2.5.9.Физический уровень.

2.5.9.1. Логический подуровень физического уровня

Физический уровень, смотри рис. 2.5.10., подключается к уровню передачи данных с одной стороны и к каналу связи - с другой стороны. Физический уровень принимает и обрабатывает, исходящие из уровня передачи данных, пакеты TLP(DLL) и DLLP перед их передачей в канал связи, а так же принимает и обрабатывает входящие пакеты TLP(PL), DLLP(PL) и PLP, полученные из канала связи перед их передачей на уровень передачи данных.

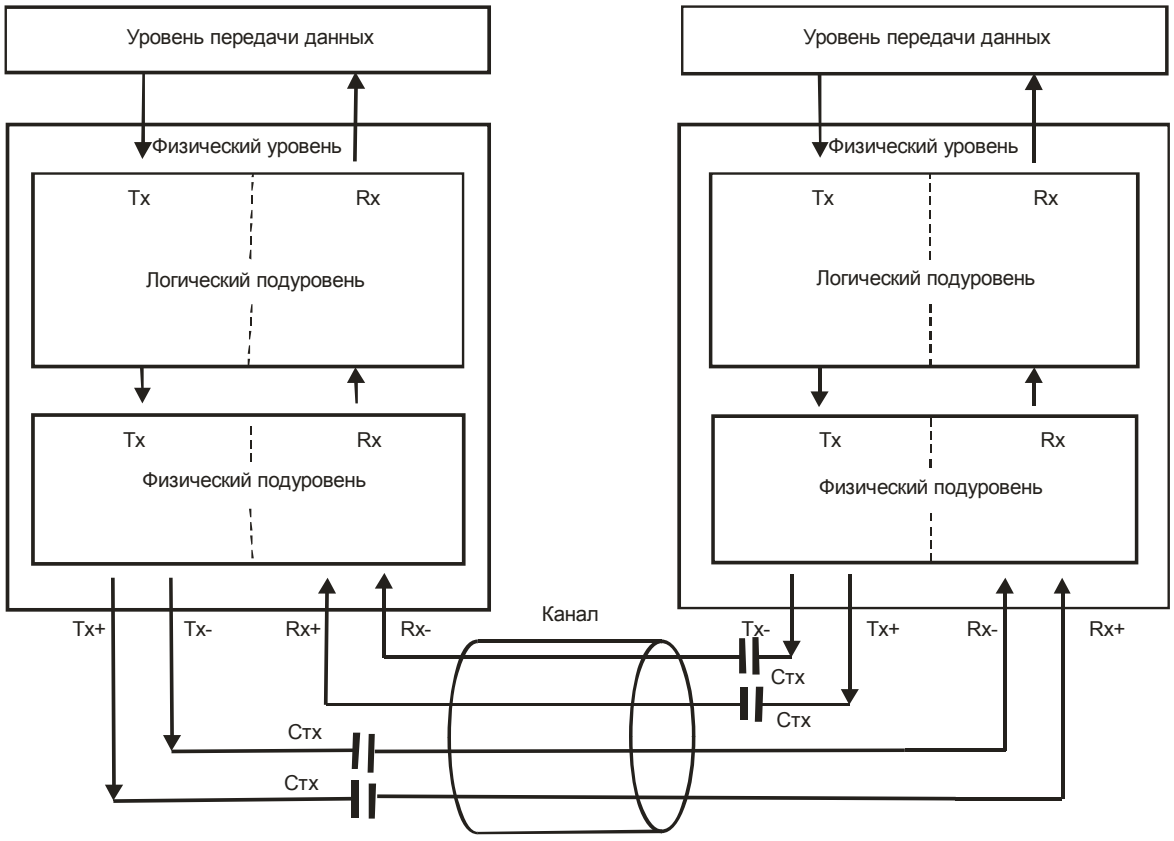

Рис. 2.5.10. Физический уровень PCI-XP

Два подуровня составляют физический уровень устройства: это логический подуровень физического уровня и электрический подуровень физического уровня, как показано на рис. 2.5.10. Оба подуровня разделены на передающую

логику Tx и принимающую логику Rx (независимые друг от друга), которые позволяют осуществлять двойную симплексную связь.

Передающая логика логического подуровня в основном обрабатывает пакеты TLP(DLL) и DLLP, приходящие с уровня канала данных, и затем преобразовывает их из параллельного представления в последовательный поток битов. Битовый поток передается в канал связи с битовой скоростью 2.5 Гбит/с по каждой дифференциальной линии передачи данных.

Принимающая логика логического подуровня преобразует последовательный битовый поток в параллельный символьный поток, обрабатывает входящие символы, собирает пакеты TLP(PL), DLLP(PL) и PLP и отправляет пакеты TLP(PL), DLLP(PL) на уровень канала данных.

Передающая часть логического подуровня физического уровня

Рис. 2.5.11. показывает элементы, составляющие передающую логику логического подуровня физического уровня: буфер передачи данных, (Tx), мультиплексор (mux), логику разделения данных (необходима только, если канал связи реализует более одной линии передачи данных), скремблеры, 8b/10b кодеры, преобразователи параллельного кода в последовательный.

Рис. 2.5.12. показывает элементы, составляющие принимающую логику: PLL приемника, последовательно-параллельный преобразователь, буфер регулируемой емкости, 8b/10b декодер, дескремблер, логика соединения данных (необходима только, если канал связи реализует более одной линии передачи данных), схема удаления управляющих знаков, буфер приема пакетов

Пакеты TLP(DLL) и DLLP с уровня передачи данных принимаются в передающий буфер пакетов, Tx. Уровень передачи данных указывает начало и конец пакета, используя сигнал «Управление» так, чтобы физический уровень мог добавить начальные и конечные кадровые знаки STP, SDP, END к пакету,

используя генератор стартовых и стоповых знаков. Tx буфер использует также сигнал «блокировка приема» чтобы остановить поток пакетов с уровня

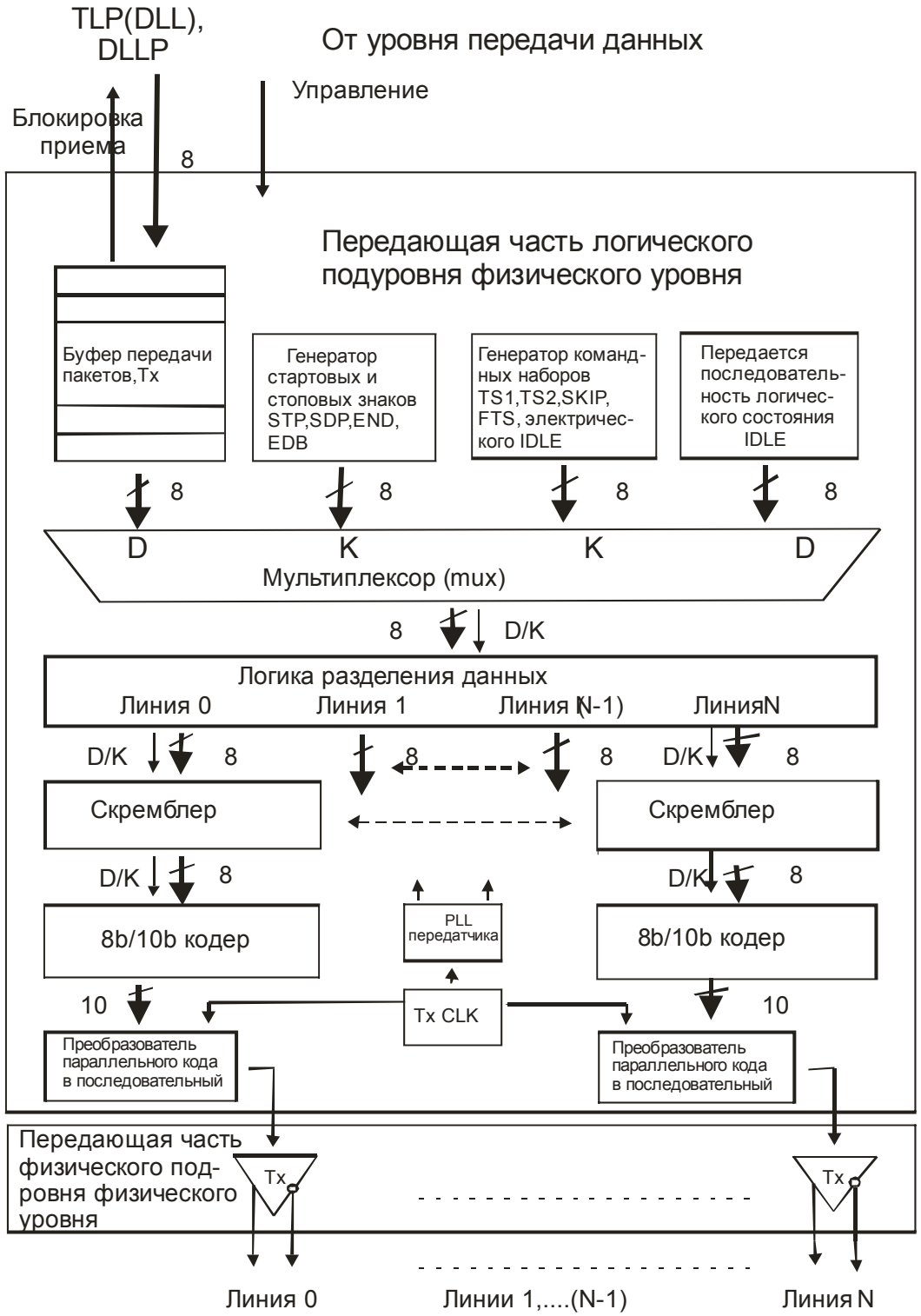

Рис. 2.5.11. Передающая логика логического подуровня физического уровня.

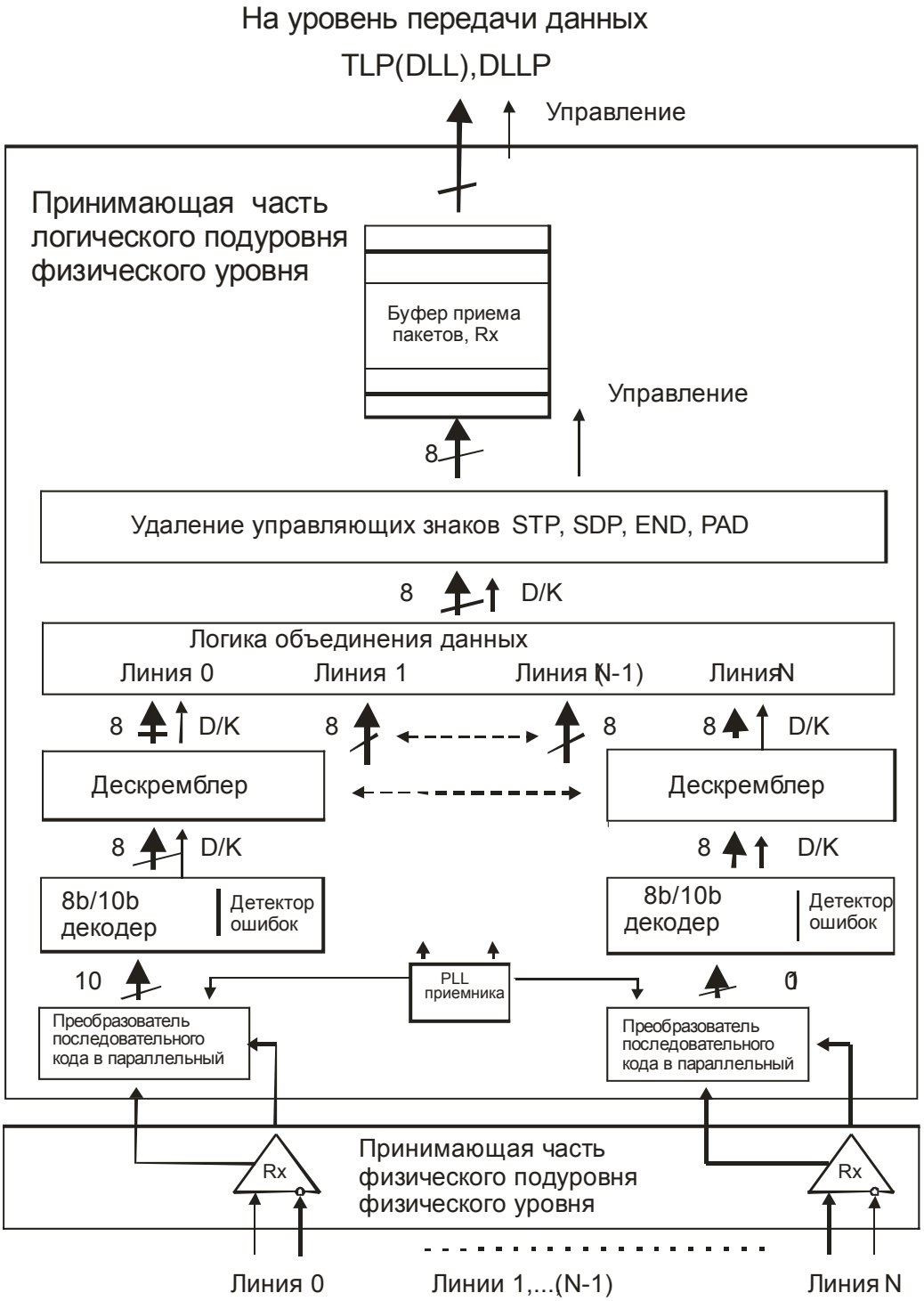

Рис. 2.5.12. Принимающая логика логического подуровня физического уровня.

передачи данных в случае, если передающий буфер Tx полностью заполняется.

С помощью мультиплексора (Mux) физический уровень обрамляет пакет TLP(DLL) или пакет DLLP стартовыми или конечными символами. Эти символы – это кадрирующие символы, по которым приемное устройство опознает начало и конец пакета.

Однако при определенных условиях мультиплексор Mux может пропускать другие входящие данные к логике разделения данных. Существует четыре вида данных на входе мультиплексора Mux :

- пакет передаваемых данных;
- стартовые и стоповые управляющие знаки;
- командный набор;
- последовательность логического ожидания.

Пакет передаваемых данных. В этом случае на вход мультиплексора поступает байтовая последовательность, соответствующая пакету TLP(DLL) или пакету DLLP, каждый байт которой рассматривается как *знак данных* или *'D' знак.*

К началу и концу каждого TLP и DLLP добавляются *стартовые и стоповые управляющие знаки* или *К знаки*. Эти знаки позволяют приемнику распознавать начало и конец пакета. Существуют два типа стартовых знаков, первый (STP) – это стартовый знак пакета TLP(DLL), второй (SDP) – стартовый знак пакета DLLP. Существуют два типа стоповых знаков; знак END - удачное завершение пакетов TLP(DLL), DLLP и знак EDB - неудачное завершение пакетов TLP(DLL). Список управляющих знаков смотри. в табл. 2.5.13.

Табл. 2.5.13. Определение и кодирование управляющих знаков.

Название знака	8b название	10b (CRD-)	10b (CRD+)	Описание
COM	K28.5 (BCh)	001111 1010	110000 0101	Первый знак в любом командном наборе. Обнаруживается приемником и используется, чтобы достигнуть символьной синхронизации во время приема командных наборов TS1/TS2
PAD	K23.7 (F7h)	111010 1000	000101 0111	Знак заполнения пакетов
SKP	K28.0 (1Ch)	001111 0100	110000 1011	Используется в командном наборе SKIP. Командный набор используется для коррекции отклонения синхросигнала
STP	K27.7 (FBh)	110110 1000	001001 0111	Знак начала пакета TLP(DLL)
SDP	K28.2 (5Ch)	001111 0101	110000 1010	Знак начала пакета DLLP
END	K29.7 (FDh)	101110 1000	010001 0111	Знак конца правильного пакета TLP(DLL)
EDB	K30.7 (FEh)	011110 1000	100001 0111	Отмечает конец ошибочного пакета TLP(DLL)
FTS	K28.1 (3Ch)	001111 1001	110000 0110	Используется в командном наборе FTS. Этот командный набор используется для перехода из состояния L0s в L0
IDLE	K28.3 (7Ch)	001111 0011	110000 1100	Используется в командном наборе электрического ожидания. Этот командный набор используется для перевода канала связи в состояние электрического ожидания.

Командный набор – это последовательность из 4 и более управляющих знаков, начинающихся со знака COM. Командные наборы, начинаются со знака K и, в зависимости от типа набора, могут содержать D или K знаки.

Например, командные наборы настроечных последовательностей 1 и 2 (TS1 и TS2) передаются по каналу связи во время настройки канала связи. Подробнее о командных наборах смотри в разделе 2.5.6.

Последовательность логического ожидания.

Когда нет пакетов для передачи в канале связи (так называемое логическое ожидание в канале связи), вместо того, чтобы оставить канал связи в отключенном состоянии или передавать пустые пакеты, передаются знаки логического ожидания. Это гарантирует передачу сигнала по каналу связи, обеспечивая таким образом блоку фазовой автоподстройки PLL приемника возможность поддерживать синхронизацию тактового генератора приемной логики с тактовым генератором передающей логики. Кроме того, приемная логика получает возможность поддерживать битовую и кадровую синхронизацию. Последовательность логического ожидания состоит в передаче знаков 00h.

Последовательность логического ожидания формируется соответствующим генератором (смотри Рис. 2.5.11.) и передается через мультиплексор. Некоторые свойства последовательности логического ожидания. Последовательность логического ожидания состоит из 8-ми битных знаков, равных 00h. Они одновременно передаются по всем дифференциальным линиям. При этом считается, что канал находится в состоянии логического ожидания. Не надо путать состояние логического ожидания с состоянием электрического ожидания, когда канал не управляется и нет передачи пакетов; именно тогда PLL приемника теряет синхронизацию. Последовательность логического ожидания скремблируется. Это предполагает то, что в канале последовательность логического ожидания

имеет псевдослучайное значение. Приемник может отличить последовательность логического ожидания от других передач пакетов, потому что последовательность логического ожидания передается между пакетами (например, последовательность логического ожидания передается после управляющих символов END или EDB, но перед STP и SDP).

Последовательность логического ожидания кодируется кодом 8b/10b. Во время передачи последовательности логического ожидания, также периодически передаются командные наборы SKIP.

Кадрированный пакет посылается в *логику разделения данных*, которая мультиплексирует данные пакета в линии передачи данных. Первый байт пакета передается по первой дифференциальной линии, следующий – по следующей и так до исчерпания доступных дифференциальных линий, затем снова по первой дифференциальной линии, второй и т.д.. Когда канал связи реализуется из нескольких дифференциальных линий, отдельные байты, образующие пакет, разделяются между 2, 4, 8, 12, 16 или 32 дифференциальными линиями с помощью логики разделения данных. Разделение данных означает, что каждый следующий знак в потоке направляется в следующую линию. Число используемых линий конфигурируется в процессе настройки канала связи.

Общие правила форматирования пакетов следующие:

- общая длина пакета (включая стартовые и стоповые знаки) каждого пакета должна быть кратна четырем знакам;
- пакеты TLP(DLL) всегда начинаются со знака STP;
- пакеты DLLP всегда начинаются со знака SDP и имеют длину 8 знаков (6 знаков + SDP +END);
- все TLP(DLL) заканчиваются END или EDB знаками;
- пакеты DLLP заканчиваются END знаком;

- знаки STP и SDP должны быть помещены в линию 0 при начале передачи пакета после передачи логического ожидания;
- если передача пакетов не начинается с логического ожидания (например, пакеты передаются друг за другом), то STP и SDP должны начинаться на линии, с номером, кратным 4;
- любое нарушение этих правил должно быть сообщено как ошибка приемника уровню передачи данных.

Следующие правила применяются, например, когда пакеты передается через канал связи x4 (с 4 линиями связи):
- STP и SDP знаки всегда передаются по линии 0;
- END и EDB знаки всегда передаются по линии 3;
- когда передается командный набор SKIP (для настройки тактового генератора в приемнике), он должен передаваться всем 4 линиям одновременно;
- когда передаются последовательности логического ожидания, они должны передаваться всем линиям одновременно;
- любое нарушение этих правил может быть сообщено как ошибка приемника канальному уровню.

Рис. 2.5.13. иллюстрирует последовательность пакетов и командных наборов, передаваемых по каналу связи x4:
- один пакет TLP(DLL);
- командный набор SKIP, передающийся по всем линиям для периодической подстройки тактового генератора приемника;
- один пакет DLLP;
- в конце передаются знаки логического ожидания по всем дифференциальным линиям, поскольку нет больше пакетов TLP(DLL) и DLLP для передачи.

Скремблер использует алгоритм псевдослучайного скремблирования каждого байта пакета. Стартовый и конечный байты кадра не скремблируются. Скремблирование удаляет повторяющиеся последовательности в битовом потоке. Проблема состоит в том, что

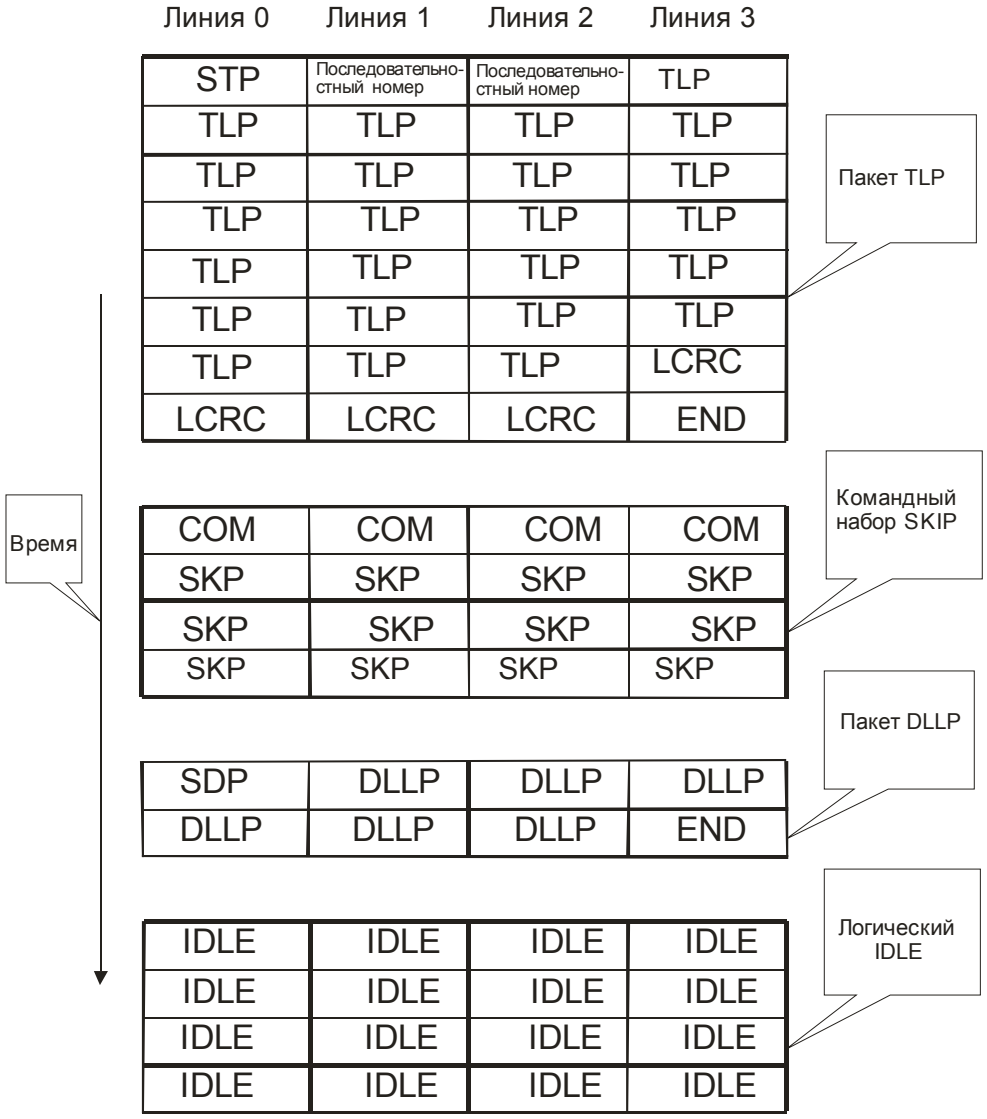

Рис. 2.5.13. Последовательность пакетов и командных наборов, передаваемых по каналу связи x4

повторяющиеся последовательности в битовом потоке приводят к тому, что значительная часть мощности, передаваемая сигналом, концентрируется на

дискретных частотах, что порождает значительный EMI шум. В случае систем с неэкранированными проводами и высокочастотной передачей в 2.5 Гбит/с, EMI шум достигает значительных величин. Скремблирование делает излучаемую мощность канала связи похожей на белый шум. Это помогает обеспечить требования злектромагнитной совместимости.

Скремблер, смотри рис. 2.5.14., реализован на 16-ти битном сдвиговом регистре с обратной связью (LFSR), который реализует полином

$$G(x) = X^{16} + X^5 + X^4 + X^3 + 1$$

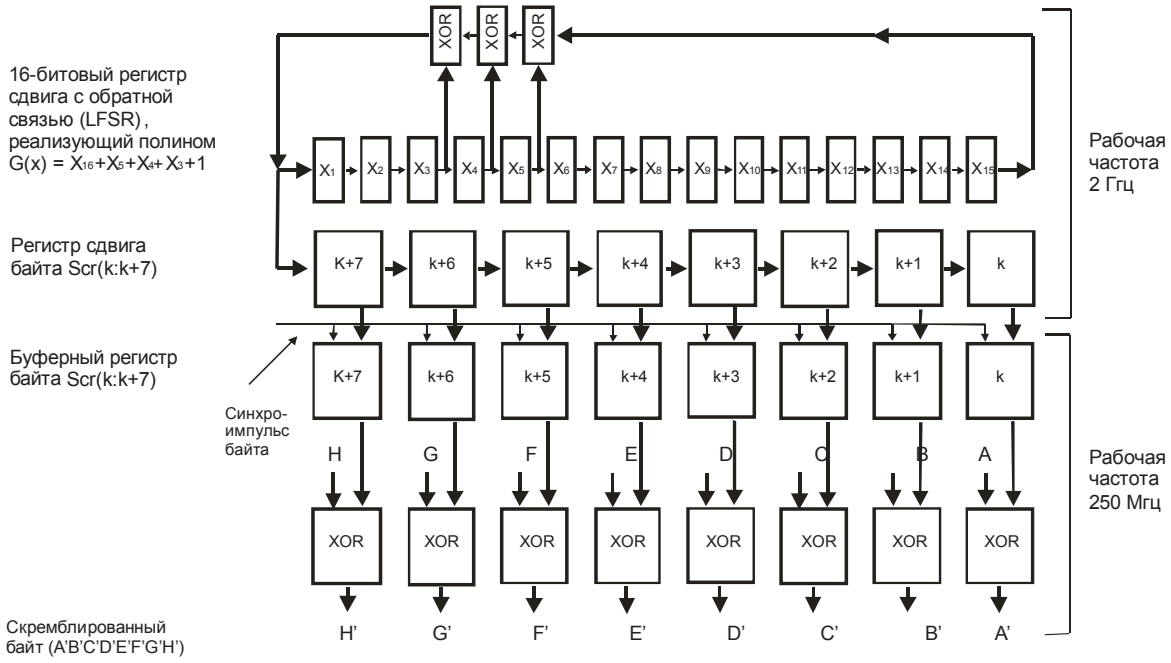

Рис 2.5.14. Скремблер

Некоторые правила реализации скремблера

1. В много линейной реализации канала связи скремблеры, связанные с каждой линией должны работать согласованно, поддерживая одно и то же значение одновременно во всех LFSR
2. Скремблирование применяется к ‘D’ знакам, составляющим пакеты TLP(DLL) и DLLP, включая последовательность логического ожидания (00h). ‘D’ знаки в командных наборах TS1 и TS2 не скремблируются.

3. ‘K’ знаки и знаки в командных наборах, таких как TS1, TS2, SKIP, FTS и электрическое ожидание не скремблируются. Эти знаки минуют логику скремблирования.
4. Знаки, относящиеся к последовательностям согласования не скремблируются.
5. Когда знак COM выходит из скремблера (COM не скремблируется) он инициализирует LFSR. Инициализированное значение 16-тибитного LFSR равно FFFFh. Таким же образом, на стороне приемника, когда знак COM входит в дескремблер, он инициализируется.
6. С одним исключением LFSR последовательно продвигается восемь раз для передачи каждого знака (D или K знака). LFSR не продвигается на SKP знаках, принадлежащих командным набором SKIP. Причина этого состоит в том, что приемник входящего пакета может добавлять или удалять SKP символы, чтобы произвести подстройку тактового генератора. Изменение числа знаков в приемнике по сравнению с переданным числом знаков приведет к потере синхронизации между значениями LFSR приемника и передатчика.

После скремблирования 8-битные знаки (8b знаки) кодируются в 10-битные символы (10b символы) с помощью логики *8b/10b кодера***.** При этом, конечно, происходит 25%-ная потеря реальной скорости передачи из-за расширения каждого 8b знака в 10b символ. *Знак* определяется как 8-битный не кодрированный байт из пакета. 8-ми битный знак поставляется в 8b/10b кодер вместе с сигналом, показывающим является ли он знаком данных (D) или управляющим знаком (К). *Символ* определяется как 10-битный кодированный эквивалент 8-ми битного знака. **Основная задача 8b/10b кодирования знаков пакета состоит в том, чтобы создать достаточную плотность переходов 1-в-0 и 0-в-1 в битовом потоке символов так, чтобы**

приемник смог выделить в потоке символов синхросигнал с помощью *контура* *фазовой автоподстройки частоты (PLL)* и тем самым решить задачу битовой синхронизации потока символов на входе приемника и задачу кадровой синхронизации входящих пакетов. Заметим, что синхросигнал, используемый для синхронизации последовательного битового потока на выходе из передатчика отдельно не передается по линии связи. Для синхронизации входящих пакетов используется синхросигнал, выделяемый приемником с помощью контура фазовой автоподстройки частоты.

Логика 8b/10b кодирования была изобретена в IBM в 1982 году и изложена в документе ANSI X3.230-1994, статья 11 (а также IEEE 802.3z, 36.2.4) и в патенте США номер 4,486,739, под названием "Byte Oriented DC Balanced 8b/10b Partitioned Block Transmission Code".

Ниже перечисляются преимущества кодирования 8b/10b:

- встроенный синхросигнал;
- баланс постоянного тока;
- кодирование специальных управляющих знаков;
- обнаружение ошибок.

Недостаток 8b/10b кодирования состоит в том, что из-за расширения каждого 8-ми битного знака в 10-ти битный символ перед передачей, реальная скорость передачи уменьшается на 25% .

Свойства 10-тибитных (10b) символов

- При передаче 10-ти битных символов, среднее число передаваемых единиц за период времени равно числу передаваемых нулей, вне зависимости от того, какой 8-ми битный знак передается, то есть передача символов сбалансирована по постоянному току.
- Битовый поток 10- битного символа никогда не содержит более пяти последовательных единиц или нулей.

- Каждый 10-ти битный символ содержит четыре 0 и шесть 1 (не обязательно смежных) или шесть 0 и четыре 1 (не обязательно смежных) или пять 0 и пять 1(не обязательно смежных).

- Каждый 10-ти битный символ разделен на два блока: первый длиной шесть бит и второй длиной четыре бита. В 6-ти битном блоке не более четырех 1 или четырех 0. В 4-х битном блоке не более трех 1 или трех 0.

- Любой символ, не обладающий вышеописанными свойствами, считается ошибочным и преемник сообщает об ошибке.

Понятие *неравенство символов* относится к разности между числом единиц и нулей в 10-тибитном символе. Когда в символе число нулей больше, чем число единиц – это *отрицательное неравенство* (например, 0101000101b).

Когда в символе число единиц больше, чем число нулей – это *положительное неравенство* (например, 1001101110b). Когда в символе одинаковое число единиц и нулей – это *нейтральное неравенство* (например, 0110100101b).

Каждый 10-ти битный символ содержит одно из следующих количеств нулей и единиц (не обязательно последовательных):

- четыре 0 и шесть 1 (+ неравенство);
- шесть 0 и четыре 1 (- неравенство);
- пять 0 и пять 1 (нейтральное неравенство)

Кодер на выходе выдает эквивалентный 10-ти битный символ вместе с *текущей проверкой четности* (Current Running Disparity, CRD), или *текущим неравенством символа* (Current Running Disparity,CRD), которое отражает баланс общего числа единиц и нулей, переданных по линии с момента ее инициализации, и имеет следующие характеристики:

- текущее состояние CRD отражает баланс единиц и нулей, переданных с момента инициализации линии;

- начальное состояние CRD (перед передачей данных) может быть положительным (+) или отрицательным (–);
- текущее состояние CRD может быть положительным (+), если передано больше единиц, чем нулей, или отрицательным (-), если передано больше нулей, чем единиц;
- каждый знак конвертируется по таблице, учитывая текущее состояние CRD;
- после того, как закодирован последний символ, CRD остается таким же, если сгенерированный 10-ти битный символ имеет *нейтральное неравенство* (н), или меняется на противоположную полярность, если сгенерированный символ имеет *положительное (+) или отрицательное (–) неравенство*.

Процедура кодирования 8b/10b

Кодирование производится путем просмотра двух пар таблиц (одна пара для кодирования знаков данных, другая пара для кодирования знаков команд), Рис.2.5.15. Выбор пары таблиц определяется значением D/K знака. Как для знаков данных, так и для знаков команд биты кодируемого знака ABCDE, поступающие на вход кодера из скремблера, с учетом значения CRD, вычисленного по результатам кодирования предшествующего знака, определяют по соответствующей таблице значения битов abcdei части символа (таблица 5b/6b кодирования). Как для знаков данных, так и для знаков команд биты кодируемого знака FGH, поступающие на вход кодера из скремблера, с учетом значения CRD, вычисленного по результатам кодирования предшествующей части abcdei знака, определяют по соответствующей таблице значения битов fghj части символа (таблица 3b/4b кодирования). Совокупность битов abcdeifghj образует символ знака D/K

ABCDEFGH. По совокупности битов abcdeifghj также вычисляется текущее неравенство символа, которое в свою очередь определяет значение CRD для кодирования следующего знака .

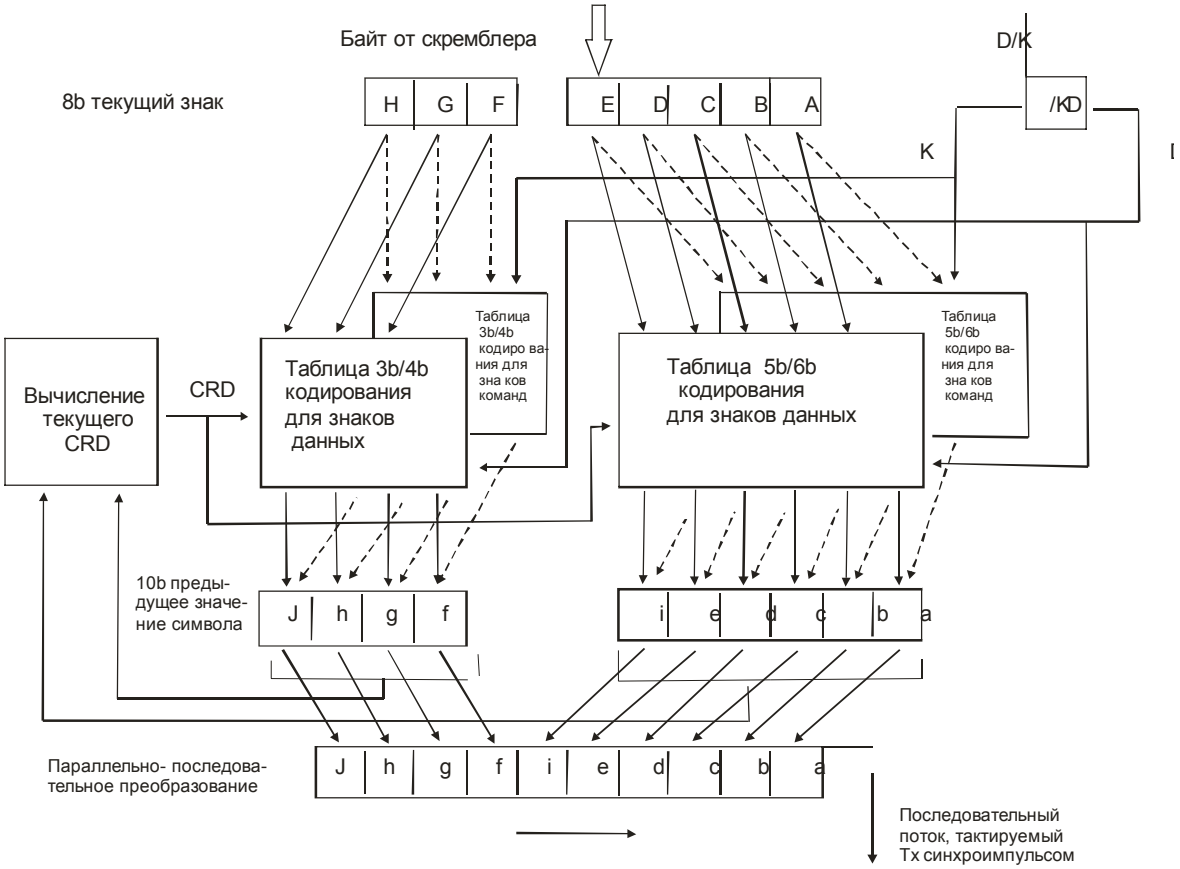

Рис. 2.5.15. Процедура кодирования 8b/10b

1. Для удобства записи знаки данных и знаки команд из двоичной формы переводятся в форму имени знака. На рис. 2.5.16. поясняется алгоритм такого перевода:

- биты в знаке обозначаются заглавными буквами от A до H;
- знак разделяется на два блока, первый знак длиной 3 бита (старшие) и второй знак длиной 5 бит (младшие);
- блоки зеркально отображаются;
- знак принимает форму Zxx.y, где:

Z - D или K для знаков данных или управляющих знаков

xx - десятичное значение 5-ти битного поля

y - десятичное значение 3-х битного поля

Выбранный для примера знак 6Ah на рис. 2.5.16. представляется как D10.3 в таблицах преобразований 8b/10b

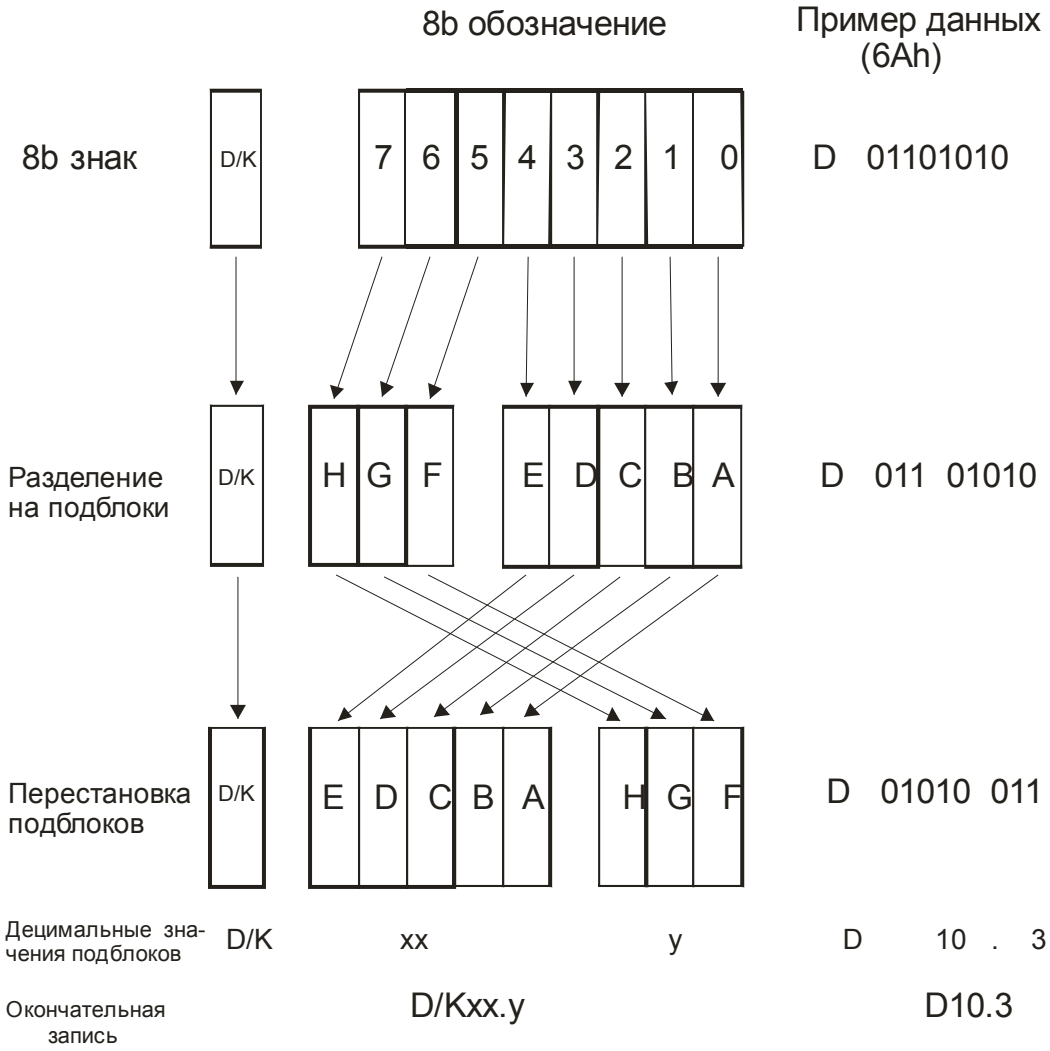

Рис. 2.5.16. Алгоритм перевода знака из двоичной формы в форму имени знака.

2. Если текущее неравенство предыдущего символа положительно (CRD +), то при кодировании 5-ти разрядного блока EDCBA из табл. 2.5.14. (для знака

данных) и из табл. 2.5.15.(для знака команды) выбираются биты abcdei. соответствующие текущему значению CRD- .

Табл. 2.5.14.. Кодирование 5 бит в 6 бит для знаков данных

Название байта данных	Некодированные биты EDCBA	Текущее RD - abcdei	Текущее RD + abcdei
D0	00000	100111	011000
D1	00001	011101	100010
D2	00010	101101	010010
D3	00011	110001	110001
D4	00100	110101	001010
D5	00101	101001	101001
D6	00110	011001	011001
D7	00111	111000	000111
D8	01000	111001	000110
D9	01001	100101	100101
D10	01010	010101	010101
D11	01011	110100	110100
D12	01100	001101	001101
D13	01101	101100	101100
D14	01110	011100	011100
D15	01111	010111	101000
D16	10000	011011	100100
D17	10001	100011	100011
D18	10010	010011	010011
D19	10011	110010	110010
D20	10100	001011	001011
D21	10101	101010	101010
D22	10110	011010	011010
D23	10111	111010	000101
D24	11000	110011	001100
D25	11001	100110	100110
D26	11010	010110	010110
D27	11011	110110	001001
D28	11100	001110	001110
D29	11101	101110	010001
D30	11110	011110	100001
D31	11111	101011	010100

Если текущее неравенство предыдущего символа отрицательно (CRD -), то при кодировании 5-ти разрядного блока EDCBA из табл. 2.5.14.(для знака

данных) и из табл. 2.5.15. (для знака команды) выбираются биты abcdei. соответствующие текущему значению CRD + .Т.е. при кодировании 5-ти разрядного блока EDCBA значение CRD меняется на противоположное.

Табл. 2.5.15. Кодирование 5 бит в 6 бит для управляющих знаков

Название байта данных	Некодированные биты EDCBA	Текущее RD - abcdei	Текущее RD + abcdei
K28	11100	001111	110000
K23	10111	111010	000101
K27	11011	110110	001001
K29	11101	101110	010001
K30	11110	011110	100001

3. Вычисляется неравенство для последовательности битов abcdei. Оно может принимать значения : положительное, отрицательное или нейтральное .

4. Если вычисленное значение неравенства положительно или отрицательно (+ или -), то при кодировании 3-х разрядного блока HGF из табл. 2.5.16. (для знака данных) и из табл. 2.5.17. (для знака команды) выбираются биты fghj, соответствующие текущему положительному или отрицательному значению CRD . Т.е CRD сохраняет предшествующее значение.

Если значение неравенства нейтральное, то при кодировании 3-х разрядного блока HGF из табл. 2.5.16. (для знака данных) и из табл. 2.5.17. (для знака команды) выбираются биты fghj, соответствующие текущему CRD, принятому при выборе битов abcdei.

4. Значения битов abcdeifghj составляют символ исходного двоичного знака D/K HGFEDCBA.

5. Вычисляется неравенство для символа abcdeifghj.

6. Если неравенство символа abcdeifghj положительное или отрицательное, то CRD предыдущего символа изменяет свое значение на противоположное. Если неравенство символа abcdeifghj нейтральное, то CRD предыдущего символа сохраняет свое значение.

Табл. 2.5.16. Кодирование 3 бита в 4 бита для знаков данных

Название байта данных	Некодированные биты HGF	Текущее RD - fghj	Текущее RD + fghj
--.0	000	1011	0100
--.1	001	1001	1001
--.2	010	0101	0101
--.3	011	1100	0011
--.4	100	1101	0010
--.5	101	1010	1010
--.6	110	0110	0110
--.7	111	1110/0111	0001/1000

Табл. 2.5.17. Кодирование 3 бита в 4 бита для управляющих знаков

Название байта данных	Некодированные биты HGF	Текущее RD - fghj	Текущее RD + fghj
--.0	000	1011	0100
--.1	001	0110	1001
--.2	010	1010	0101
--.3	011	1100	0011
--.4	100	1101	0010
--.5	101	0101	1010
--.6	110	1001	0110
--.7	111	0111	1000

Рис. 2.5.17. поясняет логику 8b/10b кодирования последовательности знаков D10.3, D27.0, D23.7, K23.7, K28.5.

Кодирование управляющих символов

Табл. 2.5.13. иллюстрирует кодирование управляющих знаков, принятых в PCI-XP. Эти знаки не скремблируются передающей логикой, но они

кодируются в 10-тибитные символы. Поскольку эти символы не скремблируются,

Признак знака	16-ричный знак	8- битный знак HGF EDCBA	8- битный знак с перестановкой подблоков EDCBA HGF	Имя знака	CRD предыдущего знака	Текущее CRD для битов abcdei	Биты части символа abcdei	Неравенство для битов abcdei	Текущее CRD для битов fghj	Биты части символа fghj	Неравенство для символа abcdeifghj	CRD предыдущего знака	Текущее CRD для битов abcdei	Биты части символа abcdei	Неравенство для битов abcdei	Текущее CRD для битов fghj	Биты части символа fghj	Неравенство для символа abcdeifghj
D	6Ah	011 01010	01010 011	D10.3	+	-	010101	н	-	1100	н	-	+	010101	н	+	0011	н
D	1Bh	000 11011	11011 000	D27.0	+	-	110110	+	+	0100	н	-	+	001001	-	-	1011	н
D	F7h	111 10111	10111 111	D23.7	+	-	111010	+	+	0001	н	-	+	000101	-	-	1110	н
K	F7h	111 10111	10111 111	K23.7	+	-	111010	+	+	1000	н	-	+	000101	-	-	0111	н
K	Bch	101 11100	11100 101	K28.5	+	-	001111	+	+	1010	+	-	+	110000	-	-	0101	-
					-							+						

Рис. 2.5.17. Кодирование последовательности знаков D10.3, D27.0, D23.7, K23.7, K28.5.

принимающая логика легко может обнаружить эти символы во входящем символьном потоке.

Управляющие символы имеют следующие свойства:

Знак COM (запятая). Знак COM используется в качестве первого символа командного набора. Напомним, что командные наборы это последовательности из 4-х и более знаков, используемые для специальных целей. 10-ти битное кодирование знака COM (K28.5) состоит из двух бит одной полярности, за которыми следуют пять бит противоположной полярности (001111 1010 или 110000 0101). Символы COM и FTS – это единственные символы, обладающие этим свойством, что облегчает их обнаружение физическим уровнем приемника. Приемник обнаруживает COM последовательность и указывает начало командного набора. В частности, знаки COM, ассоциированные с командными последовательностями TS1, TS2 или FTS используются приемником, чтобы достичь битовой и символьной синхронизации во входящем символьном потоке.

Знак PAD**.** Предположим, что в много линейном канале связи передатчик передает знак END, ассоциированный с окончанием пакета, по средней линии, например линии 3 в x8 канале связи. Если канал связи, переходит в состояние логического ожидания после передачи знака END, то знаки PAD используются, чтобы заполнить оставшиеся линии.

Знак SKP (пропуска). Знак SKP используется как часть командного набора SKIP. Командный набор SKIP передается для коррекции отклонения синхросигнала

Знак STP. Этот знак вставляется для идентификации начала пакета TLP(DLL).

Знак SDP. Этот знак вставляется для идентификации начала пакета DLLP.

Знак END**.** Этот знак вставляется для идентификации конца пакета TLP(DLL) или пакета DLLP, в которых не обнаружилось CRC ошибок.

Знак EDB (конец неправильного пакета). Это знак вставляется для идентификации конца пакета TLP(DLL), который передающее устройство (такое как переключатель) желает аннулировать. Возможен режим обработки на лету – это режим, при котором переключатель передает пакет со своего входного порта на выходной порт с минимальной задержкой без предварительной буферизации входящих пакетов. Переключатель может начать передавать пакет на лету, а затем обнаружить, что пакет испорчен. Поэтому он должен инструктировать приемник пакета отклонить его. Чтобы аннулировать ошибочный пакет TLP(DLL) переключатель подставляет знак EDB в конец пакета и инвертирует LCRC относительно вычисленного значения. Приемник, который получает такой аннулированный пакет, отклоняет его и не возвращает ни пакет DLLP ACK, ни пакет DLLP NAK.

Знак FTS (последовательность быстрой настройки). Этот знак используется как часть командного набора FTS. Командный набор FTS

передается, чтобы перевести канал связи из состояния L0s обратно в состояние L0.

Знак IDL (ожидание). Этот знак используется как часть командного набора электрического ожидания. Этот командный набор передается, чтобы проинформировать приемник, что канал связи собирается перейти в состояние L0s (также называемое состоянием электрического ожидания).

Параллельно-последовательный преобразователь

8b/10b кодер на каждой линии подает данные в *параллельно-последовательный преобразователь*, соответствующий данной линии. Параллельно-последовательный преобразователь выдает 10-тибитные символы в битовом порядке 'abcdeifghj', первым передается младший бит (a), а последним – старший (j) (как показано на рис. 2.5.15). Символы, передаваемые 8b/10b кодером, передаются в параллельно-последовательный преобразователь с частотой 250МГц. Последовательный битовый поток передается в дифференциальный драйвер линии параллельно-последовательным преобразователем с частотой 2.5 ГГц.

Дифференциальный драйвер передатчика Tx

Дифференциальный драйвер передатчика Tx, который управляет последовательным битовым потоком в линии (или оптоволокне), использует NRZ кодирование и управляет последовательным битовым потоком на скорости 2.5 Гбит/с. Выходной сигнал драйвера передатчика дифференциальный (используются две линии сигналов D+ и D-). Логическая единица передается положительной разностью напряжений между сигналами D+ и D-. Логический ноль передается отрицательной разностью напряжений между сигналами D+ и D-. Переменное напряжение двойной амплитуды, созданное передатчиком находится в диапазоне между 800мВ(мин.) и 1200мВ(макс.)

В состоянии электрического ожидания дифференциальный драйвер передатчика устанавливает переменное напряжение между 0мВ и 20мВ.

Синхросигнал передатчика (TxCLK)

Битовая последовательность с выхода параллельно-последовательного преобразователя на каждой линии передается в дифференциальный драйвер с помощью Tx синхроигнала (см. рис. 2.5.11). Частота синхросигнала Tx равна 2.5 ГГц и должна отличаться не более, чем на +/-300ppm(частей на миллион) от центральной частоты 2.5 ГГц (или всего 600ppm). Синхросигнал может отклоняться на интервал времени в один такт синхросигнала за каждые 1666 тактов синхросигнала

Правила вставки командного набора SKIP

Когда принимающая логика принимает символьный поток, иногда требуется добавить или удалить символ из него, чтобы компенсировать разницу синхронизирующих частот между передатчиком и приемником

Очевидно, что логика приемника не может произвольно подбирать символ для добавления или удаления. Это значит, что передающая логика должна периодически предавать последовательность управляющих символов, которая может использоваться для этой цели. Эта последовательность называется командным набором SKIP (смотри табл. 2.5.13), который состоит из знака COM и следующих за ним трех SKP знаков. Передатчик должен периодически передавать командные наборы SKIP. При этом выполняются следующие правила:

- командный набор SKIP должен планироваться для вставки не чаще, чем раз за 1180 символьных тактов и не реже, чем раз за 1538 символьных тактов;
- когда требуется вставить командный набор SKIP, он вставляется на границе следующего пакета (а не в середине пакета);

- если происходит передача длинного пакета, командные наборы SKIP собираются и затем последовательно вставляются на границе следующего пакета;
- в много линейной реализации командный набор SKIP должен передаваться по всем линиям одновременно (смотри рис.2.5.13);
- в течение времени, пока имеет место состояние пониженного питания, любые счетчики, используемые для планирования командных наборов SKIP, должны быть сброшены;
- командные наборы SKIP не должны передаваться, когда передается последовательность согласования.

Принимающая часть логического подуровня физического уровня

Рис. 2.5.12. поясняет принимающую часть логического подуровня физического уровня. Ниже в самом общем виде описывается обработка пакетов от момента, когда данные в символьной форме принимаются последовательно на каждой линии до момента, когда пакетный битовый поток передается на уровень передачи данных.

Рис. 2.5.18. раскрывает более детально входную часть принимающей логики для каждой линии логического подуровня физического уровня, охватывающую дифференциальные драйвер приемника и преобразователь последовательного кода в параллельный (смотри рис. 2.5.12.) Эта входная часть включает в себя:

- дифференциальный приемник;
- логику восстановления синхросигнала Rx;
- детектор символов COM и командных наборов;
- последовательно-параллельный преобразователь;
- логику устранения расфазировки между линиями (схема задержки)
- буфер переменной емкости и логика коррекции искажения синхросигнала

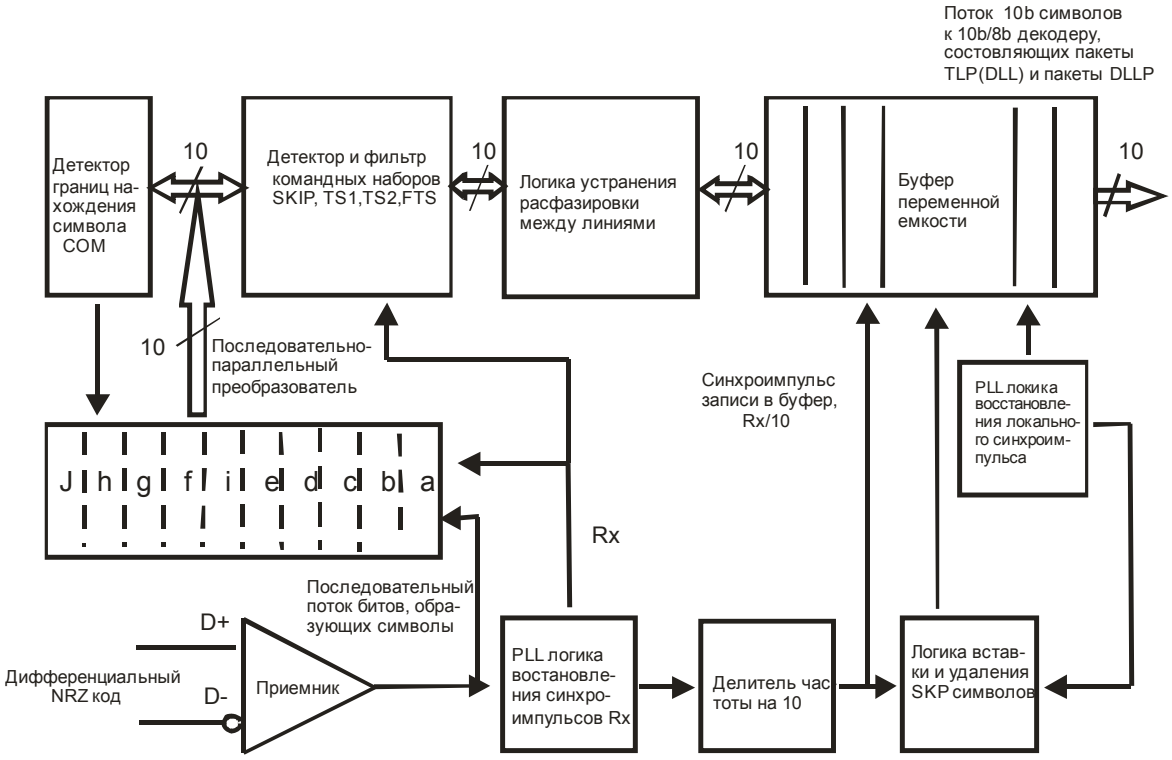

Рис. 2.5.18. Входная часть принимающей логики

Дифференциальный драйвер приемника

Дифференциальный драйвер приемника на каждой линии обнаруживает разность переменных напряжений (D+)-(D-) амплитудой > 175 мВ, но < 1200 мВ. Положительная разность соответствует логической 1, отрицательная разность соответствует логическому 0. Разность переменных напряжений (D+)-(D-) амплитудой < 65 мВ считается отсутствием сигнала и канал находится в состоянии электрического ожидания. В это время приемник блокирует свой вход, чтобы не срабатывала схема обнаружении ошибок. Разность переменных напряжений (D+)-(D-) амплитудой между 65 мВ и 175 мВ служит защитной полосой от шума.

Восстановление Rx синхросигнала

Используя PLL (цепь фазовой синхронизации), логика принимающей части восстанавливает Rx синхросигнал из последовательности битов данных

во входном потоке данных. Восстановленный синхросигнал имеет ту же частоту (2.5 ГГц), что и Tx синхросигнал, используемый передающим устройством. Rx синхросигнал используется для передачи входящего последовательного символьного потока в последовательно-параллельный преобразователь. 10-ти битный символьный поток, сгенерированный последовательно-параллельным преобразователем передается в буфер регулируемой емкости.

Достижение битовой синхронизации

Вспомним, что входящий последовательный символьный поток гарантированно имеет частые переходы из 1-в-0 и из 0-в-1, благодаря применению кодирования 8b/10b. Переходы из 1-в-0 и из 0-в-1 гарантируются по крайней мере раз в течение времени передачи 5 бит. PLL приемника использует фронты переходов в принятом битовом потоке, чтобы синхронизировать Rx синхросигнал с Tx синхросигналом, который используется в передатчике при формировании последовательного битового потока. Когда PLL приемника синхронизируется на частоте Tx синхросигнала, говорят, что приемник достиг *битовой синхронизации*.

Во время настройки канала связи передающее устройство передает длинные серии последовательных командных наборов TS1 и TS2 приемнику, и приемник использует битовые фронты в этих командных наборах, чтобы достичь битовой синхронизации. Когда канала находится в состоянии L0 (?), фронты переходов осуществляются регулярно и PLL приемника может поддерживать битовую синхронизацию

Если канал переходит в состоянии низкого питания (L0s), когда передача пакетов прекращается, PLL приемника постепенно теряет битовую синхронизацию. Передатчик посылает командный набор электрического ожидания, чтобы приемник блокировал свой вход и не допустил срабатывания схемы обнаружения ошибок.

Когда канал связи находится в состоянии L0s, передатчик посылает несколько командных наборов FTS приемнику, и приемник использует их, чтобы восстановить битовую синхронизацию. Приемнику требуется всего несколько FTS, чтобы достичь битовой синхронизации (соответственно, задержка пробуждения невысока).

Последовательно-параллельный преобразователь

Входящие последовательные данные в каждой линии передаются в последовательно-параллельный преобразователь с помощью Rx синхросигнала (смотри рис. 2.5.18). Образуемые последовательно-параллельным преобразователем 10-тибитные символы передаются в буфер переменной емкости с частотой, равной частоте Rx синхросигнала, деленный на 10.

Обнаружение границ символов (символьная синхронизация)

Когда принимающая логика начинает принимать битовый поток, это просто набор битов без маркеров и без отделения одного символа от другого. Принимающая логика должна иметь возможность определить начало и конец 10-тибитного символа. Этой цели служит символ запятой (COM). 10-тибитное кодирование символа COM (K28.5) содержит два бита одной полярности, а затем пять бит противоположной полярности (0011111010b или 1100000101b). Если не происходит ошибок, никакие другие знаки не обладают этим свойством, что облегчает их обнаружение. Вспомним, что управляющий знак COM, как и все другие управляющие знаки, не скремблируется в передатчике. Поэтому, знак COM легко обнаруживается COM детектором, который ищет два последовательных нуля или две последовательные единицы, за которыми следуют пять единиц или пять нулей соответственно. После обнаружения COM символа, COM детектор знает, что следующий символ после символа COM – это первый бит правильного 10-тибитного символа. Затем последовательно-параллельный преобразователь настраивается так, что он

может с этого момента времени выделять из потока битов правильные 10-тибитные символы. Говорят, что последовательно-параллельный преобразователь достиг с*имвольной синхронизации*

Символ COM используется для достижения символьной синхронизации в следующих случаях:

- во время настройки и перенастройки канала, когда канал устанавливается впервые; передаются командные наборы TS1 и TS2 (и каждый набор начинается с символа COM)
- когда передатчиком посылаются командные наборы FTS, чтобы приемник восстановил битовую и символьную синхронизации и изменил состояние канала с L0s в L0 .

Роль буфера переменной емкости в приемнике

Чтобы компенсировать разность частоты между Rx синхросигналом (который получается из синхросигнала Tx удаленного порта) и локальным синхросигналом используется буфер переменной емкости (смотри рис. 2.5.18) для вставки и удаления добавочных SKP символов к уже принятым в буфер переменной емкости; при этом:

- если частота синхросигнала передатчика больше, чем частота синхросигнала приемника на величину до 600ppm, SKP символ удаляется из буфера;
- если частота синхросигнала передатчика меньше, чем частота синхросигнала приемника на величину до 600ppm, SKP символ добавляется в буфер.

Устранение искажений между линиями

Проблема искажений между линиями существует только для много линейных каналов. Символы передаются одновременно по всем линиям, используя один и тот же синхросигнал, но нельзя ожидать, что сигналы поступят на вход приемной части в одно и то же время (т.е., без искажений

между линиями). Много линейный канал может иметь много причин искажений между линиями. Эти причины, в частности, следующие:

- разброс параметров дифференциальных драйверов и приемников;
- разброс сопротивлений на печатной плате;
- разброс физической длины линий;
- разброс задержек, вызванных логикой параллельно-последовательного преобразования и последовательно-параллельного преобразования.

Когда разделенные на байты последовательные битовые потоки, соответствующие пакетам поступают по всем линиям в приемники, они должны устранить искажения между линиями , чтобы правильно получить и обработать данные. Этот процесс называется устранение искажений в канале. Приемники используют командные наборы TS1 или TS2 во время настройки канала или командные наборы FTS во время выхода из L0s для этого.

8b/10b декодер

Каждая линия принимающей части логического подуровня физического уровня включает в себя 10b/8b декодер, на который подаются данные из буфера переменной емкости. 10b/8b декодер использует две таблицы для поиска (D и K знаки), чтобы декодировать поток 10-тибитных символов в 8-ми битные знаки данных (D) и знаки команд (K) плюс D/K# сигнал. Состояние D/K# сигнала показывает, что декодированный знак – это:

- знак данных (D), если полученный знак находится в D таблице, в этом случае сигнал D/K# передается высоким уровнем напряжения;
- знак команды (K), если полученный знак находится в K таблице, в этом случае сигнал D/K# передается низким уровнем напряжения.

Дескремблер

На дескремблер подаются данные из 10b/8b декодера. Дескремблер преобразует только знаки данных (D), соответствующие TLP(DLL) или DLLP (сигнал D/K# высокий уровень напряжения). Он не дескремблирует знаки

команд (K) или командные наборы. Знаки команд и командные наборы, поступающие из 10b/8b декодера, уже правильные.

Некоторые правила реализации дескремблера

В много линейном канале дескремблеры, соответствующие каждой линии должны работать согласовано, поддерживая одинаковое значение в каждой LFSR.

Дескремблирование применяется к ‘D’ знакам соответствующим TLP и DLLP, включая последовательность логического ожидания (00h). ‘D' знаки внутри командных наборов TS1 и TS2 не дескремблируются

'K' знаки и знаки командных наборов не дескремблируются. Эти знаки минуют логику дескремблирования.

Знаки, относящиеся к последовательности согласования не дескремблируются.

Когда знак COM входит в дескремблер, он инициализирует LFSR. Инициализированное значение 16-тибитного LFSR равно FFFFh.

LFSR последовательно продвигается 8 раз для каждого принятого знака (D или K) за одним исключением. LFSR не продвигается на знаке SKP, соответствующем командному набору SKIP, поскольку число переданных SKP знаков и SKP знаков выходящих из буфера переменной емкости могут различаться.

Фильтрация и проверка синхронизации пакетов

Последовательный байтовый поток, передаваемый логикой объединения данных, содержит пакеты TLP(PL), DLLP(PL), последовательности логического ожидания, управляющие знаки, такие как STP, SDP, END, EDB, и PAD, также как и другие типы командных наборов. Из этих знаков последовательности логического ожидания, управляющие знаки и командные наборы обнаруживаются и удаляются. Остаются пакеты TLP(PL) и пакеты DLLP(PL), которые посылаются в буфер приема пакетов

Rx вместе с граничными знаками, показывающими начало и конец каждого пакета TLP(PL) и пакета DLLP(PL).

Буфер приема пакетов (Rx буфер)

Буфер приема пакетов (Rx) хранит полученные пакеты TLP(DLL) и DLLP после того, как начальный и конечный знаки были удалены. Принятые пакеты готовы к отправке на уровень передачи данных.

Интерфейс между физическим уровнем и уровнем передачи данных в спецификации не определен. Поэтому разработчик может самостоятельно решать, какую разрядность шины данных реализовывать.