализации

	Элемент синтаксиса	**Таблица**	**Тип секции**			
			SI	**I**	**P, SP**	**B**
slice_data( )	mb_skip_flag	Таблица 9-13 Таблица 9-14			11–13	24–26
	mb_field_decoding_flag	Таблица 9-18	70–72	70–72	70–72	70–72
macroblock_layer( )	mb_type	Таблица 9-12 Таблица 9-13 Таблица 9-14	0–10	3–10	14–20	27–35
	coded_block_pattern (яркость)	Таблица 9-18	73–76	73–76	73–76	73–76
	coded_block_pattern (цветность)	Таблица 9-18	77–84	77–84	77–84	77–84
	mb_qp_delta	Таблица 9-17	60–63	60–63	60–63	60–63
mb_pred( )	prev_intra4x4_pred_mode_flag	Таблица 9-17	68	68	68	68
	rem_intra4x4_pred_mode	Таблица 9-17	69	69	69	69
	intra_chroma_pred_mode	Таблица 9-17	64–67	64–67	64–67	64–67
mb_pred( ) и sub_mb_pred( )	ref_idx_l0	Таблица 9-16			54–59	54–59
	ref_idx_l1	Таблица 9-16				54–59
	mvd_l0[ ][ ][0]	Таблица 9-15			40–46	40–46
	mvd_l1[ ][ ][0]	Таблица 9-15				40–46
	mvd_l0[ ][ ][1]	Таблица 9-15			47–53	47–53
	mvd_l1[ ][ ][1]	Таблица 9-15				47–53
sub_mb_pred( )	sub_mb_type	Таблица 9-13 Таблица 9-14			21–23	36–39
residual_block_cabac( )	coded_block_flag	Таблица 9-18	85–104	85–104	85–104	85–104
	significant_coeff_flag[ ]	Таблица 9-19 Таблица 9-22	105–165, 277–337	105–165, 277–337	105–165, 277–337	105–165, 277–337
	last_significant_coeff_flag[ ]	Таблица 9-20, Таблица 9-23	166–226, 338–398	166–226, 338–398	166–226, 338–398	166–226, 338–398
	coeff_abs_level_minus1[ ]	Таблица 9-21	227–275	227–275	227–275	227–275

ПРИМЕЧАНИЕ. – Значение ctxIdx, равное 276, объединяют с end_of_slice_flag и с бином mb_type, который определяет тип макроблока I_PCM. Процесс декодирования, определенный в п. 9.3.3.2.4, применяют к ctxIdx, равному 276. Однако этот

процесс декодирования можно также применять, используя процесс декодирования, определенный в п. 9.3.3.2.1. В этом случае первоначальные значения, объединенные с ctxIdx, равным 276, определены как pStateIdx = 63 и valMPS = 0, где pStateIdx = 63 представляет неадаптивное состояние вероятности**.**

Таблица 9-12 – Значения переменных m и n для ctxIdx от 0 до 10

Переменные инициализации	**ctxIdx**										
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
m	20	2	3	20	2	3	–28	–23	–6	–1	7
n	–15	54	74	–15	54	74	127	104	53	54	51

Таблица 9-13 – Значения переменных m и n для ctxIdx от 11 до 23

Значение cabac_init_idc	**Переменные инициализации**	**ctxIdx**												
		11	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**
0	**m**	23	23	21	1	0	–37	5	–13	–11	1	12	–4	17
	n	33	2	0	9	49	118	57	78	65	62	49	73	50
1	**m**	22	34	16	–2	4	–29	2	–6	–13	5	9	–3	10
	n	25	0	0	9	41	118	65	71	79	52	50	70	54
2	**m**	29	25	14	–10	–3	–27	26	–4	–24	5	6	–17	14
	n	16	0	0	51	62	99	16	85	102	57	57	73	57

Таблица 9-14 – Значения переменных m и n для ctxIdx от 24 до 39

Значение cabac_init_idc	**Переменные инициализации**	**ctxIdx**															
		24	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**
0	**m**	18	9	29	26	16	9	–46	–20	1	–13	–11	1	–6	–17	–6	9
	n	64	43	0	67	90	104	127	104	67	78	65	62	86	95	61	45
1	**m**	26	19	40	57	41	26	–45	–15	–4	–6	–13	5	6	–13	0	8
	n	34	22	0	2	36	69	127	101	76	71	79	52	69	90	52	43
2	**m**	20	20	29	54	37	12	–32	–22	–2	–4	–24	5	–6	–14	–6	4
	n	40	10	0	0	42	97	127	117	74	85	102	57	93	88	44	55

Таблица 9-15 – Значения переменных m и n для ctxIdx от 40 до 53

Значение cabac_init_idc	Переменные инициализации	ctxIdx													
		40	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**
0	**m**	–3	–6	–11	6	7	–5	2	0	–3	–10	5	4	–3	0
	n	69	81	96	55	67	86	88	58	76	94	54	69	81	88
1	**m**	–2	–5	–10	2	2	–3	–3	1	–3	–6	0	–3	–7	–5
	n	69	82	96	59	75	87	100	56	74	85	59	81	86	95
2	**m**	–11	–15	–21	19	20	4	6	1	–5	–13	5	6	–3	–1
	n	89	103	116	57	58	84	96	63	85	106	63	75	90	101

Таблица 9-16 – Значения переменных m и n для ctxIdx от 54 до 59

Значение cabac_init_idc	Переменные инициализации	ctxIdx					
		54	**55**	**56**	**57**	**58**	**59**
0	**m**	–7	–5	–4	–5	–7	1
	n	67	74	74	80	72	58
1	**m**	–1	–1	1	–2	–5	0
	n	66	77	70	86	72	61
2	**m**	3	–4	–2	–12	–7	1
	n	55	79	75	97	50	60

Таблица 9-17 – Значения переменных m и n для ctxIdx от 60 до 69

Переменные инициализации	ctxIdx									
	60	**61**	**62**	**63**	**64**	**65**	**66**	**67**	**68**	**69**
m	0	0	0	0	–9	4	0	–7	13	3
n	41	63	63	63	83	86	97	72	41	62

Таблица 9-18 – Значения переменных m и n для ctxIdx от 70 до 104

ctxIdx	Секции I и SI		Значение cabac_init_idc						ctxIdx	Секции I и SI		Значение cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n	m	n
70	0	11	0	45	13	15	7	34	**88**	–11	115	–13	108	–4	92	5	78
71	1	55	–4	78	7	51	–9	88	**89**	–12	63	–3	46	0	39	–6	55
72	0	69	–3	96	2	80	–20	127	**90**	–2	68	–1	65	0	65	4	61
73	–17	127	–27	126	–39	127	–36	127	**91**	–15	84	–1	57	–15	84	–14	83
74	–13	102	–28	98	–18	91	–17	91	**92**	–13	104	–9	93	–35	127	–37	127
75	0	82	–25	101	–17	96	–14	95	**93**	–3	70	–3	74	–2	73	–5	79
76	–7	74	–23	67	–26	81	–25	84	**94**	–8	93	–9	92	–12	104	–11	104
77	–21	107	–28	82	–35	98	–25	86	**95**	–10	90	–8	87	–9	91	–11	91
78	–27	127	–20	94	–24	102	–12	89	**96**	–30	127	–23	126	–31	127	–30	127
79	–31	127	–16	83	–23	97	–17	91	**97**	–1	74	5	54	3	55	0	65
80	–24	127	–22	110	–27	119	–31	127	**98**	–6	97	6	60	7	56	–2	79
81	–18	95	–21	91	–24	99	–14	76	**99**	–7	91	6	59	7	55	0	72
82	–27	127	–18	102	–21	110	–18	103	**100**	–20	127	6	69	8	61	–4	92
83	–21	114	–13	93	–18	102	–13	90	**101**	–4	56	–1	48	–3	53	–6	56
84	–30	127	–29	127	–36	127	–37	127	**102**	–5	82	0	68	0	68	3	68
85	–17	123	–7	92	0	80	11	80	**103**	–7	76	–4	69	–7	74	–8	71
86	–12	115	–5	89	–5	89	5	76	**104**	–22	125	–8	88	–9	88	–13	98
87	–16	122	–7	96	–7	94	2	84									

Таблица 9-19 – Значения переменных m и n для ctxIdx от 105 до 165

ctxIdx	Секции I и SI		Значение cabac_init_idc						ctxIdx	Секции I и SI		Значение cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n	m	n
105	–7	93	–2	85	–13	103	–4	86	**136**	–13	101	5	53	0	58	–5	75
106	–11	87	–6	78	–13	91	–12	88	**137**	–13	91	–2	61	–1	60	–8	80
107	–3	77	–1	75	–9	89	–5	82	**138**	–12	94	0	56	–3	61	–21	83
108	–5	71	–7	77	–14	92	–3	72	**139**	–10	88	0	56	–8	67	–21	64
109	–4	63	2	54	–8	76	–4	67	**140**	–16	84	–13	63	–25	84	–13	31
110	–4	68	5	50	–12	87	–8	72	**141**	–10	86	–5	60	–14	74	–25	64
111	–12	84	–3	68	–23	110	–16	89	**142**	–7	83	–1	62	–5	65	–29	94
112	–7	62	1	50	–24	105	–9	69	**143**	–13	87	4	57	5	52	9	75
113	–7	65	6	42	–10	78	–1	59	**144**	–19	94	–6	69	2	57	17	63
114	8	61	–4	81	–20	112	5	66	**145**	1	70	4	57	0	61	–8	74
115	5	56	1	63	–17	99	4	57	**146**	0	72	14	39	–9	69	–5	35
116	–2	66	–4	70	–78	127	–4	71	**147**	–5	74	4	51	–11	70	–2	27
117	1	64	0	67	–70	127	–2	71	**148**	18	59	13	68	18	55	13	91
118	0	61	2	57	–50	127	2	58	**149**	–8	102	3	64	–4	71	3	65
119	–2	78	–2	76	–46	127	–1	74	**150**	–15	100	1	61	0	58	–7	69
120	1	50	11	35	–4	66	–4	44	**151**	0	95	9	63	7	61	8	77
121	7	52	4	64	–5	78	–1	69	**152**	–4	75	7	50	9	41	–10	66
122	10	35	1	61	–4	71	0	62	**153**	2	72	16	39	18	25	3	62
123	0	44	11	35	–8	72	–7	51	**154**	–11	75	5	44	9	32	–3	68
124	11	38	18	25	2	59	–4	47	**155**	–3	71	4	52	5	43	–20	81
125	1	45	12	24	–1	55	–6	42	**156**	15	46	11	48	9	47	0	30
126	0	46	13	29	–7	70	–3	41	**157**	–13	69	–5	60	0	44	1	7
127	5	44	13	36	–6	75	–6	53	**158**	0	62	–1	59	0	51	–3	23
128	31	17	–10	93	–8	89	8	76	**159**	0	65	0	59	2	46	–21	74
129	1	51	–7	73	–34	119	–9	78	**160**	21	37	22	33	19	38	16	66
130	7	50	–2	73	–3	75	–11	83	**161**	–15	72	5	44	–4	66	–23	124
131	28	19	13	46	32	20	9	52	**162**	9	57	14	43	15	38	17	37
132	16	33	9	49	30	22	0	67	**163**	16	54	–1	78	12	42	44	–18
133	14	62	–7	100	–44	127	–5	90	**164**	0	62	0	60	9	34	50	–34
134	–13	108	9	53	0	54	1	67	**165**	12	72	9	69	0	89	–22	127
135	–15	100	2	53	–5	61	–15	72									

Таблица 9-20 – Значения переменных m и n для ctxIdx от 166 до 226

ctxIdx	Секции I и SI		Значение cabac_init_idc						ctxIdx	Секции I и SI		Значение f cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n	m	n
166	24	0	11	28	4	45	4	39	**197**	26	–17	28	3	36	–28	28	–3
167	15	9	2	40	10	28	0	42	**198**	30	–25	28	4	38	–28	24	10
168	8	25	3	44	10	31	7	34	**199**	28	–20	32	0	38	–27	27	0
169	13	18	0	49	33	–11	11	29	**200**	33	–23	34	–1	34	–18	34	–14
170	15	9	0	46	52	–43	8	31	**201**	37	–27	30	6	35	–16	52	–44
171	13	19	2	44	18	15	6	37	**202**	33	–23	30	6	34	–14	39	–24
172	10	37	2	51	28	0	7	42	**203**	40	–28	32	9	32	–8	19	17
173	12	18	0	47	35	–22	3	40	**204**	38	–17	31	19	37	–6	31	25
174	6	29	4	39	38	–25	8	33	**205**	33	–11	26	27	35	0	36	29
175	20	33	2	62	34	0	13	43	**206**	40	–15	26	30	30	10	24	33
176	15	30	6	46	39	–18	13	36	**207**	41	–6	37	20	28	18	34	15
177	4	45	0	54	32	–12	4	47	**208**	38	1	28	34	26	25	30	20
178	1	58	3	54	102	–94	3	55	**209**	41	17	17	70	29	41	22	73
179	0	62	2	58	0	0	2	58	**210**	30	–6	1	67	0	75	20	34
180	7	61	4	63	56	–15	6	60	**211**	27	3	5	59	2	72	19	31
181	12	38	6	51	33	–4	8	44	**212**	26	22	9	67	8	77	27	44
182	11	45	6	57	29	10	11	44	**213**	37	–16	16	30	14	35	19	16
183	15	39	7	53	37	–5	14	42	**214**	35	–4	18	32	18	31	15	36
184	11	42	6	52	51	–29	7	48	**215**	38	–8	18	35	17	35	15	36
185	13	44	6	55	39	–9	4	56	**216**	38	–3	22	29	21	30	21	28
186	16	45	11	45	52	–34	4	52	**217**	37	3	24	31	17	45	25	21
187	12	41	14	36	69	–58	13	37	**218**	38	5	23	38	20	42	30	20
188	10	49	8	53	67	–63	9	49	**219**	42	0	18	43	18	45	31	12
189	30	34	–1	82	44	–5	19	58	**220**	35	16	20	41	27	26	27	16
190	18	42	7	55	32	7	10	48	**221**	39	22	11	63	16	54	24	42
191	10	55	–3	78	55	–29	12	45	**222**	14	48	9	59	7	66	0	93
192	17	51	15	46	32	1	0	69	**223**	27	37	9	64	16	56	14	56
193	17	46	22	31	0	0	20	33	**224**	21	60	–1	94	11	73	15	57
194	0	89	–1	84	27	36	8	63	**225**	12	68	–2	89	10	67	26	38
195	26	–19	25	7	33	–25	35	–18	**226**	2	97	–9	108	–10	116	–24	127
196	22	–17	30	–7	34	–30	33	–25									

Таблица 9-21 – Значения переменных m и n для ctxIdx от 227 до 275

ctxIdx	Секции I и SI		Значение cabac_init_idc						ctxIdx	Секции I и SI		Значение cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n	m	n
227	–3	71	–6	76	–23	112	–24	115	**252**	–12	73	–6	55	–16	72	–14	75
228	–6	42	–2	44	–15	71	–22	82	**253**	–8	76	0	58	–7	69	–10	79
229	–5	50	0	45	–7	61	–9	62	**254**	–7	80	0	64	–4	69	–9	83
230	–3	54	0	52	0	53	0	53	**255**	–9	88	–3	74	–5	74	–12	92
231	–2	62	–3	64	–5	66	0	59	**256**	–17	110	–10	90	–9	86	–18	108
232	0	58	–2	59	–11	77	–14	85	**257**	–11	97	0	70	2	66	–4	79
233	1	63	–4	70	–9	80	–13	89	**258**	–20	84	–4	29	–9	34	–22	69
234	–2	72	–4	75	–9	84	–13	94	**259**	–11	79	5	31	1	32	–16	75
235	–1	74	–8	82	–10	87	–11	92	**260**	–6	73	7	42	11	31	–2	58
236	–9	91	–17	102	–34	127	–29	127	**261**	–4	74	1	59	5	52	1	58
237	–5	67	–9	77	–21	101	–21	100	**262**	–13	86	–2	58	–2	55	–13	78
238	–5	27	3	24	–3	39	–14	57	**263**	–13	96	–3	72	–2	67	–9	83
239	–3	39	0	42	–5	53	–12	67	**264**	–11	97	–3	81	0	73	–4	81
240	–2	44	0	48	–7	61	–11	71	**265**	–19	117	–11	97	–8	89	–13	99
241	0	46	0	55	–11	75	–10	77	**266**	–8	78	0	58	3	52	–13	81
242	–16	64	–6	59	–15	77	–21	85	**267**	–5	33	8	5	7	4	–6	38
243	–8	68	–7	71	–17	91	–16	88	**268**	–4	48	10	14	10	8	–13	62
244	–10	78	–12	83	–25	107	–23	104	**269**	–2	53	14	18	17	8	–6	58
245	–6	77	–11	87	–25	111	–15	98	**270**	–3	62	13	27	16	19	–2	59
246	–10	86	–30	119	–28	122	–37	127	**271**	–13	71	2	40	3	37	–16	73
247	–12	92	1	58	–11	76	–10	82	**272**	–10	79	0	58	–1	61	–10	76
248	–15	55	–3	29	–10	44	–8	48	**273**	–12	86	–3	70	–5	73	–13	86
249	–10	60	–1	36	–10	52	–8	61	**274**	–13	90	–6	79	–1	70	–9	83
250	–6	62	1	38	–10	57	–8	66	**275**	–14	97	–8	85	–4	78	–10	87
251	–4	65	2	43	–9	58	–7	70									

Таблица 9-22 – Значения переменных m и n для ctxIdx от 277 до 337

ctxIdx	Секции I и SI		Значение cabac_init_idc						ctxIdx	Секции I и SI		Значение cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n	m	n
277	–6	93	–13	106	–21	126	–22	127	**308**	–16	96	–1	51	–16	77	–10	67
278	–6	84	–16	106	–23	124	–25	127	**309**	–7	88	7	49	–2	64	1	68
279	–8	79	–10	87	–20	110	–25	120	**310**	–8	85	8	52	2	61	0	77
280	0	66	–21	114	–26	126	–27	127	**311**	–7	85	9	41	–6	67	2	64
281	–1	71	–18	110	–25	124	–19	114	**312**	–9	85	6	47	–3	64	0	68
282	0	62	–14	98	–17	105	–23	117	**313**	–13	88	2	55	2	57	–5	78
283	–2	60	–22	110	–27	121	–25	118	**314**	4	66	13	41	–3	65	7	55
284	–2	59	–21	106	–27	117	–26	117	**315**	–3	77	10	44	–3	66	5	59
285	–5	75	–18	103	–17	102	–24	113	**316**	–3	76	6	50	0	62	2	65
286	–3	62	–21	107	–26	117	–28	118	**317**	–6	76	5	53	9	51	14	54
287	–4	58	–23	108	–27	116	–31	120	**318**	10	58	13	49	–1	66	15	44
288	–9	66	–26	112	–33	122	–37	124	**319**	–1	76	4	63	–2	71	5	60
289	–1	79	–10	96	–10	95	–10	94	**320**	–1	83	6	64	–2	75	2	70
290	0	71	–12	95	–14	100	–15	102	**321**	–7	99	–2	69	–1	70	–2	76
291	3	68	–5	91	–8	95	–10	99	**322**	–14	95	–2	59	–9	72	–18	86
292	10	44	–9	93	–17	111	–13	106	**323**	2	95	6	70	14	60	12	70
293	–7	62	–22	94	–28	114	–50	127	**324**	0	76	10	44	16	37	5	64
294	15	36	–5	86	–6	89	–5	92	**325**	–5	74	9	31	0	47	–12	70
295	14	40	9	67	–2	80	17	57	**326**	0	70	12	43	18	35	11	55
296	16	27	–4	80	–4	82	–5	86	**327**	–11	75	3	53	11	37	5	56
297	12	29	–10	85	–9	85	–13	94	**328**	1	68	14	34	12	41	0	69
298	1	44	–1	70	–8	81	–12	91	**329**	0	65	10	38	10	41	2	65
299	20	36	7	60	–1	72	–2	77	**330**	–14	73	–3	52	2	48	–6	74
300	18	32	9	58	5	64	0	71	**331**	3	62	13	40	12	41	5	54
301	5	42	5	61	1	67	–1	73	**332**	4	62	17	32	13	41	7	54
302	1	48	12	50	9	56	4	64	**333**	–1	68	7	44	0	59	–6	76
303	10	62	15	50	0	69	–7	81	**334**	–13	75	7	38	3	50	–11	82
304	17	46	18	49	1	69	5	64	**335**	11	55	13	50	19	40	–2	77
305	9	64	17	54	7	69	15	57	**336**	5	64	10	57	3	66	–2	77
306	–12	104	10	41	–7	69	1	67	**337**	12	70	26	43	18	50	25	42
307	–11	97	7	46	–6	67	0	68									

Таблица 9-23 – Значения переменных m и n для ctxIdx от 338 до 398

ctxIdx	Секции I и SI		Значение cabac_init_idc						ctxIdx	Секции I и SI		Значение cabac_init_idc					
			0		1		2					0		1		2	
	m	n	m	n	m	n	m	n		m	n	m	n	m	n	m	n
338	15	6	14	11	19	–6	17	–13	**369**	32	–26	31	–4	40	–37	37	–17
339	6	19	11	14	18	–6	16	–9	**370**	37	–30	27	6	38	–30	32	1
340	7	16	9	11	14	0	17	–12	**371**	44	–32	34	8	46	–33	34	15
341	12	14	18	11	26	–12	27	–21	**372**	34	–18	30	10	42	–30	29	15
342	18	13	21	9	31	–16	37	–30	**373**	34	–15	24	22	40	–24	24	25
343	13	11	23	–2	33	–25	41	–40	**374**	40	–15	33	19	49	–29	34	22
344	13	15	32	–15	33	–22	42	–41	**375**	33	–7	22	32	38	–12	31	16
345	15	16	32	–15	37	–28	48	–47	**376**	35	–5	26	31	40	–10	35	18
346	12	23	34	–21	39	–30	39	–32	**377**	33	0	21	41	38	–3	31	28
347	13	23	39	–23	42	–30	46	–40	**378**	38	2	26	44	46	–5	33	41
348	15	20	42	–33	47	–42	52	–51	**379**	33	13	23	47	31	20	36	28
349	14	26	41	–31	45	–36	46	–41	**380**	23	35	16	65	29	30	27	47
350	14	44	46	–28	49	–34	52	–39	**381**	13	58	14	71	25	44	21	62
351	17	40	38	–12	41	–17	43	–19	**382**	29	–3	8	60	12	48	18	31
352	17	47	21	29	32	9	32	11	**383**	26	0	6	63	11	49	19	26
353	24	17	45	–24	69	–71	61	–55	**384**	22	30	17	65	26	45	36	24
354	21	21	53	–45	63	–63	56	–46	**385**	31	–7	21	24	22	22	24	23
355	25	22	48	–26	66	–64	62	–50	**386**	35	–15	23	20	23	22	27	16
356	31	27	65	–43	77	–74	81	–67	**387**	34	–3	26	23	27	21	24	30
357	22	29	43	–19	54	–39	45	–20	**388**	34	3	27	32	33	20	31	29
358	19	35	39	–10	52	–35	35	–2	**389**	36	–1	28	23	26	28	22	41
359	14	50	30	9	41	–10	28	15	**390**	34	5	28	24	30	24	22	42
360	10	57	18	26	36	0	34	1	**391**	32	11	23	40	27	34	16	60
361	7	63	20	27	40	–1	39	1	**392**	35	5	24	32	18	42	15	52
362	–2	77	0	57	30	14	30	17	**393**	34	12	28	29	25	39	14	60
363	–4	82	–14	82	28	26	20	38	**394**	39	11	23	42	18	50	3	78
364	–3	94	–5	75	23	37	18	45	**395**	30	29	19	57	12	70	–16	123
365	9	69	–19	97	12	55	15	54	**396**	34	26	22	53	21	54	21	53
366	–12	109	–35	125	11	65	0	79	**397**	29	39	22	61	14	71	22	56
367	36	–35	27	0	37	–33	36	–16	**398**	19	66	11	86	11	83	25	61
368	36	–34	28	0	39	–36	37	–14									

9.3.1.2 Процесс инициализации устройства арифметического декодирования

Процесс активируют перед декодированием первого макроблока секции или после декодирования pcm_alignment_zero_bit и всех данных pcm_byte макроблока типа I_PCM.

Выходы этого процесса – регистры инициализированного устройства декодирования codIRange и codIOffset – оба с точностью 16-битового регистра.

Статус устройства арифметического декодирования представлен переменными codIRange и codIOffset. В процедуре инициализации процесса арифметического декодирования codIRange устанавливают равным 0x01FE, а codIOffset устанавливают равным возвратному значению от read_bits(9), которое интерпретируют как 9 битов бинарного представления целого без знака с наиболее значащим битом, записанным первым.

ПРИМЕЧАНИЕ. – В данной Рекомендации | Международном стандарте при описании устройства арифметического декодирования используют точность 16-битового регистра. Однако минимальная точность регистра для переменных codIRange и codIOffset составляет 9 битов.

9.3.2 Процесс бинаризации

Вход в этот процесс – запрос элемента синтаксиса.

Выход этого процесса – бинаризация элементов синтаксиса, maxBinIdxCtx, ctxIdxOffset и bypassFlag.

Таблица 9-24 определяет тип процесса бинаризации, maxBinIdxCtx и ctxIdxOffset, связанный с каждым элементом синтаксиса.

Спецификации унарного (U) процесса бинаризации, усеченного унарного (TU) процесса бинаризации, связанного унарного процесса бинаризации k-ого порядка с кодом Exp-Golomb (UEGk) и процесса бинаризации с фиксированной длиной (FL), даны в пп. 9.3.2.1–9.3.2.4, соответственно. Прочие бинаризации определены в пп. 9.3.2.5–9.3.2.7.

За исключением секций I, бинаризация элемента синтаксиса mb_type, как это определено в п. 9.3.2.5, состоит из строк бинов, заданных связью со строками битов префиксов и суффиксов. UEGk бинаризация, как это определено в п. 9.3.2.3, которую используют для элементов синтаксиса mvd_lX (X = 0, 1) и coeff_abs_level_minus1, и бинаризация coded_block_pattern также состоят из связи со строками битов префиксов и суффиксов. Для этих процессов бинаризации строки битов префиксов и суффиксов индексируют раздельно, используя переменную binIdx, как это определено далее в п. 9.3.3. Два множества строк битов префиксов и суффиксов называют частью бинаризации префикса и частью бинаризации суффикса, соответственно.

Связанный с каждой бинаризацией или частью бинаризации элемент синтаксиса является определенным значением переменной сдвига (ctxIdxOffset) индекса контекста и определенным значением переменной maxBinIdxCtx, заданной в таблице 9-24. Если два значения для каждой из этих переменных определены для одного элемента синтаксиса в таблице 9-24, значение в верхнем ряду относится к части префикса, а значение в нижнем ряду – к части суффикса бинаризации соответствующего элемента синтаксиса.

Использование процесса DecodeBypass и переменной bypassFlag происходит следующим образом:

– Если значение ctxIdxOffset не присвоено соответствующей бинаризации или части бинаризации в таблице 9-24, что обозначено как "na", все бины в строке битов соответствующей бинаризаци или части бинаризации префикса/суффикса, должны быть декодированы с помощью процесса DecodeBypass, как это определено в п. 9.3.3.2.3. В этом случае bypassFlag устанавливают равным 1, и bypassFlag используют для указания на то, что для анализа значения бина из потока битов должен быть применен процесс DecodeBypass.

– Иначе для каждого возможного значения binIdx, вплоть до определенного значения MaxBinIdxCtx, заданного в таблице 9-24, особое значение переменной ctxIdx определено далее в п. 9.3.3. В этом случае bypassFlag устанавливают равным 0.

Возможные значения индекса контекста ctxIdx находятся в диапазоне от 0 до 398 включительно. Значение, присвоенное ctxIdxOffset, определяет нижнее значение диапазона ctxIdx, присвоенное соответствующей бинаризации или части бинаризации элемента синтаксиса.

Значение ctxIdx = ctxIdxOffset = 276 присвоено элементу синтаксиса end_of_slice_flag и бину mb_type, который определяет тип макроблока I_PCM, как описано далее в п. 9.3.3.1. Для анализа значения соответствующего бина из потока битов, должен быть использован процесс арифметического декодирования для принятия решений перед окончанием (DecodeTerminate), как это определено в п. 9.3.3.2.4.

ПРИМЕЧАНИЕ. – Бины mb_type в I секциях и бины суффикса mb_type в SI секциях, которые соответствуют тем же значениям binIdx, имеют общее значение ctxIdx. Последний бин префикса mb_type и первый бин суффикса mb_type в секциях P, SP и B могут иметь общее значение ctxIdx.

Таблица 9-24 – Элементы синтаксиса и объединенные типы бинаризации, maxBinIdxCtx и ctxIdxOffset

Элемент синтаксиса	Тип бинаризации	maxBinIdxCtx	ctxIdxOffset
mb_type (только SI секции)	префикс и суффикс, как это определено в п. 9.3.2.5	префикс: 0 суффикс: 6	префикс: 0 суффикс: 3
mb_type (только I секции)	как это определено в п. 9.3.2.5	6	3
mb_skip_flag (только P, SP секции)	FL, cMax = 1	0	11
mb_type (только P, SP секции)	префикс и суффикс, как это определено в п. 9.3.2.5	префикс: 2 суффикс: 5	префикс: 14 суффикс: 17
sub_mb_type (только P, SP секции)	как это определено в п. 9.3.2.5	2	21
mb_skip_flag (только B секции)	FL, cMax = 1	0	24
mb_type (только B секции)	префикс и суффикс, как это определено в п. 9.3.2.5	префикс: 3 суффикс: 5	префикс: 27 суффикс: 32
sub_mb_type (только B секции)	как это определено в п. 9.3.2.5	3	36
mvd_l0[ ][ ][0], mvd_l1[ ][ ][0]	префикс и суффикс, как заданные UEG3 с signedValFlag = 1, uCoff = 9	префикс: 4 суффикс: na	префикс: 40 суффикс: na (использование DecodeBypass)
mvd_l0[ ][ ][1], mvd_l1[ ][ ][1]		префикс: 4 суффикс: na	префикс: 47 суффикс: na (использование DecodeBypass)
ref_idx_l0, ref_idx_l1	U	2	54
mb_qp_delta	как это определено в п. 9.3.2.7	2	60
intra_chroma_pred_mode	TU, cMax = 3	1	64
prev_intra4x4_pred_mode_flag	FL, cMax = 1	0	68
rem_intra4x4_pred_mode	FL, cMax = 7	0	69
mb_field_decoding_flag	FL, cMax = 1	0	70
coded_block_pattern	префикс и суффикс, как это определено в п. 9.3.2.6	префикс: 3 суффикс: 1	префикс: 73 суффикс: 77
coded_block_flag	FL, cMax = 1	0	85
significant_coeff_flag (кадр, кодированный только блоками)	FL, cMax = 1	0	105
last_significant_coeff_flag (кадр, кодированный только блоками)	FL, cMax = 1	0	166
coeff_abs_level_minus1	префикс и суффикс, как заданные UEG0 с signedValFlag = 0, uCoff = 14	префикс: 1 суффикс: na	префикс: 227 суффикс: na, (использование DecodeBypass)
coeff_sign_flag	FL, cMax = 1	0	na, (использование DecodeBypass)
end_of_slice_flag	FL, cMax = 1	0	276
significant_coeff_flag (поле, кодированное только блоками)	FL, cMax = 1	0	277
last_significant_coeff_flag (поле, кодированное только блоками)	FL, cMax = 1	0	338

9.3.2.1 Унарный (U) процесс бинаризации

Вход в этот процесс – запрос U бинаризации элемента синтаксиса.

Выход этого процесса – U бинаризация элемента синтаксиса.

Строка бинов элемента синтаксиса, принимающего значение (целое без знака) synElVal, – это строка битов длиной synElVal + 1, индексированная BinIdx. Бины binIdx, меньшие synElVal, равны 1. Бин со значением binIdx, равным synElVal, равен 0.

Таблица 9-25 иллюстрирует строку бинов унарной бинаризации элемента синтаксиса.

Таблица 9-25 – Строка бинов унарной бинаризации (информативное)

Значение элемента синтаксиса	Строка бинов					
0	0					
1	1	0				
2	1	1	0			
3	1	1	1	0		
4	1	1	1	1	0	
5	1	1	1	1	1	0
…						
binIdx	0	1	2	3	4	5

9.3.2.2 Усеченный унарный (TU) процесс бинаризации

Вход в этот процесс – запрос TU бинаризации элемента синтаксиса и cMax.

Выход этого процесса – TU бинаризация элемента синтаксиса.

Для значения элемента синтаксиса (целого без знака), меньшего cMax, активируют U процесс бинаризации, как это определено в п. 9.3.2.1. Для значения элемента синтаксиса равного cMax строка бинов – это строка битов длиной cMax со всеми бинами, равными 1.

ПРИМЕЧАНИЕ. – TU бинаризацию всегда активируют со значением cMax, равным наибольшему возможному значению элемента синтаксиса, подлежащего декодированию.

9.3.2.3 Соединенный процесс бинаризации: унарный и k-ого порядка с экспоненциальным кодом Exp-Golomb (UEGk)

Вход в этот процесс – запрос UEGk бинаризации для элемента синтаксиса, signedValFlag и uCoff.

Выход этого процесса – UEGk бинаризация элемента синтаксиса.

Строка бинов UEGk – соединение префикса строки битов и суффикса строки битов. Префикс бинаризации описывают, активируя TU процесс бинаризации для части префикса Min(uCoff, Abs(synElVal)) значения элемента синтаксиса synElVal, как это определено в п. 9.3.2.2, с cMax = uCoff, где uCoff > 0.

Строку бинов UEGk находят следующим образом:

– Если одно из следующих условий – истина, то строка бинов элемента синтаксиса со значением synElVal состоит только из префикса строки битов,

  – signedValFlag равно 0, а префикс строки битов не равен строке битов длиной uCoff со всеми битами, равными 1.

  – signedValFlag равно 1, а префикс строки битов равен строке битов, которая состоит из единственного бита со значением, равным 0.

– Иначе строку бинов части суффикса UEGk со значением элемента синтаксиса synElVal описывают процессом, эквивалентным следующему псевдокоду:

```
if( Abs( synElVal ) >= uCoff ) {
    sufS = Abs( synElVal ) − uCoff
```

```
        stopLoop = 0
        do {
            if( sufS >= ( 1 << k ) ) {
                put( 1 )
                sufS = sufS − ( 1<<k )
                k++
            } else {
                put( 0 )
                while( k− − )
                    put( ( sufS >> k ) & 0x01 )
                stopLoop = 1
            }
        } while( !stopLoop )
    }
    if( signedValFlag && synElVal ! = 0)
        if( synElVal > 0 )
            put( 0 )
        else
            put( 1 ).
```

ПРИМЕЧАНИЕ. – Спецификация для кода Exp-Golomb (EGk) k-ого порядка использует 1 и 0 в обратном значении для унарной части кода Exp-Golomb 0-го порядка, как это определено в п. 9.1.

9.3.2.4 Процесс бинаризации фиксированной длины (FL)

Вход в этот процесс – запрос FL бинаризации элемента синтаксиса и cMax.

Выход этого процесса – FL бинаризация элемента синтаксиса.

FL бинаризацию создают, используя fixedLength-bit, – строку бинов с целыми без знака значения элемента синтаксиса, где fixedLength = Ceil(Log2(cMax + 1)). Индексацию бинов при FL бинаризации производят таким образом, что binIdx = 0 относится к наименее значащему биту с возрастающим значением binIdx в направлении к наиболее значащему биту.

9.3.2.5 Процесс бинаризации типов макроблока и субмакроблока

Вход в этот процесс – запрос бинаризации элементов синтаксиса mb_type или sub_mb_type.

Выход этого процесса – бинаризация элемента синтаксиса.

Схема бинаризации для декодирования типа макроблока в I секции описана в таблице 9-26.

Для типов макроблоков в SI секции, бинаризация состоит из строки бинов, определенных как соединение префикса и суффикса строки битов следующим образом.

Префикс строки битов состоит из единственного бита, который описывают как b_0 = ((mb_type = = SI) ? 0 : 1). При значении элемента синтаксиса для каждого b_0, равного 0, строка бинов состоит только из префикса строки битов. При значении элемента синтаксиса для каждого b_0, равного 1, бинаризация задана соединением префикса b_0 и суффикса строки битов, как это определено в таблице 9-26 для типа макроблока в I секции, индексированной вычитанием 1 из значения mb_type в SI секции.

Таблица 9-26 – Бинаризация типов макроблоков в I секции

Значение (имя) mb_type	Строка бинов						
0 (I_4x4)	0						
1 (I_16x16_0_0_0)	1	0	0	0	0	0	
2 (I_16x16_1_0_0)	1	0	0	0	0	1	
3 (I_16x16_2_0_0)	1	0	0	0	1	0	
4 (I_16x16_3_0_0)	1	0	0	0	1	1	
5 (I_16x16_0_1_0)	1	0	0	1	0	0	0
6 (I_16x16_1_1_0)	1	0	0	1	0	0	1
7 (I_16x16_2_1_0)	1	0	0	1	0	1	0
8 (I_16x16_3_1_0)	1	0	0	1	0	1	1
9 (I_16x16_0_2_0)	1	0	0	1	1	0	0
10 (I_16x16_1_2_0)	1	0	0	1	1	0	1
11 (I_16x16_2_2_0)	1	0	0	1	1	1	0
12 (I_16x16_3_2_0)	1	0	0	1	1	1	1
13 (I_16x16_0_0_1)	1	0	1	0	0	0	
14 (I_16x16_1_0_1)	1	0	1	0	0	1	
15 (I_16x16_2_0_1)	1	0	1	0	1	0	
16 (I_16x16_3_0_1)	1	0	1	0	1	1	
17 (I_16x16_0_1_1)	1	0	1	1	0	0	0
18 (I_16x16_1_1_1)	1	0	1	1	0	0	1
19 (I_16x16_2_1_1)	1	0	1	1	0	1	0
20 (I_16x16_3_1_1)	1	0	1	1	0	1	1
21 (I_16x16_0_2_1)	1	0	1	1	1	0	0
22 (I_16x16_1_2_1)	1	0	1	1	1	0	1
23 (I_16x16_2_2_1)	1	0	1	1	1	1	0
24 (I_16x16_3_2_1)	1	0	1	1	1	1	1
25 (I_PCM)	1	1					
binIdx	0	1	2	3	4	5	6

Схемы бинаризации типов P макроблоков в P и SP секциях и B макроблоков в B секции показаны в таблице 9-27.

Строка бинов типов I макроблоков в P и SP секциях, соответствующих значениям mb_type от 5 до 30, состоит из соединения префикса, который представлен единственным битом со значением, равным 1, как это определено в таблице 9-27, и суффикса, как это определено в таблице 9-26, индексированной вычитанием 5 из значения mb_type.

Значение mb_type, равное 4 (P_8x8ref0), не разрешено.

Для типов I макроблоков в B секциях (значения mb_type от 23 до 48) бинаризация состоит из строки бинов, определенных как соединение префикса строки битов, как это указано в таблице 9-27, и суффикса строки битов, как это указано в таблице 9-26, индексированной вычитанием 23 из значения mb_type.

Таблица 9-27 – Бинаризация типов макроблоков в P, SP и B секциях

Тип секции	**Значение (имя) mb_type**	**Строка бинов**						
P, SP секции	0 (P_L0_16x16)	0	0	0				
	1 (P_L0_L0_16x8)	0	1	1				
	2 (P_L0_L0_8x16)	0	1	0				
	3 (P_8x8)	0	0	1				
	4 (P_8x8ref0)	na						
	от 5 до 30 (Intra, только префикс)	1						
B секция	0 (B_Direct_16x16)	0						
	1 (B_L0_16x16)	1	0	0				
	2 (B_L1_16x16)	1	0	1				
	3 (B_Bi_16x16)	1	1	0	0	0	0	
	4 (B_L0_L0_16x8)	1	1	0	0	0	1	
	5 (B_L0_L0_8x16)	1	1	0	0	1	0	
	6 (B_L1_L1_16x8)	1	1	0	0	1	1	
	7 (B_L1_L1_8x16)	1	1	0	1	0	0	
	8 (B_L0_L1_16x8)	1	1	0	1	0	1	
	9 (B_L0_L1_8x16)	1	1	0	1	1	0	
	10 (B_L1_L0_16x8)	1	1	0	1	1	1	
	11 (B_L1_L0_8x16)	1	1	1	1	1	0	
	12 (B_L0_Bi_16x8)	1	1	1	0	0	0	0
	13 (B_L0_Bi_8x16)	1	1	1	0	0	0	1
	14 (B_L1_Bi_16x8)	1	1	1	0	0	1	0
	15 (B_L1_Bi_8x16)	1	1	1	0	0	1	1
	16 (B_Bi_L0_16x8)	1	1	1	0	1	0	0
	17 (B_Bi_L0_8x16)	1	1	1	0	1	0	1
	18 (B_Bi_L1_16x8)	1	1	1	0	1	1	0
	19 (B_Bi_L1_8x16)	1	1	1	0	1	1	1
	20 (B_Bi_Bi_16x8)	1	1	1	1	0	0	0
	21 (B_Bi_Bi_8x16)	1	1	1	1	0	0	1
	22 (B_8x8)	1	1	1	1	1	1	
	от 23 до 48 (Intra, только префикс)	1	1	1	1	0	1	
binIdx		0	1	2	3	4	5	6

Для секций P, SP и B спецификация бинаризации sub_mb_type задана в таблице 9-28.

Таблица 9-28 – Бинаризация типов субмакроблоков в P, SP и B секциях

Тип секции	Значение (имя) sub_mb_type	Строка бинов					
P, SP секции	0 (P_L0_8x8)	1					
	1 (P_L0_8x4)	0	0				
	2 (P_L0_4x8)	0	1	1			
	3 (P_L0_4x4)	0	1	0			
B секция	0 (B_Direct_8x8)	0					
	1 (B_L0_8x8)	1	0	0			
	2 (B_L1_8x8)	1	0	1			
	3 (B_Bi_8x8)	1	1	0	0	0	
	4 (B_L0_8x4)	1	1	0	0	1	
	5 (B_L0_4x8)	1	1	0	1	0	
	6 (B_L1_8x4)	1	1	0	1	1	
	7 (B_L1_4x8)	1	1	1	0	0	0
	8 (B_Bi_8x4)	1	1	1	0	0	1
	9 (B_Bi_4x8)	1	1	1	0	1	0
	10 (B_L0_4x4)	1	1	1	0	1	1
	11 (B_L1_4x4)	1	1	1	1	0	
	12 (B_Bi_4x4)	1	1	1	1	1	
binIdx		0	1	2	3	4	5

9.3.2.6 Процесс бинаризации для кодированного образца блока

Вход в этот процесс – запрос бинаризации элемента синтаксиса coded_block_pattern.

Выход этого процесса – бинаризация элемента синтаксиса.

Бинаризация coded_block_pattern состоит из соединения части префикса и части суффикса. Часть префикса бинаризации задана длиной FL бинаризации CodedBlockPatternLuma с cMax = 15. Часть суффикса состоит из TU бинаризации CodedBlockPatternChroma с cMax = 2. Взаимодействие между значением элемента синтаксиса coded_block_pattern и значениями CodedBlockPatternLuma и CodedBlockPatternChroma заданы, как это определено в п. 7.4.5.

9.3.2.7 Процесс бинаризации для mb_qp_delta

Вход в этот процесс – запрос бинаризации элемента синтаксиса mb_qp_delta.

Выход этого процесса – бинаризация элемента синтаксиса.

Строку бинов mb_qp_delta находят U бинаризацией отображенного значения элемента синтаксиса mb_qp_delta, где правило присвоения между значением mb_qp_delta со знаком и его отображенным значением задано в таблице 9-3.

9.3.3 Процесс декодирования потока

Вход в этот процесс – бинаризация запрошенного элемента синтаксиса, maxBinIdxCtx, bypassFlag и ctxIdxOffset, как это определено в п. 9.3.2.

Выход этого процесса – значение данного элемента синтаксиса.

Процесс определяет, как анализируют каждый бит строки битов для каждого элемента синтаксиса.

После анализа каждого бита результирующую строку битов сравнивают со всеми строками бинов процесса бинаризации элемента синтаксиса и применяют следующие условия:

– Если строка битов равна одной строке бинов, то соответствующее значение элемента синтаксиса – это выход.

– Иначе (строка битов не равна одной строке бинов), анализируют следующий бит.

При анализе каждого бина переменная binIdx возрастает на 1, начиная с binIdx, которое устанавливают на 0 для первого бина.

Если бинаризация соответствующего элемента синтаксиса состоит из бинаризации частей префикса и суффикса, переменную binIdx устанавливают равной 0 для первого бина каждой части строки бинов (части префикса или части суффикса). В этом случае, после анализа префикса строки битов, анализ суффикса строки битов (связанный с процессами бинаризации из пп. 9.3.2.3 и 9.3.2.5) активируют в зависимости от результирующего префикса строки битов, как это указано в пп. 9.3.2.3 и 9.3.2.5. Отметим, что для бинаризации элемента синтаксиса coded_block_pattern, суффикс строки битов представляют независимо от префикса строки битов длиной 4, как это определено в п. 9.3.2.6.

В зависимости от переменной bypassFlag применяют следующее:

– Если bypassFlag равно 1, должен быть выполнен процесс обхода декодирования, как это определено в п. 9.3.3.2.3, для анализа значений бинов из потока битов.

– Иначе (bypassFlag равно 0), анализ каждого бина производят по порядку с помощью следующих двух шагов:

  1. Находят заданные значения binIdx, maxBinIdxCtx и ctxIdxOffset, ctxIdx, как это определено в п. 9.3.3.1.

  2. Зная ctxIdx, декодируют значение бина из потока битов, как это определено в п. 9.3.3.2.

9.3.3.1 Процесс отыскания ctxIdx

Входы в этот процесс – binIdx, maxBinIdxCtx и ctxIdxOffset.

Выход этого процесса – ctxIdx.

Таблица 9-29 показывает присвоение приращений ctxIdx (ctxIdxInc) значениям binIdx для всех ctxIdxOffset, за исключением тех значений, которые относятся к элементам синтаксиса coded_block_flag, significant_coeff_flag, last_significant_coeff_flag и coeff_abs_level_minus1.

Значение ctxIdx, которое следует использовать со специальным значением binIdx, описано первым определением ctxIdxOffset, связанным с заданной строкой бинов или ее частью. Значение ctxIdx определяют следующим образом:

– Если ctxIdxOffset перечислено в таблице 9-29, то значение ctxIdx для binIdx представляет сумму ctxIdxOffset и ctxIdxInc, которую находят в таблице 9-29. Если в таблице 9-29 перечислено более одного значения binIdx, то процесс присвоения ctxIdxInc для binIdx описан далее в пунктах, указанных в скобках в соответствующей графе таблицы.

– Иначе (ctxIdxOffset не перечислено в таблице 9-29), ctxIdx описывают суммой следующих членов: ctxIdxOffset и ctxIdxBlockCatOffset(ctxBlockCat), как это определено в таблице 9-30, и ctxIdxInc(ctxBlockCat). В п. 9.3.3.1.3 указано, какое значение ctxBlockCat используют. В п. 9.3.3.1.1.9 определено присвоение ctxIdxInc(ctxBlockCat) для coded_block_flag, а в п. 9.3.3.1.3 определено присвоение ctxIdxInc(ctxBlockCat) для significant_coeff_flag, last_significant_coeff_flag и coeff_abs_level_minus1.

Все бины с binIdx больше, чем maxBinIdxCtx, анализируют, используя значение ctxIdx, присвоенное maxBinIdxCtx.

Все входы в таблице 9-29, отмеченные как "na", соответствуют значениям binIdx, которые не встречаются для соответствующего значения ctxIdxOffset.

Значение ctxIdx = 276 присваивают значению binIdx mb_type, указывающему на режим I_PCM. Перед завершением для анализа значений соответствующих бинов из потока битов должен применяться процесс арифметического декодирования решений, как это определено в п. 9.3.3.2.4.

Таблица 9-29 – Присвоение ctxIdxInc значению binIdx для всех значений ctxIdxOffset, за исключением тех, которые относятся к элементам синтаксиса coded_block_flag, significant_coeff_flag, last_significant_coeff_flag и coeff_abs_level_minus1

ctxIdxOffset	binIdx						
	0	**1**	**2**	**3**	**4**	**5**	**>= 6**
0	0,1,2 (п. 9.3.3.1.1.3)	na	na	na	na	na	na
3	0,1,2 (п. 9.3.3.1.1.3)	ctxIdx=276	3	4	5,6 (п. 9.3.3.1.2)	6,7 (п. 9.3.3.1.2)	7
11	0,1,2 (п. 9.3.3.1.1.1)	na	na	na	na	na	na
14	0	1	2,3 (п. 9.3.3.1.2)	na	na	na	na
17	0	ctxIdx=276	1	2	2,3 (п. 9.3.3.1.2)	3	3
21	0	1	2	na	na	na	na
24	0,1,2 (п. 9.3.3.1.1.1)	na	na	na	na	na	na
27	0,1,2 (п. 9.3.3.1.1.3)	3	4,5 (п. 9.3.3.1.2)	5	5	5	5
32	0	ctxIdx=276	1	2	2,3 (п. 9.3.3.1.2)	3	3
36	0	1	2,3 (п. 9.3.3.1.2)	3	3	3	na
40	0,1,2 (п. 9.3.3.1.1.7)	3	4	5	6	6	6
47	0,1,2 (п. 9.3.3.1.1.7)	3	4	5	6	6	6
54	0,1,2,3 (п. 9.3.3.1.1.6)	4	5	5	5	5	5
60	0,1 (п. 9.3.3.1.1.5)	2	3	3	3	3	3
64	0,1,2 (п. 9.3.3.1.1.8)	3	3	na	na	na	na
68	0	na	na	na	na	na	na
69	0	0	0	na	na	na	na
70	0,1,2 (п. 9.3.3.1.1.2)	na	na	na	na	na	na
73	0,1,2,3 (п. 9.3.3.1.1.4)	0,1,2,3 (п. 9.3.3.1.1.4)	0,1,2,3 (п. 9.3.3.1.1.4)	0,1,2,3 (п. 9.3.3.1.1.4)	na	na	na
77	0,1,2,3 (п. 9.3.3.1.1.4)	4,5,6,7 (п. 9.3.3.1.1.4)	na	na	na	na	na
276	0	na	na	na	na	na	na

Таблица 9-30 показывает значения ctxIdxBlockCatOffset в зависимости от ctxBlockCat для элементов синтаксиса coded_block_flag, significant_coeff_flag, last_significant_coeff_flag, и coeff_abs_level_minus1. Спецификация ctxBlockCat дана в таблице 9-32.

Таблица 9-30 – Присвоение ctxIdxBlockCatOffset значению ctxBlockCat для элементов синтаксиса coded_block_flag, significant_coeff_flag, last_significant_coeff_flag и coeff_abs_level_minus1

Элемент синтаксиса	ctxBlockCat (как это определено в таблице 9-32)				
	0	1	2	3	4
coded_block_flag	0	4	8	12	16
significant_coeff_flag	0	15	29	44	47
last_significant_coeff_flag	0	15	29	44	47
coeff_abs_level_minus1	0	10	20	30	39

9.3.3.1.1 Процесс присвоения ctxIdxInc с использованием ближайших элементов синтаксиса

Пункт 9.3.3.1.1.1 определяет процесс отыскания ctxIdxInc для элемента синтаксиса mb_skip_flag.

Пункт 9.3.3.1.1.2 определяет процесс отыскания ctxIdxInc для элемента синтаксиса mb_field_decoding_flag.

Пункт 9.3.3.1.1.3 определяет процесс отыскания ctxIdxInc для элемента синтаксиса mb_type.

Пункт 9.3.3.1.1.4 определяет процесс отыскания ctxIdxInc для элемента синтаксиса coded_block_pattern.

Пункт 9.3.3.1.1.5 определяет процесс отыскания ctxIdxInc для элемента синтаксиса mb_qp_delta.

Пункт 9.3.3.1.1.6 определяет процесс отыскания ctxIdxInc для элементов синтаксиса ref_idx_l0 и ref_idx_l1.

Пункт 9.3.3.1.1.7 определяет процесс отыскания ctxIdxInc для элементов синтаксиса mvd_l0 и mvd_l1.

Пункт 9.3.3.1.1.8 определяет процесс отыскания ctxIdxInc для элемента синтаксиса intra_chroma_pred_mode.

Пункт 9.3.3.1.1.9 определяет процесс отыскания ctxIdxInc для элемента синтаксиса coded_block_flag.

9.3.3.1.1.1 Процесс отыскания ctxIdxInc для элемента синтаксиса mb_skip_flag

Выход этого процесса – ctxIdxInc.

Если MbaffFrameFlag равно 1, а mb_field_decoding_flag еще не декодировано для текущей пары макроблоков с адресом верхнего макроблока address 2 * (CurrMbAddr / 2), то для элемента синтаксиса mb_field_decoding_flag должно применяться установленное правило, как это определено в п. 7.4.4.

Активируют процесс нахождения смежных макроблоков, определенных в п. 6.4.7.1, и присваивают выходу значения mbAddrA и mbAddrB.

Предположим, что переменную condTermFlagN (с N, принимающим значения A или B) получили следующим образом:

– Если mbAddrN недоступно или mb_skip_flag макроблока mbAddrN равно 1, condTermFlagN устанавливают равным 0.

– Иначе (mbAddrN доступно, а mb_skip_flag макроблока mbAddrN равно 0), condTermFlagN устанавливают равным 1.

Переменную ctxIdxInc находят как

$$ctxIdxInc = condTermFlagA + condTermFlagB. \quad (9\text{-}1)$$

9.3.3.1.1.2 Процесс отыскания ctxIdxInc элемента синтаксиса mb_field_decoding_flag

Выход этого процесса – ctxIdxInc.

Активируют процесс нахождения адресов смежных макроблоков и их доступности в кадрах MBAFF, как это определено в п. 6.4.6, и присваивают выходу значения mbAddrA и mbAddrB.

Если оба макроблока, mbAddrN и mbAddrN + 1, имеют mb_type, равное P_Skip или B_Skip, то для элемента синтаксиса mb_field_decoding_flag должно применяться установленное правило, как это определено в п. 7.4.4.

Предположим, что переменную condTermFlagN (с N, принимающим значения A или B) получили следующим образом:

– Если любое из следующих условий – истина, то condTermFlagN устанавливают равным 0:

– mbAddrN недоступно;

– макроблок mbAddrN – это макроблок кадра.

– Иначе condTermFlagN устанавливают равным 1.

Переменную ctxIdxInc находят как

ctxIdxInc = condTermFlagA + condTermFlagB. (9-2)

9.3.3.1.1.3 Процесс отыскания ctxIdxInc элемента синтаксиса mb_type

Вход в этот процесс – ctxIdxOffset.

Выход этого процесса – ctxIdxInc.

Активируют процесс нахождения смежных макроблоков, определенных в п. 6.4.7.1, и присваивают выходу значения mbAddrA и mbAddrB.

Предположим, что переменную condTermFlagN (с N, принимающим значения A или B) получили следующим образом:

– Если любое из следующих условий – истина, то condTermFlagN устанавливают равным 0:

  – mbAddrN недоступно;

  – ctxIdxOffset равно 0, а mb_type макроблока mbAddrN равно SI;

  – ctxIdxOffset равно 3, а mb_type макроблока mbAddrN равно I_4x4;

  – ctxIdxOffset равно 27, а макроблок mbAddrN пропущен;

  – ctxIdxOffset равно 27, а mb_type макроблока mbAddrN равно B_Direct_16x16.

– Иначе condTermFlagN устанавливают равным 1.

Переменную ctxIdxInc находят как

ctxIdxInc = condTermFlagA + condTermFlagB. (9-3)

9.3.3.1.1.4 Процесс отыскания ctxIdxInc элемента синтаксиса coded_block_pattern

Входы в этот процесс – ctxIdxOffset и binIdx.

Выход этого процесса – ctxIdxInc.

В зависимости от значения переменной ctxIdxOffset применяют следующее:

– Если ctxIdxOffset равно 73, то:

  – Активируют процесс нахождения смежных блоков яркости 8x8, определенных в п. 6.4.7.2, с luma8x8BlkIdx = binIdx в качестве входа и присваивают выходу значения mbAddrA, mbAddrB, luma8x8BlkIdxA и luma8x8BlkIdxB.

  – Предположим, что переменную condTermFlagN (с N, принимающим значения A или B) получили следующим образом:

    – Если любое из следующих условий – истина, то condTermFlagN устанавливают равным 0:

      – mbAddrN недоступно;

      – mb_type макроблока mbAddrN равно I_PCM;

      – макроблок mbAddrN не пропускают, и ((CodedBlockPatternLuma >> luma8x8BlkIdxN) & 1) не равно 0 для макроблока mbAddrN.

    – Иначе condTermFlagN устанавливают равным 1.

  – Переменную ctxIdxInc находят как

    ctxIdxInc = condTermFlagA + 2 * condTermFlagB. (9-4)

– Иначе (ctxIdxOffset равно 77), применяют следующее:

  – Активируют процесс нахождения смежных макроблоков, определенных в п. 6.4.7.1, и присваиваются выходу значения mbAddrA и mbAddrB.

– Предположим, что переменную condTermFlagN (с N, принимающим значения A или B) получили следующим образом:

  – Если mbAddrN доступно, а mb_type макроблока mbAddrN равно I_PCM, то condTermFlagN устанавливают равным 1.

  – Иначе, если любое из следующих условий – истина, то condTermFlagN устанавливают равным 0:

    – mbAddrN недоступно или макроблок mbAddrN пропущен;

    – binIdx равно 0 и CodedBlockPatternChroma макроблока mbAddrN равно 0;

    – binIdx равно 1 и CodedBlockPatternChroma макроблока mbAddrN не равно 2.

  – Иначе condTermFlagN устанавливают равным 1.

– Переменную ctxIdxInc находят как

  ctxIdxInc = condTermFlagA + 2 * condTermFlagB + ((binIdx = = 1) ? 4 : 0). (9-5)

ПРИМЕЧАНИЕ. – Если для макроблока используют режим предсказания Intra_16x16, то значения CodedBlockPatternLuma и CodedBlockPatternChroma макроблока находят из mb_type, как это определено в таблице 7-8.

9.3.3.1.1.5 Процесс отыскания ctxIdxInc элемента синтаксиса mb_qp_delta

Выход этого процесса – ctxIdxInc.

Предположим, что prevMbAddr – адрес макроблока из той группы макроблоков, которая предшествует текущему макроблоку в порядке декодирования. Если текущий макроблок – первый макроблок секции, то prevMbAddr помечают как недоступный.

Предположим, что переменную ctxIdxInc получили следующим образом:

– Если любое из перечисленных условий – истина, то ctxIdxInc устанавливают равным 0:

  – prevMbAddr недоступно или макроблок prevMbAddr пропущен;

  – mb_type макроблока prevMbAddr равно I_PCM;

  – макроблок prevMbAddr не кодирован в режиме предсказания Intra_16x16, а оба значения CodedBlockPatternLuma и CodedBlockPatternChroma макроблока prevMbAddr равны 0;

  – mb_qp_delta макроблока prevMbAddr равно 0.

– Иначе ctxIdxInc устанавливают равным 1.

9.3.3.1.1.6 Процесс отыскания ctxIdxInc элемента синтаксиса ref_idx_l0 и ref_idx_l1

Входы в этот процесс – mbPartIdx и суффикс списка контрольного изображения lX, где X = 0 или 1.

Выход этого процесса – ctxIdxInc.

Положим, что currSubMbType равно sub_mb_type[mbPartIdx].

Процесс нахождения смежных разделенных частей, определенных в п. 6.4.7.5, активируют с mbPartIdx, currSubMbType и subMbPartIdx = 0 в качестве входа, а выходу присваивают значения mbAddrA\mbPartIdxA и mbAddrB\mbPartIdxB.

Со значением ref_idx_lX[mbPartIdxN] (при N, равным A или B), которая определяет элемент синтаксиса макроблока mbAddrN, предположим, что переменную refIdxZeroFlagN получили следующим образом:

– Если MbaffFrameFlag равно 1, текущий макроблок – макроблок кадра, а макроблок mbAddrN – макроблок поля, то

  refIdxZeroFlagN = ((ref_idx_lX[mbPartIdxN] > 1) ? 0 : 1). (9-6)

– Иначе

  refIdxZeroFlagN = ((ref_idx_lX[mbPartIdxN] > 0) ? 0 : 1). (9-7)

Предположим, что переменная predModeEqualFlag определена следующим образом:

– Если макроблок mbAddrN имеет mb_type, равное P_8x8 или B_8x8, то применяют следующее:

– Если SubMbPredMode(sub_mb_type[mbPartIdxN]) не равно Pred_LX и не равно BiPred, то predModeEqualFlag устанавливают равным 0, где sub_mb_type определяет элемент синтаксиса макроблока mbAddrN.

– Иначе predModeEqualFlag устанавливают равным 1.

– Иначе применяют следующее:

– Если MbPartPredMode(mb_type, mbPartIdxN) не равно Pred_LX и не равно BiPred, то predModeEqualFlag устанавливают равным 0, где mb_type определяет элемент синтаксиса макроблока mbAddrN.

– Иначе predModeEqualFlag устанавливают равным 1.

Предположим, что переменную condTermFlagN (с N принимающим значения A или B) получили следующим образом:

– Если любое из перечисленных условий – истина, то condTermFlagN устанавливают равным 0:

– mbAddrN недоступно;

– макроблок mbAddrN имеет mb_type, равное P_Skip or B_Skip;

– макроблок mbAddrN кодирован в режиме Intra предсказания;

– predModeEqualFlag равно 0;

– refIdxZeroFlagN равно 1.

– Иначе condTermFlagN устанавливают равным 1.

Переменную ctxIdxInc находят как

ctxIdxInc = condTermFlagA + 2 * condTermFlagB. (9-8)

9.3.3.1.1.7 Процесс отыскания ctxIdxInc элементов синтаксиса mvd_l0 и mvd_l1

Входы в этот процесс – mbPartIdx, subMbPartIdx, суффикс списка контрольного изображения lX и ctxIdxOffset.

Выход этого процесса – ctxIdxInc.

Положим, что currSubMbType установлено равным sub_mb_type[mbPartIdx].

Процесс нахождения смежных разделенных частей, определенных в п. 6.4.7.5, активируют с mbPartIdx, currSubMbType и subMbPartIdx в качестве входа, а выходу присваивают значения mbAddrA\mbPartIdxA\subMbPartIdxA и mbAddrB\mbPartIdxB\subMbPartIdxB.

Предположим, что переменную compIdx получили следующим образом:

– Если ctxIdxOffset равно 40, то compIdx устанавливают равным 0.

– Иначе (ctxIdxOffset равно 47), compIdx устанавливают равным 1.

Предположим, что переменная predModeEqualFlag определена следующим образом:

– Если макроблок mbAddrN имеет mb_type, равное P_8x8 или B_8x8, то применяют следующее:

– Если SubMbPredMode(sub_mb_type[mbPartIdxN]) не равно Pred_LX и не равно BiPred, то predModeEqualFlag устанавливают равным 0, где sub_mb_type определяет элемент синтаксиса макроблока mbAddrN.

– Иначе predModeEqualFlag устанавливают равным 1.

– Иначе применяют следующее:

– Если MbPartPredMode(mb_type, mbPartIdxN) не равно Pred_LX и не равно BiPred, то predModeEqualFlag устанавливают равным 0, где mb_type определяет элемент синтаксиса макроблока mbAddrN.

– Иначе predModeEqualFlag устанавливают равным 1.

Предположим, что переменную absMvdCompN (с N, принимающим значения A или B) получили следующим образом:

– Если любое из следующих условий – истина, то absMvdCompN устанавливают равным 0:

– mbAddrN недоступно;

- макроблок mbAddrN имеет mb_type, равное P_Skip или B_Skip;
- макроблок mbAddrN кодирован в режиме Intra предсказания;
- predModeEqualFlag равно 0.

– Иначе применяют следующее:

- если compIdx равно 1, MbaffFrameFlag равно 1, то текущий макроблок – макроблок кадра, а макроблок mbAddrN – макроблок поля,

  absMvdCompN = Abs(mvd_lX[mbPartIdxN][subMbPartIdxN][compIdx]) * 2. (9-9)

- Иначе, если compIdx равно 1, MbaffFrameFlag равно 1, то текущий макроблок – макроблок поля, а макроблок mbAddrN – макроблок кадра,

  absMvdCompN = Abs(mvd_lX[mbPartIdxN][subMbPartIdxN][compIdx]) / 2. (9-10)

- Иначе

  absMvdCompN = Abs(mvd_lX[mbPartIdxN][subMbPartIdxN][compIdx]). (9-11)

Переменную ctxIdxInc находят следующим образом:

– Если (absMvdCompA + absMvdCompB) меньше чем 3, ctxIdxInc устанавливают равным 0.

– Иначе, если (absMvdCompA + absMvdCompB) больше чем 32, ctxIdxInc устанавливают равным 2.

– Иначе ((absMvdCompA + absMvdCompB) находится в диапазоне от 3 до 32 включительно), ctxIdxInc устанавливают равным 1.

9.3.3.1.1.8 Процесс отыскания ctxIdxInc элемента синтаксиса intra_chroma_pred_mode

Выход этого процесса – ctxIdxInc.

Активируют процесс нахождения смежных макроблоков, определенных в п. 6.4.7.1, и присваивают выходу значения mbAddrA и mbAddrB.

Предположим, что переменную condTermFlagN (с заменой N на A или B) получили следующим образом:

– Если любое из перечисленных условий – истина, то condTermFlagN устанавливают равным 0:
  - mbAddrN недоступно;
  - макроблок mbAddrN кодирован в режим Inter предсказания;
  - mb_type макроблока mbAddrN равно I_PCM;
  - intra_chroma_pred_mode макроблока mbAddrN равно 0.

– Иначе condTermFlagN устанавливают равным 1.

Переменную ctxIdxInc находят как

ctxIdxInc = condTermFlagA + condTermFlagB. (9-12)

9.3.3.1.1.9 Процесс отыскания ctxIdxInc элемента синтаксиса coded_block_flag

Вход в этот процесс – ctxBlockCat, а дополнительный вход описывают следующим образом:

– Если ctxBlockCat равно 0, дополнительный вход отсутствует.

– Иначе, если ctxBlockCat равно 1 или 2, то luma4x4BlkIdx.

– Иначе, если ctxBlockCat равно 3, то индекс компонента цветности iCbCr.

– Иначе (ctxBlockCat равно 4), то chroma4x4BlkIdx, а индекс компонента цветности compIdx.

Выход этого процесса – ctxIdxInc(ctxBlockCat).

Предположим, что переменную transBlockN (с N, принимающим значения A или B) получили следующим образом:

– Если ctxBlockCat равно 0, применяют следующее:
  - Активируют процесс нахождения смежных макроблоков, определенных в п. 6.4.7.1, и присваивают выходу значение mbAddrN (с N, принимающим значения A или B).

- Переменную transBlockN находят следующим образом:
  - Если mbAddrN доступно, а макроблок mbAddrN кодирован в режиме предсказания Intra_16x16, то блоку яркости DC макроблока mbAddrN присваивают значение transBlockN.
  - Иначе transBlockN помечают как недоступный.
- Иначе, если ctxBlockCat равно 1 или 2, применяют следующее:
  - Активируют процесс нахождения смежных блоков яркости 4x4, определенных в п. 6.4.7.3, с luma4x4BlkIdx в качестве входа, и присваиваются выходу значение mbAddrN, luma4x4BlkIdxN (с N, принимающим значения A или B).
  - Переменную transBlockN находят следующим образом:
    - Если mbAddrN доступен, макроблок mbAddrN не пропущен, mb_type макроблока mbAddrN не равно I_PCM, а ((CodedBlockPatternLuma >> (luma4x4BlkIdxN >>2)) & 1) не равно 0 для макроблока mbAddrN, то блоку яркости 4x4 с luma4x4BlkIdxN макроблока mbAddrN присваивают transBlockN.
    - Иначе transBlockN помечают как недоступный.
- Иначе, если ctxBlockCat равно 3, применяют следующее:
  - Активируют процесс нахождения смежных макроблоков, определенных в п. 6.4.7.1, и присваивают выходу значение mbAddrN (с N, принимающим значения A или B).
  - Переменную transBlockN находят следующим образом:
    - Если mbAddrN доступно, макроблок mbAddrN не пропущен, mb_type макроблока mbAddrN не равно I_PCM и CodedBlockPatternChroma не равно 0 для макроблока mbAddrN, то компоненту цветности iCbCr DC блока цветности макроблока mbAddrN присваивают значение transBlockN.
    - Иначе transBlockN помечают как недоступный.
- Иначе (ctxBlockCat равно 4), применяют следующее:
  - Активируют процесс нахождения смежных блоков цветности 4x4, определенных в п. 6.4.7.4, с chroma4x4BlkIdx в качестве входа и присваивают выходу mbAddrN значение chroma4x4BlkIdxN (с N, принимающим значения A или B).
  - Переменную transBlockN находят следующим образом:
    - Если mbAddrN доступно, макроблок mbAddrN не пропущен, mb_type макроблока mbAddrN не равно I_PCM и CodedBlockPatternChroma равно 2 макроблока mbAddrN, то компоненту цветности iCbCr блока цветности 4x4 с chroma4x4BlkIdxN макроблока mbAddrN присваивают значение transBlockN.
    - Иначе transBlockN помечают как недоступный.

Предположим, что переменную condTermFlagN (с N, принимающим значения A или B) получили следующим образом:

- Если любое из перечисленных условий – истина, то condTermFlagN устанавливают равным 0:
  - mbAddrN недоступно, а текущий макроблок кодирован в режиме Inter предсказания;
  - mbAddrN доступно, а transBlockN недоступно и mb_type макроблока mbAddrN не равно I_PCM;
  - текущий макроблок кодирован в режиме Intra предсказания, constrained_intra_pred_flag равно 1, макроблок mbAddrN доступен и кодирован в режиме Inter предсказания, а также используют данные секции разделения на части (nal_unit_type находится в диапазоне от 2 до 4 включительно).
- Иначе, если следующие условия – истина, то condTermFlagN устанавливают равным 1,
  - mbAddrN недоступно, а текущий макроблок кодирован в режиме Intra предсказания,
  - mb_type макроблока mbAddrN равно I_PCM.
- Иначе condTermFlagN устанавливают равным значению coded_block_flag преобразования блока transBlockN, который был декодирован для макроблока mbAddrN.

Переменную ctxIdxInc(ctxBlockCat) находят как

$$ctxIdxInc(ctxBlockCat) = condTermFlagA + 2 * condTermFlagB. \quad (9\text{-}13)$$

9.3.3.1.2 Процесс присвоения значения ctxIdxInc, которое используют перед декодированием значений бинов

Входы в этот процесс – ctxIdxOffset и binIdx.

Выход этого процесса – ctxIdxInc.

Таблица 9-31 содержит спецификацию ctxIdxInc для заданных значений ctxIdxOffset и binIdx.

Для каждого ctxIdxOffset binIdx значение ctxIdxInc находят, используя некоторые из ранее декодированных значений бинов (b_0, b_1, b_2, …, b_k), где значение индекса k меньше, чем значение binIdx.

Таблица 9-31 – Спецификация ctxIdxInc для особых значений ctxIdxOffset и binIdx

Значение (имя) ctxIdxOffset	binIdx	ctxIdxInc
3	4	(b_3 != 0) ? 5: 6
	5	(b_3 != 0) ? 6: 7
14	2	(b_1 != 1) ? 2: 3
17	4	(b_3 != 0) ? 2: 3
27	2	(b_1 != 0) ? 4: 5
32	4	(b_3 != 0) ? 2: 3
36	2	(b_1 != 0) ? 2: 3

9.3.3.1.3 Процесс присвоения ctxIdxInc элементам синтаксиса significant_coeff_flag, last_significant_coeff_flag и coeff_abs_level_minus1

Входы в этот процесс – ctxIdxOffset и binIdx.

Выход этого процесса – ctxIdxInc.

Процесс присвоения ctxIdxInc элементам синтаксиса significant_coeff_flag, last_significant_coeff_flag и coeff_abs_level_minus1, а также coded_block_flag зависит от категорий различных блоков, отмеченных переменной ctxBlockCat. Спецификация этих категорий блоков показана в таблице 9-32.

Таблица 9-32 – Спецификация ctxBlockCat для различных блоков

Описание блока	maxNumCoeff	ctxBlockCat
блок яркости DC уровней коэффициентов преобразования (для макроблока, кодированного в режиме предсказания Intra_16x16)	16	0
блок яркости AC уровней коэффициентов преобразования (для макроблока, кодированного в режиме предсказания Intra_16x16)	15	1
блок яркости уровней коэффициентов преобразования (для макроблока, кодированного в режиме предсказания Intra_16x16)	16	2
блок цветности DC уровней коэффициентов преобразования	4	3
блок цветности AC уровней коэффициентов преобразования	15	4

Для элементов синтаксиса significant_coeff_flag и last_significant_coeff_flag позицию сканирования scanningPos для рассматриваемого блока присваивают значению ctxIdxInc, где диапазон scanningPos находится от 0 до maxNumCoeff – 2 включительно:

$$ctxIdxInc = scanningPos. \quad (9\text{-}14)$$

Позиция сканирования для кадра, кодированного блоками, зависит от сканирования зигзагом. Позиция сканирования для поля, кодированного блоками, зависит от сканирования поля.

Положим, что numDec108odAbsLevelEq1 описывает общее число декодированных уровней коэффициентов преобразования с абсолютным значением, равным 1, и положим, что numDecodAbsLevelGt1 описывает общее число

декодированных уровней коэффициентов преобразования с абсолютным значением, большим, чем 1. Оба набора чисел относятся к тем же коэффициентам преобразования блока, где происходит текущий процесс декодирования. Далее декодирование coeff_abs_level_minus1 и ctxIdxInc для coeff_abs_level_minus1 описывают в зависимости от binIdx следующим образом:

– Если binIdx равно 0, ctxIdxInc находят как

ctxIdxInc = ((numDecodAbsLevelGt1 != 0) ? 0: Min(4, 1 + numDecodAbsLevelEq1)). (9-15)

– Иначе (binIdx больше, чем 0), ctxIdxInc находят как

ctxIdxInc = 5 + Min(4, numDecodAbsLevelGt1). (9-16)

9.3.3.2 Процесс арифметического декодирования

Входы в этот процесс – bypassFlag, ctxIdx, как указано в п. 9.3.3.1, и переменные состояний codIRange и codIOffset устройства арифметического декодирования.

Выход этого процесса – значение бина.

Рисунок 9-2 иллюстрирует весь процесс арифметического декодирования для одного бина. Для декодирования значения бина индекс контекста ctxIdx вводят в процесс арифметического декодирования DecodeBin(ctxIdx), который описывают следующим образом:

– Если bypassFlag равно 1, то активируют DecodeBypass( ), как это определено в п. 9.3.3.2.3.

– Иначе, если bypassFlag равно 0, а ctxIdx равно 276, то активируют DecodeTerminate( ), как это определено в п. 9.3.3.2.4.

– Иначе (bypassFlag равно 0, а ctxIdx не равно 276), должно быть использовано значение DecodeDecision( ), как это определено в п. 9.3.3.2.1.

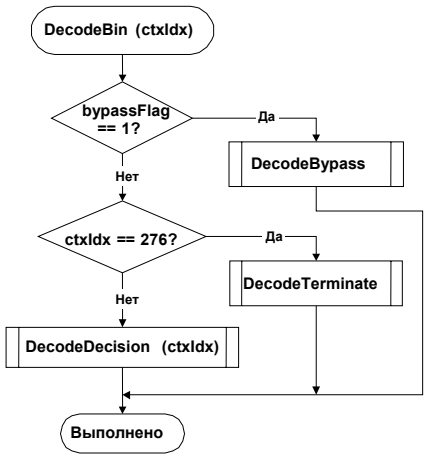

Рисунок 9-2 –Диаграмма последовательности процесса арифметического декодирования для одиночного бина (информативное)

ПРИМЕЧАНИЕ. – Арифметическое кодирование основано на принципе рекурсивного подразделения интервала. Заданная оценка вероятностей p(0) и p(1) = 1 – p(0) бинарного решения (0, 1), первоначально заданная в подинтервале с диапазоном codIRange, должна быть разделена на два подинтервала с диапазонами p(0) * codIRange и codIRange – p(0) * codIRange соответственно. В зависимости от принятого решения должен быть выбран соответствующий подинтервал в качестве нового кодового интервала, а положение бинарной кодовой строки в этом интервале должно представлять последовательность рассмотренных бинарных решений. Полезно различать наиболее вероятный символ (MPS) и наименее вероятный символ (LPS) для того, чтобы это бинарное решение могло быть идентифицировано скорее как MPS или LPS, чем как 0 или 1. Придерживаясь такой терминологии, каждый контекст описывают вероятностью p_{LPS} для LPS и значением MPS (valMPS), которое может быть 0 или 1.

В этой Рекомендации | Международном стандарте основная часть арифметического устройства обладает тремя отличительными свойствами:

– Оценка вероятности производится конечным автоматом с табличным процессом перехода от 64 различных представлений состояний вероятности { p_{LPS}(pStateIdx) | 0 <= pStateIdx < 64 } к вероятности LPS – p_{LPS}. Число состояний организуют таким образом, что вероятности состояний с индексом pStateIdx = 0 соответствует вероятность значения LPS в 0,5, с уменьшением вероятности LPS по мере возрастания индексов состояния.

– Перед вычислением нового диапазона интервала диапазон codIRange, представляющий состояние кодирующего устройства, квантуют до небольшого множества $\{Q_1,\ldots,Q_4\}$ заранее квантованных значений. Хранимая таблица, которая содержит все 64x4 заранее вычисленные результирующие значения $Q_i * p_{LPS}(pStateIdx)$, разрешает свободный порядок аппроксимации результата в виде codIRange * $p_{LPS}(pStateIdx)$.

– Для элементов синтаксиса или их частей, для которых предполагают приблизительно равномерное распределение вероятности, используют упрощенный процесс обхода кодирования и декодирования.

9.3.3.2.1 Процесс арифметического декодирования для бинарного решения

Входы в этот процесс – ctxIdx, codIRange и codIOffset.

Выходы этого процесса – декодированные значения binVal и обновленные переменные codIRange и codIOffset.

Рисунок 9-3 показывает диаграмму последовательности действий для декодирования одного решения (DecodeDecision).

1. Значение переменной codIRangeLPS находят следующим образом:

– Для заданных текущих значений codIRange переменную qCodIRangeIdx находят как

$$qCodIRangeIdx = (codIRange >> 6) \& 0x03. \quad (9\text{-}17)$$

– Для заданных значений qCodIRangeIdx и pStateIdx, которые связаны с ctxIdx, переменной RangeTabLPS, как это определено в таблице 9-33, присваивают значение codIRangeLPS:

$$codIRangeLPS = RangeTabLPS[pStateIdx][qCodIRangeIdx]. \quad (9\text{-}18)$$

2. Переменную codIRange устанавливают равной codIRange – codIRangeLPS и применяют следующее:

– Если codIOffset больше или равно codIRange, то переменную binVal устанавливают равной 1 – valMPS, codIOffset уменьшают на codIRange, а codIRange устанавливают равной codIRangeLPS.

– Иначе переменную binVal устанавливают равной valMPS.

При заданных значениях binVal состояние перехода выполняют, как это определено в п. 9.3.3.2.1.1. В зависимости от текущего значения codIRange выполняют изменение нормировки, как это определено в п. 9.3.3.2.2.

9.3.3.2.1.1 Процесс перехода состояния

Входы в этот процесс – текущее значение pStateIdx, декодированные значения binVal и valMPS контекстной переменной, связанной с ctxIdx.

Выходы этого процесса – обновленные значения pStateIdx и valMPS контекстной переменной, связанной с ctxIdx.

В зависимости от декодированных значений binVal обновление двух переменных pStateIdx и valMPS, связанных с ctxIdx, находят следующим образом:

```
if( binVal = = valMPS )
    pStateIdx = transIdxMPS( pStateIdx )
else {                                                    (9-19)
    if( pStateIdx = = 0 )
        valMPS = 1 – valMPS
    pStateIdx = transIdxLPS( pStateIdx )
}
```

Таблица 9-34 определяет правила перехода transIdxMPS( ) и transIdxLPS( ) после декодирования значений valMPS и 1 – valMPS соответственно.

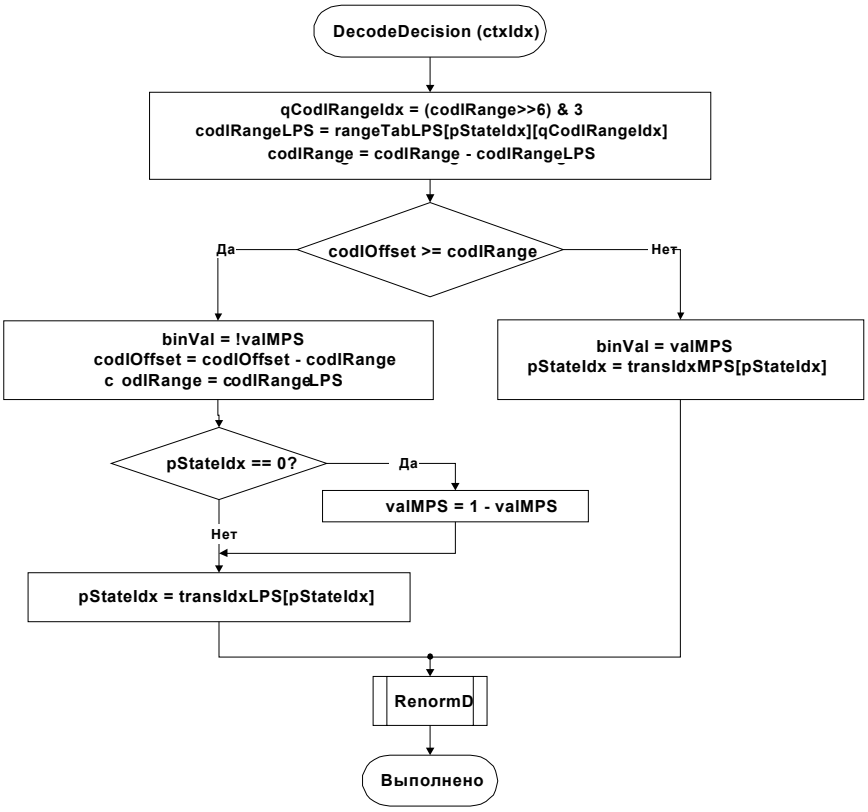

Рисунок 9-3 –Диаграмма последовательности для декодирования решения

Таблица 9-33 – Спецификация RangeTabLPS в зависимости от pStateIdx и qCodIRangeIdx

pStateIdx	qCodIRangeIdx				pStateIdx	qCodIRangeIdx			
	0	1	2	3		0	1	2	3
0	128	176	208	240	**32**	27	33	39	45
1	128	167	197	227	**33**	26	31	37	43
2	128	158	187	216	**34**	24	30	35	41
3	123	150	178	205	**35**	23	28	33	39
4	116	142	169	195	**36**	22	27	32	37
5	111	135	160	185	**37**	21	26	30	35
6	105	128	152	175	**38**	20	24	29	33
7	100	122	144	166	**39**	19	23	27	31
8	95	116	137	158	**40**	18	22	26	30
9	90	110	130	150	**41**	17	21	25	28
10	85	104	123	142	**42**	16	20	23	27
11	81	99	117	135	**43**	15	19	22	25
12	77	94	111	128	**44**	14	18	21	24
13	73	89	105	122	**45**	14	17	20	23
14	69	85	100	116	**46**	13	16	19	22
15	66	80	95	110	**47**	12	15	18	21
16	62	76	90	104	**48**	12	14	17	20
17	59	72	86	99	**49**	11	14	16	19
18	56	69	81	94	**50**	11	13	15	18
19	53	65	77	89	**51**	10	12	15	17
20	51	62	73	85	**52**	10	12	14	16
21	48	59	69	80	**53**	9	11	13	15
22	46	56	66	76	**54**	9	11	12	14
23	43	53	63	72	**55**	8	10	12	14
24	41	50	59	69	**56**	8	9	11	13
25	39	48	56	65	**57**	7	9	11	12
26	37	45	54	62	**58**	7	9	10	12
27	35	43	51	59	**59**	7	8	10	11
28	33	41	48	56	**60**	6	8	9	11
29	32	39	46	53	**61**	6	7	9	10
30	30	37	43	50	**62**	6	7	8	9
31	29	35	41	48	**63**	2	2	2	2

Таблица 9-34 – Таблица перехода состояний

pStateIdx	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
transIdxLPS	0	0	1	2	2	4	4	5	6	7	8	9	9	11	11	12
transIdxMPS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
pStateIdx	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
transIdxLPS	13	13	15	15	16	16	18	18	19	19	21	21	22	22	23	24
transIdxMPS	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
pStateIdx	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**
transIdxLPS	24	25	26	26	27	27	28	29	29	30	30	30	31	32	32	33
transIdxMPS	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
pStateIdx	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**	**60**	**61**	**62**	**63**
transIdxLPS	33	33	34	34	35	35	35	36	36	36	37	37	37	38	38	63
transIdxMPS	49	50	51	52	53	54	55	56	57	58	59	60	61	62	62	63

9.3.3.2.2 Процесс изменения нормировки в устройстве арифметического декодирования

Входы в этот процесс – биты из данных секции и переменные codIRange и codIOffset.

Выходы этого процесса – обновленные переменные codIRange и codIOffset.

Диаграмма последовательности изменения нормировки показана на рисунке 9-4. Текущее значение codIRange сначала сравнивают с 0x0100, а дальнейшие шаги следующие:

– Если codIRange больше или равно 0x0100, то изменения нормировки не требуется, и процесс RenormD заканчивается.

– Иначе (codIRange меньше, чем 0x0100), вводят цикл изменения нормировки. В этом цикле значение codIRange удваивают, т. е. сдвигают влево на 1, а единичный бит сдвигают на codIOffset, используя read_bits(1).

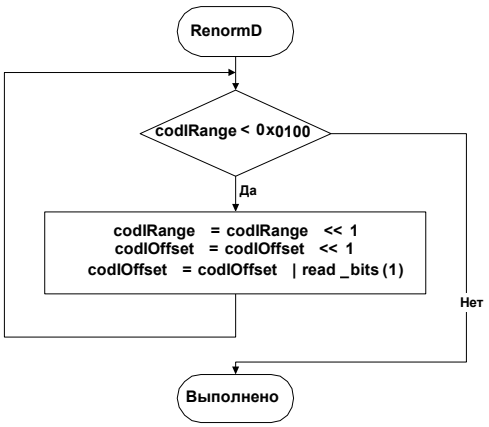

Рисунок 9-4 – Диаграмма последовательности изменения нормировки

9.3.3.2.3 Процесс обхода декодирования бинарных решений

Входы в этот процесс – биты из данных секции и переменные codIRange и codIOffset.

Выходы этого процесса – обновленная переменная codIOffset и декодированное значение binVal.

Процесс обхода декодирования активируют, если bypassFlag равно 1. Рисунок 9-5 показывает диаграмму последовательности соответствующего процесса.

Сначала значение codIOffset удваивают, т. е. сдвигают влево на 1, а единичный бит сдвигают на codIOffset, используя read_bits(1). Затем значение codIOffset сравнивают со значением codIRange, а дальнейшие шаги определены следующим образом:

– Если codIOffset больше или равно codIRange, то переменную binVal устанавливают равной 1, а codIOffset уменьшают на codIRange.

– Иначе (codIOffset меньше, чем codIRange), переменную binVal устанавливают на 0.

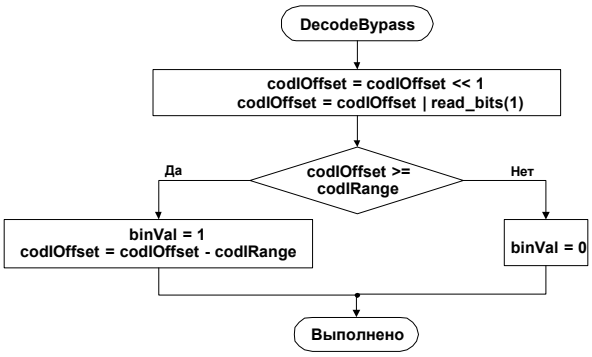

Рисунок 9-5 – Диаграмма последовательности процесса обхода декодирования

9.3.3.2.4 Процесс декодирования бинарного решения перед завершением

Входы в этот процесс – биты из данных секции и переменные codIRange и codIOffset.

Выходы этого процесса – обновленные переменные codIRange и codIOffset и декодированное значение binVal.

Эту особую рутинную процедуру декодирования применяют к декодированию end_of_slice_flag и к бину, который указывает соответствующий режим I_PCM значения ctxIdx, равного 276. Рисунок 9-6 показывает диаграмму последовательности соответствующего процесса декодирования, который описывают следующим образом.

Сначала значение codIRange уменьшают на 2. Затем значение codIOffset сравнивают со значением codIRange, а дальнейшие шаги определены следующим образом:

– Если codIOffset больше или равно codIRange, то переменную binVal устанавливают равной 1, изменения нормировки не производят, а декодирование CABAC закончено. Последний введенный в регистр codIOffset бит равен 1. При декодировании end_of_slice_flag последний введенный в регистр codIOffset бит интерпретируют как rbsp_stop_one_bit.

– Иначе (codIOffset меньше codIRange), переменную binVal устанавливают на 0, процесс изменения нормировки выполняют, как это определено в п. 9.3.3.2.2.

ПРИМЕЧАНИЕ. – Эту процедуру можно также выполнить, используя DecodeDecision(ctxIdx) с ctxIdx = 276. В этом случае там, где декодированное значение равно 1, более семи битов могут быть считаны с помощью DecodeDecision(ctxIdx), а процесс декодирования должен настраиваться указателем потока битов, должным образом декодированным согласно последующим элементам синтаксиса.

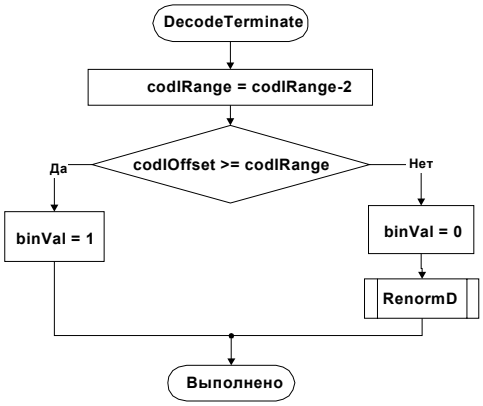

Рисунок 9-6 – Диаграмма последовательности декодирования решения перед завершением

9.3.4 Процесс арифметического кодирования (информативное)

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Входы в этот процесс – решения, которые следует кодировать и записать.

Выходы этого процесса – биты, которые записаны в RBSP.

Этот информативный пункт описывает арифметическое кодирующее устройство, которое соответствует устройству арифметического декодирования, описанному в п. 9.3.3.2. Кодирующее устройство является по существу симметричным устройству декодирования, т. е. процедуры выполняют в том же самом порядке. В этом разделе описаны следующие процедуры: InitEncoder, EncodeDecision, EncodeBypass, EncodeTerminate, которым соответствуют InitDecoder, DecodeDecision, DecodeBypass и DecodeTerminate соответственно. Состояние устройства арифметического кодирования представлено значением переменной codILow, указывающей на нижний конец подинтервала, а значение переменной codIRange определяет соответствующий диапазон подинтервала.

9.3.4.1 Процесс инициализации устройства арифметического кодирования (информативное)

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Процесс активируют перед кодированием первого макроблока секции и после кодирования pcm_alignment_zero_bit и всех данных pcm_byte макроблока типа I_PCM.

Выходы этого процесса – значения codILow, codIRange, firstBitFlag, bitsOutstanding и symCnt устройства арифметического кодирования.

Во время процедуры инициализации кодера codILow устанавливают равным 0, а codIRange устанавливают равным 0x01FE. Далее firstBitFlag устанавливают равным 1, а счетчики bitsOutstanding и symCnt устанавливают равными 0.

ПРИМЕЧАНИЕ. – Минимальная точность регистра, которую требуют для codILow, составляет 10 битов, а для CodIRange – 9 битов. Точность, которая требуется для счетчиков bitsOutstanding и symCnt, должна быть существенно выше, чтобы предотвратить переполнение соответствующих регистров. Если MaxBinCountInSlice указывает на максимальное общее число бинарных решений для кодирования в одной секции, то минимальная точность регистра, которая требуется для переменных bitsOutstanding и symCnt, задана значением Ceil(Log2(MaxBinCountInSlice + 1)).

9.3.4.2 Процесс кодирования бинарного решения (информативное)

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Входы в этот процесс – индекс контекста ctxIdx, значение binVal, которое следует кодировать, и переменные codIRange, codILow и symCnt.

Выходы этого процесса – переменные codIRange, codILow и symCnt.

Рисунок 9-7 показывает диаграмму последовательности для кодирования одного решения. На первом этапе переменную codIRangeLPS находят следующим образом.

При заданном текущем значении codIRange codIRange отображают в индекс qCodIRangeIdx квантованного значения codIRange, используя равенство 9-17. Значение qCodIRangeIdx и значение pStateIdx, объединенные с ctxIdx, используют для определения значения переменной rangeTabLPS (как это показано в таблице 9-33), которой присваивают значение codIRangeLPS. Значение codIRange – codIRangeLPS присваивают codIRange.

На втором этапе значение binVal сравнивают с valMPS, объединенным с ctxIdx. Если binVal отличается от valMPS, значение codIRange добавляют к codILow, а codIRange устанавливают равным значению codIRangeLPS. При заданном кодированном решении переход состояния выполняют, как это определено в п. 9.3.3.2.1.1. В зависимости от текущего значения codIRange производят изменение нормировки, как это определено в п. 9.3.4.3. Наконец, переменная symCnt увеличивается на 1.

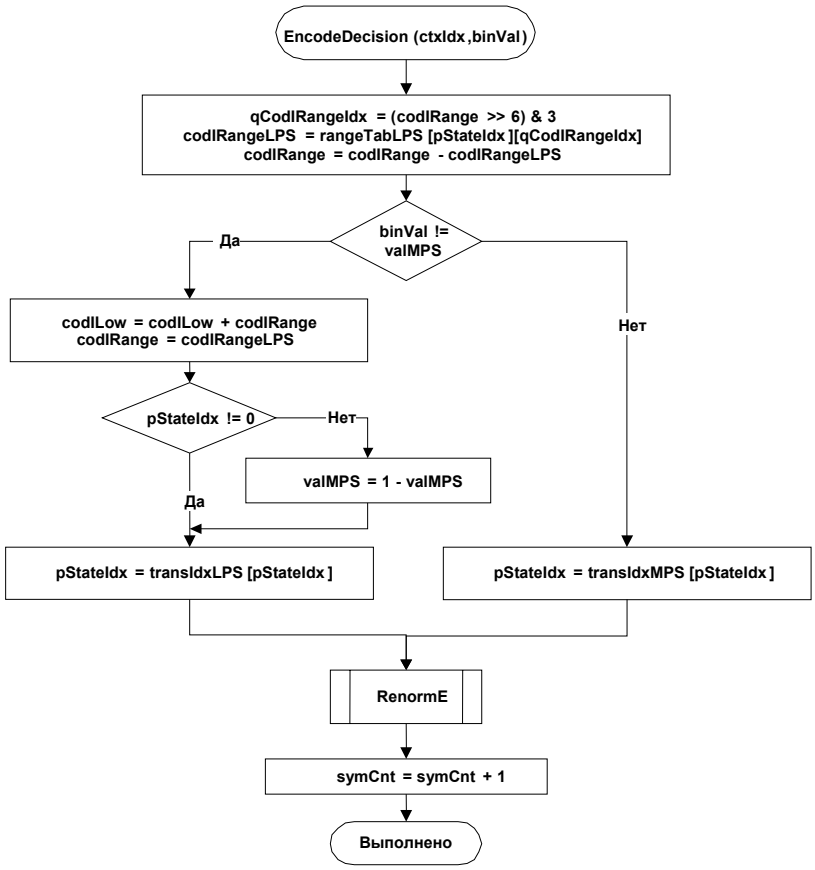

Рисунок 9-7 – Диаграмма последовательности кодирования решения

9.3.4.3 Процесс изменения нормировки в устройстве арифметического кодирования (информативное)

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Входы в этот процесс – переменные codIRange, codILow, firstBitFlag и bitsOutstanding.

Выходы этого процесса – нуль или более битов, записанных в RBSP, и обновленные переменные codIRange, codILow, firstBitFlag и bitsOutstanding.

Изменение нормировки проиллюстрировано на рисунке 9-8.

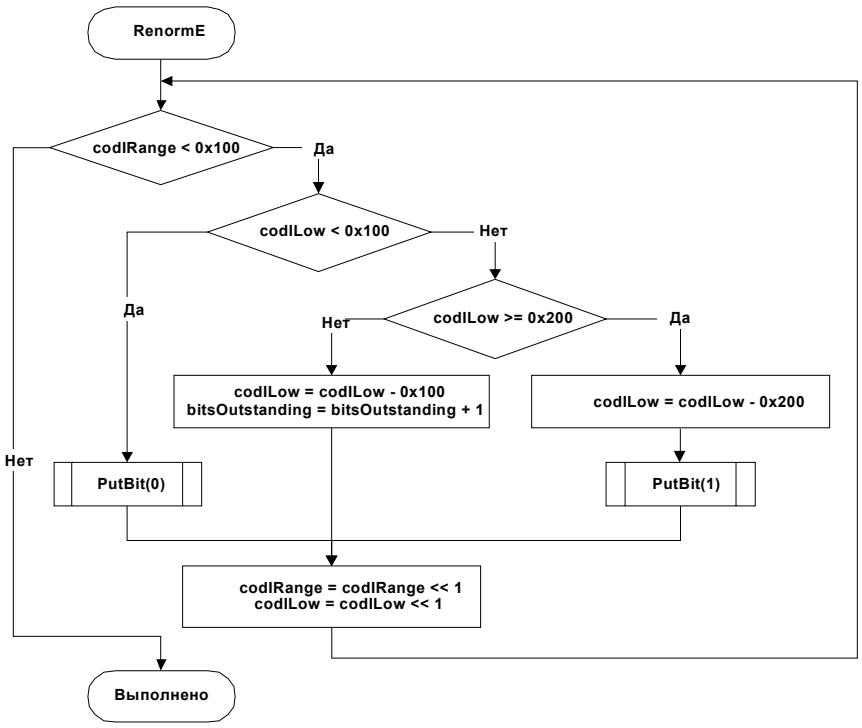

Рисунок 9-8 – Диаграмма последовательности изменения нормировки в кодере

Процедура PutBit( ), показанная на рисунке 9-9, обеспечивает перенос управления. При этом используют функцию WriteBits(B, N), которая записывает N битов со значением (B) в поток битов и продвигает указатель потока битов вперед на N битов. Действие этой функции предполагает существование указателя потока битов с указанием позиции следующего бита, который следует записать в поток битов с помощью процесса кодирования.

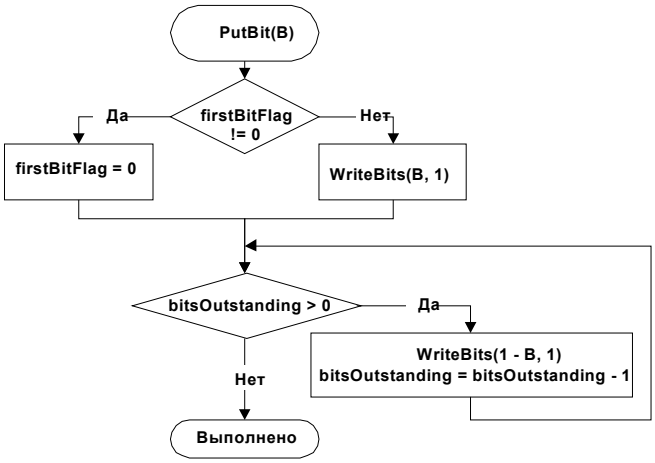

Рисунок 9-9 – Диаграмма последовательности PutBit(B)

9.3.4.4 Процесс обхода кодирования бинарных решений (информативное)

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Входы в этот процесс – переменные binVal, codILow, codIRange, bitsOutstanding и symCnt.

Выход этого процесса – бит, записанный в RBSP, и обновленные переменные codILow, bitsOutstanding и symCnt.

Этот процесс кодирования применим ко всем бинарным решениям со значением bypassFlag, равным 1. Изменение нормировки включено в спецификацию этого процесса, как показано на рисунке 9-10.

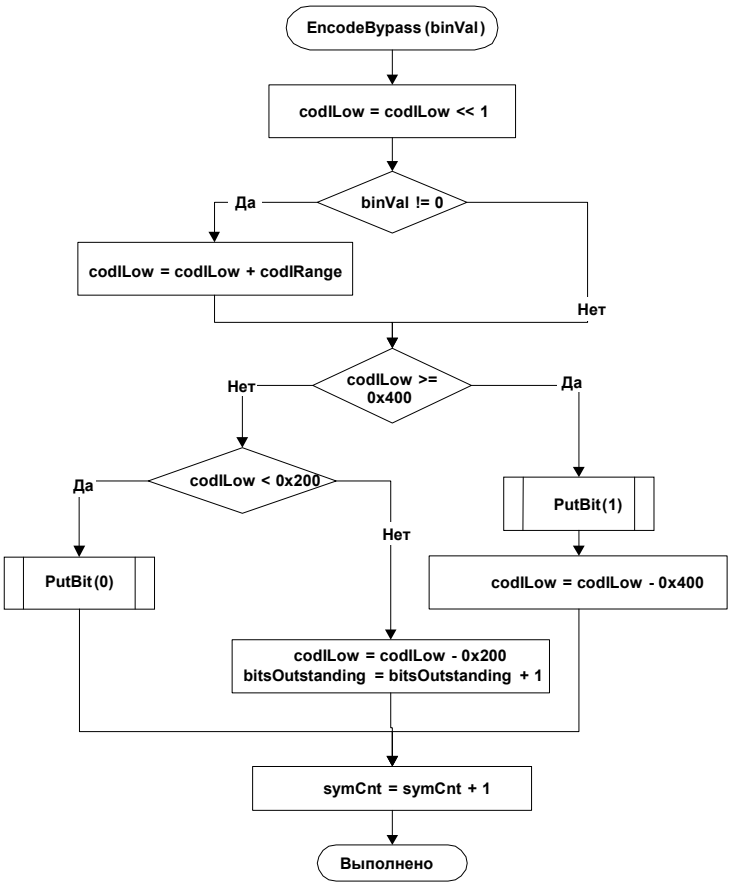

Рисунок 9-10 – Диаграмма последовательности обхода кодирования

9.3.4.5 Процесс кодирования бинарного решения перед завершением (информативное)

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Входы в этот процесс – переменные binVal, codIRange, codILow, bitsOutstanding и symCnt.

Выходы этого процесса – нуль или более битов, записанных в RBSP, и обновленные переменные codILow, codIRange, bitsOutstanding и symCnt.

Общепринятый порядок кодирования, показанный на рисунке 9-11, применим к кодированию значения end_of_slice_flag и бина, указывающего тип I_PCM: mb_type. Оба значения связаны с ctxIdx, равным 276.

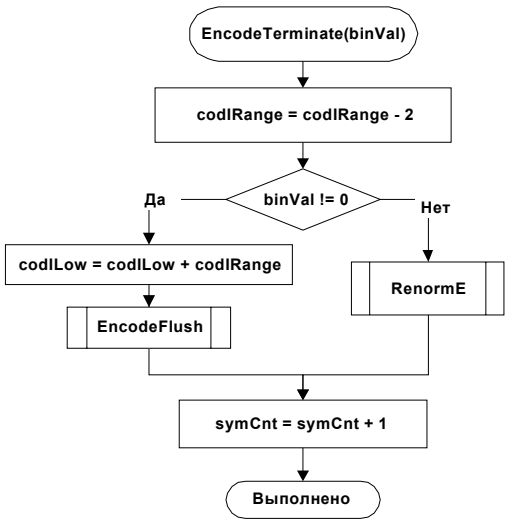

Рисунок 9-11 – Диаграмма последовательности кодирования решения перед завершением

Если предназначенное для кодирования значение binVal равно 1, то кодирование CABAC завершают и выполняют процедуру сдвига, показанную на рисунке 9-12. В процедуре сдвига последний бит, записанный с помощью WriteBits(B, N), равен 1. При кодировании end_of_slice_flag, последний бит интерпретируют как rbsp_stop_one_bit.

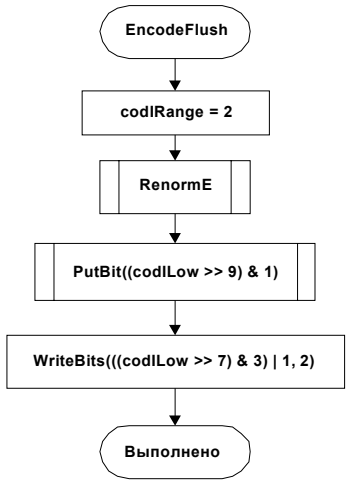

Рисунок 9-12 – Диаграмма последовательности сдвига при завершении

9.3.4.6 Процесс стаффинга байтов (информативное)

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Процесс активируют после кодирования последнего макроблока последней секции изображения и после инкапсуляции.

Входы в этот процесс – число байтов NumBytesInVclNALunits из всех частей NAL VCL изображения, число макроблоков PicSizeInMbs в изображении и число бинарных символов BinCountsInNALunits, полученных в результате кодирования содержания всех частей NAL VCL изображения.

Выходы этого процесса – нуль или более байтов, присоединенных к блоку NAL.

Предположим, что переменная k установлена равной Ceil((Ceil((3 * BinCountsInNALunits – 3 * 96 * PicSizeInMbs)/32) – NumBytesInVclNALunits)/3). В зависимости от переменной k применяют следующее:

– Если k меньше или равно 0, к блоку NAL не присоединяют cabac_zero_word.

– Иначе (k больше 0), после инкапсуляции к блоку NAL k раз присоединяют 3-байтовую последовательность 0x000003, где первые два байта 0x0000 представляют значение cabac_zero_word, а третий байт 0x03 представляет значение emulation_prevention_three_byte.

Приложение A

Параметры и уровни

(Это Приложение является составной частью данной Рекомендации | Международного стандарта)

Параметры и уровни определяют ограничения на потоки битов и поэтому ограничивают возможности, необходимые для декодирования потоков битов. Параметры и уровни можно также использовать для указания совместимости применений отдельных декодеров.

ПРИМЕЧАНИЕ. – В эту Рекомендацию | Международный стандарт не включены отдельно выбранные "опции" декодера, так как это увеличило бы трудности совместимости.

Каждый параметр определяет подмножество алгоритмических особенностей и ограничений, которые должны поддерживать все декодеры, соответствующие этим параметрам.

ПРИМЕЧАНИЕ. – Для того чтобы сделать возможным применение любых конкретных свойств, заложенных в параметрах, кодеры не требуются.

Каждый уровень определяет набор ограничений на значения, которые могут быть наложены элементами синтаксиса этой Рекомендации | Международного стандарта. Тот же набор определений уровней используют со всеми параметрами, но частные применения могут поддерживать различный уровень для каждого поддерживаемого параметра. Уровни для любых заданных параметров, как правило, соответствуют обработке декодером нагрузки и возможностям его памяти.

A.1 Требования к возможностям видеодекодера

Возможности видеодекодеров, соответствующих этой Рекомендации | Международному стандарту, определены в понятиях способности декодировать видеопотоки, соответствующие ограничениям на параметры и уровни, определенные в этом Приложении. Для каждого такого параметра также должны быть расширены уровни поддержки этих параметров.

Особые значения определены в этом Приложении для элементов синтаксиса profile_idc и level_idc. Все прочие значения profile_idc и level_idc зарезервированы для будущего использования МСЭ-Т | ИСО/МЭК.

ПРИМЕЧАНИЕ. – Декодеры не должны считать зарезервированные значения profile_idc или level_idc, которые могут находиться между значениями, определенными в этой Рекомендации | Международном стандарте, как некоторые промежуточные возможности между определенными параметрами или уровнями. В действительности отсутствуют ограничения на метод, который был выбран МСЭ-Т | ИСО/МЭК для использования в будущем подобных зарезервированных значений.

A.2 Параметры

A.2.1 Основные параметры

На потоки битов, соответствующие основным параметрам, должны быть наложены следующие ограничения:

- Могут быть представлены только типы секций I и P.
- Потоки блоков NAL не должны содержать значения nal_unit_type в диапазоне от 2 до 4 включительно.
- Последовательность набора параметров должна иметь frame_mbs_only_flag, равное 1.
- Набор параметров изображения должен иметь weighted_pred_flag и weighted_bipred_idc, оба равные 0.
- Набор параметров изображения должен иметь entropy_coding_mode_flag, равное 0.
- Набор параметров изображения должен иметь num_slice_groups_minus1 в диапазоне от 0 до 7 включительно.
- Для основных параметров из п. A.3. должен соблюдаться специальный уровень ограничений.

Соответствие потока битов основным параметрам определено значением profile_idc, равным 66.

Декодеры, соответствующие основным параметрам со специальным уровнем, должны быть в состоянии декодировать все потоки битов, в которых profile_idc равно 66 или constraint_set0_flag равно 1, а также в которых level_idc представляет уровень, меньший или равный специальному уровню.

A.2.2 Главные параметры

На потоки битов, соответствующие главным параметрам, должны быть наложены следующие ограничения:

- Могут быть представлены только типы секций I, P и B.
- Потоки блоков NAL не должны содержать значения nal_unit_type в диапазоне от 2 до 4 включительно.
- Произвольный порядок секций не допустим.
- Набор параметров изображения должен иметь значение num_slice_groups_minus1, только равное 0.
- Набор параметров изображения должен иметь значение redundant_pic_cnt_present_flag, только равное 0.

– Для главных параметров из пункта A.3 должен соблюдаться специальный уровень ограничений.

Соответствие потока битов главным параметрам определено значением profile_idc, равным 77.

Декодеры, соответствующие главным параметрам со специальным уровнем, должны быть в состоянии декодировать все потоки битов, в которых profile_idc равно 77 или constraint_set1_flag равно 1 и в которых level_idc представляет уровень, меньший или равный специальному уровню.

A.2.3 Расширенные параметры

На потоки битов, соответствующие расширенным параметрам, должны быть наложены следующие ограничения:

– Последовательность набора параметров должна иметь direct_8x8_inference_flag, равное 1.

– Набор параметров изображения должен иметь значение entropy_coding_mode_flag, равное 0.

– Набор параметров изображения должен иметь значение num_slice_groups_minus1 в диапазоне от 0 до 7 включительно.

– Для расширенных параметров из пункта A.3 должен соблюдаться специальный уровень ограничений.

Соответствие потока битов расширенным параметрам определено значением profile_idc, равным 88.

Декодеры, соответствующие расширенным параметрам со специальным уровнем, должны быть в состоянии декодировать все потоки битов, в которых profile_idc равно 88 или constraint_set2_flag равно 1 и в которых level_idc представляет уровень, меньший или равный специальному уровню.

Декодеры, соответствующие расширенным параметрам со специальным уровнем, должны быть также в состоянии декодировать все потоки битов, в которых profile_idc равно 66 или constraint_set0_flag равно 1 и в которых level_idc представляет уровень, меньший или равный специальному уровню.

A.3 Уровни

В этом Приложении для выражения ограничений определено следующее:

– Положим, что блок доступа n – это n-й блок доступа в порядке декодирования с первым блоком доступа 0.

– Положим, что изображение n – это исходное кодированное изображение или соответствующее декодированное изображение блока доступа n.

A.3.1 Границы уровней с независимыми параметрами

Положим, что переменную fR находят следующим образом:

– Если изображение n – это кадр, fR устанавливают равным 1 ÷ 172.

– Иначе (изображение n – это поле), fR устанавливают равным 1 ÷ (172 * 2).

На потоки битов, соответствующие любым параметрам со специальным уровнем, должны быть наложены следующие ограничения:

a) Номинальное время удаления блока доступа n (с n > 0) из CPB, как это определено в п. C.1.2, удовлетворяет ограничению $t_{r,n}(n) - t_r(n - 1)$ больше или равно Max(PicSizeInMbs ÷ MaxMBPS, fR), где MaxMBPS – значение, определенное в таблице A-1, применяют к изображению n, а PicSizeInMbs – это число макроблоков в изображении n.

b) Разность между последовательными временами выхода изображений из DPB, как это определено в п. C.2.2, удовлетворяет ограничению $\Delta t_{o,dpb}(n) >= Max(PicSizeInMbs \div MaxMBPS, fR)$, где MaxMBPS – значение, указанное в таблице A-1 для изображения n, а PicSizeInMbs – число макроблоков изображения n, при условии, что изображение n – это изображение, которое является выходным и не последним изображением в потоке битов на выходе.

c) Сумма переменных NumBytesInNALunit для блока доступа 0 меньше или равна $256 * ChromaFormatFactor * (PicSizeInMbs + MaxMBPS * (t_r(0) - t_{r,n}(0))) \div MinCR$, где MaxMBPS и MinCR – значения, определенные в таблице A-1, которые применимы к изображению 0, а PicSizeInMbs – число макроблоков в изображении 0.

d) Сумма переменных NumBytesInNALunit для блока доступа n (с n > 0) меньше или равна $256 * ChromaFormatFactor * MaxMBPS * (t_r(n) - t_r(n - 1)) \div MinCR$, где MaxMBPS и MinCR – значения, определенные в таблице A-1, которые применимы к изображению n.

e) PicWidthInMbs * FrameHeightInMbs <= MaxFS, где MaxFS определено в таблице A-1.

f) PicWidthInMbs <= Sqrt(MaxFS * 8).

g) FrameHeightInMbs <= Sqrt(MaxFS * 8).

h) max_dec_frame_buffering <= MaxDpbSize, где MaxDpbSize равно Min(1024 * MaxDPB / (PicWidthInMbs * FrameHeightInMbs * 256 * ChromaFormatFactor), 16), а MaxDPB задано в таблице A-1 в блоках по 1024 байтов. Значение max_dec_frame_buffering также называют размером DPB.

i) Если vcl_hrd_parameters_present_flag равно 1, то параметры VCL HRD BitRate[SchedSelIdx] <= 1000 * MaxBR и CpbSize[SchedSelIdx] <= 1000 * MaxCPB, по крайней мере, для одного значения SchedSelIdx, где BitRate[SchedSelIdx] задано уравнением E-13, а CpbSize[SchedSelIdx] задано уравнением E-14. Значения MaxBR и MaxCPB определены в таблице A-1 в блоках 1000 битов/с и 1000 битов, соответственно. Поток битов должен удовлетворять этим условиям, по крайней мере, для одного значения SchedSelIdx в диапазоне от 0 до cpb_cnt_minus1 включительно. CpbSize[SchedSelIdx] также называют размером CPB.

j) Если nal_hrd_parameters_present_flag равно 1, то параметры NAL HRD, BitRate[SchedSelIdx] <= 1200 * MaxBR и CpbSize[SchedSelIdx] <= 1200 * MaxCPB, по крайней мере, для одного значения SchedSelIdx, где BitRate[SchedSelIdx] задано уравнением E-13, а CpbSize[SchedSelIdx] задано уравнением E-14. Значения MaxBR и MaxCPB определены в таблице A-1 в блоках 1200 бит/с и 1200 битов, соответственно. Поток битов должен удовлетворять этим условиям, по крайней мере, для одного значения SchedSelIdx в диапазоне от 0 до cpb_cnt_minus1.

k) Диапазон компонентов вектора вертикального движения не должен превосходить MaxVmvR в блоках образцов кадров яркости, где MaxVmvR определено в таблице A-1.

l) Диапазон вектора горизонтального движения не должен превосходить диапазона от –2048 до 2047,75 включительно в блоках образцов яркости.

m) Число векторов движения на два последовательных макроблока в порядке декодирования (это также применимо к общему числу от последнего макроблока секции и до первого макроблока следующей секции в порядке декодирования) не должно превышать MaxMvsPer2Mb, где MaxMvsPer2Mb определено в таблице A-1. Число векторов движения для каждого макроблока – это значение переменной MvCnt после завершения процесса intra (внутреннего) или inter (внешнего) предсказания макроблока.

n) Число битов данных macroblock_layer( ) для любого макроблока не превышает 128 + 2048 * ChromaFormatFactor. В зависимости от entropy_coding_mode_flag биты данных macroblock_layer( ) подсчитывают следующим образом:

   – Если entropy_coding_mode_flag равно 0, число битов данных macroblock_layer( ) задано числом битов в структуре синтаксиса макроблока macroblock_layer( ).

   – Иначе (entropy_coding_mode_flag равно 1), число битов данных макроблока macroblock_layer( ) задано числом раз read_bits(1), указанным в пп. 9.3.3.2.2 и 9.3.3.2.3, если синтаксический анализ macroblock_layer( ) связан с этим макроблоком.

Ниже в таблице A-1 указаны границы каждого уровня. Входы в таблице A-1, помеченные как "–", указывают на отсутствие соответствующей границы.

Соответствие конкретному уровню должно быть определено установкой элемента синтаксиса level_idc, равного значению десятикратного номера уровня, указанного в таблице A-1.

Таблица A-1 – Границы уровней

Номер уровня	Макс. скорость обработки макроблока MaxMBPS (Мб/с)	Макс. размер кадра MaxFS (Мб)	Макс. размер буфера декодированного изображения MaxDPB (1024 байта)	Макс. скорость видео-сигнала MaxBR (1000 бит/с или 1200 бит/с)	Макс. размер CPB MaxCPB (1000 бит или 1200 бит)	Диапазон вертикального компонента MV MaxVmvR (образцы кадров яркости)	Мин. коэфф. сжатия MinCR	Макс. число векторов движения на два последовательн. MB MaxMvsPer2Mb
1	1 485	99	148,5	64	175	[–64, +63,75]	2	–
1.1	3 000	396	337,5	192	500	[–128, +127,75]	2	–
1.2	6 000	396	891,0	384	1 000	[–128, +127,75]	2	–
1.3	11 880	396	891,0	768	2 000	[–128, +127,75]	2	–
2	11 880	396	891,0	2 000	2 000	[–128, +127,75]	2	–
2.1	19 800	792	1 782,0	4 000	4 000	[–256, +255,75]	2	–
2.2	20 250	1 620	3 037,5	4 000	4 000	[–256, +255,75]	2	–
3	40 500	1 620	3 037,5	10 000	10 000	[–256, +255,75]	2	32
3.1	108 000	3 600	6 750,0	14 000	14 000	[–512, +511,75]	4	16
3.2	216 000	5 120	7 680,0	20 000	20 000	[–512, +511,75]	4	16
4	245 760	8 192	12 288,0	20 000	25 000	[–512, +511,75]	4	16
4.1	245 760	8 192	12 288,0	50 000	62 500	[–512, +511,75]	2	16
4.2	491 520	8 192	12 288,0	50 000	62 500	[–512, +511,75]	2	16
5	589 824	22 080	41 400,0	135 000	135 000	[–512, +511,75]	2	16
5.1	983 040	36 864	69 120,0	240 000	240 000	[–512, +511,75]	2	16

Уровни с нецелыми номерами в таблице A-1 считают "промежуточными уровнями".

ПРИМЕЧАНИЕ. – Все уровни имеют одинаковый статус, но некоторые приложения могут выбирать для использования только целочисленные уровни.

Информативный п. A.3.3 показывает действие этих границ на скорость кадров для разных примеров форматов изображений.

A.3.2 Границы уровней специальных параметров

a) В потоках битов, соответствующих главным параметрам, время удаления блока доступа 0 должно удовлетворять требованию на число секций в изображении 0: меньше или равно (PicSizeInMbs + MaxMBPS * (t_r(0) – $t_{r,n}$(0))) ÷ SliceRate, где SliceRate – значение, определенное в таблице A-3, которое используют для изображения 0.

b) В потоках битов, соответствующих главным параметрам, разность между последовательными временами удаления блоков доступа n и n – 1 (с n > 0) должна удовлетворять ограничению на число секций в изображении n: меньше или равно MaxMBPS * (t_r(n) – t_r(n – 1)) ÷ SliceRate, где SliceRate – значение, определенное в таблице A-3, которое используют для изображения n.

c) В потоках битов, соответствующих главным параметрам, последовательность набора параметров должна иметь direct_8x8_inference_flag, равное 1, для всех уровней, определенных в таблице A-3.

ПРИМЕЧАНИЕ. – Значение direct_8x8_inference_flag не имеет отношения к основным параметрам, поскольку этот параметр не разрешает использовать секции типа B (определенные в п. A.2.1), а direct_8x8_inference_flag, равное 1, не разрешает это для всех уровней расширенных параметров (определенных в п. A.2.3).

d) В потоках битов, соответствующих главным и расширенным параметрам, последовательность набора параметров должна иметь frame_mbs_only_flag, равное 1, для уровней, определенных в таблице A-3 для главных параметров, а в таблице A-4 – для расширенных параметров.

ПРИМЕЧАНИЕ. – Значение frame_mbs_only_flag равно 1 для всех уровней основных параметров (определенных в п. A.2.1).

e) В потоках битов, соответствующих основным и расширенным параметрам, значение sub_mb_type в макроблоках B не должно быть равно B_Bi_8x4, B_Bi_4x8, или B_Bi_4x4 для уровней, в которых MinLumaBiPredSize показано как 8x8 в таблице A-3 для основных параметров и в таблице A-4 для расширенных параметров.

f) В потоках битов, соответствующих основным и расширенным параметрам, значение ($xInt_{max} - xInt_{min} + 6$) * ($yInt_{max} - yInt_{min} + 6$) <= MaxSubMbRectSize в макроблоках, кодированных с mb_type, равным P_8x8, P_8x8ref0 или B_8x8 для всех вызываемых процедур процесса, определенных в п. 8.4.2.2.1 и используемых, чтобы создать массив предсказанных образцов яркости для одного из списков (списка 0 или списка 1) для каждого субмакроблока 8x8, где NumSubMbPart(sub_mb_type) > 1, а MaxSubMbRectSize определено в таблице A-2 для основных параметров и в таблице A-4 для расширенных параметров, и

- $xInt_{min}$ – это минимальное значение $xInt_L$ среди всех предсказанных образцов яркости для субмакроблока;
- $xInt_{max}$ – это максимальное значение $xInt_L$ среди всех предсказанных образцов яркости для субмакроблока;
- $yInt_{min}$ – это минимальное значение $yInt_L$ среди всех предсказанных образцов яркости для субмакроблока;
- $yInt_{max}$ – это максимальное значение $yInt_L$ среди всех предсказанных образцов яркости для субмакроблока.

A.3.2.1 Границы основных параметров

Таблица A-2 определяет для каждого уровня границы, которые являются специальными для потоков битов, соответствующих основным параметрам. Входы в таблице A-2, помеченные как "–", указывают на отсутствие соответствующей границы.

Таблица A-2 – Границы уровней основных параметров

Номер уровня	MaxSubMbRectSize
1	576
1.1	576
1.2	576
1.3	576
2	576
2.1	576
2.2	576
3	576
3.1	–
3.2	–
4	–
4.1	–
4.2	–
5	–
5.1	–

A.3.2.2 Границы главных параметров

Таблица A-3 определяет для каждого уровня границы, которые являются специальными для потоков битов, соответствующих главным параметрам. Входы в таблице A-3, помеченные как "–", указывают на отсутствие соответствующей границы.

Таблица A-3 – Границы уровней главных параметров

Номер уровня	SliceRate	MinLumaBiPredSize	direct_8x8_inference_flag	frame_mbs_only_flag
1	–	–	–	1
1.1	–	–	–	1
1.2	–	–	–	1
1.3	–	–	–	1
2	–	–	–	1
2.1	–	–	–	–
2.2	–	–	–	–
3	22	–	1	–
3.1	60	8x8	1	–
3.2	60	8x8	1	–
4	60	8x8	1	–
4.1	24	8x8	1	–
4.2	24	8x8	1	1
5	24	8x8	1	1
5.1	24	8x8	1	1

A.3.2.3 Границы расширенных параметров

Таблица A-4 определяет для каждого уровня границы, которые являются специальными для потоков битов, соответствующих расширенным параметрам. Входы в таблице A-4, помеченные как "–", указывают на отсутствие соответствующей границы.

Таблица A-4 – Границы уровней расширенных параметров

Номер уровня	MaxSubMbRectSize	MinLumaBiPredSize	frame_mbs_only_flag
1	576	–	1
1.1	576	–	1
1.2	576	–	1
1.3	576	–	1
2	576	–	1
2.1	576	–	–
2.2	576	–	–
3	576	–	–
3.1	–	8x8	–
3.2	–	8x8	–
4	–	8x8	–
4.1	–	8x8	–
4.2	–	8x8	1
5	–	8x8	1
5.1	–	8x8	1

A.3.3 Влияние границы уровня на скорость кадров (информативное)

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Таблица A-5 – Максимальная скорость кадров (кадров в секунду) для некоторых примеров размеров кадров

Номер уровня:					**1**	**1.1**	**1.2**	**1.3**	**2**	**2.1**	**2.2**
Макс. размер кадра (макроблоков):					**99**	**396**	**396**	**396**	**396**	**792**	**1 620**
Макс. макроблоков/с:					**1 485**	**3 000**	**6 000**	**11 880**	**11 880**	**19 800**	**20 250**
Макс. размер кадра (образцов):					25 344	101 376	101 376	101 376	101 376	202 752	414 720
Макс. образцов/с:					380 160	768 000	1 536 000	3 041 280	3 041 280	5 068 800	5 184 000
Формат	**Ширина яркости**	**Высота яркости**	**Всего MB**	**Образцов яркости**							
SQCIF	**128**	**96**	48	12 288	30,9	62,5	125,0	172,0	172,0	172,0	172,0
QCIF	**176**	**144**	99	25 344	15,0	30,3	60,6	120,0	120,0	172,0	172,0
QVGA	**320**	**240**	300	76 800	–	10,0	20,0	39,6	39,6	66,0	67,5
525 SIF	**352**	**240**	330	84 480	–	9,1	18,2	36,0	36,0	60,0	61,4
CIF	**352**	**288**	396	101 376	–	7,6	15,2	30,0	30,0	50,0	51,1
525 HHR	**352**	**480**	660	168 960	–	–	–	–	–	30,0	30,7
625 HHR	**352**	**576**	792	202 752	–	–	–	–	–	25,0	25,6
VGA	**640**	**480**	1 200	307 200	–	–	–	–	–	–	16,9
525 4SIF	**704**	**480**	1 320	337 920	–	–	–	–	–	–	15,3
525 SD	**720**	**480**	1 350	345 600	–	–	–	–	–	–	15,0
4CIF	**704**	**576**	1 584	405 504	–	–	–	–	–	–	12,8
625 SD	**720**	**576**	1 620	414 720	–	–	–	–	–	–	12,5
SVGA	**800**	**600**	1 900	486 400	–	–	–	–	–	–	–
XGA	**1024**	**768**	3 072	786 432	–	–	–	–	–	–	–
720p HD	**1280**	**720**	3 600	921 600	–	–	–	–	–	–	–
4VGA	**1280**	**960**	4 800	1 228 800	–	–	–	–	–	–	–
SXGA	**1280**	**1024**	5 120	1 310 720	–	–	–	–	–	–	–
525 16SIF	**1408**	**960**	5 280	1 351 680	–	–	–	–	–	–	–
16CIF	**1408**	**1152**	6 336	1 622 016	–	–	–	–	–	–	–
4SVGA	**1600**	**1200**	7 500	1 920 000	–	–	–	–	–	–	–
1080 HD	**1920**	**1088**	8 160	2 088 960	–	–	–	–	–	–	–
2Kx1K	**2048**	**1024**	8 192	2 097 152	–	–	–	–	–	–	–
4XGA	**2048**	**1536**	12 288	3 145 728	–	–	–	–	–	–	–
16VGA	**2560**	**1920**	19 200	4 915 200	–	–	–	–	–	–	–
3616x1536 (2.35:1)	**3616**	**1536**	21 696	5 554 176	–	–	–	–	–	–	–
3672x1536 (2.39:1)	**3680**	**1536**	22 080	5 652 480	–	–	–	–	–	–	–
4Kx2K	**4096**	**2048**	32 768	8 388 608	–	–	–	–	–	–	–
4096x2304 (16:9)	**4096**	**2304**	36 864	9 437 184	–	–	–	–	–	–	–

Таблица A-5 (продолжение) – Максимальная скорость кадров (кадров в секунду) для некоторых примеров размеров кадров

Номер уровня:					**3**	**3,1**	**3,2**	**4**	**4,1**	**4,2**
Макс. размер кадра (макроблоков):					**1 620**	**3 600**	**5 120**	**8 192**	**8 192**	**8 192**
Макс. макроблоков/с:					**40 500**	**108 000**	**216 000**	**245 760**	**245 760**	**589 824**
Макс. размер кадра (образцов):					414 720	921 600	1 310 720	2 097 152	2 097 152	2 097 152
Макс. образцов/с:					10 368 000	27 648 000	55 296 000	62 914 560	62 914 560	125 829 120
Формат	**Ширина яркости**	**Высота яркости**	**Всего MB**	**Образцов яркости**						
SQCIF	**128**	**96**	48	12 288	172,0	172,0	172,0	172,0	172,0	172,0
QCIF	**176**	**144**	99	25 344	172,0	172,0	172,0	172,0	172,0	172,0
QVGA	**320**	**240**	300	76 800	135,0	172,0	172,0	172,0	172,0	172,0
525 SIF	**352**	**240**	330	84 480	122,7	172,0	172,0	172,0	172,0	172,0
CIF	**352**	**288**	396	101 376	102,3	172,0	172,0	172,0	172,0	172,0
525 HHR	**352**	**480**	660	168 960	61,4	163,6	172,0	172,0	172,0	172,0
625 HHR	**352**	**576**	792	202 752	51,1	136,4	172,0	172,0	172,0	172,0
VGA	**640**	**480**	1 200	307 200	33,8	90,0	172,0	172,0	172,0	172,0
525 4SIF	**704**	**480**	1 320	337 920	30,7	81,8	163,6	172,0	172,0	172,0
525 SD	**720**	**480**	1 350	345 600	30,0	80,0	160,0	172,0	172,0	172,0
4CIF	**704**	**576**	1 584	405 504	25,6	68,2	136,4	155,2	155,2	172,0
625 SD	**720**	**576**	1 620	414 720	25,0	66,7	133,3	151,7	151,7	172,0
SVGA	**800**	**600**	1 900	486 400	–	56,8	113,7	129,3	129,3	172,0
XGA	**1024**	**768**	3 072	786 432	–	35,2	70,3	80,0	80,0	160,0
720p HD	**1280**	**720**	3 600	921 600	–	30,0	60,0	68,3	68,3	136,5
4VGA	**1280**	**960**	4 800	1 228 800	–	–	45,0	51,2	51,2	102,4
SXGA	**1280**	**1024**	5 120	1 310 720	–	–	42,2	48,0	48,0	96,0
525 16SIF	**1408**	**960**	5 280	1 351 680	–	–	–	46,5	46,5	93,1
16CIF	**1408**	**1152**	6 336	1 622 016	–	–	–	38,8	38,8	77,6
4SVGA	**1600**	**1200**	7 500	1 920 000	–	–	–	32,8	32,8	65,5
1080 HD	**1920**	**1088**	8 160	2 088 960	–	–	–	30,1	30,1	60,2
2Kx1K	**2048**	**1024**	8 192	2 097 152	–	–	–	30,0	30,0	60,0
4XGA	**2048**	**1536**	12 288	3 145 728	–	–	–	–	–	–
16VGA	**2560**	**1920**	19 200	4 915 200	–	–	–	–	–	–
3616x1536 (2.35:1)	**3616**	**1536**	21 696	5 554 176	–	–	–	–	–	–
3672x1536 (2.39:1)	**3680**	**1536**	22 080	5 652 480	–	–	–	–	–	–
4Kx2K	**4096**	**2048**	32 768	8 388 608	–	–	–	–	–	–
4096x2304 (16:9)	**4096**	**2304**	36 864	9 437 184	–	–	–	–	–	–

Таблица A-5 (окончание) – Максимальная скорость кадров (кадров в секунду) для некоторых примеров размеров кадров

Номер уровня:					**5**	**5,1**
Макс. размер кадра (макроблоков):					**21 696**	**36 864**
Макс. макроблоков/с:					**589 824**	**983 040**
Макс. размер кадра (образцов):					5 554 176	9 437 184
Макс. образцов/с:					150 994 944	251 658 240
Формат	**Ширина яркости**	**Высота яркости**	**Всего MB**	**Образцов яркости**		
SQCIF	**128**	**96**	48	12 288	172,0	172,0
QCIF	**176**	**144**	99	25 344	172,0	172,0
QVGA	**320**	**240**	300	76 800	172,0	172,0
525 SIF	**352**	**240**	330	84 480	172,0	172,0
CIF	**352**	**288**	396	101 376	172,0	172,0
525 HHR	**352**	**480**	660	168 960	172,0	172,0
625 HHR	**352**	**576**	792	202 752	172,0	172,0
VGA	**640**	**480**	1 200	307 200	172,0	172,0
525 4SIF	**704**	**480**	1 320	337 920	172,0	172,0
525 SD	**720**	**480**	1 350	345 600	172,0	172,0
4CIF	**704**	**576**	1 584	405 504	172,0	172,0
625 SD	**720**	**576**	1 620	414 720	172,0	172,0
SVGA	**800**	**600**	1 900	486 400	172,0	172,0
XGA	**1024**	**768**	3 072	786 432	172,0	172,0
720p HD	**1280**	**720**	3 600	921 600	163,8	172,0
4VGA	**1280**	**960**	4 800	1 228 800	122,9	172,0
SXGA	**1280**	**1024**	5 120	1 310 720	115,2	172,0
525 16SIF	**1408**	**960**	5 280	1 351 680	111,7	172,0
16CIF	**1408**	**1152**	6 336	1 622 016	93,1	155,2
4SVGA	**1600**	**1200**	7 500	1 920 000	78,6	131,1
1080 HD	**1920**	**1088**	8 160	2 088 960	72,3	120,5
2Kx1K	**2048**	**1024**	8 192	2 097 152	72,0	120,0
4XGA	**2048**	**1536**	12 288	3 145 728	48,0	80,0
16VGA	**2560**	**1920**	19 200	4 915 200	30,7	51,2
3616x1536 (2.35:1)	**3616**	**1536**	21 696	5 554 176	27,2	45,3
3672x1536 (2.39:1)	**3680**	**1536**	22 080	5 652 480	26,7	44,5
4Kx2K	**4096**	**2048**	32 768	8 388 608	–	30,0
4096x2304 (16:9)	**4096**	**2304**	36 864	9 437 184	–	26,7

Необходимо отметить следующее:

– Эта Рекомендация | Международный стандарт представляет спецификацию с переменным размером кадров. Конкретный размер кадра в таблице A-5 является просто пояснительным примером.

– Указанное в таблице A-5 значение "525" относится к типичному использованию аналогового сканирования 525 линий (из которых примерно 480 линий содержат область видимого изображения), а "625" относится к использованию аналогового сканирования 625 линий (из которых примерно 576 линий содержат область видимого изображения).

– XGA также известно как (aka) XVGA, 4SVGA aka UXGA, 16XGA aka 4Kx3K, CIF aka 625 SIF, 625 HHR aka 2CIF aka half 625 D-1, aka half 625 ITU-R BT.601, 525 SD aka 525 D-1 aka 525 ITU-R BT.601, 625 SD aka 625 D-1 aka 625 ITU-R BT.601.

– Приведенные скорости кадров справедливы для режимов последовательного сканирования. Скорости кадров также справедливы для чередующегося кодирования видеосигнала для тех случаев, когда высота кадра кратна 32.

Приложение B

Формат потока байтов

(Это Приложение является составной частью данной Рекомендации | Международного стандарта)

В этом Приложении определены синтаксис и семантика формата потока байтов, определенного для использования приложениями, которые доставляют некоторые или все потоки блоков NAL в виде упорядоченного потока байтов или битов, внутри которых расположены границы блоков NAL, необходимые для возможности различить шаблоны данных, как, например, в системах согласно Рекомендации МСЭ-Т H.222.0 | ИСО/МЭК 13818-1 или Рекомендации МСЭ-Т H.320. Для доставки с ориентацией на биты порядок битов в формате потока байтов определен при старте битом MSB первого байта, предшествующего LSB первого байта, за которым следует MSB второго байта и т. д.

Формат потока байтов состоит из последовательности потока байтов со структурой синтаксиса блоков NAL. Каждый поток байтов со структурой синтаксиса блоков NAL содержит один префикс кода старта, за которым следует одна структура синтаксиса nal_unit(NumBytesInNALunit). Поток может также содержать (при некоторых обстоятельствах) дополнительный элемент синтаксиса zero_byte, а также один или более дополнительных элементов синтаксиса trailing_zero_8bits. Если это – первый поток байтов блоков NAL в потоке битов, он может также содержать один или более элементов синтаксиса leading_zero_8bits.

B.1 Синтаксис и семантика блоков NAL потока байтов

B.1.1 Синтаксис блоков NAL потока байтов

byte_stream_nal_unit(NumBytesInNALunit) {	C	Дескриптор
while(next_bits(24) != 0x000001 && next_bits(32) != 0x00000001)		
leading_zero_8bits /* равно 0x00 */		f(8)
if(next_bits(24) != 0x000001)		
zero_byte /* равно 0x00 */		f(8)
if(more_data_in_byte_stream( )) {		
start_code_prefix_one_3bytes /* равно 0x000001 */		f(24)
nal_unit(NumBytesInNALunit)		
}		
while(more_data_in_byte_stream( ) && next_bits(24) != 0x000001 && next_bits(32) != 0x00000001)		
trailing_zero_8bits /* равно 0x00 */		f(8)
}		

B.1.2 Семантика блоков NAL потока байтов

Порядок блоков NAL в потоке байтов должен следовать порядку декодирования блоков NAL, находящихся в потоке байтов (см. п. 7.4.1.2). Содержание каждого блока NAL потока байтов объединяют с тем же блоком доступа, что и блок NAL, который содержится среди блоков NAL потока байтов (см. п. 7.4.1.2.3).

leading_zero_8bits – байт, равный 0x00.

ПРИМЕЧАНИЕ. – Элемент синтаксиса leading_zero_8bits может быть представлен только в первом блоке NAL потока байтов из потока битов, так как (как показано на диаграмме синтаксиса в п. B.1.1) любые байты, равные 0x00, которые следуют за структурой синтаксиса блока NAL и предшествуют 4-байтовой последовательности 0x00000001 (что можно интерпретировать как zero_byte с последующим start_code_prefix_one_3bytes), должны рассматриваться как элементы синтаксиса trailing_zero_8bits, составляющие часть предшествующих блоков NAL потока байтов.

zero_byte – единичный байт, равный 0x00.

Если любое из следующих условий выполнено, то должен быть представлен элемент синтаксиса zero_byte:

– значение nal_unit_type в the nal_unit( ) равно 7 (установка параметров последовательности) или 8 (установка параметров изображения);

– структура синтаксиса блока NAL в потоке байтов содержит первый блок NAL из блока доступа в порядке декодирования, как это определено в п. 7.4.1.2.3.

start_code_prefix_one_3bytes – последовательность из 3 байтов фиксированной величины, равной 0x000001. Этот элемент синтаксиса называют префиксом кода старта (запуска).

trailing_zero_8bits – байт, равный 0x00.

B.2 Процесс декодирования блоков NAL потока байтов

Вход в этот процесс состоит из упорядоченного потока байтов, содержащего последовательность структуры синтаксиса блоков NAL потока байтов.

Выход этого процесса состоит из последовательности структур синтаксиса блоков NAL.

В начале процесса декодирования декодер инициализирует свою текущую позицию в потоке байтов от начала этого потока. Далее декодер извлекает и отбрасывает каждый элемент синтаксиса leading_zero_8bits (если такой присутствует), продвигая текущую позицию в потоке байтов вперед каждый раз на один байт, пока текущая позиция в потоке байтов не станет такой, что следующие четыре байта в потоке битов сформируют 4-байтовую последовательность 0x00000001.

Далее декодер повторно выполняет следующий пошаговый процесс, чтобы извлечь и декодировать каждую структуру синтаксиса блока NAL в потоке байтов, пока не будет достигнут конец (что устанавливают с помощью не определяемых здесь средств) и декодирован последний блок NAL в потоке байтов:

1. Если следующие 4 байта в потоке битов формируют 4-байтовую последовательность 0x00000001, то следующий байт в потоке байтов (который является элементом синтаксиса zero_byte) извлекают и отбрасывают, а текущую позицию в потоке байтов устанавливают равной позиции байта, следующего за отброшенным.
2. Следующую 3-байтовую последовательность в потоке байтов (которой является start_code_prefix_one_3bytes) извлекают и отбрасывают, а текущую позицию в потоке байтов устанавливают равной позиции байта, следующего за 3-байтовой последовательностью.
3. Значение NumBytesInNALunit устанавливают равным числу байтов, которое начинается с байта на текущей позиции в потоке байтов и продолжается до (и включая) последний байт, который предшествует положению с любыми из следующих условий:
    a. Последующая байт-ориентированная 3-байтовая последовательность равна 0x000000, или
    b. Последующая байт-ориентированная 3-байтовая последовательность равна 0x000001, или
    c. Конец потока байтов, что было установлено не определяемыми здесь средствами.
4. NumBytesInNALunit байтов удалены из потока битов, а текущая позиция в потоке байтов продвинута на NumBytesInNALunit байтов. Эту последовательность байтов nal_unit(NumBytesInNALunit) декодируют, используя процесс декодирования блока NAL.
5. Если текущая позиция в потоке байтов не является концом потока байтов (что устанавливают не определяемые здесь средства) и следующие байты в потоке байтов не начинаются с 3-байтовой последовательности, равной 0x000001, а последующие байты в потоке байтов не начинаются с 4-байтовой последовательности, равной 0x00000001, то декодер извлекает и отбрасывает каждый элемент синтаксиса trailing_zero_8bits, продвигающий текущую позицию в потоке байтов вперед каждый раз на один байт, до тех пор, пока текущая позиция в потоке байтов не станет такой, что следующие байты в потоке байтов не сформируют 4-байтовую последовательность 0x00000001, или пока не появится конец потока байтов (что устанавливают не определяемые здесь средства).

B.3 Восстановление декодером выравнивания байтов (информативное)

Этот пункт не является составной частью данной Рекомендации | Международного стандарта.

Многие приложения поставляют данные в декодер способом, который, по сути, является выровненным по байтам и поэтому не нуждается в описанной в этом пункте процедуре обнаружения ориентированного бита для выравнивания байтов.

Декодер называют выровненным по байтам с потоком битов, если декодер в состоянии определить, являются ли позиции данных в потоке битов выровненными по байтам. Если у декодера отсутствует выравнивание байтов относительно потока байтов кодера, то декодер может проверить входящий поток битов по бинарному шаблону '00000000 00000000 00000000 00000001' (31 последовательный бит, равный 0, за которым следует бит, равный 1). Бит, непосредственно следующий за этим шаблоном, – это первый бит выровненного байта, за которым следует префикс кода старта. После проверки этого шаблона декодер будет выровненным по байтам с кодером и будет занимать позицию в начале блока NAL в потоке байтов.

Как только произойдет выравнивание по байтам с кодером, декодер может проверить входящий поток байтов на следующие 3-байтовые последовательности: 0x000001 и 0x000003.

Если 3-байтовая последовательность 0x000001 обнаружена, это и есть префикс кода старта.

Если обнаружена 3-байтовая последовательность 0x000003, третий байт (0x03) – emulation_prevention_three_byte должен быть отброшен, как это определено в п. 7.4.1.

Процедура обнаружения выравнивания байтов, описанная в этом пункте, функционально эквивалентна отысканию последовательности байтов для трех следующих друг за другом нулевых байтов (0x000000), которые начинаются с любой позиции выравнивания. Обнаружение такого шаблона указывает, что следующий ненулевой байт означает конец префикса кода старта. (Поскольку согласованный поток байтов не может содержать более 23 последовательных битов с нулевыми значениями без включения 31 или более последовательных нулевых битов, позволяющих обнаружить сигнал 0x000000 относительно любой начальной позиции выравнивания.) Первый ненулевой бит в следующем ненулевом байте – это последний бит выровненного байта и последний бит префикса кода старта.

Приложение C

Гипотетический контрольный декодер

(Это Приложение является составной частью данной Рекомендации | Международного стандарта)

Это Приложение определяет гипотетический контрольный декодер (HRD) и его использование для контроля соответствия потока битов и декодера.

Под действие HRD попадают два типа потоков битов для проверки на соответствие этой Рекомендации | Международному стандарту. Первый тип потока битов, называемый потоком битов Типа I, это поток блоков NAL, содержащий только блоки NAL VCL и блоки NAL данных заполнения для всех блоков доступа в потоке битов. Второй тип, называемый потоком битов Типа II, содержит вдобавок к блокам NAL VCL и блокам NAL данных заполнения для всех блоков доступа в потоке битов, по крайней мере, одно из следующих:

- дополнительные не блоки NAL VCL и иные, чем блоки NAL данных заполнения;
- все элементы синтаксиса leading_zero_8bits, zero_byte, start_code_prefix_one_3bytes и trailing_zero_8bits, которые формируют поток блоков NAL из потока байтов (как определено в Приложении B).

Рисунок C-1 показывает точки соответствия типов потоков битов, которые контролирует HRD.

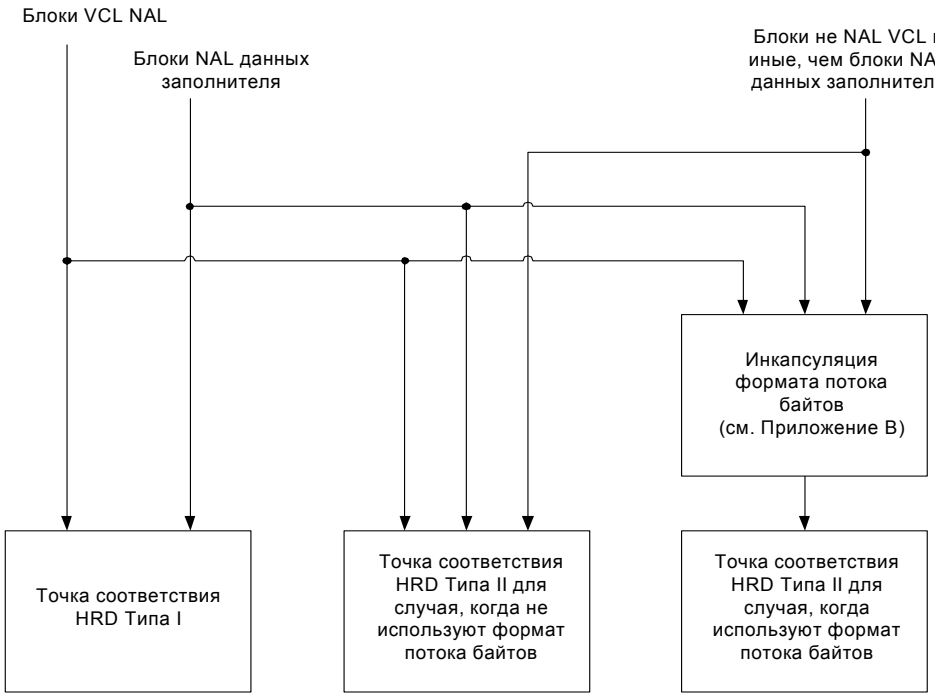

Рисунок C-1 – Структура потоков байтов и потоков блоков NAL для проверки соответствия с помощью HRD

Элементы синтаксиса не блоков NAL VCL (или их значения для некоторых элементов синтаксиса), которые требуются для HRD, рассмотрены в пунктах семантики в разделе 7 и в Приложениях D и E.

Используют два типа набора параметров HRD. Наборы параметров HRD сообщают с помощью удобной визуальной информации (как описано в пп. E.1 и E.2), к какой части структуры синтаксиса относится последовательность набора параметров.

Чтобы проверить соответствие потока битов с помощью HRD, все наборы параметров последовательности, а также наборы параметров изображения относят к блокам NAL VCL, а соответствующий период буферизации и сообщения SEI о синхронизации изображения должны быть переданы в HRD своевременно либо в потоке битов (не блоками NAL VCL), либо другими средствами, не описанными в этой Рекомендации | Международном стандарте.

В Приложениях C, D и E спецификация для "присутствия" не блоков NAL VCL также выполняется, если эти блоки NAL (или некоторые из них) поступают в декодеры (или в HRD) с помощью других средств, не описанных в

этой Рекомендации | Международном стандарте. Поэтому при подсчете битов учитывают только те биты, которые действительно присутствуют в потоке битов.

ПРИМЕЧАНИЕ. – В качестве примера синхронизацию не блока NAL VCL (которую передают средствами иными, чем для блоков NAL, присутствующими в потоке битов) можно выполнить, указав на два пункта в потоке битов, между которыми не блок NAL VCL мог бы присутствовать, если бы кодер решил передавать его в потоке битов.

Если содержание не блока NAL VCL передают приложению некоторыми средствами иными, чем присутствием в потоке битов, то представление содержания не блока NAL VCL не требует использования того же синтаксиса, который описан в этом Приложении.

ПРИМЕЧАНИЕ. – Если информация HRD содержится в потоке битов, то возможно проверить соответствие потока битов требованиям этого пункта, основываясь исключительно на информации, которая имеется в потоке битов. Если информация HRD не содержится в потоке битов, как в случае для всех "автономных" потоков битов Типа I, соответствие можно проверить, если данные HRD поставлены какими-то другими средствами, не рассмотренными в этой Рекомендации | Международном стандарте.

HRD состоит из буфера кодированного изображения (CPB), процесса мгновенного декодирования, буфера декодированного изображения (DPB) и разделения на кадры или поля (кадрирование) на выходе, как показано на рисунке C-2.

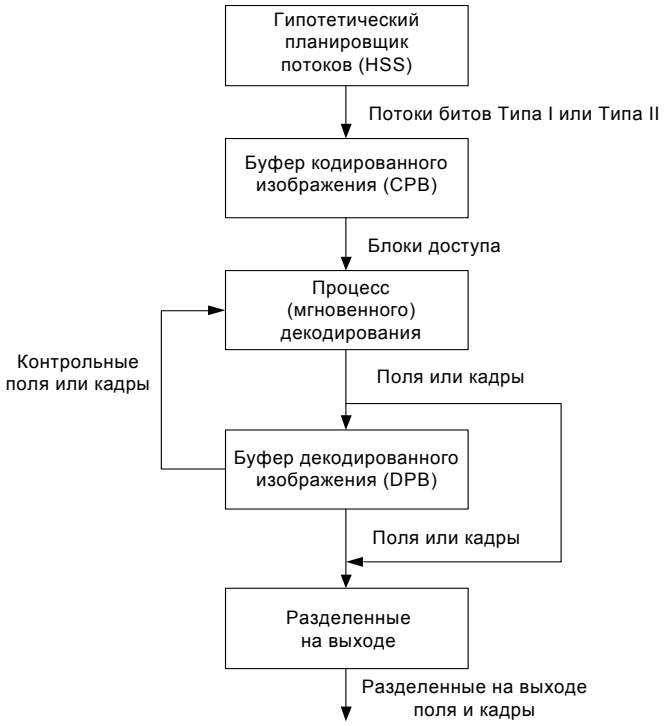

Рисунок C-2 – Модель буфера HRD

Размер CPB (число битов) определено CpbSize[SchedSelIdx] в Приложении E. Размер DPB (число буферов кадров) определено как Max(1, max_dec_frame_buffering) в Приложении E.

Декодер HRD работает следующим образом. Данные, объединенные с блоками доступа, которые поступают в CPB согласно определенному расписанию, расставляют с помощью HSS. Данные, объединенные с каждым блоком доступа, удаляют и мгновенно декодируют процессом мгновенного декодирования в моменты удаления из CPB. Каждое декодированное изображение помещают в DPB в моменты удаления из CPB, если это не выход CPB в момент удаления и если это неконтрольное изображение. Если изображение размещено в DPB, его позднее удаляют из DPB во время выхода DPB или во время, при котором его помечают как "не использовано для контроля".

Работа CPB описана в п. C.1. Работа мгновенного декодера описана в разделах 8 и 9. Работа DPB описана в п. C.2. Выход операции кадрирования описан в п. C.2.2.

Информация HSS и HRD, которая касается числа перечисленных расписаний доставки и их связи со скоростями битов и размерами буфера, описана в пп. E.1.1, E.1.2, E.2.1 и E.2.2. HRD инициализируют, как это определено сообщением SEI о периоде буферизации в пп. D.1.1 и D.2.1. Моменты удаления блоков доступа из CPB и моменты удаления на выходе DPB указаны в сообщении SEI о синхронизации изображения, как это определено в пп. D.1.2 и

D.2.2. Вся информация о распределении во времени, которая касается специального блока доступа, должна поступать до времени удаления из CPB блока доступа.

HRD используют для контроля соответствия потоков битов и декодеров, как это определено в пп. C.3 и C.4, соответственно.

ПРИМЕЧАНИЕ. – Хотя соответствие гарантировано в предположении, что все скорости кадров и тактовых импульсов, использованных для генерации потока битов, точно соответствуют сигналам в потоке битов, в реальной системе каждое из них может отклоняться от этих сигналов или установленных значений.

Вся арифметика в этом Приложении выполнена над реальными значениями, с тем чтобы избежать размножения ошибок округления. Например, число битов в CPB непосредственно перед и после удаления блока доступа не обязательно является целым числом.

Переменную t_c называют тактом системных часов (clock tick) и находят следующим образом:

$$t_c = num\_units\_in\_tick \div time\_scale. \quad (C\text{-}1)$$

Следующие пункты определены для выражения ограничений в этом Приложении:

– Положим, что блок доступа n – это n-й блок доступа в порядке декодирования с первым блоком доступа, который является блоком доступа 0.

– Положим, что изображение n – исходное кодированное изображение или исходное декодированное изображение блока доступа n.

C.1 Работа буфера кодированного изображения (CPB)

Спецификация в этом пункте применима независимо от каждой установки параметров CPB и от точек соответствия для обоих типов, Типа I и Типа II, показанных на рисунке C-1.

C.1.1 Синхронизация поступления потока битов

HRD можно инициализировать в любой из периодов сообщения SEI о периоде буферизации. Перед инициализацией CPB пусто.

ПРИМЕЧАНИЕ. – После инициализации HRD не инициализируют снова при последующем сообщении SEI о периоде буферизации.

Каждый блок доступа считают блоком доступа n, где число n идентифицирует конкретный блок доступа. Блок доступа, связанный с сообщением SEI о периоде буферизации, при котором происходит инициализация CPB, считают блоком доступа 0. Это значение возрастает на 1 для каждого последующего блока доступа в порядке декодирования.

Время, когда первый бит блока доступа n начинает поступать в CPB, считают начальным временем поступления $t_{ai}(n)$.

Начальное время поступления блоков доступа находят следующим образом:

– Если блок доступа – это блок доступа 0, $t_{ai}(0) = 0$.

– Иначе (блок доступа – это блок доступа n с $n > 0$), применяют следующее:

  – Если cbr_flag[SchedSelIdx] равно 1, начальное время поступления блока доступа n равно конечному времени поступления (которое находят ниже) блока доступа n – 1, т. е.

$$t_{ai}(n) = t_{af}(n - 1). \quad (C\text{-}2)$$

  – Иначе (cbr_flag[SchedSelIdx] равно 0), начальное время поступления блока доступа n находят как

$$t_{ai}(n) = Max(t_{af}(n - 1), t_{ai,earliest}(n)), \quad (C\text{-}3)$$

где $t_{ai,earliest}(n)$ находят следующим образом:

  – Если блок доступа n – не первый блок доступа последующего периода буферизации, то $t_{ai,earliest}(n)$ находят как

$$t_{ai,earliest}(n) = t_{r,n}(n) - (initial\_cpb\_removal\_delay[SchedSelIdx] + initial\_cpb\_removal\_delay\_offset[SchedSelIdx]) \div 90000, \quad (C\text{-}4)$$

где $t_{r,n}(n)$ – номинальное время удаления блока доступа n из CPB, как это определено в п. C.1.2, а initial_cpb_removal_delay[SchedSelIdx] и initial_cpb_removal_delay_offset[SchedSelIdx] определены в предыдущем сообщении SEI о периоде буферизации.

  – Иначе (блок доступа n – не первый блок доступа в последующем периоде буферизации), $t_{ai,earliest}(n)$ находят как

$$t_{ai,earliest}(n) = t_{r,n}(n) - (initial\_cpb\_removal\_delay[SchedSelIdx] \div 90000) \quad (C\text{-}5)$$

со значением initial_cpb_removal_delay[SchedSelIdx], определенным в сообщении SEI о периоде буферизации, связанном с блоком доступа n.

Конечное время поступления блока доступа n находят как

$$t_{af}(n) = t_{ai}(n) + b(n) \div BitRate[SchedSelIdx], \quad (C\text{-}6)$$

где b(n) – размер в битах блока доступа n, считая биты блоков NAL VCL и блоков NAL с данными заполнения для точек соответствия Типу I или все биты потока битов Типа II для точек соответствия Типу II, где точки соответствия Типу I и Типу II показаны на рисунке C-1.

Значения SchedSelIdx, BitRate[SchedSelIdx] и CpbSize[SchedSelIdx] ограничены следующим образом:

– Если блок доступа n и блок доступа n – 1 – части разных кодированных видеопоследовательностей, а содержание действующих наборов параметров последовательности двух кодированных видеопоследовательностей различаются, то HSS выбирает значение SchedSelIdx1 из SchedSelIdx среди значений SchedSelIdx, обеспечивая для кодированной видеопоследовательности, содержащей блок доступа n, эти результаты в значениях BitRate[SchedSelIdx1] или CpbSize[SchedSelIdx1] для второй из двух кодированных видеопоследовательностей (которая содержит блок доступа n). Значения BitRate[SchedSelIdx1] или CpbSize[SchedSelIdx1] могут отличаться от значений BitRate[SchedSelIdx0] или CpbSize[SchedSelIdx0] для значения SchedSelIdx0 из SchedSelIdx, которое было использовано в кодированной видеопоследовательности с блоком доступа n – 1.

– Иначе HSS продолжает работу с предыдущими значениями SchedSelIdx, BitRate[SchedSelIdx] и CpbSize[SchedSelIdx].

Если HSS выбирает значения BitRate[SchedSelIdx] или CpbSize[SchedSelIdx], которые отличаются от первоначальных значений блока доступа, применяют следующее:

– переменная BitRate[SchedSelIdx] вступает в действие в момент $t_{ai}(n)$;

– переменная CpbSize[SchedSelIdx] вступает в действие следующим образом:

  – Если новое значение CpbSize[SchedSelIdx] превосходит прежний размер CPB, оно вступает в действие в момент $t_{ai}(n)$.

  – Иначе новое значение CpbSize[SchedSelIdx] вступает в действие в момент $t_r(n)$.

C.1.2 Синхронизация удаления кодированного изображения

Для блока доступа 0 номинальное время удаления блока доступа из CPB определено как

$$t_{r,n}(0) = initial\_cpb\_removal\_delay[SchedSelIdx] \div 90000. \quad (C\text{-}7)$$

Для первого блока доступа в период буферизации, при котором не происходит инициализации HRD, номинальное время удаления блока доступа из CPB определяют как

$$t_{r,n}(n) = t_{r,n}(n_b) + t_c * cpb\_removal\_delay(n), \quad (C\text{-}8)$$

где $t_{r,n}(n_b)$ – номинальное время удаления первого блока доступа предыдущего периода буферизации, а cpb_removal_delay(n) – значение cpb_removal_delay, определенное в сообщении SEI о синхронизации изображения, связанного с блоком доступа n.

Если блок доступа n – первый блок доступа периода буферизации, то n_b устанавливают равным значению n во время удаления блока доступа n.

Номинальное время удаления $t_{r,n}(n)$ блока доступа n, который не является первым блоком доступа периода буферизации, задают как

$$t_{r,n}(n) = t_{r,n}(n_b) + t_c * cpb\_removal\_delay(n), \quad (C\text{-}9)$$

где $t_{r,n}(n_b)$ – номинальное время удаления первого блока доступа текущего периода буферизации, а cpb_removal_delay(n) – значение cpb_removal_delay, определенное в сообщении SEI о синхронизации изображения, связанного с блоком доступа n.

Время удаления блока доступа n определено следующим образом:

– Если low_delay_hrd_flag равно 0 или $t_{r,n}(n) >= t_{af}(n)$, то время удаления блока доступа n определено как

$$t_r(n) = t_{r,n}(n). \quad (C\text{-}10)$$

– Иначе (low_delay_hrd_flag равно 1 и $t_{r,n}(n) < t_{af}(n)$), время удаления блока доступа n определено как

$$t_r(n) = t_{r,n}(n) + t_c * Ceil((t_{af}(n) – t_{r,n}(n)) \div t_c). \quad (C\text{-}11)$$

ПРИМЕЧАНИЕ. – Последний случай указывает, что размер блока доступа n, b(n) настолько велик, что его невозможно удалить за номинальное время удаления.

C.2 Работа буфера декодированного изображения (DPB)

Буфер декодированного изображения содержит буферы кадров. Каждый из буферов кадров может содержать декодированный кадр, декодированную дополнительную пару полей или единственное (непарное) декодированное поле, которые помечены как "использовано для контроля" (контрольных изображений) или удерживаются для будущего выхода (измененного порядка или задержанного изображения). Перед инициализацией DPB пусто (заполнение DPB устанавливают на нуль). Все последующие разделы этого пункта имеют дело со временем $t_r(n)$ и с перечисленными последовательностями.

C.2.1 Декодирование промежутков в frame_num и хранение "несуществующих" кадров

В случае применения промежутки в значении frame_num обнаруживают процессом декодирования, а генерированные кадры отмечают и вводят в DPB, как это определено ниже.

Промежутки в frame_num обнаруживают процессом декодирования, а генерированные кадры отмечают, как это определено в п. 8.2.5.2.

После отметки каждого сгенерированного кадра каждое изображение m, отмеченное в процессе отметки в "раздвижном окне" как "не использованное для контроля", удаляют из DPB, если буфер также отмечен как "несуществующий" или если выходное время DPB меньше чем или равно времени удаления CPB текущего изображения n, т. е. $t_{o,dpb}(m) <= t_r(n)$. Если кадр или последнее поле в кадре буфера удалено из DPB, то заполнение DPB определяют единицей. "Несуществующий" сгенерированный кадр вводят в DPB, а заполнение DPB увеличивается на единицу.

C.2.2 Декодирование и выход изображения

Изображение n декодируют, а время выхода DPB $t_{o,dpb}(n)$ находят как

$$t_{o,dpb}(n) = t_r(n) + t_c * dpb\_output\_delay(n). \quad (C-12)$$

Выход текущего изображения определен следующим образом:

– Если $t_{o,dpb}(n) = t_r(n)$, то текущее изображение – это выход.
  ПРИМЕЧАНИЕ. – Если текущее изображение – это контрольное изображение, оно будет сохранено в DPB.

– Иначе ($t_{o,dpb}(n) > t_r(n)$), текущее изображение – это выход с задержкой, и изображение будет сохранено в DPB (как это определено в п. C.2.4). Выход этого изображения происходит во время $t_{o,dpb}(n)$, если только не будет указано, что это – не выход с помощью декодирования или вмешательства команды no_output_of_prior_pics_flag, равной 1, за время, предшествующее $t_{o,dpb}(n)$.

Выход изображения должен быть разделен на кадры (или поля), используя прямоугольник кадрирования, определенный установкой параметров для этой последовательности.

Если изображение n – это изображение, которое является выходом, но не последнее изображение в потоке битов, который представляет выход, то значение $\Delta t_{o,dpb}(n)$ определяют как:

$$\Delta t_{o,dpb}(n) = t_{o,dpb}(n_n) – t_{o,dpb}(n), \quad (C-13)$$

где n_n указывает на изображение, которое следует за изображением n в порядке выхода.

Декодированное изображение временно сохраняют (но не в DPB).

C.2.3 Удаление изображения из DPB перед возможным введением текущего изображения

Удаление изображения из DPB перед возможным введением текущего изображения происходит следующим образом:

– Если декодированное изображение – это IDR изображение, то применяют следующее:
  – Все контрольные изображения в DPB отмечают как "не использовано для контроля", как это определено в пп. 8.2.5.3 и 8.2.5.4.
  – Если IDR изображение – это не первое декодированное IDR изображение, а значения PicWidthInMbs, или FrameHeightInMbs, или max_dec_frame_buffering, найденные из действующей установки параметров последовательности, отличаются от значений PicWidthInMbs, или FrameHeightInMbs, или max_dec_frame_buffering, найденных из установки параметров последовательности, которая была действующей соответственно для предыдущей последовательности, то в HRD значение no_output_of_prior_pics_flag принимают равным 1 независимо от действительного значения no_output_of_prior_pics_flag.
    ПРИМЕЧАНИЕ. – При использовании декодера следует попытаться управлять изменениями размеров кадра или DPB, но более осторожно, чем в HRD по отношению к изменениям в PicWidthInMbs или FrameHeightInMbs.
  – Если no_output_of_prior_pics_flag равно 1 или предполагают равным 1, то все буферы кадров в DPB опустошают без выдачи на выход изображений, которые там содержались, а заполнение DPB устанавливают на 0.

– Иначе (декодированное изображение – это не IDR изображение), применяют следующее:

– Если заголовок секции текущего изображения включает memory_management_control_operation, равное 5, то все контрольные изображения в DPB помечают как "не использовано для контроля".

– Иначе (заголовок секции текущего изображения не включает memory_management_control_operation, равное 5), активируют процесс разметки декодированного контрольного изображения, определенный в п. 8.2.5.

Все изображения m в DPB, для которых все нижеследующие условия являются истиной, удаляют из DPB:

– изображение m помечено как "не использовано для контроля" или изображение m – это неконтрольное изображение. Если изображение – это контрольный кадр, то считают, что он помечен как "не использовано для контроля", если только оба его поля уже помечены как "не использовано для контроля";

– изображение m помечено как "несуществующее" или время выхода из DPB меньше чем или равно времени удаления из CPB текущего изображения n, т. е. $t_{o,dpb}(m) <= t_r(n)$.

Если кадр или последнее поле в буфере кадров удалены из DPB, то заполнение DPB уменьшают на единицу.

C.2.4 Разметка и хранение текущего декодированного изображения

C.2.4.1 Разметка и хранение контрольного декодированного изображения в DPB

Если текущее изображение – это контрольное изображение, то его хранят в DPB следующим образом:

– Если текущее декодированное изображение является вторым полем (в порядке декодирования) дополнительной пары контрольных полей, а первое поле пары все еще находится в DPB, то текущее декодированное изображение хранят в том же буфере кадров, что и первое поле пары.

– Иначе текущее декодированное изображение хранят в пустом буфере кадров, а заполнение DPB возрастает на единицу.

C.2.4.2 Хранение неконтрольного изображения в DPB

Если текущее изображение – это неконтрольное изображение и для текущего изображения n имеет место $t_{o,dpb}(n) > t_r(n)$, то его хранят в DPB следующим образом:

– Если текущее декодированное изображение – это второе поле (в порядке декодирования) дополнительной пары неконтрольных полей, а первое поле пары все еще находится в DPB, текущее декодированное изображение хранят в том же буфере кадров, что и первое поле пары.

– Иначе текущее декодированное изображение хранят в пустом буфере кадров, а заполнение DPB возрастает на единицу.

C.3 Соответствие потока битов

Поток битов кодированных данных, соответствующих этой Рекомендации | Международному стандарту, должен выполнять следующие требования.

Поток битов создают вне рамок этого Приложения, согласно синтаксису, семантике и ограничениям, определенным в этой Рекомендации | Международном стандарте.

Поток битов проверяют с помощью HRD, как это определено ниже:

Для потоков битов Типа I число проверок выполняют cpb_cnt_minus1 + 1 раз, где cpb_cnt_minus1 – это либо элемент синтаксиса hrd_parameters( ), следующий за vcl_hrd_parameters_present_flag, либо определенный другими средствами приложения, которые не рассмотрены в этой Рекомендации | Международном стандарте. Для каждой комбинации скорости битов и размера CPB, определенных значением hrd_parameters( ), следующим за vcl_hrd_parameters_present_flag, выполняют одну проверку. Каждая из этих проверок руководствуется точкой соответствия для Типа I, показанной на рисунке C-1.

Для потоков битов Типа II существует два набора проверок. Число проверок первого набора равно cpb_cnt_minus1 + 1, где cpb_cnt_minus1 – это либо элемент синтаксиса hrd_parameters( ), следующий за vcl_hrd_parameters_present_flag, либо определенный другими средствами приложения, которые не рассмотрены в этой Рекомендации | Международном стандарте. Для каждой комбинации скорости битов и размера CPB выполняют одну проверку. Каждая из этих проверок руководствуется точкой соответствия для Типа I, показанной на рисунке C-1. Для этих проверок в качестве входной скорости битов и памяти CPB учитывают только блоки VCL и NAL с данными заполнения.

Число проверок второго набора для потока битов Типа II равно cpb_cnt_minus1 + 1, где cpb_cnt_minus1 – это либо элемент синтаксиса hrd_parameters( ), следующий за nal_hrd_parameters_present_flag, либо определенный другими средствами приложения, которые не рассмотрены в этой Рекомендации | Международном стандарте. Для каждой комбинации скорости битов и размера CPB, определенной значением hrd_parameters( ), следующим за nal_hrd_parameters_present_flag, выполняют одну проверку. Каждая из этих проверок руководствуется точкой

соответствия для Типа II, показанной на рисунке C-1. Для этих проверок учитывают все блоки NAL (поток блоков NAL Типа II) или все байты (из потока байтов) в качестве входной скорости битов и памяти CPB.

ПРИМЕЧАНИЕ. – Параметры NAL HRD, установленные значением SchedSelIdx точки соответствия Типа II, показанной на рисунке C-1, достаточны также, чтобы установить соответствие VCL HRD для точки соответствия Типа I, показанной на рисунке C-1, для тех же значений initial_cpb_removal_delay[SchedSelIdx], BitRate[SchedSelIdx] и CpbSize[SchedSelIdx] для случая VBR (cbr_flag[SchedSelIdx] равно 0). Так происходит, потому что поток данных в точке соответствия Типа I является подмножеством потока данных в точке соответствия Типа II, а также потому, что в случае VBR буферу CPB разрешено стать пустым и оставаться пустым до тех пор, пока по расписанию не начнет поступать следующее изображение. Например, если параметры NAL HRD предусмотрены для точки соответствия Типа II таким образом, что не только находятся в границах набора параметров для характеристик соответствия NAL HRD подпункту j п. A.3.1, но также находятся в границах набора параметров для характеристик соответствия VCL HRD подпункту i п. A.3.1, то соответствие VCL HRD для точки соответствия Типа I также уверенно попадет в границы подпункта i пункта A.3.1.

Для соответствующих потоков битов все следующие условия должны выполняться для каждой из проверок:

– Для каждого блока доступа n с n>0, связанного с сообщением SEI о периоде буферизации, и с $\Delta t_{g,90}(n)$, определенным как

$$\Delta t_{g,90}(n) = 90\,000 * (t_{r,n}(n) - t_{af}(n - 1)), \quad \text{(C-14)}$$

значение initial_cpb_removal_delay[SchedSelIdx] должно быть ограничено следующим образом:

– Если cbr_flag[SchedSelIdx] равно 0,

$$\text{initial\_cpb\_removal\_delay[SchedSelIdx]} <= \text{Ceil}(\Delta t_{g,90}(n)). \quad \text{(C-15)}$$

– Иначе (cbr_flag[SchedSelIdx] равно 1),

$$\text{Floor}(\Delta t_{g,90}(n)) <= \text{initial\_cpb\_removal\_delay[SchedSelIdx]} <= \text{Ceil}(\Delta t_{g,90}(n)). \quad \text{(C-16)}$$

ПРИМЕЧАНИЕ. – Точное число битов в CPB во время удаления каждого изображения может зависеть от того, какой период буферизации в сообщение SEI был выбран для инициализации HRD. Кодеры должны принимать это во внимание, чтобы обеспечить соблюдение всех определенных ограничений, независимо от того, какой период буферизации в сообщении SEI был выбран для инициализации HRD, поскольку HRD можно инициализировать в один из периодов буферизации сообщений SEI.

– Переполнение CPB определено как условие, при котором общее число битов в CPB больше, чем размер CPB. Буфер CPB никогда не должен переполняться.

– Переполнение CPB определено как условие, при котором $t_{r,n}(n)$ меньше, чем $t_{af}(n)$. Если low_delay_hrd_flag равно 0, буфер CPB никогда не должен переполняться.

– Номинальное время удалений изображений из CPB (начиная со второго изображения в порядке декодирования) должно удовлетворять ограничениям на $t_{r,n}(n)$ и $t_r(n)$, выраженным в пп. A.3.1 и A.3.2 для параметров и уровней, определенных в потоке битов.

– Сразу же после добавления любого декодированного изображения к DPB заполнение DPB должно быть меньше или равно размеру DPB, как это ограничено Приложениями A, D и E для параметров и уровней, определенных в потоке битов.

– Если это необходимо для предсказания, то все контрольные изображения должны быть представлены в DPB. Каждое изображение должно быть представлено в DPB во время поступления на выход DPB, если только оно вообще не хранилось в DPB или не было удалено из DPB до времени поступления на выход одним из процессов, определенных в п. C.2.

– Значение $\Delta t_{o,dpb}(n)$, данное уравнением C-13, которое выражает разность между временем выхода изображения и временем изображения, следующим за ним сразу в порядке выхода, должно удовлетворять ограничению, выраженному в п. A.3.1 для параметров и уровней, определенных в потоке битов.

C.4 Соответствие декодера

Декодер, соответствующий этой Рекомендации | Международному стандарту, должен удовлетворять следующим требованиям.

Декодер, заявленный как соответствующий особым параметрам и уровням, должен успешно декодировать все соответствующие потоки битов (спецификация соответствия приведена в п. C.3), при условии, что все наборы параметров последовательностей и изображений относятся к блокам NAL VCL, а соответствующие сообщения SEI о периоде буферизации и о синхронизации изображения поступают в декодер своевременно либо в потоке битов (не блоками NAL VCL), либо от внешних устройств, не определенных этой Рекомендацией | Международным стандартом.

Существуют два типа соответствия, которые должны быть заявлены для декодера: соответствие времени выхода и соответствие порядка выхода.

Чтобы проверить соответствие декодера, тестовые потоки битов, соответствующие заявленным параметрами и уровням, как это описано в п. C.3, вводятся планировщиком гипотетического потока (HSS) как в HRD, так и в испытуемый декодер (DUT). Все изображения на выходе HRD должны быть также выходом DUT. Для каждого

изображения на выходе HRD значения всех образцов, которые появляются на выходе DUT для соответствующего изображения, должны быть равны значениям образцов на выходе HRD.

Для соответствия синхронизации выхода декодера HSS работает, как описано выше, с вводом таблиц, выбранных только из подмножества значений SchedSelIdx, для которых скорость битов и размер CPB ограничены, как это указано в Приложении A, для определенных параметров и уровней, или с вводом "интерполированных" таблиц, как это определено ниже, для которых скорость битов и размер CPB ограничены, как указано в Приложении A. Такой же ввод таблиц используют для HRD и DUT.

Если параметры HRD и сообщение SEI о периоде буферизации представлены со значением cpb_cnt_minus1, большим 0, декодер должен быть в состоянии декодировать поток битов как поступающий от HSS, действуя с использованием "интерполированных" таблиц, определенных как имеющих пиковую скорость битов r, размер CPB c(r) и начальную задержку удаления CPB (f(r) ÷ r), следующим образом:

$$\alpha = (r - BitRate[SchedSelIdx - 1]) \div (BitRate[SchedSelIdx] - BitRate[SchedSelIdx - 1]), \quad \text{(C-17)}$$

$$c(r) = \alpha * CpbSize[SchedSelIdx] + (1 - \alpha) * CpbSize[SchedSelIdx-1], \quad \text{(C-18)}$$

$$f(r) = \alpha * initial\_cpb\_removal\_delay[SchedSelIdx] * BitRate[SchedSelIdx] + (1 - \alpha) * initial\_cpb\_removal\_delay[SchedSelIdx - 1] * BitRate[SchedSelIdx - 1], \quad \text{(C-19)}$$

для любого SchedSelIdx > 0 и r такого, что BitRate[SchedSelIdx – 1] <= r <= BitRate[SchedSelIdx], с тем, чтобы r и c(r) находились в границах, как указано в Приложении A, максимальной скорости битов и размера буфера для определенных параметров и уровней.

ПРИМЕЧАНИЕ. – Значение initial_cpb_removal_delay[SchedSelIdx] может отличаться от такого же значения периода буферизации и его следует вычислить еще раз.

Для соответствия декодера синхронизации выхода в HRD, как это было описано выше, используют синхронизацию (относительно ввода первого бита) выхода изображения одинаково как для HRD, так и для DUT, вплоть до фиксированной задержки.

Для соответствия декодера порядку выхода HSS вводит поток битов в DUT "по требованию" от DUT, подразумевая, что HSS вводит биты (в порядке декодирования) только в том случае, если DUT требует больше битов для продолжения обработки. Как описано ниже, для этого используется буфер HRD, а HSS вводит поток битов в HRD с помощью одной из таблиц, определенных в потоке битов, или с помощью "интерполированной" таблицы, такой, что скорость битов и размер CPB ограничены, как это определено в Приложении A. Порядок выхода изображения должен быть одинаковым для HRD и для DUT.

ПРИМЕЧАНИЕ. – Это означает, что для этого теста буфер кодированного изображения DUT может быть таким же, как размер наибольшего блока доступа.

Для соответствия декодера порядку выхода размер CPB HRD равен CpbSize[SchedSelIdx] для выбранной таблицы, а размер DPB равен MaxDpbSize. Время удаления из CPB для HRD равно времени поступления заключительного бита, а декодирование происходит безотлагательно. Работа буфера DPB этого HRD описана ниже.

C.4.1 Работа DPB по организации очередности выхода

Буфер декодированного изображения содержит буферы кадров. Каждый из буферов кадров может содержать декодированный кадр, декодированную дополнительную пару полей или единственное (непарное) декодированное поле, которое помечено как "использовано для контроля" (контрольного изображения), или каждый из буферов можно содержать для будущего выхода (изображения с измененным порядком). При инициализации HRD заполнение DPB, измеряемое в кадрах, устанавливают на 0. Все следующие шаги происходят одновременно, если блок доступа удален из CPB в перечисленном порядке.

C.4.2 Декодирование промежутков в frame_num и хранение "несуществующих" изображений

В случае применения промежутки в значении frame_num обнаруживают процессом декодирования, а необходимое число "несуществующих" кадров учитывают в порядке, определенном генерацией значения UnusedShortTermFrameNum в уравнении 7-10, и отмечают, как это определено в п. 8.2.5.2. Каждый "несуществующий" кадр хранят в DPB следующим образом:

- Если не имеется пустого буфера кадров (т. е. заполнение DPB равно размеру DPB), процесс "пульсации", определенный в п. C.4.5.3, активируют повторно до тех пор, пока не образуется пустой буфер кадров, в котором можно хранить "несуществующий" кадр.
- "Несуществующий" кадр хранят в пустом буфере кадров и помечают как "не требуется для выхода", а заполнение DPB увеличивают на единицу.

C.4.3 Декодирование изображения

Первично кодированное изображение n является декодированным, и его временно хранят (но не в DPB).

C.4.4 Удаление изображения из DPB перед возможным введением текущего изображения

Удаление изображения из DPB перед возможным введением текущего изображения происходит следующим образом:

- Если декодированное изображение – это IDR изображение, то применяют следующее:
  - Все контрольные изображения в DPB помечают как "не использовано для контроля", как это определено в п. 8.2.5.

- Если IDR изображение – это не первое декодированное IDR изображение, а значения PicWidthInMbs, или FrameHeightInMbs, или max_dec_frame_buffering, найденные из действующей установки параметров последовательности, отличаются от значений PicWidthInMbs, или FrameHeightInMbs, или max_dec_frame_buffering, найденных из установки параметров последовательности, которая была действующей соответственно для предыдущей последовательности, то в HRD значение no_output_of_prior_pics_flag принимают равным 1 независимо от действительного значения no_output_of_prior_pics_flag.

  ПРИМЕЧАНИЕ. – При использовании декодера следует попытаться управлять изменениями размеров кадра или DPB, но более осторожно, чем в HRD по отношению к изменениям в PicWidthInMbs или FrameHeightInMbs.

- Если no_output_of_prior_pics_flag равно 1 или предполагают равным 1, то все буферы кадров в DPB опустошают без выдачи на выход изображений, которые там содержались, а заполнение DPB устанавливают на 0.

– Иначе (декодированное изображение – это не IDR изображение), активируют процесс разметки декодированного контрольного изображения, определенный в п. 8.2.5. Буферы кадров, содержащие кадр, или дополнительную пару полей или непарное поле, которые помечают как "не требуется для выхода" и "не использовано для контроля", опустошают (без выхода), а заполнение DPB уменьшают на число опустошенных буферов кадров.

Если текущее изображение – это IDR изображение, а значение no_output_of_prior_pics_flag не равно 1 и не предполагают равным 1, или если текущее изображение имеет memory_management_control_operation равное 5, то все не пустые буферы кадров в DPB повторно опустошают, запуская процесс "пульсации", определенный в п. C.4.5.3, а заполнение DPB устанавливают на 0.

C.4.5 Разметка и хранение текущего декодированного изображения

C.4.5.1 Разметка и хранение контрольного декодированного изображения в DPB

Если текущее изображение – это контрольное изображение, то его хранят в DPB следующим образом:

- Если текущее декодированное изображение является вторым полем (в порядке декодирования) дополнительной пары контрольных полей, а первое поле пары все еще находится в DPB, то текущее декодированное изображение хранят в том же буфере кадра, что и первое поле пары.
- Иначе выполняют следующие операции:
  - Если отсутствует пустой буфер кадров (т. е. заполнение DPB равно размеру DPB), процесс "пульсации", определенный в п. C.4.5.3, повторно активируют до тех пор, пока не появится пустой буфер кадров, в котором можно хранить текущее декодированное изображение.
  - Текущее декодированное изображение хранят в пустом буфере кадров и помечают как "не требуется для выхода", а заполнение DPB увеличивают на единицу.

C.4.5.2 Разметка и хранение неконтрольного декодированного изображения в DPB

Если текущее изображение – это неконтрольное изображение, то выполняют следующие действия:

- Если текущее декодированное изображение – это второе поле (в порядке декодирования) дополнительной неконтрольной пары полей, а первое поле пары еще находится в DPB, то текущее изображение хранят в том же буфере кадров, что и первое поле пары.
- Иначе повторно выполняют следующие действия до тех пор, пока текущее декодированное изображение не будет разделено на кадры (или поля) на выходе или сохранено в DPB:
  - Если отсутствует пустой буфер кадров (т. е. заполнение DPB равно размеру DPB), применяют следующее:
    - Если текущее изображение не содержит более низкого значения PicOrderCnt( ), чем все изображения в DPB, которые помечены как "необходимые для выхода", то выполняют процесс "пульсации", определенный в п. C.4.5.3.
    - Иначе (текущее изображение содержит более низкие значения PicOrderCnt( ), чем все изображения в DPB, которые помечены как "необходимые для выхода"), текущее изображение кадрируют (разделяют на кадры или поля), используя прямоугольник кадрирования, определенный в установке параметров последовательности для данной последовательности, а разделенное на кадры изображение является выходом.
  - Иначе (существует пустой буфер кадров, т. е. заполнение DPB меньше, чем размер DPB), текущее декодированное изображение хранят в пустом буфер кадров и помечают как "необходимое для выхода ", а заполнение DPB увеличивается на единицу.

C.4.5.3 Процесс "пульсации"

Процесс "пульсации" активируют в следующих случаях:

- Отсутствует пустой буфер кадров (т. е. заполнение DPB равно размеру DPB), а пустой буфер кадров требуется для хранения предполагаемого "несуществующего" кадра, как это определено в п. C.4.2.
- Текущее изображение – это IDR изображение, а значение no_output_of_prior_pics_flag не равно 1, и не предполагают, что оно будет равно 1, как это определено в п. C.4.4.
- Текущее изображение имеет memory_management_control_operation, равное 5, как это определено в п. C.4.4.

– Отсутствует пустой буфер кадров (т. е. заполнение DPB равно размеру DPB), а пустой буфер кадров требуется для хранения декодированного (не IDR) контрольного изображения, как это определено в п. C.4.5.1.

– Отсутствует пустой буфер кадров (т. е. заполнение DPB равно размеру DPB). Текущее изображение не является неконтрольным изображением, которое не является вторым полем дополнительной неконтрольной пары полей, а в DPB существуют изображения, помеченные как "необходимые для выхода", которые предшествуют текущему неконтрольному изображению в порядке выхода, как это определено в п. C.4.5.2. Поэтому для хранения текущего изображения необходим пустой буфер.

Процесс "пульсации" заключается в следующем:

– Следующим образом выбирают изображение или пару дополнительных контрольных полей, которые являются первыми на выходе:
  – Выбирают такой буфер кадров, который содержит изображение, имеющее наименьшее значение PicOrderCnt( ) из всех изображений в DPB, помеченных как "необходимые для выхода".
  – Если этот буфер кадров содержит дополнительную пару неконтрольных полей с обоими полями, помеченными как "необходимые для выхода", и оба поля имеют то же значение PicOrderCnt( ), то первое из этих полей в порядке декодирования рассматривают первым как выход.
  – Иначе, если этот буфер кадров содержит дополнительную пару контрольных полей с обоими полями, помеченными как "необходимые для выхода", и оба поля имеют то же значение PicOrderCnt( ), то собственно дополнительную пару контрольных полей рассматривают как выход.
  – Иначе изображение в этом буфере кадров, которое имеет наименьшее значение PicOrderCnt( ), рассматривают первым как выход.
– Если единственное изображение рассматривают первым как выход, то это изображение кадрируют, используя прямоугольник кадрирования, определенный в установке параметров последовательности, а кадрированное изображение считают выходом и помечают как "не требуется для выхода".
– Иначе (дополнительную пару контрольных полей первую рассматривают как выход), кадрируют оба поля из этой дополнительной пары контрольных полей, используя прямоугольник кадрирования, определенный в установке параметров последовательности, оба поля дополнительной пары контрольных полей вместе составляют выход, и оба поля дополнительной пары контрольных полей помечают как "не требуется для выхода".
– Буфер кадров, который включает изображение или дополнительную пару контрольных полей, которые были разделены на кадры, а также выход проверяют, и если любое из следующих условий удовлетворено, то буфер кадров опустошают, а заполнение DPB уменьшают на единицу:
  – Буфер кадров содержит неконтрольное непарное поле.
  – Буфер кадров содержит неконтрольный кадр.
  – Буфер кадров содержит дополнительную пару неконтрольных полей, а оба поля помечены как "не требуется для выхода".
  – Буфер кадров содержит непарное контрольное поле, помеченное как "не использовано для контроля".
  – Буфер кадров содержит контрольный кадр с двумя полями, помеченными как "не использовано для контроля".
  – Буфер кадров содержит дополнительную пару контрольных полей, и оба поля помечены как "не использовано для контроля" и "не требуется для выхода".

Приложение D

Дополнительная расширенная информация

(Это Приложение является составной частью данной Рекомендации | Международного стандарта)

Это Приложение определяет синтаксис и семантику полезной нагрузки сообщения SEI.

Сообщения SEI помогают в процессах, связанных с декодированием, отображением и для других целей. Однако сообщения SEI не требуются для конструирования образцов яркости или цветности процессом декодирования. Соответствие декодеров не требуется, чтобы обработать эту информацию для согласования порядка выхода с этой Рекомендацией | Международным стандартом (см. Приложение C для спецификации соответствия). Некоторая информация сообщения SEI требуется для проверки соответствия потока битов, а также соответствия декодера для синхронизации выхода.

В Приложении D спецификация присутствия сообщений SEI также выполняется, если эти сообщения (или некоторое их подмножество) передают в декодеры (или в HRD) другими средствами, не определенными в этой Рекомендации | Международном стандарте. Если сообщения SEI присутствуют в потоке битов, они должны подчиняться синтаксису и семантике, которые определены в пп. 7.3.2.3 и 7.4.2.3, также в этом Приложении. Если содержание сообщения SEI передают приложению какими-нибудь средствами иными, чем те, которые имеются в потоке битов, то для представления содержания сообщения SEI не требуется использование того же синтаксиса, который определен в этом Приложении. Для подсчета битов учитывают только соответствующие биты, которые действительно находятся в потоке битов.

D.1 Синтаксис полезной нагрузки SEI

sei_payload(payloadType, payloadSize) {	C	Дескриптор
if(payloadType = = 0)		
buffering_period(payloadSize)	5	
else if(payloadType = = 1)		
pic_timing(payloadSize)	5	
else if(payloadType = = 2)		
pan_scan_rect(payloadSize)	5	
else if(payloadType = = 3)		
filler_payload(payloadSize)	5	
else if(payloadType = = 4)		
user_data_registered_itu_t_t35(payloadSize)	5	
else if(payloadType = = 5)		
user_data_unregistered(payloadSize)	5	
else if(payloadType = = 6)		
recovery_point(payloadSize)	5	
else if(payloadType = = 7)		
dec_ref_pic_marking_repetition(payloadSize)	5	
else if(payloadType = = 8)		
spare_pic(payloadSize)	5	
else if(payloadType = = 9)		
scene_info(payloadSize)	5	
else if(payloadType = = 10)		
sub_seq_info(payloadSize)	5	
else if(payloadType = = 11)		
sub_seq_layer_characteristics(payloadSize)	5	
else if(payloadType = = 12)		
sub_seq_characteristics(payloadSize)	5	
else if(payloadType = = 13)		
full_frame_freeze(payloadSize)	5	
else if(payloadType = = 14)		
full_frame_freeze_release(payloadSize)	5	
else if(payloadType = = 15)		
full_frame_snapshot(payloadSize)	5	
else if(payloadType = = 16)		
progressive_refinement_segment_start(payloadSize)	5	
else if(payloadType = = 17)		
progressive_refinement_segment_end(payloadSize)	5	
else if(payloadType = = 18)		
motion_constrained_slice_group_set(payloadSize)	5	
else		
reserved_sei_message(payloadSize)	5	
if(!byte_aligned( )) {		
bit_equal_to_one /* равно 1 */	5	f(1)
while(!byte_aligned( ))		
bit_equal_to_zero /* equal to 0 */	5	f(1)
}		
}		

D.1.1 Синтаксис сообщения SEI о периоде буферизации

buffering_period(payloadSize) {	C	Дескриптор
seq_parameter_set_id	5	ue(v)
if(NalHrdBpPresentFlag) {		
for(SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++) {		
initial_cpb_removal_delay[SchedSelIdx **]**	5	u(v)
initial_cpb_removal_delay_offset[SchedSelIdx **]**	5	u(v)
}		
}		
if(VclHrdBpPresentFlag) {		
for(SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++) {		
initial_cpb_removal_delay[SchedSelIdx **]**	5	u(v)
initial_cpb_removal_delay_offset[SchedSelIdx **]**	5	u(v)
}		
}		
}		

D.1.2 Синтаксис сообщения SEI о синхронизации изображения

pic_timing(payloadSize) {	C	Дескриптор
if(CpbDpbDelaysPresentFlag) {		
cpb_removal_delay	5	u(v)
dpb_output_delay	5	u(v)
}		
if(pic_struct_present_flag) {		
pic_struct	5	u(4)
for(i = 0; i < NumClockTS ; i++) {		
clock_timestamp_flag[i]	5	u(1)
if(clock_timestamp_flag[i]) {		
ct_type	5	u(2)
nuit_field_based_flag	5	u(1)
counting_type	5	u(5)
full_timestamp_flag	5	u(1)
discontinuity_flag	5	u(1)
cnt_dropped_flag	5	u(1)
n_frames	5	u(8)
if(full_timestamp_flag) {		
seconds_value /* 0..59 */	5	u(6)
minutes_value /* 0..59 */	5	u(6)
hours_value /* 0..23 */	5	u(5)
} else {		
seconds_flag	5	u(1)
if(seconds_flag) {		

seconds_value /* диапазон 0..59 */	5	u(6)
minutes_flag	5	u(1)
if(minutes_flag) {		
minutes_value /* 0..59 */	5	u(6)
hours_flag	5	u(1)
if(hours_flag)		
hours_value /* 0..23 */	5	u(5)
}		
}		
}		
if(time_offset_length > 0)		
time_offset	5	i(v)
}		
}		
}		
}		

D.1.3 Синтаксис сообщения SEI о прямоугольнике сканирования

pan_scan_rect(payloadSize) {	**C**	**Дескриптор**
pan_scan_rect_id	5	ue(v)
pan_scan_rect_cancel_flag	5	u(1)
if(!pan_scan_rect_cancel_flag) {		
pan_scan_cnt_minus1	5	ue(v)
for(i = 0; i <= pan_scan_cnt_minus1; i++) {		
pan_scan_rect_left_offset[i **]**	5	se(v)
pan_scan_rect_right_offset[i **]**	5	se(v)
pan_scan_rect_top_offset[i **]**	5	se(v)
pan_scan_rect_bottom_offset[i **]**	5	se(v)
}		
pan_scan_rect_repetition_period	5	ue(v)
}		
}		

D.1.4 Синтаксис сообщения SEI о полезной нагрузке заполнителя

filler_payload(payloadSize) {	**C**	**Дескриптор**
for(k = 0; k < payloadSize; k++)		
ff_byte /* equal to 0xFF */	5	f(8)
}		

D.1.5 Синтаксис сообщения SEI о данных пользователя, зарегистрированных по Рекомендации МСЭ-Т T.35

user_data_registered_itu_t_t35(payloadSize) {	C	Дескриптор
itu_t_t35_country_code	5	b(8)
if(itu_t_t35_country_code != 0xFF)		
i = 1		
else {		
itu_t_t35_country_code_extension_byte	5	b(8)
i = 2		
}		
do {		
itu_t_t35_payload_byte	5	b(8)
i++		
} while(i < payloadSize)		
}		

D.1.6 Синтаксис сообщения SEI о незарегистрированных данных пользователя

user_data_unregistered(payloadSize) {	C	Дескриптор
uuid_iso_iec_11578	**5**	u(128)
for(i = 16; i < payloadSize; i++)		
user_data_payload_byte	5	b(8)
}		

D.1.7 Синтаксис сообщения SEI о пункте восстановления

recovery_point(payloadSize) {	C	Дескриптор
recovery_frame_cnt	5	ue(v)
exact_match_flag	5	u(1)
broken_link_flag	5	u(1)
changing_slice_group_idc	5	u(2)
}		

D.1.8 Синтаксис сообщения SEI о повторной разметке декодированного контрольного изображения

dec_ref_pic_marking_repetition(payloadSize) {	C	Дескриптор
original_idr_flag	5	u(1)
original_frame_num	5	ue(v)
if(!frame_mbs_only_flag) {		
original_field_pic_flag	5	u(1)
if(original_field_pic_flag)		
original_bottom_field_flag	5	u(1)
}		
dec_ref_pic_marking( )	5	
}		

D.1.9 Синтаксис сообщения SEI о резервном изображении

spare_pic(payloadSize) {	C	Дескриптор
target_frame_num	5	ue(v)
spare_field_flag	5	u(1)
if(spare_field_flag)		
target_bottom_field_flag	5	u(1)
num_spare_pics_minus1	5	ue(v)
for(i = 0; i < num_spare_pics_minus1 + 1; i++) {		
delta_spare_frame_num[i **]**	5	ue(v)
if(spare_field_flag)		
spare_bottom_field_flag[i **]**	5	u(1)
spare_area_idc[i **]**	5	ue(v)
if(spare_area_idc[i] = = 1)		
for(j = 0; j < PicSizeInMapUnits; j++)		
spare_unit_flag[i **][** j **]**	5	u(1)
else if(spare_area_idc[i] = = 2) {		
mapUnitCnt = 0		
for(j=0; mapUnitCnt < PicSizeInMapUnits; j++) {		
zero_run_length[i **][** j **]**	5	ue(v)
mapUnitCnt += zero_run_length[i][j] + 1		
}		
}		
}		
}		

D.1.10 Синтаксис сообщения SEI о сценической информации

scene_info(payloadSize) {	C	Дескриптор
scene_info_present_flag	5	u(1)
if(scene_info_present_flag) {		
scene_id	5	ue(v)
scene_transition_type	5	ue(v)
if(scene_transition_type > 3)		
second_scene_id	5	ue(v)
}		
}		

D.1.11 Синтаксис сообщения SEI об информации субпоследовательности

sub_seq_info(payloadSize) {	C	Дескриптор
sub_seq_layer_num	5	ue(v)
sub_seq_id	5	ue(v)
first_ref_pic_flag	5	u(1)
leading_non_ref_pic_flag	5	u(1)
last_pic_flag	5	u(1)
sub_seq_frame_num_flag	5	u(1)
if(sub_seq_frame_num_flag)		
sub_seq_frame_num	5	ue(v)
}		

D.1.12 Синтаксис сообщения SEI о характеристиках слоя субпоследовательности

sub_seq_layer_characteristics(payloadSize) {	C	Дескриптор
num_sub_seq_layers_minus1	5	ue(v)
for(layer = 0; layer <= num_sub_seq_layers_minus1; layer++) {		
accurate_statistics_flag	5	u(1)
average_bit_rate	5	u(16)
average_frame_rate	5	u(16)
}		
}		

D.1.13 Синтаксис сообщения SEI о характеристиках субпоследовательности

sub_seq_characteristics(payloadSize) {	C	Дескриптор
sub_seq_layer_num	5	ue(v)
sub_seq_id	5	ue(v)
duration_flag	5	u(1)
if(duration_flag)		
sub_seq_duration	5	u(32)
average_rate_flag	5	u(1)
if(average_rate_flag) {		
accurate_statistics_flag	5	u(1)
average_bit_rate	5	u(16)
average_frame_rate	5	u(16)
}		
num_referenced_subseqs	5	ue(v)
for(n = 0; n < num_referenced_subseqs; n++) {		
ref_sub_seq_layer_num	5	ue(v)
ref_sub_seq_id	5	ue(v)
ref_sub_seq_direction	5	u(1)
}		
}		

D.1.14 Синтаксис сообщения SEI о полностью фиксированном кадре

full_frame_freeze(payloadSize) {	C	Дескриптор
full_frame_freeze_repetition_period	5	ue(v)
}		

D.1.15 Синтаксис сообщения SEI об отмене режима полностью фиксированного кадра

full_frame_freeze_release(payloadSize) {	C	Дескриптор
}		

D.1.16 Синтаксис сообщения SEI о стоп-кадре

full_frame_snapshot(payloadSize) {	C	Дескриптор
snapshot_id	5	ue(v)
}		

D.1.17 Синтаксис сообщения SEI о запуске сегмента последовательного улучшения

progressive_refinement_segment_start(payloadSize) {	C	Дескриптор
progressive_refinement_id	5	ue(v)
num_refinement_steps_minus1	5	ue(v)
}		

D.1.18 Синтаксис сообщения SEI об окончании сегмента последовательного улучшения

progressive_refinement_segment_end(payloadSize) {	C	Дескриптор
progressive_refinement_id	5	ue(v)
}		

D.1.19 Синтаксис сообщения SEI о наборе группы секций ограниченного движения

motion_constrained_slice_group_set(payloadSize) {	C	Дескриптор
num_slice_groups_in_set_minus1	5	ue(v)
for(i = 0; i <= num_slice_groups_in_set_minus1; i++)		
slice_group_id[i **]**	5	u(v)
exact_sample_value_match_flag	5	u(1)
pan_scan_rect_flag	5	u(1)
if(pan_scan_rect_flag)		
pan_scan_rect_id	5	ue(v)
}		

D.1.20 Синтаксис зарезервированных сообщений SEI

reserved_sei_message(payloadSize) {	C	Дескриптор
for(i = 0; i < payloadSize; i++)		
reserved_sei_message_payload_byte	5	b(8)
}		

D.2 Семантика полезной нагрузки SEI

D.2.1 Семантика сообщения SEI о периоде буферизации

Если NalHrdBpPresentFlag или VclHrdBpPresentFlag равно 1, то сообщение SEI о периоде буферизации может быть связано с любым блоком доступа в потоке битов. Сообщение SEI о периоде буферизации должно быть связано с каждым IDR блоком доступа, а каждый блок доступа связан с сообщением SEI о пункте восстановления.

ПРИМЕЧАНИЕ. – Для некоторых приложений частое появление сообщения SEI о периоде буферизации может быть нежелательным.

Период буферизации определен как набор блоков доступа между двумя событиями – сообщениями SEI о периоде буферизации – в порядке декодирования.

seq_parameter_set_id определяет установку параметров последовательности, которая содержит последовательность атрибутов HRD. Значение seq_parameter_set_id должно быть равно значению seq_parameter_set_id в установке параметров изображения, на которую ссылается начальное кодированное изображение, связанное с сообщением SEI о периоде буферизации. Значение seq_parameter_set_id должно быть в диапазоне от 0 до 31 включительно.

initial_cpb_removal_delay[SchedSelIdx] определяет задержку для SchedSelIdx-ого буфера CPB между временем поступления в CPB первого бита кодированных данных, связанных с блоком доступа, который, в свою очередь, связан с сообщением SEI о периоде буферизации, и временем удаления из CPB кодированных данных, связанных с тем же блоком доступа за первый период буферизации после инициализации HRD. Элемент синтаксиса имеет длину в битах, заданную начальным значением initial_cpb_removal_delay_length_minus1 + 1. Он расположен в блоках с тактовой частотой 90 кГц. Значение initial_cpb_removal_delay[SchedSelIdx] не должно быть равно 0 и не должно превышать 90 000 * (CpbSize[SchedSelIdx] ÷ BitRate[SchedSelIdx]), что является временным эквивалентом размера CPB для синхронизирующих импульсов 90 кГц.

initial_cpb_removal_delay_offset[SchedSelIdx] используют для SchedSelIdx-ого буфера CPB в комбинации с cpb_removal_delay для определения начального времени доставки кодированных блоков доступа в CPB. Значение initial_cpb_removal_delay_offset[SchedSelIdx] записано в единицах тактов 90 кГц. Элемент синтаксиса initial_cpb_removal_delay_offset[SchedSelIdx] – это фиксированная длина кода, у которого длина в битах задана initial_cpb_removal_delay_length_minus1 + 1. Этот элемент синтаксиса не используют декодеры, и он необходим только для доставки планировщика (HSS), определенного в Приложении C.

Вдобавок к собственно кодированной видеопоследовательности сумма initial_cpb_removal_delay[SchedSelIdx] и initial_cpb_removal_delay_offset[SchedSelIdx] должна быть константой для каждого значения SchedSelIdx.

D.2.2 Семантика сообщения SEI о синхронизации изображения

Если CpbDpbDelaysPresentFlag равно 1, сообщение SEI о синхронизации изображения должно быть связано с каждым блоком доступа в потоке битов.

cpb_removal_delay определяет, сколько тактов системных часов (см. п. E.2.1) требуется ожидать после удаления из CPB блока доступа, связанного с самым последним сообщением SEI о периоде буферизации перед удалением из буфера блока доступа данных, связанных с сообщением SEI о синхронизации изображения. Это значение также используют, чтобы вычислить самое ранее возможное время поступления данных блока доступа в CPB для HSS, как это определено в Приложении C. Элемент синтаксиса является кодом с фиксированной длиной, чья длина в битах задана cpb_removal_delay_length_minus1 + 1. Значение cpb_removal_delay – это остаток $2^{(\text{cpb\_removal\_delay\_length\_minus1} + 1)}$ счетчика.

Значение cpb_removal_delay для первого изображения в потоке битов должно быть равно 0.

dpb_output_delay используют для вычисления времени выхода из DPB изображения. Оно определяет, сколько тактов системных часов требуется ожидать после удаления из CPB блока доступа перед тем, как декодированное изображение может появиться на выходе DPB (см. п. C.2).

ПРИМЕЧАНИЕ. – Изображение не удаляют из DPB в назначенное время, если оно все еще помечено как "использовано для краткосрочного контроля" или "использовано для долгосрочного контроля".

ПРИМЕЧАНИЕ. – Для декодированного изображения определено только одно значение dpb_output_delay.

Размер элемента синтаксиса dpb_output_delay задан в битах dpb_output_delay_length_minus1 + 1. Если max_dec_frame_buffering равно 0, то dpb_output_delay должно быть равно 0.

Время выхода, полученное из dpb_output_delay для любого изображения, которое является выходом от выходного сигнала синхронизации согласованного декодера, как это определено в п. C.2, должно предшествовать времени выхода, полученного из dpb_output_delay для всех изображений в любой кодированной видеопоследовательности в порядке декодирования.

Время выхода, полученное из dpb_output_delay второго поля в порядке декодирования дополнительной пары неконтрольных полей, не должно превышать времени выхода, полученного из dpb_output_delay первого поля той же дополнительной пары неконтрольных полей.

Порядок изображения на выходе, установленный значениями этого элемента синтаксиса, должен быть таким же, как порядок, установленный значениями PicOrderCnt( ), как это определено в пп. C.4.1–C.4.5, за исключением тех случаев, когда два поля дополнительной пары контрольных полей имеют то же значение PicOrderCnt( ) и различные времена выхода.

Для изображений, которые не появляются на выходе с помощью процесса "пульсаций" из п. C.4.5, поскольку они предшествуют в порядке декодирования IDR изображению с no_output_of_prior_pics_flag, равным 1 (или который считают равным 1), времена выхода, полученные из dpb_output_delay, должны возрастать с ростом значения PicOrderCnt( ) относительно всех изображений в той же самой кодированной видеопоследовательности, следующей за любым изображением с memory_management_control_operation, равным 5.

pic_struct указывает, действительно ли изображение может быть выведено на дисплей в виде кадра или одного или более полей, согласно таблице D-1. Дублирование кадров (pic_struct равно 7) указывает, что кадр может быть выведен на дисплей последовательно два раза, а утроение кадров (pic_struct равно 8) указывает, что кадр может быть выведен на дисплей последовательно три раза.

ПРИМЕЧАНИЕ. – Дублирование кадров можно ослабить для отображения, например, 25 строк видео на 50-строчный экран и 29,97 строк видео на 59,94-строчный экран. Использование дублирования и утроения кадров в комбинации с каждым другим кадром может облегчить отображение 23,98 строк видео на 59,94-строчный экран.

Таблица D-1 – Интерпретация pic_struct

Значение	**Указанный вывод изображения**	**Ограничения**	**NumClockTS**
0	кадр	field_pic_flag должно быть 0	1
1	верхнее поле	field_pic_flag должно быть 1, bottom_field_flag должно быть 0	1
2	нижнее поле	field_pic_flag должно быть 1, bottom_field_flag должно быть 1	1
3	верхнее поле, нижнее поле – в таком порядке	field_pic_flag должно быть 0	2
4	нижнее поле, верхнее поле – в таком порядке	field_pic_flag должно быть 0	2
5	верхнее поле, нижнее поле, верхнее поле повторно – в таком порядке	field_pic_flag должно быть 0	3
6	нижнее поле, верхнее поле, нижнее поле повторно – в таком порядке	field_pic_flag должно быть 0	3
7	дублирование кадра	field_pic_flag должно быть 0 fixed_frame_rate_flag должно быть 1	2
8	утроение кадра	field_pic_flag должно быть 0 fixed_frame_rate_flag должно быть 1	3
9..15	зарезервировано		

NumClockTS определено с помощью pic_struct, как это показано в таблице D-1. Имеется до NumClockTS разных наборов информационных отметок времени для изображения, как это определено значением clock_timestamp_flag[i], в каждом наборе. Наборы информационных отметок времени применяют к полю (полям) или к кадру (кадрам), связанным с изображение с помощью pic_struct.

Содержание элементов синтаксиса отметок времени указывает на время возникновения, захвата или альтернативного, идеального отображения. Указанное время вычисляют как

$$clockTimestamp = ((hH * 60 + mM) * 60 + sS) * time\_scale + nFrames * (num\_units\_in\_tick * (1 + nuit\_field\_based\_flag)) + tOffset, \quad \text{(D-1)}$$

в единицах тактовых импульсов синхронизатора с задающей частотой, равной time_scale Гц, относительно некоторой неопределенной точки, для которой clockTimestamp равно 0. Значение clockTimestamp не влияет на порядок выхода и синхронизации выхода DPB. Если два или более кадров с pic_struct, равным 0, следуют друг за другом в порядке выхода и имеют равные значения clockTimestamp, то это означает, что кадры представляют то же содержание и что последний такой кадр предпочтительней для представления в порядке выхода.

ПРИМЕЧАНИЕ. – Индикация времени clockTimestamp может сделать более простым вывод на устройства с обновленными скоростями, чем индикация, подстроенная под временные отметки на выходе DPB.

clock_timestamp_flag[i **]**, равное 1, указывает, что элементы синтаксиса числа отметок времени присутствуют и следуют сразу за элементами синтаксиса. Значение clock_timestamp_flag[i], равное 0, указывает, что связанные с отметками времени элементы синтаксиса отсутствуют. Если NumClockTS больше 1, а clock_timestamp_flag[i] равно 1 более чем для одного значения i, то значение clockTimestamp должно быть неубывающим при возрастании значений i.

ct_type указывает на тип сканирования (чередующийся или последовательный) исходного материала следующим образом:

Два поля кодированного кадра могу иметь разные значения ct_type.

Если clockTimestamp равно для двух полей противоположной четности, которые последовательны в порядке выхода, и оба поля с ct_type, равным 0 (последовательно), или ct_type, равным 2 (неизвестно), то это указывает, что оба поля поступили от одного и того же последовательного кадра. Два последовательных поля в порядке выхода должны иметь различные значения clockTimestamp, если значение ct_type для каждого поля равно 1 (чередование).

Таблица D-2 – Отображение ct_type на сканирование исходного изображения

Зн241 ачение	**Сканирование исходного изображения**
0	последовательное
1	чередующееся
2	неизвестное
3	зарезервировано

nuit_field_based_flag: Используют для вычисления clockTimestamp, как это определено в уравнении D-1.

counting_type: Определяет метод разделения на кадры значений n_frames, как это определено в таблице D-3.

Таблица D-3 – Определение значений counting_type

Значение	Интерпретация
0	нет разделения на кадры вычисленных значений n_frames, и не используют time_offset
1	нет разделения на кадры вычисленных значений n_frames
2	разделение на кадры отдельных нулевых вычисленных значений n_frames
3	разделение на кадры отдельных вычисленных значений n_frames MaxFPS-1
4	разделение на кадры двух наименьших (значение 0 и 1) вычисленных значений n_frames, если seconds_value равно 0, а minutes_value – не целое кратное 10
5	разделение на кадры неопределенных отдельных значений n_frames
6	разделение на кадры неопределенного числа неопределенных значений n_frames
7..31	зарезервировано

full_timestamp_flag, равное 1, определяет, что элемент синтаксиса n_frames следует за seconds_value, minutes_value и hours_value. Значение full_timestamp_flag, равное 0, определяет, что элемент синтаксиса n_frames следует за seconds_flag.

discontinuity_flag, равное 0, указывает, что разность между значением clockTimestamp и значением clockTimestamp, вычисленную из прежних отметок времени в порядке выхода, можно интерпретировать как разность временем возникновения или захвата объединенных кадров или полей. Значение discontinuity_flag, равное 1, указывает, что разность между текущим значением clockTimestamp и значением clockTimestamp, вычисленную из прежних отметок времени в порядке выхода, нельзя интерпретировать как разность времени возникновения или захвата объединенных кадров или полей. Если discontinuity_flag равно 0, то значение clockTimestamp должно быть больше чем или равно всем значениям clockTimestamp, присутствующим в предшествующем изображении в порядке выхода из DPB.

cnt_dropped_flag определяет пропуск одного или более значений n_frames, используя метод вычислений, определенный counting_type.

n_frames определяет значение nFrames, использованное для вычисления clockTimestamp. n_frames должно быть меньше чем

$$MaxFPS = Ceil(time\_scale \div num\_units\_in\_tick). \quad (D\text{-}2)$$

ПРИМЕЧАНИЕ. – n_frames – это счетчик на основе кадров. Для специального обозначения синхронизации поля используют time_offset для указания отличия clockTimestamp для каждого поля.

Если counting_type равно 2, а cnt_dropped_flag равно 1, то n_frames должно быть равно 1, а значение n_frames для прежнего изображения в порядке выхода не должно быть равно 0 до тех пор, пока discontinuity_flag не будет равен 1.

ПРИМЕЧАНИЕ. – Если counting_type равно 2, необходимых значительных увеличений величин tOffset в уравнении D-1, когда используют постоянные не целочисленные скорости кадров (например, 12,5 кадров в секунду с time_scale, равным 25, и num_units_in_tick, равным 2, а nuit_field_based_flag, равным 0), можно избежать, пропуская иногда при подсчетах значение n_frames, равное 0 (например, подсчитывая n_frames от 0 до 12, а затем увеличивая seconds_value и считая n_frames от 1 до 12, а затем увеличивая seconds_value и counting n_frames от 0 до 12, и т. д.).

Если counting_type равно 3, а cnt_dropped_flag равно 1, то n_frames должно быть равно 0, а значение n_frames для предыдущего изображения в порядке выхода не должно быть равно MaxFPS – 1 до тех пор, пока discontinuity_flag не будет равен 1.

ПРИМЕЧАНИЕ. – Если counting_type равно 3, необходимых значительных увеличений величин tOffset в уравнении D-1, когда используют постоянные не целочисленные скорости кадров (например, 12,5 кадров в секунду с time_scale, равным 25, и num_units_in_tick, равным 2, а nuit_field_based_flag, равным 0), можно избежать, пропуская иногда при подсчетах значение n_frames, равное 0 (например, подсчитывая n_frames от 0 до 12, а затем увеличивая seconds_value и считая n_frames от 1 до 12, а затем увеличивая seconds_value и counting n_frames от 0 до 12, и т. д.).

Если counting_type равно 4, а cnt_dropped_flag равно 1, то n_frames должно быть равно 2, а особое значение sS должно быть нулем, особое значение mM не должно быть целым кратным 10, n_frames для предыдущего изображения в порядке выхода не должно быть равно 0 или 1 до тех пор, пока discontinuity_flag не будет равен 1.

ПРИМЕЧАНИЕ. – Если counting_type равно 4, необходимые значительные увеличения величин tOffset в уравнении D-1, когда используют постоянные не целочисленные скорости кадров (например, 30000÷1001 кадров в секунду с time_scale, равным 60000, num_units_in_tick, равным 1001, а nuit_field_based_flag, равным 1) можно снизить, пропуская иногда при подсчетах значение n_frames равное MaxFPS (например, подсчитывая n_frames от 0 до 29, а затем увеличивая seconds_value и считая n_frames от 0 до 29, и т. д., до тех пор пока seconds_value не будет нулем, а minutes_value не будет целым кратным 10, а затем считая n_frames от 2 до 29, затем увеличивая seconds_value и считая n_frames от 0 до 29, и т. д.). Этот метод подсчета хорошо известен в промышленности, и на него часто ссылаются как на метод "NTSC drop-frame".

Если counting_type равно 5 или 6, а cnt_dropped_flag равно 1, то n_frames не должно быть равно 1 плюс значение n_frames для предыдущего изображения в порядке выхода по модулю MaxFPS до тех пор, пока discontinuity_flag не будет равен 1.

ПРИМЕЧАНИЕ. – Если counting_type равно 5 или 6, необходимых значительных увеличений величин tOffset в уравнении D-1, когда используют постоянные не целочисленные скорости кадров, можно избежать, пропуская иногда при подсчетах значение n_frames. Особые значения n_frames, которые пропускают, не являются определенными, если counting_type равно 5 или 6.

seconds_flag, равное 1, указывает, что seconds_value и minutes_flag присутствуют, если full_timestamp_flag равно 0. Значение seconds_flag, равное 0, указывает, что seconds_value и minutes_flag отсутствуют.

seconds_value указывает, что значение sS используют для вычисления clockTimestamp. Значение seconds_value должно быть в диапазоне от 0 до 59 включительно. Если seconds_value не присутствует, то для вычисления clockTimestamp должно быть использовано предыдущее значение seconds_value в порядке декодирования вместо sS.

minutes_flag, равное 1, указывает, что minutes_value и hours_flag присутствуют, если full_timestamp_flag равно 0, а seconds_flag равно 1. Значение minutes_flag, равное 0, указывает, что minutes_value и hours_flag отсутствуют.

minutes_value указывает, что значение mM используют для вычисления clockTimestamp. Значение minutes_value должно быть в диапазоне от 0 до 59 включительно. Если minutes_value не присутствует, то для вычисления clockTimestamp должно быть использовано предыдущее значение minutes_value в порядке декодирования вместо mM.

hours_flag, равное 1, указывает, что hours_value присутствует, если full_timestamp_flag равно 0 и seconds_flag равно 1, а minutes_flag равно 1.

hours_value указывает, что для вычисления clockTimestamp используют значение hH. Значение hours_value должно быть в диапазоне от 0 до 23 включительно. Если hours_value отсутствует, то для вычисления clockTimestamp должно быть использовано предыдущее значение hours_value в порядке декодирования в виде hH.

time_offset указывает, что значение tOffset используют для вычисления clockTimestamp. Число битов, используемое для представления time_offset, должно быть равно time_offset_length. Если time_offset отсутствует, то для вычисления clockTimestamp должно быть использовано значение 0 вместо tOffset.

D.2.3 Семантика сообщения SEI о прямоугольнике сканирования

Элементы синтаксиса сообщения SEI о прямоугольнике сканирования определяют координаты прямоугольника относительно разделения (на кадры) прямоугольника при установке параметров последовательности. Каждая координата этого прямоугольника определена в единицах одной шестнадцатой образца относительно опорной сетки яркости.

pan_scan_rect_id содержит идентифицирующий номер, который может быть использован для идентификации цели прямоугольника сканирования (например, для идентификации прямоугольника как площади, которая должна быть видна на конкретном экране, или как площади, на которой присутствует на сцене конкретный актер). Значение pan_scan_rect_id должно быть в диапазоне от 0 до $2^{32} - 1$ включительно.

Значения pan_scan_rect_id от 0 до 255 и от 512 до $2^{31} - 1$ может быть использовано, как это определено приложением. Значения pan_scan_rect_id от 256 до 511 и от 2^{31} до $2^{32} - 1$ зарезервированы для будущего использования МСЭ-Т | ИСО/МЭК. Декодеры, встречающие значение pan_scan_rect_id в диапазоне от 256 до 511 или в диапазоне от 2^{31} до $2^{32} - 1$, должны его игнорировать (удалять из потока битов и отбрасывать).

pan_scan_rect_cancel_flag, равное 1, указывает, что сообщение SEI аннулирует последействие предыдущего сообщения SEI о прямоугольнике сканирования. Значение pan_scan_rect_cancel_flag, равное 0, указывает, что сообщение SEI не аннулирует последействие предыдущего сообщения SEI о прямоугольнике сканирования и что информация о прямоугольнике сканирования следует далее.

pan_scan_cnt_minus1 определяет число прямоугольников сканирования, которые присутствуют в сообщении SEI. Значение pan_scan_cnt_minus1 должно быть в диапазоне от 0 до 2 включительно. Значение pan_scan_cnt_minus1, равное 0, указывает, что присутствует единственный прямоугольник сканирования, который использует все поля декодированного изображения. Значение pan_scan_cnt_minus1 должно быть равно 0, если текущее изображение – это поле. Значение pan_scan_cnt_minus1, равное 1, указывает, что присутствуют два прямоугольника сканирования, первый из которых используют для первого поля изображения в порядке выхода, а второй – для второго поля изображения в порядке выхода. Значение pan_scan_cnt_minus1, равное 2, указывает, что присутствуют три прямоугольника сканирования. Первый используют для первого поля изображения в порядке выхода, второй – для второго поля изображения в порядке выхода, а третий используют для повторения первого поля как третьего в порядке выхода.

pan_scan_rect_left_offset[i **]**, **pan_scan_rect_right_offset[** i **]**, **pan_scan_rect_top_offset[** i **]** и **pan_scan_rect_bottom_offset[** i **]** определяют положение прямоугольника сканирования, как целые величины со знаками в единицах одной шестнадцатой образца относительно опорной сетки яркости. Значения каждого из этих четырех элементов синтаксиса должно быть в диапазоне от -2^{31} до $2^{31} - 1$ включительно.

Прямоугольник сканирования определен в единицах одной шестнадцатой образца относительно опорной сетки яркости как площадь прямоугольника с координатами следующим образом:

– Если frame_mbs_only_flag равно 1, прямоугольник сканирования имеет горизонтальные координаты кадра яркости от 32 * frame_crop_left_offset + pan_scan_rect_left_offset[i] до 32 * (8 * PicWidthInMbs – frame_crop_right_offset) + pan_scan_rect_right_offset[i] – 1 и вертикальные координаты от 32 * frame_crop_top_offset + pan_scan_rect_top_offset[i] до 32 * (8 * PicHeightInMbs – frame_crop_bottom_offset) + pan_scan_rect_bottom_offset[i] – 1 включительно. В этом случае значение 32 * frame_crop_left_offset + pan_scan_rect_left_offset[i] должно быть меньше чем или равно 32 * (8 * PicWidthInMbs – frame_crop_right_offset) + pan_scan_rect_right_offset[i] – 1; а значение 32 * frame_crop_top_offset + pan_scan_rect_top_offset[i] должно быть меньше чем или равно 32 * (8 * PicHeightInMbs – frame_crop_bottom_offset) + pan_scan_rect_bottom_offset[i] – 1.

– Иначе (frame_mbs_only_flag равно 0), прямоугольник сканирования имеет горизонтальные координаты кадра яркости от 32 * frame_crop_left_offset + pan_scan_rect_left_offset[i] до 32 * (8 * PicWidthInMbs – frame_crop_right_offset) + pan_scan_rect_right_offset[i] – 1 и вертикальные координаты от 64 * frame_crop_top_offset + pan_scan_rect_top_offset[i] до 64 * (4 * PicHeightInMbs – frame_crop_bottom_offset) + pan_scan_rect_bottom_offset[i] – 1 включительно. В этом случае значение 32 * frame_crop_left_offset + pan_scan_rect_left_offset[i] должно быть меньше чем или равно 32 * (8 * PicWidthInMbs – frame_crop_right_offset) + pan_scan_rect_right_offset[i] – 1; а значение 64 * frame_crop_top_offset + pan_scan_rect_top_offset[i] должно быть меньше чем или равно 64 * (4 * PicHeightInMbs – frame_crop_bottom_offset) + pan_scan_rect_bottom_offset[i] – 1.

Если площадь прямоугольника сканирования включает образцы вне разделенного (на кадры) прямоугольника, то область вне этого прямоугольника может быть заполнена синтезированным контентом (например, черным или нейтральным серым видеоконтентом) для вывода на экран.

pan_scan_rect_repetition_period указывает, действительно ли другое сообщение SEI о прямоугольнике сканирования с таким же значением pan_scan_rect_id должно присутствовать в потоке битов, и определяет интервал подсчета порядка изображения, с которым оно должно присутствовать. Значение pan_scan_rect_repetition_period должно быть в диапазоне от 0 до 16 384 включительно. Если pan_scan_cnt_minus1 больше 0, то pan_scan_rect_repetition_period не должно быть больше 1.

pan_scan_rect_repetition_period, равное 0, определяет, что информация о прямоугольнике сканирования применима только к текущему декодированному изображению.

pan_scan_rect_repetition_period, равное 1, определяет, что информацию о прямоугольнике сканирования повторяют в порядке выхода, пока любое из следующих условий не станет истиной:

- Начинается новая кодированная видеопоследовательность.
- Изображение в блоке доступа, которое содержит сообщение SEI о прямоугольнике сканирования с тем же значением pan_scan_rect_id, является выходом, имеющим PicOrderCnt( ) больше, чем PicOrderCnt(CurrPic).

pan_scan_rect_repetition_period, равное 0 или 1, указывает, что другое сообщение SEI о прямоугольнике сканирования с тем же значением pan_scan_rect_id может присутствовать, а может и не присутствовать.

pan_scan_rect_repetition_period больше 1 определяет, что информацию о прямоугольнике сканирования повторяют в порядке выхода, пока любое из следующих условий не станет истиной:

- Начинается новая кодированная видеопоследовательность.
- Изображение в блоке доступа, которое содержит сообщение SEI о прямоугольнике сканирования с тем же значением pan_scan_rect_id, является выходом, имеющим PicOrderCnt( ) больше, чем PicOrderCnt(CurrPic) + pan_scan_rect_repetition_period.

pan_scan_rect_repetition_period больше 1 указывает, что другое сообщение SEI о прямоугольнике сканирования с тем же значением pan_scan_rect_id должно присутствовать в блоке доступа для изображения, которое служит выходом с PicOrderCnt( ) меньшим чем или равным PicOrderCnt(CurrPic) + pan_scan_rect_repetition_period, до тех пор пока не начнется новая кодированная видеопоследовательность без такого изображения на выходе.

D.2.4 Семантика сообщения SEI о полезной нагрузке заполнителя

Это сообщение содержит ряд байтов payloadSize значения 0xFF, которое может отбрасываться.

ff_byte должно быть байтом со значением 0xFF.

D.2.5 Семантика сообщения SEI о данных пользователя, зарегистрированных по Рекомендации МСЭ-Т Т.35

Это сообщение содержит данные пользователя, зарегистрированные, как это определено в Рекомендации МСЭ-Т Т.35. Содержание этих данных не определено данной Рекомендацией | Международным стандартом.

itu_t_t35_country_code должно быть байтом, имеющим значение определенного кода страны, согласно Приложению А к Рекомендации МСЭ-Т Т.35.

itu_t_t35_country_code_extension_byte должно быть байтом, имеющим значение определенного кода страны, согласно Приложению В к Рекомендации МСЭ-Т Т.35.

itu_t_t35_payload_byte должно быть байтом, который содержит данные, зарегистрированные, как это определено Рекомендацией МСЭ-Т Т.35.

Код провайдера терминала в соответствии с Рекомендацией МСЭ-Т Т.35 и ориентированный код провайдера терминала должны содержаться в первом или более байтах itu_t_t35_payload_byte в формате, определенном администрацией, которая публикует код провайдера терминала. Любые остающиеся данные itu_t_t35_payload_byte data должны быть данными с синтаксисом и семантикой, которые определены идентифицированными по МСЭ-Т Т.35 кодом страны и кодом провайдера терминала.

D.2.6 Семантика сообщения SEI о незарегистрированных данных пользователя

Это сообщение содержит идентифицированные UUID незарегистрированные данные пользователя, содержание которых не определено этой Рекомендацией | Международным стандартом.

uuid_iso_iec_11578 должно иметь значение, определенное как UUID, согласно процедуре Приложения А ИСО/МЭК 11578:1996.

user_data_payload_byte должно быть байтом, который содержит данные с синтаксисом и семантикой, как это определено производителем UUID.

D.2.7 Семантика сообщения SEI о пункте восстановления

Сообщение SEI о пункте восстановления помогает декодеру определить, когда процесс декодирования обеспечит вывод изображений на экран после инициирования произвольного доступа или после того, как кодер укажет на разомкнутое звено связи в последовательности. Если процесс декодирования начинают в порядке декодирования с блока доступа, связанного с сообщением SEI о пункте восстановления, то обо всех декодированных изображениях в этом или последующем пункте в порядке выхода, определенном в этом сообщении SEI, будет указано как о правильных или приблизительно правильных по содержанию. Декодированные изображения, полученные произвольным доступом к изображению или ранее и связанные с сообщением SEI о пункте восстановления, не требуют исправления содержания до тех пор, пока не поступит указания от пункта восстановления. Действие процесса декодирования, которое начинается с изображения, связанного с сообщением SEI о пункте восстановления, может содержать ссылки на изображения, недоступные буферу декодированного изображения.

Кроме того, используя broken_link_flag, сообщение SEI о пункте восстановления может указать декодеру местоположение некоторых изображений в потоке битов, которые возникают в результате серьезных визуальных дефектов изображения при выводе на экран, если даже процесс декодирования начался в месте расположения предыдущего IDR блока доступа в порядке декодирования.

ПРИМЕЧАНИЕ. – Значение broken_link_flag может быть использовано кодерами, чтобы указать место расположения пункта, после которого процесс декодирования при декодировании некоторых изображений может вызвать ссылки на изображения, которые будучи доступными в процессе декодирования не являются изображениями, которые можно было бы использовать как контрольные, если бы поток битов был бы кодирован изначально (т. е. благодаря операции секционирования, которую выполнили при генерации потока битов).

Пункт восстановления устанавливает, в единицах приращений frame_num, следующего за frame_num текущего блока доступа в месте сообщения SEI.

ПРИМЕЧАНИЕ. – Если в потоке битов присутствует информация HRD, то сообщение SEI о периоде буферизации должно быть объединено с блоком доступа, связанного с сообщением SEI о пункте восстановления, чтобы после произвольного доступа выполнить инициализацию модели буфера HRD.

recovery_frame_cnt определяет пункт восстановления выходных изображений в порядке выхода. Все декодированные изображения в порядке выхода указаны как правильные или приблизительно правильные по содержанию, начиная с положения в порядке выхода контрольного изображения, имеющего frame_num, равным frame_num блоков NAL текущего блока доступа, с приращением recovery_frame_cnt по арифметике модуля MaxFrameNum. Значение recovery_frame_cnt должно быть в диапазоне от 0 до MaxFrameNum – 1 включительно.

exact_match_flag указывает, действительно ли декодированные изображения – в указанном пункте восстановления и далее в порядке выхода, – которые находят, запуская процесс декодирования в блоке доступа, связанного с сообщением SEI о пункте восстановления, должны в точности совпадать с изображениями, которые могли бы быть созданы запуском процесса декодирования в месте расположения предыдущего IDR блока доступа в потоке блоков NAL. Значение 0 указывает, что не требуется точного совпадения, а значение 1 указывает, что совпадение должно быть точным.

Если декодирование начинают с места сообщения SEI о пункте восстановления, все ссылки на недоступность контрольных изображений должны восприниматься как ссылки на изображения, которые содержат только макроблоки, использующие режимы Intra предсказания, и образцы, заданные значениями Y, равным 128, Cb, равным 128, и Cr, равным 128 (средний уровень серого), с целью определения соответствия значения exact_match_flag.

ПРИМЕЧАНИЕ. – При выполнении произвольного доступа декодеры должны учитывать все ссылки на недоступные контрольные изображения как ссылки на изображения, которые сдержат только intra макроблоки и образцы, заданные значениями Y, равным 128, Cb, равным 128, и Cr, равным 128 (средний уровень серого), независимо от значения exact_match_flag.

Если exact_match_flag равно 0, качество аппроксимации в пункте восстановления выбирают процессом кодирования, не определенным этой Рекомендацией | Международным стандартом.

broken_link_flag указывает на присутствие или отсутствие разомкнутого звена связи в потоке блоков NAL в месте сообщения SEI о пункте восстановления. Этому значению присвоена следующая семантика:

– Если broken_link_flag равно 1, то изображения, созданные запуском процесса декодирования в месте предыдущего IDR блока доступа, могут содержать визуальные дефекты, нежелательные до такой степени, что декодированные изображения в этом и последующих блоках доступа, связанных с сообщением SEI о пункте восстановления в порядке декодирования, не смогут быть выведены на экран в порядке выхода до определенного пункта восстановления.

– Иначе (broken_link_flag равно 0), не дано никаких указаний на любое потенциальное присутствие нежелательных визуальных дефектов.

Независимо от значения broken_link_flag следующие до определенного пункта восстановления в порядке выхода изображения определены таким образом, чтобы правильно или приблизительно правильно передать содержание.

ПРИМЕЧАНИЕ. – Если субпоследовательность информации сообщения SEI присутствует вместе с сообщением SEI о пункте восстановления, в котором broken_link_flag равно 1, и если sub_seq_layer_num равно 0, то sub_seq_id должно отличаться от самого последнего значения sub_seq_id для sub_seq_layer_num, равного 0, которое было декодировано до места сообщения SEI о пункте восстановления. Если broken_link_flag равно 0, то значение sub_seq_id в субпоследовательности слоя 0 следует оставить неизменным.

changing_slice_group_idc, равное 0, указывает, что декодированные изображения правильны или приблизительно правильны по содержанию в этом и следующем пункте восстановления в порядке выхода, если все макроблоки первичных кодированных изображений декодированы в период изменения группы секции, т. е. в промежутке между

блоком доступа, связанного с сообщением SEI о пункте восстановления (включительно), и указанным пунктом восстановления (включительно) в порядке декодирования. Значение changing_slice_group_idc должно быть равно 0, если num_slice_groups_minus1 равно 0 в любом первичном кодированном изображении в период изменения группы секции.

Если changing_slice_group_idc равно 1 или 2, то num_slice_groups_minus1 должно быть равно 1, а тип отображения макроблока в группу секции 3, 4 или 5 должен быть применен к каждому первичному кодированному изображению в период изменения группы секции.

changing_slice_group_idc, равное 1, указывает, что в период изменения группы секции никакие значения образцов вне декодированных макроблоков в группе секции 0 не используют для inter предсказания любого макроблока в группе секции 0. Кроме того, changing_slice_group_idc, равное 1, указывает, что если все макроблоки в группе секции 0 в период изменения группы секции декодированы, то декодированные изображения будут правильными или приблизительно правильными по содержанию в этом и в следующем указанном пункте восстановления в порядке выхода, независимо от того, декодирован ли любой макроблок в группе секции 1 в период изменения группы секции.

changing_slice_group_idc, равное 2, указывает, что в период изменения группы секции никакие значения образцов вне декодированных макроблоков в группе секции 1 не используют для inter предсказания любого макроблока в группе секции 1. Кроме того, changing_slice_group_idc, равное 2, указывает, что если все макроблоки в группе секции 1 в период изменения группы секции декодированы, то декодированные изображения будут правильными или приблизительно правильными по содержанию в этом и в следующем указанном пункте восстановления в порядке выхода, независимо от того, декодирован ли любой макроблок в группе секции 1 в период изменения группы секции.

changing_slice_group_idc должно быть в диапазоне от 0 до 2 включительно.

D.2.8 Семантика сообщения SEI о повторной разметке декодированного контрольного изображения

Сообщение SEI о повторной разметке декодированного контрольного изображения используют, чтобы повторить структуру синтаксиса разметки декодированного контрольного изображения, которая находилась в заголовке секции более раннего изображения в последовательности порядка декодирования.

original_idr_flag должно быть равено 1, если структура синтаксиса разметки декодированного контрольного изображения появилась ранее в IDR изображении. Значение original_idr_flag должно быть равно 0, если структура синтаксиса разметки повторного декодированного контрольного изображения не появилась ранее в IDR изображении.

original_frame_num должно быть равно значению frame_num изображения, в котором ранее появилась структура синтаксиса разметки повторного декодированного контрольного изображения. Изображение, отмеченное как original_frame_num, – это предыдущее кодированное изображение с определенным значением frame_num. Значение original_frame_num, использованное для ссылки на изображение с memory_management_control_operation, равным 5, должно быть нулем.

original_field_pic_flag должно быть равно field_pic_flag изображения, в котором ранее появилась структура синтаксиса разметки повторного декодированного контрольного изображения.

original_bottom_field_flag должно быть равно bottom_field_flag изображения, в котором ранее появилась структура синтаксиса разметки повторного декодированного контрольного изображения.

dec_ref_pic_marking( ) должна содержать копию структуры синтаксиса разметки декодированного контрольного изображения, у которого значение frame_num было original_frame_num. Значение nal_unit_type, используемое для соответствия повторного значения dec_ref_pic_marking( ) структуры синтаксиса, должно быть nal_unit_type в заголовке секции(ий) изображения, у которого значение frame_num было original_frame_num (т. е. значение nal_unit_type, использованное в п. 7.3.3.3, должно считаться равным 5, если original_idr_flag равно 1, и не должно считаться равным 5, если original_idr_flag равно 0).

D.2.9 Семантика сообщения SEI о резервном изображении

Это сообщение SEI указывает, что отдельные блоки отображения группы секции, которые называют резервными блоками отображения группы секции, в одном или более декодированных контрольных изображениях имеют сходство с так называемыми блоками отображения группы секции в определенном декодированном изображении, которое называют целевым изображением. Резервный блок отображения группы секции может быть использован для замены близко расположенного неправильно декодированного блока отображения группы секции в целевом изображении. Декодированное изображение, которое содержит резервные блоки отображения группы секции, называют резервным изображением.

Для всех резервных изображений, идентифицированных в сообщении SEI о резервном изображении, значение frame_mbs_only_flag должно быть равно значению frame_mbs_only_flag целевого изображения в том же сообщении SEI. Резервные изображения в сообщении SEI ограничены следующим образом:

– Если целевое изображение – это декодированное поле, то все резервные изображения, идентифицированные в том же сообщении SEI, должно быть декодированными полями.

– Иначе (целевое изображение – это декодированный кадр), все резервные изображения, идентифицированные в том же сообщении SEI, должно быть декодированными кадрами.

Для всех резервных изображений, идентифицированных в сообщении SEI о резервном изображении, значения pic_width_in_mbs_minus1 и pic_height_in_map_units_minus1 должны быть равны значениям pic_width_in_mbs_minus1 и pic_height_in_map_units_minus1, соответственно, целевого изображения в том же сообщении SEI. Изображение, связанное (как это определено в п. 7.4.1.2.3) с этим сообщением, должно появляться в порядке декодирования после целевого изображения.

target_frame_num указывает значение frame_num целевого изображения.

spare_field_flag, равное 0, указывает, что целевое изображение и резервные изображения – это декодированные кадры. Значение spare_field_flag, равное 1, указывает, что целевое изображение и резервные изображения – это декодированные поля.

target_bottom_field_flag, равно 0, указывает, что целевое изображение – это верхнее поле. target_bottom_field_flag, равное 1, указывает, что целевое изображение – это нижнее поле.

Целевое изображение – это декодированное контрольное изображение, чье соответствие первичному кодированному изображению предшествует в порядке декодирования текущему изображению и в котором значения frame_num, field_pic_flag (если присутствует) и bottom_field_flag (если присутствует) равны target_frame_num, spare_field_flag и target_bottom_field_flag, соответственно.

num_spare_pics_minus1 указывает число резервных изображений для определенного целевого изображения. Число резервных изображений равно num_spare_pics_minus1 + 1. Значение num_spare_pics_minus1 должно быть в диапазоне от 0 до 15 включительно.

delta_spare_frame_num[i **]** используют для идентификации резервного изображения, которое содержит i-й набор резервных блоков отображения группы секции, здесь и далее называемых i-м резервным изображением, как определено ниже. Значение delta_spare_frame_num[i] должно быть в диапазоне от 0 до MaxFrameNum – 1 – !spare_field_flag включительно.

Значение frame_num i-ого резервного изображения, т. е. spareFrameNum[i], находят следующим образом для всех значений i от 0 до num_spare_pics_minus1 включительно:

```
candidateSpareFrameNum = target_frame_num – !spare_field_flag
for ( i = 0; i <= num_spare_pics_minus1; i++ ) {
    if( candidateSpareFrameNum < 0 )
        candidateSpareFrameNum = MaxFrameNum – 1
    spareFrameNum[ i ] = candidateSpareFrameNum – delta_spare_frame_num[ i ]
    if( spareFrameNum[ i ] < 0 )
        spareFrameNum[ i ] = MaxFrameNum + spareFrameNum[ i ]
    candidateSpareFrameNum = spareFrameNum[ i ] – !spare_field_flag
}
```
(D-3)

spare_bottom_field_flag[i **]**, равное 0, указывает, что i-е резервное изображение – это верхнее поле. spare_bottom_field_flag[i], равное 1, указывает, что i-ое резервное изображение – это нижнее поле.

Нулевое (0-е) резервное изображение – это декодированное контрольное изображение, чье соответствие первичному кодированному изображению предшествует в порядке декодирования целевому изображению, и в котором значения frame_num, field_pic_flag (если присутствует) и bottom_field_flag (если присутствует) равны spareFrameNum[0], spare_field_flag и spare_bottom_field_flag[0], соответственно. i-е резервное изображение – это декодированное контрольное изображение, чье соответствие первичному кодированному изображению в порядке декодирования предшествует (i – 1)-е резервное изображение и в котором значения frame_num, field_pic_flag (если присутствует) и bottom_field_flag (если присутствует) равны spareFrameNum[i], spare_field_flag и spare_bottom_field_flag[i], соответственно.

spare_area_idc[i **]** указывает на метод, использованный для идентификации резервных блоков отображения группы секции в i-м резервном изображении. Значение spare_area_idc[i] должно быть в диапазоне от 0 до 2 включительно. Значение spare_area_idc[i], равное 0, указывает, что все блоки отображения группы секции в i-м резервном изображении – это резервные блоки. Значение spare_area_idc[i], равное 1, указывает, что значение элемента синтаксиса spare_unit_flag[i][j] используют для идентификации резервных блоков отображения группы секции. Значение spare_area_idc[i], равное 2, указывает, что элемент синтаксиса zero_run_length[i][j] используют для отыскания значения spareUnitFlagInBoxOutOrder[i][j], как описано ниже.

spare_unit_flag[i **][** j **]**, равное 0, указывает, что j-й блок отображения группы секции при растровом порядке сканирования в i-м резервном изображении – это резервный блок. Значение spare_unit_flag[i][j] равное 1, указывает, что j-й блок отображения группы секции при растровом порядке сканирования в i-м резервном изображении – это не резервный блок.

zero_run_length[i **][** j **]** используют для отыскания значения spareUnitFlagInBoxOutOrder[i][j], если spare_area_idc[i] равно 2. В этом случае резервные блоки отображения группы секции, идентифицированные в spareUnitFlagInBoxOutOrder[i][j], появляются в счетчике по часовой стрелке, как это определено в п. 8.2.2.4, для каждого резервного изображения. Значение spareUnitFlagInBoxOutOrder[i][j], равное 0, указывает, что j-й блок отображения группы секции в счетчике по часовой стрелке в i-м резервном изображении – это резервный блок. Значение spareUnitFlagInBoxOutOrder[i][j], равное 1, указывает, что j-й блок отображения группы секции в счетчике по часовой стрелке в i-м резервном изображении – это не резервный блок.

Если spare_area_idc[0], равно 2, то spareUnitFlagInBoxOutOrder[0][j] находят следующим образом:

```
for( j = 0, loop = 0; j < PicSizeInMapUnits; loop++ ) {
  for( k = 0; k < zero_run_length[ 0 ][ loop ]; k++ )
    spareUnitFlagInBoxOutOrder[ 0 ][ j++ ] = 0
  spareUnitFlagInBoxOutOrder[ 0 ][ j++ ] = 1
}
```
(D-4)

Если spare_area_idc[i] равно 2 и значение i больше 0, то spareUnitFlagInBoxOutOrder[i][j] находят следующим образом:

```
for( j = 0, loop = 0; j < PicSizeInMapUnits; loop++ ) {
  for( k = 0; k < zero_run_length[ i ][ loop ]; k++ )
    spareUnitFlagInBoxOutOrder[ i ][ j ] = spareUnitFlagInBoxOutOrder[ i – 1 ][ j++ ]          (D-5)
  spareUnitFlagInBoxOutOrder[ i ][ j ] = !spareUnitFlagInBoxOutOrder[ i – 1 ][ j++ ]
}
```

D.2.10 Семантика сообщения SEI о сценической информации

Сцену и сценический переход здесь и далее определяют как набор последовательных изображений в порядке выхода.

ПРИМЕЧАНИЕ. – Декодированные изображения в одной сцене, как правило, имеют одно и то же содержание. Сообщение SEI о сценической информации используют для отметки изображений в идентификаторах сцены и для указаний об изменениях сцены. Это сообщение определяет, как для отмеченных изображений были созданы источники изображений. Декодер может использовать эту информацию, чтобы выбрать подходящий алгоритм для подавления ошибок при передаче. Например, может быть использован специальный алгоритм для подавления ошибок передачи, которые встречаются в изображениях при постепенных изменениях сцены. Более того, сообщение SEI о сценической информации может быть использовано как способ, определенный приложением, например, для индексации сцен в кодированной последовательности.

Сообщение SEI о сценической информации помечает все изображения в порядке декодирования от исходного кодированного изображения до того (включительно), с которым связано данное сообщение SEI, как это определено в п. 7.4.1.2.3, а также до того (исключая) исходного кодированного изображения, с которым связано следующее сообщение SEI о сценической информации (если такое присутствует) в порядке декодирования, или (иначе) до последнего блока доступа в потоке битов (включительно). Эти изображения здесь и далее считаются целевыми изображениями.

scene_info_present_flag, равное 0, указывает, что не определены сцена или сценический переход, к которым относятся целевые изображения. Значение scene_info_present_flag, равное 1, указывает, что данные целевые изображения принадлежат к этой же сцене или сценическому переходу.

scene_id идентифицирует сцену, к которой принадлежат целевые изображения. Если значение scene_transition_type целевого изображения меньше 4 и предыдущее изображение в порядке выхода отмечено значением scene_transition_type, меньшим 4, а значение scene_id такое же, как значение scene_id предыдущего изображения в порядке выхода, то это указывает, что исходная сцена для целевых изображений и исходная сцена для предыдущего изображения (в порядке выхода) рассматриваются кодером как одна и та же сцена. Если значение scene_transition_type целевого изображения больше 3 и предыдущее изображение в порядке выхода отмечено значением scene_transition_type, меньшим 4, а значение scene_id такое же, как значение scene_id предыдущего изображения в порядке выхода, то это указывает, что исходная сцена для целевых изображений и исходная сцена для предыдущего изображения (в порядке выхода) рассматриваются кодером как одна и та же сцена. Если значение scene_id не равно значению scene_id предыдущего изображения в порядке выхода, это означает, что целевые изображения и предыдущее изображение (в порядке выхода) рассматриваются кодером как разные сцены.

Значение scene_id должно быть в диапазоне от 0 до $2^{32} – 1$ включительно. Значения scene_id в диапазоне от 0 до 255 включительно и в диапазоне от 512 до $2^{31} – 1$ включительно может быть использовано, как определено приложением. Значения scene_id в диапазоне от 256 до 511 включительно и в диапазоне от 2^{31} до $2^{32} – 1$ включительно зарезервированы для будущего использования МСЭ-Т | ИСО/МЭК. Декодеры, учитывающие значения scene_id в диапазоне от 256 до 511 включительно или в диапазоне от 2^{31} до $2^{32} – 1$ включительно, должны их игнорировать (удалять из потока битов и отбрасывать).

scene_transition_type определяет, к какому типу сценического перехода (если такой имеется) принадлежат целевые изображения. Действующие значения scene_transition_type показаны в таблице D-4.

Таблица D-4 – Значения scene_transition_type

Значение	**Описание**
0	Нет перехода
1	Затухание к черному
2	Увеличение от черного
3	Неопределенный переход от или к постоянному цвету
4	Растворение
5	Стирание
6	Неопределенная смесь из двух сцен

Если scene_transition_type больше 3, целевые изображения включают содержание обеих сцен, отмеченных своими значениями scene_id, а также следующую сцену в порядке выхода, которая отмечена second_scene_id (см. ниже). Термин "текущая сцена" используют для указания на сцену, отмеченную как scene_id. Термин "следующая сцена"

используют для указания на сцену, отмеченную как second_scene_id. Нет необходимости каждое последующее изображение в порядке выхода отмечать значением scene_id, равным second_scene_id в текущем сообщении SEI.

Типы сценических переходов определены следующим образом:

"Нет перехода" определяет, что целевые изображения не участвуют в плавном сценическом переходе.

ПРИМЕЧАНИЕ. – Если два последовательных изображения в порядке выхода имеют тип scene_transition_type, равный 0, и разные значения scene_id, то происходит разделение сцены между этими двумя изображениями.

"Затухание к черному" указывает, что целевые изображения являются частью последовательности изображений в порядке выхода, вовлеченной в сценический переход затухания к черному, т. е. яркость образцов сцены постепенно падает до нуля, а цветность образцов сцены постепенно достигает 128.

ПРИМЕЧАНИЕ. – Если два изображения отмечены как принадлежащие к одному сценическому переходу, а тип их сценического перехода – "Затухание к черному", то более позднее изображение в порядке выхода будет более темным, чем предыдущее.

"Увеличение от черного" указывает, что целевые изображения являются частью последовательности изображений в порядке выхода, вовлеченной в сценический переход увеличения от черного, т. е. яркость образцов сцены постепенно возрастает от нуля, а цветность образцов сцены постепенно отклоняется от 128.

ПРИМЕЧАНИЕ. – Если два изображения отмечены как принадлежащие к одному сценическому переходу, а тип их сценического перехода – "Увеличение от черного", то более позднее изображение в порядке выхода будет более светлым, чем предыдущее.

"Растворение" указывает, что значения образца каждого целевого изображения (перед кодированием) были созданы вычислением суммы совместных взвешенных значений образцов изображений из текущей сцены и из следующей сцены. Вес текущей сцены постепенно уменьшается от полного уровня до нуля, а вес следующей сцены постепенно возрастает от нуля до полного уровня. Если два изображения отмечены как принадлежащие к одному сценическому переходу, а их тип scene_transition_type – это "Растворение", то вес текущей сцены изображения в порядке выхода будет меньше веса текущего сцены предыдущего изображения, а вес следующей сцены в порядке выхода – больше веса следующей сцены изображения.

"Стирание" указывает, что некоторые значения образца каждого целевого изображения (перед кодированием) были созданы копированием совместных значений образца изображения в текущей сцене и оставшихся значений образцов каждого из целевых изображений (перед кодированием), которые были созданы копированием совместных значений образца изображения в следующей сцене. Если два изображения отмечены как принадлежащие к одному сценическому переходу, а их тип scene_transition_type – это "Стирание", то число образцов, скопированных из следующей сцены более позднего изображения (в порядке выхода), будет больше числа образцов, скопированных из следующей сцены для предыдущего изображения.

second_scene_id идентифицирует следующую сцену в постепенном сценическом переходе, в котором участвуют целевые изображения. Значение second_scene_id не должно быть равно значению scene_id. Значение second_scene_id не должно быть равно значению scene_id в предыдущем изображении в порядке выхода. Если следующее изображение в порядке выхода помечено значением scene_transition_type, меньшим 4, а значение second_scene_id – то же, что и scene_id следующего изображения (в порядке выхода), это указывает кодеру считать одну из исходных сцен целевым изображением, а исходную сцену следующего изображения (в порядке выхода) – той же самой сценой. Если значение second_scene_id не равно значению scene_id или second_scene_id (если такие присутствуют) следующего изображения в порядке выхода, это указывает на то, что кодер считает целевые изображения и следующее изображение (в порядке выхода) принадлежащими к разным исходным сценам.

Если значение scene_id изображения равно значению scene_id следующего изображения в порядке выхода, а значение scene_transition_type обоих этих изображений меньше 4, это указывает кодеру считать эти два изображения принадлежащими к одной и той же исходной сцене. Если значения scene_id, scene_transition_type и second_scene_id (если присутствует) изображения равно значениям scene_id, scene_transition_type и second_scene_id (соответственно) следующего изображения (в порядке выхода), а значение scene_transition_type больше 0, это указывает, что кодер считает эти два изображения возникающими из одного и того же исходного постепенного сценического перехода.

Значение second_scene_id должно быть в диапазоне от 0 до $2^{32} - 1$ включительно. Значения second_scene_id в диапазоне от 0 до 255 включительно и в диапазоне от 512 до $2^{31} - 1$ включительно могут быть использованы, как определено приложением. Значения second_scene_id в диапазоне от 256 до 511 включительно и в диапазоне от 2^{31} до $2^{32} - 1$ включительно зарезервированы для будущего использования МСЭ-Т | ИСО/МЭК. Декодеры, учитывающие значения second_scene_id в диапазоне от 256 до 511 включительно или в диапазоне от 2^{31} до $2^{32} - 1$ включительно, должны их игнорировать (удалять из потока битов и отбрасывать).

D.2.11 Семантика сообщения SEI об информации субпоследовательности

Сообщение SEI об информации субпоследовательности используют для указания положения изображения в иерархии зависимости данных, которые состоят из субпоследовательностей и слоев субпоследовательностей.

Слой субпоследовательности содержит в последовательности подмножество кодированных изображений. Слои субпоследовательностей нумеруют неотрицательными целыми числами. Слой, имеющий больший номер, является высшим по отношению к слою с меньшим номером. Слои упорядочены по их иерархической зависимости друг от друга таким образом, что любое изображение в слое не должно быть предсказано от любого изображения более высокого слоя.

ПРИМЕЧАНИЕ. – Другими словами, любое изображение в слое 0 не должно быть предсказано от любого изображения в слое 1 или более высоком. Изображения в слое 1 могут быть предсказаны из слоя 0, изображения в слое 2 могут быть предсказаны из слоев 0 и 1, и т. д.

ПРИМЕЧАНИЕ: Ожидается, что с увеличением номера слоя будет происходить последовательное повышение качества.

Субпоследовательность – это множество кодированных изображений в слое субпоследовательности. Изображение должно располагаться в одном и только в одном слое субпоследовательности. Любое изображение в субпоследовательности не должно быть предсказано от любого изображения в другой субпоследовательности в том же или более высоком слое субпоследовательности. Субпоследовательность в слое 0 может быть декодирована независимо от любого изображения, которое не принадлежит к этой субпоследовательности.

Сообщение SEI об информации субпоследовательности касается текущего блока доступа. Исходное кодированное изображение в блоке доступа здесь и далее считается текущим изображением.

Сообщение SEI об информации субпоследовательности не должно поступать до тех пор, пока значение gaps_in_frame_num_value_allowed_flag в установке параметров последовательности, на которую ссылается сообщение SEI, не будет равно 1.

sub_seq_layer_num определяет номер слоя субпоследовательности текущего изображения. Если sub_seq_layer_num больше 0, операции управления памятью не должны использоваться в любом заголовке секции текущего изображения. Если текущее изображение расположено в субпоследовательности с первым изображением в порядке декодирования – IDR изображением, значение sub_seq_layer_num должно быть равно 0. Для непарных контрольных полей значение sub_seq_layer_num должно быть равно 0. Значение sub_seq_layer_num должно быть в диапазоне от 0 до 255 включительно.

sub_seq_id идентифицирует субпоследовательность в слое. Если текущее изображение расположено в субпоследовательности с первым изображением в порядке декодирования – IDR изображением, – значение sub_seq_id должно быть таким же, как значение idr_pic_id IDR изображения. Значение sub_seq_id должно быть в диапазоне от 0 до 65 535 включительно.

first_ref_pic_flag, равное 1, определяет, что текущее изображение – это первое контрольное изображение субпоследовательности в порядке декодирования. Если текущее изображение – это не первое контрольное изображение субпоследовательности в порядке декодирования, first_ref_pic_flag должно быть равно 0.

leading_non_ref_pic_flag, равное 1, определяет, что текущее изображение – это неконтрольное изображение, предшествующее любому контрольному изображению в порядке декодирования в субпоследовательности, или что субпоследовательность содержит неконтрольные изображения. Если текущее изображение – это контрольное изображение или если текущее изображение – это неконтрольное изображение, следующее, по крайней мере, за одним контрольным изображением в порядке декодирования в субпоследовательности, то leading_non_ref_pic_flag должно быть равно 0.

last_pic_flag, равное 1, указывает, что текущее изображение – это не последнее изображение субпоследовательности (в порядке декодирования), включая все контрольные и неконтрольные изображения субпоследовательности. Если текущее изображение – это не последнее изображение субпоследовательности (в порядке декодирования), last_pic_flag должно быть равно 0.

Текущее изображение присваивают субпоследовательности следующим образом:

- Если одно или более условий – истина, то текущее изображение – это первое изображение субпоследовательности в порядке декодирования:
  - отсутствует более раннее изображение в порядке декодирования, отмеченное таким же значением sub_seq_id and sub_seq_layer_num, что и в текущем изображении;
  - значение leading_non_ref_pic_flag равно 1, а значение leading_non_ref_pic_flag равно 0 в предыдущем изображении в порядке декодирования с теми же значениями sub_seq_id и sub_seq_layer_num, что и в текущем изображении;
  - значение first_ref_pic_flag равно 1, а значение leading_non_ref_pic_flag равно 0 в предыдущем изображении в порядке декодирования с теми же значениями sub_seq_id и sub_seq_layer_num, что и в текущем изображении;
  - значение last_pic_flag равно 1 в предыдущем изображении в порядке декодирования с теми же значениями sub_seq_id и sub_seq_layer_num, что и в текущем изображении.
- Иначе текущее изображение принадлежит к той же субпоследовательности, что и предыдущее изображение в порядке декодирования с теми же значениями sub_seq_id и sub_seq_layer_num, что и в текущем изображении.

sub_seq_frame_num_flag, равное 0, указывает, что sub_seq_frame_num отсутствует. Значение sub_seq_frame_num_flag, равное 1, указывает, что sub_seq_frame_num присутствует.

sub_seq_frame_num должно быть равно 0 для первого контрольного изображения субпоследовательности и для любого неконтрольного изображения, предшествующего первому контрольному изображению субпоследовательности в порядке декодирования. Значение sub_seq_frame_num ограничено следующим образом:

- Если текущеее изображение – это не второе поле дополнительной пары полей, то sub_seq_frame_num должно быть увеличено на 1 в операции по модулю MaxFrameNum относительно предыдущего контрольного изображения в порядке декодирования, которое принадлежит к этой субпоследовательности.

– Иначе (текущее изображение – это второе поле дополнительной пары полей), значение sub_seq_frame_num должно быть таким же, как значение sub_seq_frame_num первого поля дополнительной пары полей.

sub_seq_frame_num должно быть в диапазоне от 0 до MaxFrameNum – 1 включительно.

Если текущее изображение – это IDR изображение, то в слое субпоследовательностей 0 должна быть запущена новая субпоследовательность. Таким образом, sub_seq_layer_num должно быть равно 0, идентификатор sub_seq_id должен отличаться от его значения в предыдущей субпоследовательности слоя 0, first_ref_pic_flag должно быть равно 1, а leading_non_ref_pic_flag должно быть равно 0.

Если сообщение SEI об информации субпоследовательности присутствует для обоих кодированных полей дополнительной пары полей, то значения sub_seq_layer_num, sub_seq_id, leading_non_ref_pic_flag и sub_seq_frame_num (если присутствуют) должны быть идентичны для обоих изображений.

Если сообщение SEI об информации субпоследовательности присутствует только для одного кодированного поля дополнительной пары полей, то значения sub_seq_layer_num, sub_seq_id, leading_non_ref_pic_flag и sub_seq_frame_num (если присутствуют) должны быть также применимы к другому кодированному полю дополнительной пары полей.

D.2.12 Семантика сообщения SEI о характеристиках слоя субпоследовательности

Сообщение SEI о характеристиках слоя субпоследовательности определяет характеристики слоя субпоследовательности.

num_sub_seq_layers_minus1 плюс 1 определяет номер слоя субпоследовательности в данной последовательности. Значение num_sub_seq_layers_minus1 должно быть в диапазоне от 0 до 255 включительно.

Пара значений average_bit_rate и average_frame_rate характеризует каждый слой субпоследовательности. Первая пара average_bit_rate и average_frame_rate определяет характеристики слоя субпоследовательности 0. Если присутствует, то вторая пара определяет совместно характеристики слоя субпоследовательности 0 и 1. Каждая пара в порядке декодирования определяет характеристики диапазона слоя субпоследовательности от номера слоя 0 до номера слоя, определенного циклом счетчика. Эти значения вступают в силу от пункта, где они были декодированы, и до обновления этих декодированных значений.

accurate_statistics_flag, равное 1, указывает, что значения average_bit_rate и average_frame_rate округлены до статистически корректных значений. Значение accurate_statistics_flag, равное 0, указывает, что average_bit_rate и average_frame_rate – это оценки, которые могут отклоняться от точных значений.

Если accurate_statistics_flag равно 0, качество аппроксимации, использованное при вычислении значений average_bit_rate и average_frame_rate, выбрано процессом кодирования и не определено в этой Рекомендации | Международном стандарте.

average_bit_rate указывает среднюю скорость битов в блоках 1000 бит/с. При вычислениях принимают во внимание все блоки NAL в диапазоне слоев определенных выше субпоследовательностей. Общую среднюю скорость битов находят в соответствии со временем удаления блока доступа, определенного в Приложении C к этой Рекомендации | Международному стандарту. Далее величина bTotal – это число битов во всех блоках NAL, следующих за сообщением SEI о характеристиках слоя субпоследовательности (включая биты блоков NAL текущего блока доступа) и предшествующих следующему блоку доступа (в порядке декодирования), включая сообщение SEI о характеристиках слоя субпоследовательности (если такое имеется), или (иначе) – это конец потока. Величина t_1 – это время удаления (в секундах) текущего блока доступа, а t_2 – это время удаления (в секундах) самого последнего блока доступа (в порядке декодирования) перед следующим сообщением SEI о характеристиках слоя субпоследовательности (если такое имеется), или (иначе) – это конец потока.

Если accurate_statistics_flag равно 1, должны быть выполнены следующие условия:

– Если t_1 не равно t_2, следующее условие должно быть истиной:

$$average\_bit\_rate == Round(bTotal \div ((t_2 - t_1) * 1000))). \quad (D\text{-}6)$$

– Иначе (t_1 равно t_2), следующее условие должно быть истиной:

$$average\_bit\_rate == 0. \quad (D\text{-}7)$$

average_frame_rate указывает среднюю скорость кадров в блоках кадров/(256 секунд). При вычислениях принимают во внимание все блоки NAL в диапазоне слоев определенных выше субпоследовательностей. Далее величина fTotal – это число кадров, дополнительных пар полей и непарных полей между текущим изображением (включительно) и следующим сообщением SEI о характеристиках слоя субпоследовательности (если такое имеется), или (иначе) – это конец потока. Величина t_1 – это время удаления (в секундах) текущего блока доступа, а t_2 – это время удаления (в секундах) самого последнего блока доступа (в порядке декодирования) перед следующим сообщением SEI о характеристиках слоя субпоследовательности (если такое имеется), или (иначе) – это конец потока.

Если accurate_statistics_flag равно 1, должны быть выполнены следующие условия:

– Если t_1 не равно t_2, следующее условие должно быть истиной:

$$average\_frame\_rate == Round(fTotal * 256 \div (t_2 - t_1)). \quad (D\text{-}8)$$

– Иначе (t_1 равно t_2), следующее условие должно быть истиной:

average_frame_rate = = 0. (D-9)

D.2.13 Семантика сообщения SEI о характеристиках субпоследовательности

Сообщение SEI о характеристиках субпоследовательности указывает на характеристики субпоследовательности. Оно также указывает на зависимости inter предсказания между субпоследовательностями. Это сообщение должно содержаться в первом блоке доступа в порядке декодирования субпоследовательности, к которой относится сообщение SEI. Эту субпоследовательность здесь и далее считают целевой субпоследовательностью.

sub_seq_layer_num идентифицирует номер слоя данной субпоследовательности по отношению к целевой. Значение sub_seq_layer_num должно быть в диапазоне от 0 до 255 включительно.

sub_seq_id идентифицирует целевую субпоследовательность. Значение sub_seq_id должно быть в диапазоне от 0 до 65535 включительно.

duration_flag, равное 0, указывает, что продолжительность целевой субпоследовательности не определена.

sub_seq_duration определяет продолжительность целевой субпоследовательности для тактовой частоты задающего генератора 90 кГц.

average_rate_flag, равное 0, указывает, что средняя скорость битов и средняя скорость кадров целевой субпоследовательности не определены.

accurate_statistics_flag указывает, насколько надежны значения average_bit_rate и average_frame_rate. accurate_statistics_flag, равное 1, указывает что average_bit_rate и average_frame_rate округлены от статистически корректного значения. Значение accurate_statistics_flag, равное 0, указывает, что average_bit_rate и average_frame_rate – это оценки, которые могут отклоняться от точных значений.

average_bit_rate указывает среднюю скорость битов в блоках 1000 бит/с целевой субпоследовательности. При вычислениях принимают во внимание все блоки NAL целевой субпоследовательности. Среднюю скорость битов находят в соответствии с временем удаления блока доступа, определенного в п. C.1.2. Далее nB – это число битов в всех блоках NAL в субпоследовательности. t_1 – это время удаления (в секундах) текущего блока доступа субпоследовательности (в порядке декодирования), а t_2 – это время удаления (в секундах) самого последнего блока доступа субпоследовательности (в порядке декодирования).

Если accurate_statistics_flag равно 1, то должно быть выполнены следующее условия:

– Если t_1 не равно t_2, то следующее условие должно быть истиной:

average_bit_rate = = Round(nB ÷ ((t_2 – t_1) * 1000)). (D-10)

– Иначе (t_1 равно t_2), следующее условие должно быть истиной:

average_bit_rate = = 0. (D-11)

average_frame_rate указывает среднюю скорость кадров в блоках кадров/(256 секунд) целевой субпоследовательности. При вычислениях принимают во внимание все блоки NAL целевой субпоследовательности. Среднюю скорость кадров находят в соответствии с временем удаления блока доступа, определенного в п. C.1.2. Далее fC – это число кадров, дополнительных пар полей и непарных полей в субпоследовательности. t_1 – это время удаления (в секундах) текущего блока доступа субпоследовательности (в порядке декодирования), а t_2 – это время удаления (в секундах) самого последнего блока доступа субпоследовательности (в порядке декодирования).

Если accurate_statistics_flag равно 1, то должны быть выполнены следующие условия:

– Если t_1 не равно t_2, то следующее условие должно быть истиной:

average_frame_rate = = Round(fC * 256 ÷ (t_2 – t_1)). (D-12)

– Иначе (t_1 равно t_2), следующее условие должно быть истиной:

average_frame_rate = = 0. (D-13)

num_referenced_subseqs определяет число субпоследовательностей, которые содержат изображения, использованные как контрольные изображения для inter предсказания изображений целевой субпоследовательности. Значение num_referenced_subseqs должно быть в диапазоне от 0 до 255 включительно.

ref_sub_seq_layer_num, **ref_sub_seq_id** и **ref_sub_seq_direction** идентифицируют субпоследовательность, которая содержит изображения, использованные как контрольные изображения для inter предсказания изображений целевой субпоследовательности. В зависимости от ref_sub_seq_direction применяют следующее:

– Если ref_sub_seq_direction равно 0, множество возможных субпоследовательностей состоит из таких субпоследовательностей, у которых sub_seq_id равно ref_sub_seq_id и которые расположены в слое субпоследовательностей с sub_seq_layer_num равным ref_sub_seq_layer_num, а первое изображение в порядке декодирования предшествует первому изображению целевой субпоследовательности в порядке декодирования.
– Иначе (ref_sub_seq_direction равно 1), множество возможных субпоследовательностей состоит из таких субпоследовательностей, у которых sub_seq_id равно ref_sub_seq_id и которые расположены в слое субпоследовательностей с sub_seq_layer_num равным ref_sub_seq_layer_num, а первое изображение в порядке декодирования следует за первым изображением целевой субпоследовательности в порядке декодирования.

Субпоследовательность, использованная как контрольная для целевой субпоследовательности, является одним из кандидатов, у которых первое изображение в порядке декодирования будет ближайшим к первому изображению целевой субпоследовательности.

D.2.14 Семантика сообщения SEI о полностью фиксированном кадре

Сообщение SEI о полностью фиксированном кадре указывает, что все содержание до отображения видеокадра на экране в порядке выхода следует сохранять без изменений, т. е. без обновления отображения, используя содержание текущего декодированного изображения.

full_frame_freeze_repetition_period указывает, должно ли другое сообщение SEI о полностью фиксированном кадре присутствовать в потоке битов и определяет порядок счета интервала изображения, внутри которого может присутствовать другое сообщение SEI о полностью фиксированном кадре или сообщение SEI об отмене режима полностью фиксированного кадра. Значение full_frame_freeze_repetition_period должно быть в диапазоне от 0 до 16 384 включительно.

full_frame_freeze_repetition_period, равное 0, определяет, что сообщение SEI о полностью фиксированном кадре применимо только к текущему декодированному изображению.

full_frame_freeze_repetition_period, равное 1, определяет, что сообщение SEI о полностью фиксированном кадре повторяют в порядке выхода до тех пор, пока любое из следующих условий является истиной:

– Начинается новая кодированная видеопоследовательность.
– Изображение в блоке доступа, в котором содержится сообщение SEI о полностью фиксированном кадре, является выходом со значением PicOrderCnt( ) большим, чем PicOrderCnt(CurrPic).

full_frame_freeze_repetition_period больше 1 определяет, что сообщение SEI о полностью фиксированном кадре повторяют, пока любое из следующих условий является истиной:

– Начинается новая кодированная видеопоследовательность.
– Изображение в блоке доступа, в котором содержится сообщение SEI о полностью фиксированном кадре или об отмене режима полностью фиксированного кадра, является выходом со значением PicOrderCnt( ) большим, чем PicOrderCnt(CurrPic) + full_frame_freeze_repetition_period.

full_frame_freeze_repetition_period больше 1 указывает, что другое сообщение SEI о полностью фиксированном кадре или об отмене режима полностью фиксированного кадра должно присутствовать в блоке доступа для изображения, которое является выходом со значением PicOrderCnt( ) меньшим чем или равным PicOrderCnt(CurrPic) + full_frame_freeze_repetition_period, до тех пор пока не начнется новая кодированная видеопоследовательность без такого изображения.

D.2.15 Семантика сообщения SEI об отмене режима полностью фиксированного кадра

Сообщение SEI об отмене режима полностью фиксированного кадра указывает, что следует возобновить обновление отображенного видеокадра, начиная с содержания текущего декодированного изображения и продолжая последующими изображениями в порядке выхода. Сообщение SEI о полностью фиксированном кадре отменяет действие любого сообщения SEI о полностью фиксированном кадре, которое было отправлено с изображениями, предшествующими текущему изображению в порядке выхода.

D.2.16 Семантика сообщения SEI о стоп-кадре

Сообщение SEI о стоп-кадре указывает, что текущий кадр отмечен для использования, как это определено приложением, в качестве стоп-кадра неподвижного видеоизображения.

snapshot_id определяет идентификационный номер стоп-кадра. Значение snapshot_id должно быть в диапазоне от 0 до $2^{32} – 1$ включительно.

Значения snapshot_id в диапазоне от 0 до 255 включительно и в диапазоне от 512 до $2^{31} – 1$ включительно может быть использовано, как определено приложением. Значения snapshot_id в диапазоне от 256 до 511 включительно и в диапазоне от 2^{31} до $2^{32} – 1$ включительны зарезервированы для будущего использования МСЭ-Т | ИСО/МЭК. Декодеры, учитывающие значение snapshot_id в диапазоне от 256 до 511 включительно или в диапазоне от 2^{31} до $2^{32} – 1$ включительно, должны игнорировать snapshot_id (удалять из потока битов и отбрасывать).

D.2.17 Семантика сообщения SEI о запуске сегмента последовательного улучшения

Сообщение SEI о запуске сегмента последовательного улучшения определяет начало ряда следующих друг за другом кодированных изображений, помеченных как текущее изображение, за которыми идет последовательность одного или более изображений, характерных в большей мере улучшением качества текущего изображения, чем представлением непрерывной сцены движения.

Помеченный набор следующих друг за другом кодированных изображений должен продолжаться, пока одно из следующих условий является истиной. Если одно из нижеперечисленных условий становится истиной, то следующая секция, предназначенная для декодирования, больше не принадлежит помеченному набору следующих друг за другом кодированных изображений:

1. Следующая секция, предназначенная для декодирования, принадлежит IDR изображению.
2. Значение num_refinement_steps_minus1 больше 0, а значение frame_num следующей предназначенной для декодирования секции – это (currFrameNum + num_refinement_steps_minus1 + 1) % MaxFrameNum, где currFrameNum – значение frame_num изображения в блоке доступа, содержащем сообщение SEI.
3. Значение num_refinement_steps_minus1 равно 0, и декодирован конец сообщения SEI о запуске сегмента последовательного улучшения с тем же значением progressive_refinement_id, что и в этом сообщение SEI.

Порядок декодирования изображений в помеченном наборе следующих друг за другом изображений должен быть таким же, как порядок на выходе. Значение **progressive_refinement_id** указывает на идентификационный номер последовательной операции улучшения. Значение progressive_refinement_id должно быть в диапазоне от 0 до $2^{32} – 1$ включительно.

Значения progressive_refinement_id в диапазоне от 0 до 255 включительно и в диапазоне от 512 до $2^{31} – 1$ включительно могут быть использованы, как это определено приложением. Значения progressive_refinement_id в диапазоне от 256 до 511 включительно и в диапазоне от 2^{31} до $2^{32} – 1$ включительно зарезервированы для будущего использования МСЭ-Т | ИСО/МЭК. Декодеры, учитывающие значение progressive_refinement_id в диапазоне от 256 до 511 включительно или в диапазоне от 2^{31} до $2^{32} – 1$ включительно, должны игнорировать progressive_refinement_id (удалять из потока битов и отбрасывать).

num_refinement_steps_minus1 определяет число контрольных кадров в помеченном наборе следующих друг за другом кодированных изображений следующим образом:

– Если num_refinement_steps_minus1 равно 0, число контрольных кадров в помеченном наборе следующих друг за другом кодированных изображений неизвестно.
– Иначе число контрольных кадров в помеченном наборе следующих друг за другом кодированных изображений равно num_refinement_steps_minus1 + 1.

num_refinement_steps_minus1 должно быть в диапазоне от 0 до MaxFrameNum – 1 включительно.

D.2.18 Семантика сообщения SEI об окончании сегмента последовательного улучшения

Сообщение SEI об окончании сегмента последовательного улучшения заканчивает набор следующих друг за другом кодированных изображений, которые были помечены для использования сообщения SEI о запуске сегмента последовательного улучшения как исходного изображения, за которым поступала последовательность одного или более изображений с улучшением качества исходного изображения. Этим заканчивается текущее изображение.

progressive_refinement_id определяет идентификационный номер операции последовательного улучшения. Значение progressive_refinement_id должно быть в диапазоне от 0 до $2^{32} – 1$ включительно.

Сообщение SEI об окончании сегмента последовательного улучшения определяет конец любого сегмента последовательного улучшения, первоначально инициированного сообщением SEI о запуске, с тем же значением progressive_refinement_id.

Значения progressive_refinement_id в диапазоне от 0 до 255 включительно и в диапазоне от 512 до $2^{31} – 1$ включительно могут быть использованы, как это определено приложением. Значения progressive_refinement_id в диапазоне от 256 до 511 включительно и в диапазоне от 2^{31} до $2^{32} – 1$ включительно зарезервированы для будущего использования МСЭ-Т | ИСО/МЭК. Декодеры, учитывающие значение progressive_refinement_id в диапазоне от 256 до 511 включительно или в диапазоне от 2^{31} до $2^{32} – 1$ включительно, должны игнорировать progressive_refinement_id (удалять из потока битов и отбрасывать).

D.2.19 Семантика сообщения SEI о наборе группы секций ограниченного движения

Это сообщение SEI указывает, что inter предсказание в границах группы секции ограничено, как определено ниже. Если таковое присутствует, это сообщение должно появляться только там, где это связано с блоком доступа IDR, как это определено в п. 7.4.1.2.3.

Набор целевых изображений для этого сообщения SEI содержит все последовательные исходные кодированные изображения в порядке декодирования, начиная со связанного с ними исходного кодированного IDR изображения (включительно) и заканчивая следующим исходным кодированным IDR изображением (исключая) или самым последним исходным кодированным изображением в потоке битов (включительно) в порядке декодирования, если за этим не следует исходное кодированное IDR изображение. Набор группы секций – это собрание одной или более групп секций, идентифицированных значением элемента синтаксиса slice_group_id[i].

Это сообщение SEI указывает, что для каждого изображения в наборе целевых изображений процесс inter предсказания ограничен следующим образом. Отсутствует значение образца вне набора группы секций и отсутствует значение образца в положении фрагмента образца, которые находят с помощью одного или более значений образца вне набора группы секций, использованных для inter предсказания любого образца в этой группе секций.

num_slice_groups_in_set_minus1 + 1 определяет число групп секций в наборе из групп секций. Разрешенный диапазон num_slice_groups_in_set_minus1 от 0 до num_slice_groups_minus1 включительно. Разрешенный диапазон num_slice_groups_minus1 определен в Приложении A.

slice_group_id[i **]** идентифицирует группу(ы) секций, которые находятся в наборе групп секций. Разрешенный диапазон от 0 до num_slice_groups_in_set_minus1 включительно. Размер элемента синтаксиса slice_group_id[i] равен Ceil(Log2(num_slice_groups_minus1 + 1)) битов.

exact_sample_value_match_flag, равное 0, указывает, что в наборе целевых изображений, когда макроблоки, не принадлежащие к этому набору групп секций еще не декодированы, значение каждого образца в группе секций не должно быть точно таким же, как значение этого же образца в случае, когда все макроблоки декодированы. Значение exact_sample_value_match_flag, равное 1, указывает, что в наборе целевых изображений, когда макроблоки, не принадлежащие к этому набору групп секций, еще не декодированы, значение каждого образца в группе секций должно быть точно таким же, как значение этого же образца в случае, когда все макроблоки декодированы.

ПРИМЕЧАНИЕ. – Если disable_deblocking_filter_idc равно 2 во всех секциях набора целевых изображений, то exact_sample_value_match_flag должно быть 1.

pan_scan_rect_flag, равное 0, указывает, что значение pan_scan_rect_id отсутствует. Значение pan_scan_rect_flag, равное 1, указывает, что значение pan_scan_rect_id присутствует.

pan_scan_rect_id указывает, что определенный набор группы секций, по меньшей мере, покрывает прямоугольник сканирования, идентифицированный значением pan_scan_rect_id в наборе целевых изображений.

ПРИМЕЧАНИЕ. – Множество сообщений SEI motion_constrained_slice_group_set может быть связано с тем же самым IDR изображением. Следовательно, в наборе целевых изображений активной может быть более одной группы секций.

ПРИМЕЧАНИЕ. – В наборе целевых изображений размер, форма и расположение группы секций могут изменяться.

D.2.20 Семантика зарезервированного сообщения SEI

Это сообщение состоит из данных, зарезервированных для будущих применений МСЭ-Т | ИСО/МЭК, совместимых с ранее использованными. Кодеры, соответствующие данной Рекомендации | Международному стандарту, не должны отправлять зарезервированные сообщения SEI до тех пор, пока использование таких сообщений не будет определено МСЭ-Т | ИСО/МЭК. Декодеры, соответствующие данной Рекомендации | Международному стандарту, которые сталкиваются с зарезервированными сообщениями SEI, должны отбрасывать их содержимое без какого-либо их воздействия на процесс декодирования, за исключением случаев, описанных в будущих Рекомендациях | Международных стандартах и определенных МСЭ-Т | ИСО/МЭК.

reserved_sei_message_payload_byte – это байт, зарезервированный для будущего использования МСЭ-Т | ИСО/МЭК.

Приложение E

Удобная в использовании визуальная информация (VUI)

(Это Приложение является составной частью данной Рекомендации | Международного стандарта)

Это Приложение определяет синтаксис и семантику параметров VUI при установке параметров последовательностей.

Параметры VUI не требуются для создания образцов яркости или цветности процессом декодирования. Также не требуется соответствие декодеров данной Рекомендации | Международному стандарту (см. Приложение C о спецификации соответствия) для обработки этой информации в порядке выхода. Однако некоторые параметры VUI требуются для проверки на соответствие потока битов и синхронизации декодера по выходу.

В Приложении E спецификация присутствия параметров VUI также выполняется, если эти параметры (или некоторое их подмножество) передаются декодерам (или HRD) с помощью других средств, не определенных этой Рекомендацией | Международным стандартом. При наличии в потоке битов параметры VUI должны следовать синтаксису и семантике, определенным в пп. 7.3.2.1 и 7.4.2.1 и в этом Приложении. Если для некоторых приложений содержание параметров VUI передают какими-нибудь средствами, отличными от тех, которые присутствуют в потоке битов, представление содержания параметров VUI не требует использования того же синтаксиса, который определен в данном Приложении. Для целей подсчета битов пригодны только соответствующие биты, присутствующие в потоке битов.

E.1 Синтаксис VUI

E.1.1 Синтаксис параметров VUI

vui_parameters( ) {	C	Дескриптор
aspect_ratio_info_present_flag	0	u(1)
if(aspect_ratio_info_present_flag) {		
aspect_ratio_idc	0	u(8)
if(aspect_ratio_idc = = Extended_SAR) {		
sar_width	0	u(16)
sar_height	0	u(16)
}		
}		
overscan_info_present_flag	0	u(1)
if(overscan_info_present_flag)		
overscan_appropriate_flag	0	u(1)
video_signal_type_present_flag	0	u(1)
if(video_signal_type_present_flag) {		
video_format	0	u(3)
video_full_range_flag	0	u(1)
colour_description_present_flag	0	u(1)
if(colour_description_present_flag) {		
colour_primaries	0	u(8)
transfer_characteristics	0	u(8)
matrix_coefficients	0	u(8)
}		
}		
chroma_loc_info_present_flag	0	u(1)
if(chroma_loc_info_present_flag) {		
chroma_sample_loc_type_top_field	0	ue(v)
chroma_sample_loc_type_bottom_field	0	ue(v)
}		
timing_info_present_flag	0	u(1)
if(timing_info_present_flag) {		
num_units_in_tick	0	u(32)
time_scale	0	u(32)
fixed_frame_rate_flag	0	u(1)
}		
nal_hrd_parameters_present_flag	0	u(1)
if(nal_hrd_parameters_present_flag)		
hrd_parameters( )		
vcl_hrd_parameters_present_flag	0	u(1)
if(vcl_hrd_parameters_present_flag)		
hrd_parameters( )		
if(nal_hrd_parameters_present_flag | | vcl_hrd_parameters_present_flag)		
low_delay_hrd_flag	0	u(1)
pic_struct_present_flag	0	u(1)
bitstream_restriction_flag	0	u(1)
if(bitstream_restriction_flag) {		

motion_vectors_over_pic_boundaries_flag	0	u(1)
max_bytes_per_pic_denom	0	ue(v)
max_bits_per_mb_denom	0	ue(v)
log2_max_mv_length_horizontal	0	ue(v)
log2_max_mv_length_vertical	0	ue(v)
num_reorder_frames	0	ue(v)
max_dec_frame_buffering	0	ue(v)
}		
}		

E.1.2 Синтаксис параметров HRD

hrd_parameters( ) {	C	Дескриптор
cpb_cnt_minus1	0	ue(v)
bit_rate_scale	0	u(4)
cpb_size_scale	0	u(4)
for(SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++) {		
bit_rate_value_minus1[SchedSelIdx **]**	0	ue(v)
cpb_size_value_minus1[SchedSelIdx **]**	0	ue(v)
cbr_flag[SchedSelIdx **]**	0	u(1)
}		
initial_cpb_removal_delay_length_minus1	0	u(5)
cpb_removal_delay_length_minus1	0	u(5)
dpb_output_delay_length_minus1	0	u(5)
time_offset_length	0	u(5)
}		

E.2 Семантика VUI

E.2.1 Семантика параметров VUI

aspect_ratio_info_present_flag равное 1, определяет, что присутствует aspect_ratio_idc. aspect_ratio_info_present_flag, равное 0, определяет, что aspect_ratio_idc отсутствует.

aspect_ratio_idc определяет значение коэффициента пропорциональности образца к яркости образцов. Таблица E-1 показывает значение этого кода. Если aspect_ratio_idc указывает на Extended_SAR, коэффициент пропорциональности образца представляют значением sar_width и sar_height. Если элемент синтаксиса aspect_ratio_idc отсутствует, значение aspect_ratio_idc должно быть принято равным 0.

Таблица E-1 – Значение указателя коэффициента пропорциональности образца

aspect_ratio_idc	Коэффициент пропорциональности образца	(информативное) Примеры использования
0	Неопределенный	
1	1:1 ("квадрат")	1280x720 16:9 кадр без выхода развертки за пределы полезной площади экрана 1920x1080 16:9 кадр без выхода развертки за пределы полезной площади экрана (разделение от 1920x1088) 640x480 4:3 9 кадр без выхода развертки за пределы полезной площади экрана
2	12:11	720x576 4:3 кадр с выходом горизонтальной развертки за пределы полезной площади экрана 352x288 4:3 кадр без выхода развертки за пределы полезной площади экрана
3	10:11	720x480 4:3 кадр с выходом горизонтальной развертки за пределы полезной площади экрана 352x240 4:3 кадр без выхода развертки за пределы полезной площади экрана
4	16:11	720x576 16:9 кадр с выходом горизонтальной развертки за пределы полезной площади экрана 540x576 4:3 кадр с выходом горизонтальной развертки за пределы полезной площади экрана
5	40:33	720x480 16:9 кадр с выходом горизонтальной развертки за пределы полезной площади экрана 540x480 4:3 кадр с выходом горизонтальной развертки за пределы полезной площади экрана
6	24:11	352x576 4:3 кадр без выхода развертки за пределы полезной площади экрана 540x576 16:9 кадр с выходом горизонтальной развертки за пределы полезной площади экрана
7	20:11	352x480 4: кадр без выхода развертки за пределы полезной площади экрана 480x480 16:9 кадр с выходом горизонтальной развертки за пределы полезной площади экрана
8	32:11	352x576 16:9 кадр без выхода развертки за пределы полезной площади экрана
9	80:33	352x480 16:9 кадр без выхода развертки за пределы полезной площади экрана
10	18:11	480x576 4:3 кадр с выходом горизонтальной развертки за пределы полезной площади экрана
11	15:11	480x480 4:3 кадр с выходом горизонтальной развертки за пределы полезной площади экрана
12	64:33	540x576 16:9 кадр с выходом горизонтальной развертки за пределы полезной площади экрана
13	160:99	540x480 16:9 кадр с выходом горизонтальной развертки за пределы полезной площади экрана
14..254	Зарезервировано	
255	Extended_SAR	

sar_width определяет горизонтальные размеры образца (в произвольных блоках).

sar_height определяет коэффициент пропорциональности вертикальных размеров образца (в тех же произвольных блоках, что и sar_width).

sar_width и sar_height должны быть выражены в относительных простых числах или равными 0. Если aspect_ratio_idc равно 0 или sar_width равно 0 или sar_height равно 0, то в этой Рекомендации | Международном стандарте коэффициент пропорциональности образца должен рассматриваться как неопределенный.

overscan_info_present_flag, равное 1, указывает, что присутствует overscan_appropriate_flag. Если overscan_info_present_flag равно 0 или отсутствует, то предпочтительный метод отображения видеосигнала не определен.

overscan_appropriate_flag, равное 1, указывает, что разделенные на блоки декодированные изображения на выходе пригодны для отображения при использовании кадров без выхода развертки за пределы полезной площади экрана. Значение overscan_appropriate_flag, равное 0, указывает, что разделенные на блоки декодированные изображения на выходе содержат важную визуальную информацию во всей области за краями прямоугольника разделения изображения таким образом, что разделенные декодированные изображения на выходе нельзя вывести на экран с выходом горизонтальной развертки за пределы полезной площади экрана. Вместо этого эти изображения следует выводить на экран, используя либо точное совпадение между площадями экрана и прямоугольника разделения, либо с низким качеством сканирования.

ПРИМЕЧАНИЕ. – Например, overscan_appropriate_flag, равное 1, можно использовать для развлекательных телевизионных программ или для "живого" показа людей во время видеоконференций. Значение overscan_appropriate_flag, равное 0, может быть использовано для фиксации содержимого экрана или камеры наблюдения.

video_signal_type_present_flag, равное 1, указывает, что присутствуют значения video_format, video_full_range_flag и colour_description_present_flag. Значение video_signal_type_present_flag, равное 0, указывает, что синтаксические структуры video_format, video_full_range_flag и colour_description_present_flag отсутствуют.

video_format указывает на показанный в таблице E-2 формат представления изображений перед их кодированием в соответствии с данной Рекомендацией | Международным стандартом. Если элемент синтаксиса video_format не присутствует, значение video_format должно быть принято равным 5.

Таблица E-2 – Значение video_format

video_format	Значение
0	Компонент
1	PAL
2	NTSC
3	SECAM
4	MAC
5	Неопределенный видеоформат
6	Зарезервировано
7	Зарезервировано

video_full_range_flag определяет сигналы уровня черного и диапазона яркости и цветности, найденных из аналоговых компонентов сигналов E'_Y, E'_{PB} и E'_{PR} следующим образом:

– Если video_full_range_flag равно 0,

$$Y = Round(219 * E'_Y + 16) \quad (E-1)$$

$$Cb = Round(224 * E'_{PB} + 128) \quad (E-2)$$

$$Cr = Round(224 * E'_{PR} + 128). \quad (E-3)$$

– Иначе (video_full_range_flag равно 1),

$$Y = Round(255 * E'_Y) \quad (E-4)$$

$$Cb = Round(255 * E'_{PB} + 128) \quad (E-5)$$

$$Cr = Round(255 * E'_{PR} + 128). \quad (E-6)$$

Если элемент синтаксиса video_full_range_flag отсутствует, то значение video_full_range_flag должно быть принято равным 0.

colour_description_present_flag, равное 1, определяет, что colour_primaries, transfer_characteristics и matrix_coefficients присутствуют. Значение colour_description_present_flag, равное 0, определяет, что colour_primaries, transfer_characteristics и matrix_coefficients отсутствуют.

colour_primaries указывает координаты цветности исходных основных цветов, показанных в таблице E-3 в терминах определения CIE 1931 параметров x и y, как это указано в ISO/CIE 10527.

Таблица E-3 – Основные цвета

Значение	**Основные цвета**		
0	Зарезервировано		
1	Рекомендация МСЭ-R BT.709		
	основные цвета	x	y
	зеленый	0,300	0,600
	синий	0,150	0,060
	красный	0,640	0,330
	белый D65	0,3127	0,3290
2	Не определена Характеристики изображения неизвестны или определены приложением.		
3	Зарезервировано		
4	Рекомендация МСЭ-R BT.470-2 Система M		
	основные цвета	x	y
	зеленый	0,21	0,71
	синий	0,14	0,08
	красный	0,67	0,33
	белый C	0,310	0,316
5	Рекомендация МСЭ-R BT.470-2 Системы B, G		
	основные цвета	x	y
	зеленый	0,29	0,60
	синий	0,15	0,06
	красный	0,64	0,33
	белый D65	0,3127	0,3290
6	Общество кино- и телеинженеров 170M		
	основные цвета	x	y
	зеленый	0,310	0,595
	blue	0,155	0,070
	красный	0,630	0,340
	белый D65	0,3127	0,3290
7	Общество по движущимся изображениям и телевизионной технике 240M (1987 г.)		
	основные цвета	x	y
	зеленый	0,310	0,595
	синий	0,155	0,070
	красный	0,630	0,340
	белый D65	0,3127	0,3290
8	Обычная пленка (цветовые фильтры с использованием источника света C)		
	основные цвета	x	y
	зеленый	0,243	0,692 (Wratten 58)
	синий	0,145	0,049 (Wratten 47)
	красный	0,681	0,319 (Wratten 25)
	белый C	0,310	0,316
9–255	Зарезервировано		

Если элемент синтаксиса colour_primaries отсутствует, то значение colour_primaries должно быть принято равным 2 (цветность не определена или определена приложением).

transfer_characteristics указывает на опто-электронную характеристику передачи исходного изображения, как это определено в таблице E-4 в виде функции от линейной оптической интенсивности на входе L_c в аналоговом диапазоне от 0 до 1.

Таблица E-4 – Характеристика передачи

Значение	Характеристика передачи
0	Зарезервировано
1	Рекомендация МСЭ-R BT.709 $V = 1{,}099\ L_c^{0{,}45} - 0{,}099$ для $1 >= L_c >= 0{,}018$ $V = 4{,}500\ L_c$ для $0{,}018 > L_c$
2	Не определена Характеристики изображения неизвестны или определены приложением.
3	Зарезервировано
4	Рекомендация МСЭ-R BT.470-2 Система M Полагают гамму экрана 2,2
5	Рекомендация МСЭ-R BT.470-2 Системы B, G Полагают гамму экрана 2,8
6	Общество кино- и телеинженеров 170M $V = 1{,}099\ L_c^{0{,}45} - 0{,}099$ для $1 >= L_c >= 0{,}018$ $V = 4{,}500\ L_c$ для $0{,}018 > L_c$
7	Общество кино- и телеинженеров (1987) $V = 1{,}1115\ L_c^{0{,}45} - 0{,}1115$ для $L_c >= 0{,}0228$ $V = 4{,}0\ L_c$ для $0{,}0228 > L_c$
8	Линейная характеристика передачи $V = L_c$
9	Логарифмическая характеристика передачи (100:1 диапазона) $V = 1{,}0 - \text{Log10}(L_c) \div 2$ для $1 >= L_c >= 0{,}01$ $V = 0{,}0$ для $0{,}01 > L_c$
10	Логарифмическая характеристика передачи (316,22777:1 диапазона) $V = 1{,}0 - \text{Log10}(L_c) \div 2{,}5$ для $1 >= L_c >= 0{,}0031622777$ $V = 0{,}0$ для $0{,}0031622777 > L_c$
11–255	Зарезервировано

Если элемент синтаксиса transfer_characteristics отсутствует, то значение transfer_characteristics должно быть принято равным 2 (характеристики передачи не определены или определяются приложением).

matrix_coefficients описывает матрицу коэффициентов, используемую для отыскания сигналов яркости и цветности из основных тонов зеленого, синего и красного, как это определено в таблице E-5.

Используют следующие определения:

E'_R, E'_G и E'_B – аналоговые сигналы со значениями в диапазоне от 0 до 1.

Белый определен как имеющий E'_R, равное 1, E'_G, равное 1, и E'_B, равное 1.

Далее:

$$E'_Y = K_R * E'_R + (1 - K_R - K_B) * E'_G + K_B * E'_B \quad \text{(E-7)}$$

$$E'_{PB} = 0{,}5 * (E'_B - E'_Y) \div (1 - K_B) \quad \text{(E-8)}$$

$$E'_{PR} = 0{,}5 * (E'_R - E'_Y) \div (1 - K_R). \quad \text{(E-9)}$$

ПРИМЕЧАНИЕ. – Тогда E'_Y – аналоговый сигнал со значениями в диапазоне от 0 до 1, E'_{PB} и E'_{PR} – аналоговые сигналы со значениями в диапазоне от –0,5 до 0,5, а белый эквивалентно задан значениями $E'_Y = 1$, $E'_{PB} = 0$, $E'_{PR} = 0$.

Таблица E-5 – Матрица коэффициентов

Значение	**Матрица**
0	Зарезервировано
1	Рекомендация МСЭ-R BT.709 $K_R = 0,2126$; $K_B = 0,0722$
2	Не определено Характеристики изображения неизвестны или определены приложением.
3	Зарезервировано
4	Федеральная комиссия связи (FCC) $K_R = 0,30$; $K_B = 0,11$
5	Рекомендация МСЭ-R BT.470-2 Системы B, G: $K_R = 0,299$; $K_B = 0,114$
6	Общество кино- и телеинженеров 170M $K_R = 0,299$; $K_B = 0,114$
7	Общество кино- и телеинженеров 240M (1987 г.) $K_R = 0,212$; $K_B = 0,087$
8–255	Зарезервировано

Если элемент синтаксиса matrix_ coefficients отсутствует, то значение matrix_coefficients должно быть принято равным 2.

chroma_loc_info_present_flag, равное 1, определяет, что chroma_sample_loc_type_top_field и chroma_sample_loc_type_bottom_field присутствуют. Значение chroma_loc_info_present_flag, равное 0, определяет, что chroma_sample_loc_type_top_field и chroma_sample_loc_type_bottom_field отсутствуют.

chroma_sample_loc_type_top_field и **chroma_sample_loc_type_bottom_field** определяют расположение образцов цветности для верхнего и нижнего поля, как показано на рисунке E-1. Значение chroma_sample_loc_type_top_field и chroma_sample_loc_type_bottom_field должно быть в диапазоне от 0 до 5 включительно. Если chroma_sample_loc_type_top_field и chroma_sample_loc_type_bottom_field отсутствуют, то значения chroma_sample_loc_type_top_field и chroma_sample_loc_type_bottom_field должны быть приняты равными 0.

ПРИМЕЧАНИЕ. – При последовательном кодировании исходного материала поля chroma_sample_loc_type_top_field и chroma_sample_loc_type_bottom_field должны иметь одно и то же значение.

Рисунок E-1 – Расположение образцов цветности для верхнего и нижнего полей как функции chroma_sample_loc_type_top_field и chroma_sample_loc_type_bottom_field

timing_info_present_flag, равное 1, указывает, что num_units_in_tick, time_scale и fixed_frame_rate_flag присутствует в потоке битов. Значение timing_info_present_flag, равное 0, указывает, что num_units_in_tick, time_scale и fixed_frame_rate_flag отсутствуют в потоке битов.

num_units_in_tick – это число блоков синхронизации задающего генератора, работающего на частоте time_scale Гц, которое соответствует одному приращению (называемому временным тактом) счетчика тактов. num_units_in_tick должно быть больше 0. Временной такт – это минимальный интервал времени, который может быть представлен в кодированных данных. Например, если тактовая частота видеосигна равна 30000 ÷ 1001 Гц, то time_scale может быть 30 000, num_units_in_tick может быть 1001. См. уравнение C-1.

time_scale – это число блоков синхронизации, которые поступают в одну секунду. Например, система координации времени, которая измеряет время, используя тактовую частоту 27 МГц, имеет time_scale, равное 27 000 000. Значение time_scale должно быть больше 0.

fixed_frame_rate_flag, равное 1, указывает, что временное расстояние на выходе HRD между двумя любыми последовательными изображениями в порядке выхода ограничено следующим образом. Значение fixed_frame_rate_flag, равное 0, указывает, что не наложено никаких ограничений на временное расстояние на выходе HRD между двумя любыми последовательными изображениями в порядке выхода.

Если fixed_frame_rate_flag равно 1 для всех n, где n указывает на n-е изображение в порядке выхода, а изображение n – это не последнее изображение в потоке битов в порядке выхода, то значение $\Delta t_{fi,dpb}(n)$ определяют как

$$\Delta t_{fi,dpb}(n) = \Delta t_{o,dpb}(n) \div \text{DeltaTfiDivisor}, \quad \text{(E-10)}$$

где $\Delta t_{o,dpb}(n)$ определено в уравнении C-13, а DeltaTfiDivisor приведено в таблице E-6, основанной на значениях pic_struct_present_flag, field_pic_flag и pic_struct для изображения n. Входы в таблице E-6, отмеченные знаком "–", указывают на отсутствие зависимости DeltaTfiDivisor от соответствующего элемента синтаксиса.

Если fixed_frame_rate_flag равно 1, то значение, вычисленное для $\Delta t_{fi,dpb}(n)$, должно быть одинаковым для всех $n > 0$ и должно быть равно num_units_in_tick ÷ time_scale.

Таблица E-6 – Делитель для вычисления $\Delta t_{fi,dpb}(n)$

pic_struct_present_flag	field_pic_flag	pic_struct	DeltaTfiDivisor
0	1	–	1
1	–	1	1
1	–	2	1
0	0	–	2
1	–	0	2
1	–	3	2
1	–	4	2
1	–	5	3
1	–	6	3
1	–	7	4
1	–	8	6

nal_hrd_parameters_present_flag, равное 1, указывает, что параметры NAL HRD (принадлежащие к Типу II соответствия потока битов) присутствуют. Значение nal_hrd_parameters_present_flag, равное 0, указывает, что параметры NAL HRD отсутствуют.

ПРИМЕЧАНИЕ. – Если nal_hrd_parameters_present_flag равно 0, то соответствие потока битов нельзя проверить без присутствия параметров NAL HRD, включая последовательность параметров информации NAL HRD и все сообщения SEI о периодах буферизации и синхронизациии изображения, не определенных в этой Рекомендации | Международном стандарте другими средствами.

Если nal_hrd_parameters_present_flag равно 1, то параметры NAL HRD (в пп. E.1.2 и E.2.2) непосредственно следуют за флагом.

Переменную NalHrdBpPresentFlag находят следующим образом:

– Если любое из следующих утверждений – истина, то значение NalHrdBpPresentFlag должно быть установлено равным 1:
  – nal_hrd_parameters_present_flag присутствует в потоке битов и равно 1;
  – необходимость представлять периоды буферизации работы NAL HRD в потоке битов в сообщениях SEI о периоде буферизации определяется приложением с помощью средств, не определенных в этой Рекомендации | Международном стандарте.
– Иначе значение NalHrdBpPresentFlag должно быть установлено равным 0.

vcl_hrd_parameters_present_flag, равное 1, определяет, что параметры VCL HRD (принадлежащие ко всем типам соответствия потока битов) присутствуют. Значение vcl_hrd_parameters_present_flag, равное 0, определяет, что параметры VCL HRD отсутствуют.

ПРИМЕЧАНИЕ. – Если vcl_hrd_parameters_present_flag равно 0, то соответствие потока битов нельзя проверить без присутствия параметров NAL HRD и всех сообщений SEI о периодах буферизации и синхронизациии изображения, не определенных в этой Рекомендации | Международном стандарте другими средствами.

Если vcl_hrd_parameters_present_flag равно 1, то параметры NAL HRD (в пп. E.1.2 и E.2.2) непосредственно следуют за флагом.

Переменную VclHrdBpPresentFlag находят следующим образом:

– Если любое из следующих утверждений – истина, то значение VclHrdBpPresentFlag должно быть установлено равным 1:
  – vcl_hrd_parameters_present_flag присутствует в потоке битов и равно 1;
  – необходимость представлять периоды буферизации работы VCL HRD в потоке битов в сообщениях SEI о периоде буферизации определяется приложением с помощью некоторых средств, не определенных в этой Рекомендации | Международном стандарте.
– Иначе значение VclHrdBpPresentFlag должно быть установлено равным 0.

Переменную CpbDpbDelaysPresentFlag находят следующим образом:

– Если любое из следующих утверждений – истина, то значение CpbDpbDelaysPresentFlag должно быть установлено равным 1:
  – nal_hrd_parameters_present_flag присутствует в потоке битов и равно 1;
  – vcl_hrd_parameters_present_flag присутствует в потоке битов и равно 1;
  – необходимость представлять задержки на выходе CPB и DPB в потоке битов в сообщениях SEI о синхронизации изображения определяется приложением с помощью некоторых средств, не определенных в этой Рекомендации | Международном стандарте.

– Иначе значение CpbDpbDelaysPresentFlag должно быть установлено равным 0.

low_delay_hrd_flag определяет режим работы HRD, как это определено в Приложении C. Если fixed_frame_rate_flag равно 1, то low_delay_hrd_flag должно быть равно 0.

ПРИМЕЧАНИЕ. – Если low_delay_hrd_flag равно 1, то разрешены "большие изображения", которые нарушают номинальное время удалений CPB из-за числа битов, использованных блоком доступа. Предполагается, но не требуется, чтобы такие "большие изображения" встречались только случайно.

pic_struct_present_flag, равное 1, указывает, что сообщения SEI о синхронизации изображения (п. D.2.2), которые включают элемент синтаксиса pic_struct, присутствуют. Значение pic_struct_present_flag, равное 0, указывает, что элемент синтаксиса pic_struct отсутствует в сообщениях SEI о синхронизации изображения.

bitstream_restriction_flag, равное 1, указывает на присутствие ограничений на последовательность потока битов. Значение bitstream_restriction_flag, равное 0, указывает на отсутствие ограничений на последовательность потока битов.

motion_vectors_over_pic_boundaries_flag, равное 0, указывает на отсутствие образца за границами изображения и на отсутствие образца в положении фрагмента образца, чьи значения находят, используя один или более образцов за границами изображения с помощью inter предсказания любого образца. Значение motion_vectors_over_pic_boundaries_flag, равное 1, указывает, что один или более образцов за границами изображения могут быть использованы в inter предсказании. Если элемент синтаксиса motion_vectors_over_pic_boundaries_flag отсутствует, то значение motion_vectors_over_pic_boundaries_flag должно быть принято равным 1.

max_bytes_per_pic_denom указывает число байтов, не превосходящих сумму размеров блоков NAL VCL, связанных с любым кодированным изображением в последовательности.

Число байтов, которые представляют изображение в потоке блоков NAL, определено для этой цели как общее число байтов данных блока NAL VCL (т. е. общее число переменных NumBytesInNALunit блоков NAL VCL) для данного изображения. Значение max_bytes_per_pic_denom должно быть в диапазоне от 0 до 16 включительно.

В зависимости от max_bytes_per_pic_denom применяют следующее:

– Если max_bytes_per_pic_denom равно 0, не указывают никаких пределов.

– Иначе (max_bytes_per_pic_denom не равно 0), никакое кодированное изображение не должно быть представлено в последовательности более чем следующим числом байтов:

$$(\text{PicSizeInMbs} * 256 * \text{ChromaFormatFactor}) \div \text{max\_bytes\_per\_pic\_denom}. \quad \text{(E-11)}$$

Если элемент синтаксиса max_bytes_per_pic_denom отсутствует, то значение max_bytes_per_pic_denom должно быть принято равным 2.

max_bits_per_mb_denom указывает максимальное число кодированных битов данных macroblock_layer( ) для любого макроблока в любом изображении последовательности. Значение max_bits_per_mb_denom должно быть в диапазоне от 0 до 16 включительно.

В зависимости от max_bits_per_mb_denom применяют следующее:

– Если max_bits_per_mb_denom равно 0, то не определены никакие пределы.

– Иначе (max_bits_per_mb_denom не равно 0), никакое кодированное значение macroblock_layer( ) не должно быть представлено в потоке битов более чем следующим числом битов:

$$(2048 * \text{ChromaFormatFactor} + 128) \div \text{max\_bits\_per\_mb\_denom}. \quad \text{(E-12)}$$

В зависимости от entropy_coding_mode_flag биты данных macroblock_layer( ) учитывают следующим образом:

– Если entropy_coding_mode_flag равно 0, то число битов данных macroblock_layer( ) задано числом битов в структуре синтаксиса macroblock_layer( ) макроблока.

– Иначе (entropy_coding_mode_flag равно 1), число битов данных макроблока macroblock_layer( ) задано числом раз read_bits(1), описанным в пп. 9.3.3.2.2 и 9.3.3.2.3, если анализ значения macroblock_layer( ) связан с макроблоком.

Если max_bits_per_mb_denom отсутствует, то значение max_bits_per_mb_denom должно быть принято равным 1.

log2_max_mv_length_horizontal и **log2_max_mv_length_vertical** указывают максимальное абсолютное значение декодированных компонентов горизонтального и вертикального векторов движения, соответственно, в ¼ блоков образцов яркости для всех изображений в последовательности. Величина n указывает, что никакое значение компонента вектора движения не должно превышать диапазон от -2^n до $2^n - 1$ включительно в блоках размещения образца яркости ¼. Значение log2_max_mv_length_horizontal должно быть в диапазоне от 0 до 16 включительно. Значение log2_max_mv_length_vertical должно быть в диапазоне от 0 до 16 включительно. Если log2_max_mv_length_horizontal отсутствует, то значения log2_max_mv_length_horizontal и log2_max_mv_length_vertical должны быть приняты равными 16.

ПРИМЕЧАНИЕ. – Максимальное абсолютное значение декодированных компонентов горизонтального и вертикального векторов движения ограничено также границами параметров и уровней, как это определено в Приложении A.

num_reorder_frames указывает максимальное число кадров, дополнительных пар полей или непарных полей, которые предшествуют любому кадру, дополнительной паре полей или непарным полям в последовательности в порядке декодирования и следуют этому порядку на выходе. Значение num_reorder_frames должно быть в диапазоне от 0 до max_dec_frame_buffering включительно. Если элемент синтаксиса num_reorder_frames отсутствует, то значение num_reorder_frames должно быть принято равным max_dec_frame_buffering.

max_dec_frame_buffering определяет требуемый размер буфера (DPB) декодированного изображения HRD в единицах буферов кадров. Последовательность не должна требовать буфера декодированного изображения с размером больше чем Max(1, max_dec_frame_buffering) буферов кадров, чтобы обеспечить выход декодированных изображений за время выхода, определенного сообщением SEI о синхронизации изображения dpb_output_delay. Значение max_dec_frame_buffering должно быть в диапазоне от num_ref_frames до MaxDpbSize (как это определено в п. A.3.1) включительно. Если элемент синтаксиса max_dec_frame_buffering отсутствует, то значение max_dec_frame_buffering должно быть принято равным MaxDpbSize.

E.2.2 Семантика параметров HRD

cpb_cnt_minus1 плюс 1 определяет число спецификаций альтернативных буферов CPB в потоке битов. Значение cpb_cnt_minus1 должно быть в диапазоне от 0 до 31 включительно. Если low_delay_hrd_flag равно 1, то cpb_cnt_minus1 должно быть равно 0. Если cpb_cnt_minus1 отсутствует, это значение должно считаться равным 0.

bit_rate_scale (совместно с bit_rate_value_minus1[SchedSelIdx]) определяет максимальную скорость битов на входе SchedSelIdx-ого буфера CPB.

cpb_size_scale (совместно с cpb_size_value_minus1[SchedSelIdx]) определяет размер CPB для SchedSelIdx-ого буфера CPB.

bit_rate_value_minus1[SchedSelIdx **]** (совместно с bit_rate_scale) определяет максимальную скорость битов на входе SchedSelIdx-ого буфера CPB. Значение bit_rate_value_minus1[SchedSelIdx] должно быть в диапазоне от 0 до $2^{32} - 2$ включительно. Для любой величины SchedSelIdx > 0 значение bit_rate_value_minus1[SchedSelIdx] должно быть больше, чем bit_rate_value_minus1[SchedSelIdx – 1]. Скорость в битах в секунду задают как

$$BitRate[SchedSelIdx] = (bit\_rate\_value\_minus1[SchedSelIdx] + 1) * 2^{(6 + bit\_rate\_scale)}. \quad (E\text{-}13)$$

Если элемент синтаксиса bit_rate_value_minus1[SchedSelIdx] отсутствует, то BitRate[SchedSelIdx] должно быть принято равным 1000 * MaxBR для параметров VCL HRD.

Если элемент синтаксиса bit_rate_value_minus1[SchedSelIdx] отсутствует, то BitRate[SchedSelIdx] должно быть принято равным 1200 * MaxBR для параметров NAL HRD.

cpb_size_value_minus1[SchedSelIdx **]** используют совместно с cpb_size_scale для определения размера SchedSelIdx-ого буфера CPB. Значение cpb_size_value_minus1[SchedSelIdx] должно быть в диапазоне от 0 до $2^{32} - 2$ включительно. Для любой величины SchedSelIdx больше, чем 0, значение cpb_size_value_minus1[SchedSelIdx] должно быть меньше чем или равно cpb_size_value_minus1[SchedSelIdx – 1].

Размер буфера CPB в битах задают как

$$CpbSize[SchedSelIdx] = (cpb\_size\_value\_minus1[SchedSelIdx] + 1) * 2^{(4 + cpb\_size\_scale)}. \quad (E\text{-}14)$$

Если элемент синтаксиса cpb_size_value_minus1[SchedSelIdx] отсутствует, то CpbSize[SchedSelIdx] должно быть принято равным 1000 * MaxCPB для параметров VCL HRD.

Если элемент синтаксиса cpb_size_value_minus1[SchedSelIdx] отсутствует, то CpbSize[SchedSelIdx] должно быть принято равным 1200 * MaxCPB для параметров NAL HRD.

Для параметров VCL HRD должно быть, по крайней мере, одно значение SchedSelIdx, для которого BitRate[SchedSelIdx] <= 1000*MaxBR и CpbSize[SchedSelIdx] <= 1000 * MaxCPB (как это определено в п. A.3.1).

Для параметров NAL HRD должно быть, по крайней мере, одно значение SchedSelIdx, для которого CpbSize[SchedSelIdx] <= 1200*MaxCPB и BitRate[SchedSelIdx] <= 1200*MaxBR.

cbr_flag[SchedSelIdx **]**, равное 0, определяет, что при декодировании данного потока битов декодером HRD, используя спецификацию SchedSelIdx-го буфера CPB, планировщик доставки гипотетического потока (HSS) работает в режиме переменной скорости битов. Значение cbr_flag[SchedSelIdx], равное 1, определяет, что HSS работает в режиме постоянной скорости битов (CBR). Если элемент синтаксиса cbr_flag[SchedSelIdx] отсутствует, то значение cbr_flag должно быть принято равным 0.

initial_cpb_removal_delay_length_minus1 определяет длину в битах элементов синтаксиса initial_cpb_removal_delay[SchedSelIdx] и initial_cpb_removal_delay_offset[SchedSelIdx] в сообщении SEI о периоде буферизации. Длина initial_cpb_removal_delay[SchedSelIdx] и initial_cpb_removal_delay_offset[SchedSelIdx] равна initial_cpb_removal_delay_length_minus1 + 1. Если элемент синтаксиса initial_cpb_removal_delay_length_minus1 присутствует более чем в одной структуре синтаксиса hrd_parameters( ) среди структур синтаксиса параметров VUI, то значение параметров initial_cpb_removal_delay_length_minus1 должно быть равным в обеих структурах синтаксиса hrd_parameters( ). Если элемент синтаксиса initial_cpb_removal_delay_length_minus1 отсутствует, то он должен быть принят равным 23.

cpb_removal_delay_length_minus1 определяет длину в битах элемента синтаксиса cpb_removal_delay. Длина элемента синтаксиса cpb_removal_delay в сообщении SEI о синхронизации изображения равна cpb_removal_delay_length_minus1 + 1. Если элемент синтаксиса cpb_removal_delay_length_minus1 присутствует более чем в одной структуре синтаксиса hrd_parameters( ) среди структур синтаксиса параметров VUI, то значение параметров cpb_removal_delay_length_minus1 должно быть равным в обеих структурах синтаксиса hrd_parameters( ). Если элемент синтаксиса cpb_removal_delay_length_minus1 отсутствует, то он должен быть принят равным 23.

dpb_output_delay_length_minus1 определяет длину в битах элемента синтаксиса dpb_output_delay. Длина элемент синтаксиса dpb_output_delay в сообщении SEI о синхронизации изображения равна dpb_output_delay_length_minus1 + 1. Если элемент синтаксиса dpb_output_delay_length_minus1 присутствует более чем в одной структуре синтаксиса hrd_parameters( ) среди структур синтаксиса параметров VUI, то значение параметров dpb_output_delay_length_minus1 должно быть равным в обеих структурах синтаксиса hrd_parameters( ). Если элемент синтаксиса dpb_output_delay_length_minus1 отсутствует, то он должен быть принят равным 23.

time_offset_length больше, чем 0, определяет длину в битах элемента синтаксиса time_offset. Значение time_offset_length равное 0, указывает на отсутствие элемента синтаксиса time_offset. Если элемент синтаксиса time_offset_length присутствует более чем в одной структуре синтаксиса hrd_parameters( ) среди структур синтаксиса параметров VUI, то значение параметров time_offset_length должно быть равным в обеих структурах синтаксиса hrd_parameters( ). Если элемент синтаксиса time_offset_length отсутствует, то он должен быть принят равным 24.

СЕРИИ РЕКОМЕНДАЦИЙ МСЭ-Т

Серия A	Организация работы МСЭ-Т
Серия B	Средства выражения: определения, символы, классификация
Серия C	Общая статистика электросвязи
Серия D	Общие принципы тарификации
Серия E	Общая эксплуатация сети, телефонная служба, функционирование служб и человеческие факторы
Серия F	Нетелефонные службы электросвязи
Серия G	Системы и среда передачи, цифровые системы и сети
Серия H	**Аудиовизуальные и мультимедийные системы**
Серия I	Цифровая сеть с интеграцией служб
Серия J	Кабельные сети и передача сигналов телевизионных и звуковых программ и других мультимедийных сигналов
Серия K	Защита от помех
Серия L	Конструкция, прокладка и защита кабелей и других элементов линейно-кабельных сооружений
Серия M	TMN и техническое обслуживание сетей: международные системы передачи, телефонные, телеграфные, факсимильные и арендованные каналы
Серия N	Техническое обслуживание: международные каналы передачи звуковых и телевизионных программ
Серия O	Требования к измерительной аппаратуре
Серия P	Качество телефонной передачи, телефонные установки, сети местных линий
Серия Q	Коммутация и сигнализация
Серия R	Телеграфная передача
Серия S	Оконечное оборудование для телеграфных служб
Серия T	Оконечное оборудование для телематических служб
Серия U	Телеграфная коммутация
Серия V	Передача данных по телефонной сети
Серия X	Сети передачи данных и взаимосвязь открытых систем
Серия Y	Глобальная информационная инфраструктура, аспекты межсетевого протокола и сети последующих поколений
Серия Z	Языки и общие аспекты программного обеспечения для систем электросвязи

Отпечатано в Швейцарии
Женева, 2005 г.