Fiber Channel кратко

https://korchma-kazak.ru/berries/aspekty-ispolzovaniya-8b-10b-kodirovaniya-kak-rabotaet-fc-al-kod-s-proverkoi.html

Пропускная способность интерфейса передачи данных (Transfer rate) — это максимальная скорость, с которой данные могут передаваться через данный интерфейс. Не стоит путать пропускную способность интерфейса с пропускной способностью отдельных устройств, которые к нему подключены. Многие интерфейсы не способны передавать данные с максимально возможной скоростью, это связано с присущими им накладными расходами. Некоторые адаптеры обладают аппаратными возможностями для обработки данных без загрузки CPU, что позволяет повысить производительность, управляемость и надежность передачи данных через интерфейс. Для интерфейсов, перечисленных в таблице, пропускные способности считаются из расчета передачи по одному порту в полудуплексном режиме.

I. Отличия схем кодирования в битах и байтах

Пропускные способности для систем хранения данных обычно указываются в MB/s. Множество интерфейсов использует схему кодирования 8b/10b, которая отображает восьмибитные байты в символы размером 10 бит для передачи на физическом уровне, при этом дополнительные биты используются для управления. Поэтому для таких интерфейсов пропускная способность в MB/s считается как пропускная способность в Mb/s, деленная на 10. Накладные расходы при схеме кодирования 8b/10b составляют 20% (10-8)/10.

Начиная с 10Gb Ethernet и с 10Gb Fibre Channel (для ISL), используется схема кодирования 64b/66b с улучшенной эффективностью. Схема 64b/66b проектировалась для высокой пропускной способности FC и InfiniBand. Само по себе кодирование 64b/66b не совместимо с 8b/10b, но иногда устройства на аппаратном уровне могут поддерживать реализацию старой схемы.

16Gb Fibre Channel имеет линейную скорость 14.025Gbps, но со схемой кодирования 64b/66b имеет в два раза большую пропускную способность, чем 8Gb Fibre Channel, линейная скорость которого равна 8.5Gbps. Кодирование 64b/66b влечет за собой 3% накладных расходов (66-64)/66.

PCIe версий 1.x и 2.x использует схему кодирования 8b/10b. PCIe версии 3 использует 128b/130b, что дает всего 1.5% дополнительных расходов.

II. Схемы кодирования

Схема кодирования	Накладные расходы	Интерфейсы
8b/10b	20%	1GbE, FC (вплоть до 8Gb), IB (SDR, DDR, QDR), PCIe (1.0 и 2.0), SAS (1.0, 2.0 и 3.0), SATA, USB (вплоть до 3.0)
64b/66b	3%	10GbE, 100GbE, FC(10/16/32Gb), FCoE, IB (FDR и EDR), Thunderbolt 2
128b/130b	1.5%	PCIe (3.0 и 4.0)
128b/132b	3%	USB 3.1 Gen 2 (10Gbps)
128b/150b	14.7%	SAS 4.0

III. Пропускная способность Fibre Channel

Интерфейс	Пропускная способность	Линейная скорость	Схема кодирования	Хост-адаптер
1Gb FC	100 MB/s	1.0625 GBaud	8b/10b	PCI-X
2Gb FC	200 MB/s	2.125 GBaud	8b/10b	PCI-X
4Gb FC	400 MB/s	4.25 GBaud	8b/10b	PCI-X 2.0 или PCIe 1.0 x4
8Gb FC	800 MB/s	8.5 GBaud	8b/10b	PCI-X 1.0 x8 или PCIe 2.0 x4
16Gb FC	1600 MB/s	14.025 GBaud	64b/66b	PCI-X 2.0 x8 или PCIe 3.0 x4
32Gb FC	3200 MB/s	28.05 GBaud	64b/66b	PCIe 3.0 x8
64Gb FC	6400 MB/s	28.9 GBaud	64b/66b	PCIe 4.0

IV. Пропускная способность InfiniBand

	1X	4X	12X	Схема кодирования	Хост-адаптер
SDR	2 Gb/s	8 Gb/s	24 Gb/s	8b/10b	PCIe 1.0 x8
DDR	4 Gb/s	16 Gb/s	48 Gb/s	8b/10b	PCIe 1.0 x16 или PCIe 2.0 x8
QDR	8 Gb/s	32 Gb/s	96 Gb/s	8b/10b	PCIe 2.0 x8
FDR-10* только Mellanox	10.31 Gb/s	41.25 Gb/s	123.75 Gb/s	64b/66b	PCIe 3.0 x8
FDR	13.64 Gb/s	54.55 Gb/s	163.64 Gb/s	64b/66b	PCIe 3.0 x8

EDR	25 Gb/s	100 Gb/s	300 Gb/s	64b/66b	PCIe 3.0 x16

v. Сравнительная таблица интерфейсов

Интерфейс	**Количество устройств**	**Максимальное расстояние (м)**	**Тип кабеля**	**Реализация контроллера интерфейса**	**Скорость передачи (MB/s)**	**Аттрибуты интерфейса**
FC	16M	10 (медный кабель) или более 10 км (оптоволоконный кабель)	Медный или Оптоволоконный	Дополнительный адаптер (HBA)	100, 200, 400, 800, 1600, 3200	двухпортовый
FCoE	16M	10 (медный кабель) или очень большое (оптоволоконный кабель)	Медный или Оптоволоконный	Конвергентный сетевой адаптер (CNA) или сетевой адаптер 10GbE (NIC)	1150, 4600	двухпортовый
Infiniband	48M	15 (медный кабель) или очень большое (оптоволоконный кабель)	Медный или Оптоволоконный	Дополнительный адаптер (HCA)	1000, 2000, 4000, 7000, 12500	
iSCSI	Много	расстояние соотв. кабелю Ethernet	Медный или Оптоволоконный	Сетевой адаптер (NIC) или дополнительный адаптер (HBA)	100, 1000, 2500, 4000	
SAS (пассивный)	16K	10	Медный		300, 600, 1200	полнодуплексный, двухпортовый
SAS (активный)	16K	20	Медный	Встроен в чипсет или дополнительный адаптер (HBA)	300, 600, 1200	полнодуплексный, двухпортовый
SAS (активный)	16K	100	Оптоволоконный	Встроен в чипсет или дополнительный адаптер (HBA)	300, 600, 1200, 2400	полнодуплексный, двухпортовый
SATA	1	1	Медный	Встроен в чипсет или дополнительный адаптер (HBA)	150, 300, 600	полудуплексный, однопортовый

Thunderbolt	6	4	Медный	Встроен в чипсет	1000, 2000, 4000	
USB	127	5	Медный или Беспроводное соединение	Встроен в чипсет или дополнительный адаптер (Adapter card)	0.15, 1.5, 48, 500, 1000	однопортовый

Любое устройство (компьютер, сервер, принтер, накопитель и т. д.), имеющее возможность обмениваться данными с использованием технологии Fibre Channel, называется N_порт (Node port) или просто узел. В настоящее время межузловой обмен происходит по полнодуплексным последовательным соединениям со скоростью 1.0625 GBit/s.

Множество связанных между собой N_портов Fibre Channel образуют среду распространения сигнала. Во избежание смешения с устоявшимися понятиями разработчики данной технологии решили не использовать для ее обозначения слово «network», и предпочли ввести новый термин «fabric». Учитывая, что нам до сих пор ни разу не попадались удачные примеры перевода этого слова на русский язык, предлагаем хотя бы в рамках данной статьи называть такую среду распространения «решеткой».

Понятие решетки аппаратно независимо и не рассматривает топологию физических межсоединений между ее отдельными узлами. Единственное ограничение заключается в максимально допустимом размере адресного пространства 2 24 , т. е. более 16 миллионов узлов (эквивалентно сети класса A).

Архитектурная модель Fibre Channel в деталях описывает параметры соединений и протоколы обмена между отдельными узлами решетки. Примерно как и в случае сетевых протоколов, эта модель может быть представлена в виде многослойного стека функциональных уровней, хотя и без непосредственного соответствия с OSI. Пять функциональных уровней архитектуры Fibre Channel определяют физический интерфейс, схему кодировки, сигнальный протокол, общие процедуры и связь с приложениями. Нумерация идет от самого низкого аппаратного уровня FC-0, отвечающего за параметры физического соединения до верхнего программного FC-4, взаимодействующего с приложениями более высокого уровня.

VI. Среда передачи и физические интерфейсы (fc-0)

На данном уровне задаются физические параметры полнодуплексного последовательного соединения между портами. В качестве среды передачи может быть использована витая пара, коаксиальный или твинаксиальный кабели, а также многомодовое или одномодовое волокно. Источниками света могут служить как мощные длинноволновые OFC лазеры, так и менее дорогие non-OFC. Для снижения задержек и уменьшения электрических и температурных перепадов по соединению идет постоянная и равномерная передача и прием сигнала. Кроме того, в целях предотвращения низкочастотных токов структура передаваемого сигнала должна быть сбалансирована постоянным чередованием 0 и 1 вне зависимости от присутствия в нем полезных данных, что упрощает точное распознавание сигнала и снижает количество ошибок. Упрощенно говоря, в FC-0 определяется способ передачи полученных с более высокого уровня бинарных последовательностей. Нелишне также отметить, что именно данный уровень позволяет изменять скорости передачи от 250 Mbit/s до 8 Gbit/s, не затрагивая остальные более высокие уровни.

VII. Протокол передачи (fc-1)

Протокол передачи определяет, как «вплести» данные в нижележащие сигналы FC-0, как установить соединения между портами и как, в случае необходимости, исправить обнаруженную ошибку. На этом уровне с помощью кодировки IBM 8b/10b 8-битовые порции данных преобразуются в 10-битовые сбалансированные по количеству 0 и 1 последовательности, что требуется для корректной работы FC-0. Для этого каждый планируемый к передаче байт (любой из 256 символов ASCII) преобразуется в четыре возможных комбинации для 10-битового представления, после чего выбираются две наиболее сбалансированные. Здесь действуют два простых правила - не менее 4 нулей и единиц в 10-битовой последовательности и не более четырех 0 или 1 подряд. В итоге, из двух предложенных ему на выбор 10-битовых последовательностей уровень FC-0 передает ту, первый символ которой отличается от последнего символа предыдущей. Таким образом, кодировка 8b/10b выполняет функцию НЧ фильтра, не пропуская низкочастотную составляющую и существенно облегчая работу приемника.

На этом уровне также используется символ K28.5. Эта 10-битовая последовательность не может быть получена из символов ASCII путем преобразования 8b/10b, поэтому может смело применяться в качестве служебной. Разумеется, для соответствия объявленным правилам передачи и этот символ существует как в виде позитива (1100000101), так и негатива (0011111010). В соответствии с теми же правилами

последовательность из пяти 0 или 1 не может встретиться при передаче данных и однозначно определяется, как служебный сигнал.

В результате применяемых на уровнях FC-0/1 алгоритмов вероятность возникновения ошибки на переданный бит (BER - Bit Error Rate) составляет ничтожную величину 10 - 12 , что на три порядка лучше, чем для применяемых в SCSI или Ethernet асинхронных способов передачи.

VIII. Сигнальный протокол (fc-2)

Вот мы и дошли до самого интересного, составляющего основное ядро Fibre Channel. Сигнальный протокол определяет иерархическую структуру посылок для установления связей между работающими через FC-AL приложениями. Основными объектами этого уровня являются слова (words), кадры (frames), пакеты (sequences) и обмены (exchanges).

Базовым элементом и минимальной единицей передачи является слово, но пересылка данных между узлами FC-AL требует помещения слов в некий контейнер. Ничего удивительного, что такой контейнер назван кадром.

Один или несколько последовательных кадров, несущих помещенную в них связанную информацию в виде файла данных, графики, программы или же IP-пакета, называются пакетом, который представляет из себя однонаправленную посылку от передающего узла принимающему.

Набор пакетов, которыми обмениваются узлы для обслуживания работающего через них приложения, называется обменом. Обмен, понимаемый как диалог между двумя приложениями высокого уровня, является двунаправленным, и после своего начала может оставаться открытым сколь угодно долго. Такая синтаксическая конструкция является отличительной чертой Fibre Channel и позволяет ему поддерживать огромное количество протоколов одновременно.
Здесь нужно обратить внимание, что сама решетка Fiber Channel умеет работает исключительно с кадрами, доставляя их между отправляющим и получающим узлами. Она даже не подозревает о более сложных синтаксических конструкциях в виде пакетов или обменов, которые создаются и разбираются самими узлами.

IX. Слова (words)

Передача в Fibre Channel идет строго 4-байтовыми словами, т. е. 40-битовыми последовательностями на уровне FC-0. Слова идут слитно и постоянно вне зависимости от наличия информации для передачи, когда передающий порт генерирует слова-пустышки (IDLE) в соответствии с требованиями уровня FC-0.

Для пересылки данных все байты представлены в виде символов ASCI. Если первый байт слова, вернее даже, его первые 10 бит на уровне FC-1 замещаются на служебный символ K28.5, то оно распознается, как служебное (ordered set) и, в зависимости от содержимого трех остальных байт, приобретает различную смысловую нагрузку. Примерами служебных слов могут быть IDLE (постоянная передача, когда все равно больше нечем заняться), ARB (запрос на арбитраж в петле), SOF (начало кадра) и EOF (конец кадра).

X. Кадры (frames)

Служат «транспортными контейнерами» для пересылок между отдельными узлами решетки. Начало кадра определяется служебным словом SOF (Start Of Frame), структуру которого мы рассматривали ранее. Непосредственно за SOF располагаются шесть слов заголовка и от 0 до 528 смысловых слов. Кадр завершается контрольным словом CRC (Cyclic Redundancy Check) и EOF (End Of Frame). Другими словами, размер каждого кадра может варьировать от 9 до 537 слов, что позволяет его гибко подстраивать под объем передаваемой информации.

Следующее сразу за SOF первое слово заголовка (header word 0) содержит адрес маршрутизации по решетке Fibre Channel. Остальные пять слов заголовка содержат уникальные идентификаторы последовательностей (header word 1), обменов (header word 2), определяют номер кадра в этих конструкциях (header word 3), относительное смещение в оперативной памяти (header word 4) и тип сообщений (header word 5) - SCSI, IP, AV, VI etc.

Далее размещаются смысловые слова, ради которых все это и затевалось. Их количество в кадре может варьировать от 0 до 528 и определяется передающим портом с учетом собственных возможностей и возможностей остальных портов, информацию о чем он получает во время процедуры подключения (log-in).

Служебное слово CRC является всегда предпоследним в кадре и служит для проверки правильности передачи заголовка и смысловых слов на основе содержащихся в нем контрольных 4 байт (32 бит).

Как уже все догадались, кадр закрывается с помощью слова EOF. И только после получения EOF порт опознает предыдущее слово, как CRC, ведь в нем все 4 байта являются контрольными и служебному символу K28.5 попросту не хватило места.

После столь детального разбора структуры кадра пришло время рассказать про одну маленькую деталь, о которой обычно умалчивают разработчики Fibre Channel в публикациях, рассчитанных на массовую аудиторию потенциальных пользователей. Ее некоторая скандальность заключается в том, что после завершения кадра N_порт обязан также передать не меньше 6 слов IDLE.

В основе требования передачи шести слов IDLE между кадрами заложена та же самая идея, что и при выборе кодировки 8b/10b - пожертвовать некоторой частью полосы канала для получения простого, недорогого и вместе с тем надежного механизма передачи. Кодировка 8b/10b «съедает» 20% пропускной способности канала в обмен на простой и надежный механизм распознавания слов. Шесть слов IDLE между кадрами делают примерно то же самое для обработки самих кадров. Дело в том, что в процессе передвижения кадра по решетке все встречающиеся ему на пути узлы имеют слегка отличающиеся собственные опорные частоты. В итоге, узел с несколько меньшей собственной частотой посылает кадры несколько медленнее, чем принимает, поэтому во избежание «наползания» кадров друг на друга он просто время от времени выкидывает слова IDLE между ними. С той же целью более «быстрый» узел также время от времени может вставлять недостающие IDLE, чтобы «не наступать на пятки» своему соседу.

Надеемся, что читатели специализированного издания поймут всю красоту такого подхода, когда можно не предъявлять высокие требования к кварцевым резонаторам разных производителей, к тому же работающим в разных температурных условиях.

Здесь сразу уместен вопрос о реальной пропускной способности Fibre Channel. Уже на ранних этапах разработки стандарта стало очевидным, что производители и пользователи вкладывают в это определение различный смысл, поэтому во избежание путаницы было принято соломоново решение. В настоящее время производители оборудования должны указывать аппаратную скорость компонентов в мегабитах или

гигабитах в секунду, в то время как пользователи должны руководствоваться реальной скоростью передачи данных, измеряемой мегабайтами в секунду.

Если вспомнить о 20% избыточности 8b/10b, то в данном случае 1 Gbit/s = 100 MBytes/s. А чтобы иметь возможность обеспечить поток пользовательских данных 100 MBytes/s с учетом избыточности CRC, адресных заголовков и других служебных слов, сейчас на уровне FC-0 используется скорость аппаратной передачи 1.0625 Gbit/s.

Таким образом, учитывая полнодуплексную реализацию Fibre Channel, можно говорить о представлении пользователю 200 MBytes/s в случае симметичности потоков данных, т. е. в контексте технологии FC-AL 1.0625 Gbit/s = 200 MBytes/s.

XI. Пакеты (sequences)

Синтаксическая конструкция пакетов FC-AL определяется как серия из одного или нескольких кадров, несущих отдельные порции единого блока информации. К примеру, SCSI команда вполне помещается в один кадр и, в данном случае, рассматривается, как пакет. Совершенно такая же ситуация и с 2 GB блоком данных, только пакет в этом случае будет состоять из 960 млн кадров. Блок данных всегда передается последовательно, что определяется протоколами более высокого уровня. Тем не менее, если при движении по решетке некоторые кадры будут задержаны, принимающий порт способен корректно собрать принятые кадры в блок данных на основании содержащихся в заголовке SEQ_ID (идентификатор последовательности) и SEQ_CNT (порядковый номер кадра в последовательности).

XII. Обмены (exchanges)

Каждое взаимодействие между приложениями через Fibre Channel происходит в контексте обмена. Каждый обмен имеет инициатора (originator) и ответчика (responder). Для начала обмена инциатор посылает первый кадр первого пакета обмена ответчику. Содержимое кадра может составлять, к примеру, SCSI команду. В заголовке этого кадра инициатор присваивает значение OX_ID (exchange originator identification). После этого все кадры данного обмена будут возвращаться от ответчика с этим же OX_ID, что позволит инициатору получать контекстную информацию о приложениях и протоколах более высокого уровня из собственной таблицы соответствий. Одновременно с этим ответчик приваивает собственное значение RX_ID (exchange responder identification) в первом кадре своего первого пакета в пределах

данного обмена. После получения этого кадра инициатором все дальнейшие кадры содержат уникальные идентификаторы сторон в контексте данного обмена, что позволяет точно установить принадлежность кадров при нескольких одновременных обменах. К примеру, кадры пакетов обмена SCSI командами могут приходить вперемешку с кадрами пакетов обмена IP, но порт сможет рассортировать их «на лету», отделив мух от котлет. Когда обмен завершен, соответствующие значения OX_ID и RX_ID освобождаются для использования в будущих обменах. Таким образом N_порт FC-AL может рассматриваться, как многопротокольный маршрутизатор.

Каждый порт способен начать и поддерживать до 64 тыс. конкурентных обменов. Одновременно с этим он способен отвечать еще на 64 тыс. обменов с их поддержкой.

Оригинальность синтаксической структуры Fiber Channel, поддерживаемая словами 1-5 заголовка, разительно отличает его от остальных протоколов передачи. Во-первых, порт имеет возможность конкурентной поддержки различных протоколов высокого уровня. Во-вторых, управление типами передачи выполняется на аппаратном уровне с минимальными микросекундными задержками. И, самое главное, все это происходит без участия системной шины, центрального процессора и операционной системы, т. е. не создавая потенциально узкие места.

XIII. Общие процедуры (fc-3)

Этот уровень зарезервирован под описание общих процедур при наличии двух или более N_портов в хосте. Одним из примеров такой процедуры является образование «группы захвата» (hunt group), когда два или более N_портов объединяются под единым адресом, что позволяет увеличить пропускную способность канала от порта до Fibre Channel решетки.

XIV. Отображение протоколов (fc-4)

Как и все предыдущие уровни, FC-4 также является чисто аппаратным и отвечает за преобразование различных протоколов в сигнальный протокол FC-AL и обратно. На основании заголовка пришедшего кадра содержащиеся в нем данные преобразуются и помещаются в область оперативной памяти, выделенную для приложения. Понятно, что вряд-ли эта функция может быть реализована программно при скоростях обмена 200 Mbytes/s, вследствие чего на этом уровне хранится аппаратный набор логических

матриц (profiles), определяющих бинарное соответствие FC-AL и наследуемых протоколов высокого уровня.

По нашему глубокому убеждению, именно качество стандартизации логических матриц данного уровня будет определять коммерческий успех технологии Fibre Channel в целом. Дело в том, что преобразование одного протокола в другой может производиться различными методами и на первых порах производителям не всегда удавалось договориться, чей же метод эффективнее и быстрее, особенно когда каждый из них уже потратил деньги на разработку собственного набора микросхем. Тем не менее, на сегодняшний день методы преобразования SCSI (для жестких дисков) и IP уже определены и приняты в виде стандарта. В ближайшее время будет завершена работа по принятию таких же единых методов преобразования для недисковых устройств SCSI, а также FDDI, Token Ring, ATM, AV, VI, IBM SBCCS, HIPPI и многих других. После того, как метод преобразования конкретного протокола принимается, его алгоритм становится доступным для любого производителя, что делает бессмысленным использование собственного метода и гарантирует совместимость различных устройств.

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).

3. ‘D’ знаки в командных наборах TS1 и TS2 не скремблируются.

4. ‘K’ знаки и знаки в командных наборах, таких как TS1, TS2, SKIP, FTS и электрическое ожидание не скремблируются. Эти знаки минуют логику скремблирования.

5. Знаки, относящиеся к последовательностям согласования не скремблируются.

6. 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.

7. 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 кодированный байт из пакета. Знак поставляется в 8b/10b кодер вместе с сигналом, показывающим является ли он знаком данных (D) или управляющим знаком (K).

Вопрос 28

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.

Ниже перечисляются преимущества кодирования 8b/10b:

Встроенный синхросигнал;- баланс постоянного тока;

Кодирование специальных управляющих знаков;- обнаружение ошибок.

Недостаток8b/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, 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.

Вопрос 29

В отличие от интерфейсов с топологией связей типа общая магистраль, таких как PCI и PCI-X, где информационный поток виден каждому устройству, и где задачами

маршрутизации “озабочены” главным образом мосты, устройства интерфейса PCI-XP, для которых характерна древовидная топология связей (топология связей состоит из независимых двухточечных дифференциальных линий связи, соединяющих каждое устройство с одним или несколькими соседями), вынуждены решать *задачу маршрутизации* .

Отметим, что только пакеты TLP используют информацию о маршрутизации, которая содержится в заголовке пакета, в то время, как пакеты DLLP и PLP образуют местный трафик связи между соседними устройствами и поэтому такие пакеты информации о маршрутизации не содержат. Поэтому говоря о маршрутизации, будем иметь в виду маршрутизацию транзакционных пакетов.

Когда пакет TLP(PL) достигает приемного устройства, последнее проверяет пакет на отсутствие ошибок в нем, а затем принимает одно из трех решений:

Принять и использовать пакет;

Переслать пакет на соответствующий выходной порт;

Не принимать пакет, поскольку устройство не является ни адресатом пакета, ни переключателем, за которым находится адресат пакета.

В устройствах типа конечная точка задача маршрутизации ограничивается принятием или непринятием пакета. Отметим, что до начала работы системы, через *механизм конфигурирования* должна быть запрограммирована стратегия системной маршрутизации.

В интерфейсе PCI-XP адресация поддерживается в четырех независимых адресных пространствах: адресное пространство памяти, адресное пространство устройств ввода-вывода, адресное пространство конфигурации и адресное пространство сообщений. Доступ к этим адресным пространствам выполняется с использованием разделенных (Split) транзакционных запросов и транзакций выполнения, смотри табл.2.5.1.

Все варианты TLP, нацеленные на любое из адресных пространств, используют одну из трех возможных *схем маршрутизации* :

Адресная маршрутизация,

ID маршрутизация (определяемая индентификационным номером ID)

Неявная маршрутизация.

Адресная маршрутизация предполагает, что в заголовке пакета TLP указывается конкретный адрес, а каждое устройство PCI-XP системы "знает" свой массив адресного пространства и " разбирает " пакеты из общего трафика, адресованные только ему. *ID маршрутизация* использует номер шины, номер устройства и номер функции для адресации кокретного устройства PCI-XP системы.*Неявная маршрутизация* основывается на информации, закодированной в заголовке пакета, указывающей на предназначенность пакета устройству с известным положением (например, корневому комплексу, следующему приемнику).

Стандарт Gigabit Ethernet с использованием в качестве среды передачи данных кабеля 5-й категории (неэкранированная витая пара), описанный в разделе IEEE 802.3ab, был окончательно утвержден 28 июня 1999 года.

Прошло время, и сейчас уже можно говорить, что гигабитный Ethernet по «меди» прочно вошел в историю развития локальных сетей. Резкое падение цен как на гигабитные сетевые адаптеры 1000Base-T, так и на гигабитные модули к коммутаторам постепенно привело к тому, что установка подобных адаптеров в серверы становится стандартом де-факто. К примеру, некоторые производители серверов уже стали интегрировать гигабитные адаптеры 1000Base-T на серверные материнские платы, а количество компаний, производящих такие адаптеры, в начале этого года достигло 25. Кроме того, стали выпускаться и адаптеры, предназначенные для установки в рабочие станции (они отличаются тем, что рассчитаны на 32-битную 33-мегагерцевую PCI-шину). Все это позволяет с уверенностью говорить, что через год-два гигабитные сетевые адаптеры станут столь же распространенными, как сейчас адаптеры Fast Ethernet.

Рассмотрим принципиальные нововведения, воплощенные в стандарте IEEE 802.3ab и позволившие достичь столь высокой скорости передачи, при сохранении неизменным максимального расстояния между двумя компьютерами в 100 м, как это было в стандарте Fast Ethernet.

Прежде всего напомним, что сетевые адаптеры работают на физическом и канальном уровнях семиуровневой модели OSI (Open System Interconnection). Канальный уровень принято разделять на два подуровня: MAC и LCC. Подуровень MAC (Media Access Control) - это подуровень управления доступом к среде передачи данных, обеспечивающий корректное совместное использование общей разделяемой среды передачи данных, предоставляя ее в соответствии с определенным алгоритмом в распоряжение той или иной станции. Подуровень LCC (Logical Link Control) отвечает за передачу кадров между узлами с различной степенью надежности, а также реализует функции интерфейса с прилегающим к нему третьим (сетевым) уровнем.

Все отличия между Ethernet и Fast Ethernet сосредоточены только на физическом уровне. При этом MAC и LCC не претерпели каких-либо изменений.

Физический уровень можно условно разделить на три элемента: уровень согласования, независимый от среды интерфейс (Media Independent Interface, MII) и устройство физического уровня (Physical layer device, PHY). Устройство физического уровня также можно поделить на несколько подуровней: подуровень физического кодирования, подуровень физического присоединения (Physical Medium Attachment), подуровень зависимости физической среды (Physical Medium Dependent) и подуровень автопереговоров о скорости передачи данных (Auto-Negotiation).

Если отличия между Ethernet и Fast Ethernet минимальны и не затрагивают MAC-уровня, то при разработке стандарта Gigabit Ethernet 1000Base-T разработчикам пришлось не только внести изменения в физический уровень, но и затронуть MAC-уровень (рис. 1).

Тем не менее между всеми тремя технологиями осталось много общего. Прежде всего это метод доступа к среде передачи данных CSMA/CD, полудуплексный и полнодуплексный режимы работы, а также форматы кадров Ethernet. В то же время использование витой пары кабеля 5-й категории потребовало внести серьезные изменения в реализацию физического уровня адаптера.

Первой проблемой реализации скорости 1 Гбит/с стало обеспечение приемлемого диаметра сети при работе в полудуплексном режиме работы. Как известно, минимальный размер кадра в сетях Ethernet и Fast Ethernet составляет 64 байта. Однако размер кадра в 64 байта при скорости передачи в 1 Гбайт/с приводит к тому, что для надежного распознавания коллизий необходимо, чтобы максимальный

диаметр сети (расстояние между двумя наиболее удаленными друг от друга компьютерами) составлял не более 25 м. Дело в том, что успешное распознавание коллизий возможно только в том случае, если время между посылкой двух последовательных кадров минимальной длины больше, чем двойное время распространения сигнала между двумя максимально удаленными друг от друга узлами в сети. Поэтому, чтобы обеспечить максимальный диаметр сети в 200 м (два кабеля по 100 м и коммутатор), минимальная длина кадра в стандарте Gigabit Ethernet была увеличена до 512 байт. Для увеличения длины кадра до требуемой сетевой адаптер дополняет поле данных до длины 448 байт так называемым расширением (extention). Поле расширения - это поле, заполненное запрещенными символами, которые невозможно принять за коды данных (рис. 2). В то же время увеличение минимальной длины кадра негативно сказывается при передаче коротких служебных сообщений, например квитанций, так как полезная передаваемая информация становится существенно меньше общей передаваемой информации. С целью сокращения накладных расходов при использовании длинных кадров для передачи коротких квитанций стандартом Gigabit Ethernet допускается возможность передачи нескольких кадров подряд в режиме монопольного захвата среды, то есть без передачи среды другим станциям. Такой монопольный режим захвата называется Burst Mode. В этом режиме станция может передавать подряд несколько кадров с общей длиной не более 8192 байт (BurstLength).

Как уже отмечалось, наряду с изменением MAC-уровня, достижение гигабитных скоростей передачи стало возможным благодаря существенному изменению физического уровня, то есть самой технологии представления данных (кодирования) при передаче данных по витой паре.

Для того чтобы разобраться с теми изменениями, которые были сделаны на физическом уровне, вспомним, что представляет собой кабель для передачи данных и какие помехи возникают при передаче сигналов.

Неэкранированный кабель 5-й категории состоит из четырех пар проводов, причем каждая пара перекручена между собой. Такой кабель рассчитан для работы на частоте 100 МГц (рис. 3).

Из курса физики известно, что любой кабель обладает, кроме активного, также емкостным и индуктивным сопротивлениями, причем два последних зависят от частоты сигнала. Все три типа сопротивления определяют так называемый импеданс

цепи. Наличие импеданса приводит к тому, что при распространении сигнала по кабелю он постепенно затухает, теряя часть своей первоначальной мощности.

Если взаимная индукция вычисляется в начале кабеля, то соответствующий тип помех будет называться NEXT (Near-end crosstalk loss). Если же помехи, вызванные взаимной индукцией, рассматриваются в конце кабеля, то они называются FEXT (Far-end crosstalk loss - рис. 4).

Кроме того, при распространении сигнала возникает и другой тип помех, связанный с рассогласованием входного импеданса сетевого адаптера и кабеля. В результате подобного рассогласования возникает отражение сигнала, что также приводит к образованию шума.

Передача сигналов в описанных выше условиях помех требует использования хитроумных способов, позволяющих обеспечить необходимую скорость передачи и в то же время гарантировать безошибочное распознавание передаваемых сигналов.

Прежде всего напомним, какие способы используются для представления информационных сигналов.

При цифровом кодировании битовых «нулей» и «единиц» используют либо потенциальные, либо импульсные коды. В потенциальных кодах (рис. 5) для представления логических нулей и единиц используют только значение потенциала сигнала. Например, единицу представляют в виде потенциала высокого уровня, а нуль - в виде потенциала низкого уровня. Импульсные коды позволяют представлять биты перепадом потенциала определенного направления. Так, перепад потенциала от низкого уровня к высокому может соответствовать логическому нулю.

При использовании прямоугольных импульсов для передачи данных необходимо выбрать такой способ кодирования, который бы одновременно удовлетворял нескольким требованиям.

Во-первых, имел бы при одной и той же битовой скорости наименьшую ширину спектра результирующего сигнала.

Во-вторых, обладал бы способностью распознавать ошибки.

В-третьих, обеспечивал бы синхронизацию между приемником и передатчиком.

XV. Код NRZ

В простейшем случае потенциального кодирования логическую единицу можно представлять высоким потенциалом, а логический нуль - низким. Подобный способ представления сигнала получил название «кодирование без возврата к нулю, или кодирование NRZ (Non Return to Zero)». Под термином «без возврата» в данном случае понимается то, что на протяжении всего тактового интервала не происходит изменения уровня сигнала. Метод NRZ прост в реализации, обладает хорошей распознаваемостью ошибок, но не обладает свойством самосинхронизации. Отсутствие самосинхронизации приводит к тому, что при появлении длинных последовательностей нулей или единиц приемник лишен возможности определять по входному сигналу те моменты времени, когда нужно в очередной раз считывать данные. Поэтому незначительное рассогласование тактовых частот приемника и передатчика может приводить к появлению ошибок, если приемник считывает данные не в тот момент времени, когда это нужно. Особенно критично такое явление при высоких скоростях передачи, когда время одного импульса чрезвычайно мало (при скорости передачи 100 Мбит/с время одного импульса составляет 10 нс). Другим недостатком кода NRZ является наличие низкочастотной составляющей в спектре сигнала при появлении длинных последовательностей нулей или единиц. Поэтому код NRZ не используется в чистом виде для передачи данных.

XVI. Код NRZI

Другим типом кодирования является несколько видоизмененный NRZ-код, называемый NRZI (Non Return to Zero with one Inverted). Код NRZI является простейшей реализацией принципа кодирования сменой уровня сигнала или дифференциального кодирования. При таком кодировании при передаче нуля уровень сигнала не меняется, то есть потенциал сигнала остается таким же, как и в предыдущем такте. При передаче единицы потенциал инвертируется на противоположный. Сравнение кодов NRZ и NRZI показывает, что код NRZI обладает лучшей самосинхронизацией в том случае, если в кодируемой информации логических единиц больше, чем логических нулей. Таким образом, этот код позволяет «бороться»

с длинными последовательностями единиц, но не обеспечивает должной самосинхронизации при появлении длинных последовательностей логических нулей.

XVII. Манчестерский код

В манчестерском коде для кодирования нулей и единиц используется перепад потенциала, то есть кодирование осуществляется фронтом импульса. Перепад потенциала происходит на середине тактового импульса, при этом единица кодируется перепадом от низкого потенциала к высокому, а нуль - наоборот. В начале каждого такта в случае появления нескольких нулей или единиц подряд может возникать служебный перепад потенциала.

Из всех рассмотренных нами кодов манчестерский обладает лучшей самосинхронизацией, поскольку перепад сигнала происходит как минимум один раз за такт. Именно поэтому манчестерский код используется в сетях Ethernet со скоростью передачи 10 Мбит/с (10Base 5, 10Base 2, 10Base-T).

XVIII. Код MLT-3

Код MLT-3 (Multi Level Transmission-3) реализуется аналогично коду NRZI. Изменение уровня линейного сигнала происходит только в том случае, если на вход кодера поступает единица, однако в отличие от кода NRZI алгоритм формирования выбран таким образом, чтобы два соседних изменения всегда имели противоположные направления. Недостаток кода MLT-3 такой же, как и у кода NRZI, - отсутствие должной синхронизации при появлении длинных последовательностей логических нулей.

Как уже отмечалось, различные коды отличаются друг от друга не только степенью самосинхронизации, но и шириной спектра. Ширина спектра сигнала определяется в первую очередь теми гармониками, которые дают основной энергетический вклад в формирование сигнала. Основную гармонику легко рассчитать для каждого типа кода. В коде NRZ или NRZI максимальная частота основной гармоники (рис. 6) соответствует периодической последовательности логических нулей и единиц, то есть

когда не встречается подряд нескольких нулей или единиц. В этом случае период основной гармоники равен временному интервалу двух битов, то есть при скорости передачи 100 Мбит/с частота основной гармоники должна быть 50 Гц.

В манчестерском коде максимальная частота основной гармоники соответствует ситуации, когда на вход кодера поступает длинная последовательность нулей. В этом случае период основной гармоники равен временному интервалу одного бита, то есть при скорости передачи 100 Мбит/с максимальная частота основной гармоники будет 100 Гц.

В коде MLT-3 максимальная частота основной гармоники (рис. 7) достигается при подаче на вход кодера длинных последовательностей логических единиц. В этом случае период основной гармоники соответствует временному интервалу четырех битов. Следовательно, при скорости передачи 100 Мбит/с максимальная частота основной гармоники будет равна 25 МГц.

Как уже отмечалось, манчестерское кодирование используется в сетях Ethrnet 10 Мбит/с, что связано и с хорошими самосинхронизирующими свойствами кода, и с допустимой максимальной частотой основной гармоники, которая при работе на скорости 10 Мбит/с составит 10 МГц. Этого значения достаточно для кабеля не только 5-й, но и 3-й категории, которая рассчитана на частоту 20 МГц.

В то же время использование манчестерского кодирования для более высокоскоростных сетей (100 Мбит/с, 1 Гбит/с) является неприемлемым, так как кабели не рассчитаны на работу при столь высоких частотах. Поэтому используются другие коды (NRZI и MLT-3), но для улучшения самосинхронизирующих свойств кода они подвергаются дополнительной обработке.

XIX. Избыточные коды

Такая дополнительная обработка заключается в логическом блочном кодировании, когда одна группа бит по определенному алгоритму заменяется другой группой. Наиболее распространенными типами подобного кодирования являются избыточные коды 4B/5B, 8B/6T и 8B/10T.

В этих кодах исходные группы бит заменяются на новые, но более длинные группы. В коде 4B/5B группе из четырех бит ставится в соответствие группа из пяти бит. Возникает вопрос - для чего нужны все эти усложнения? Дело в том, что такое кодирование является избыточным. К примеру, в коде 4B/5B в исходной последовательности из четырех бит существует 16 различных битовых комбинаций нулей и единиц, а в группе из пяти бит таких комбинаций уже 32. Поэтому в результирующем коде можно выбрать 16 таких комбинаций, которые не содержат большого количества нулей (напомним, что в исходных кодах NRZI и MLT-3 длинные последовательности нулей приводят к потере синхронизации). При этом остальные не используемые комбинации можно считать запрещенными последовательностями. Таким образом, кроме улучшения самосинхронизирующих свойств исходного кода избыточное кодирование позволяет приемнику распознавать ошибки, так как появление запрещенной последовательности бит свидетельствует о возникновении ошибки. Соответствие исходных и результирующих кодов приведено в табл. 1 .

Из таблицы видно, что после использования избыточного кода 4B/5B в результирующих последовательностях не встречается более двух нулей подряд, что гарантирует самосинхронизацию битовой последовательности.

В коде 8B/6T последовательность восьми бит исходной информации заменяется последовательностью из шести сигналов, каждый из которых может принимать три состояния. В восьмибитной последовательности имеется 256 различных состояний, а в последовательности шести трехуровневых сигналов таких состояний уже 729 (3 6 =729), поэтому 473 состояния считаются запрещенными.

В коде 8B/10T каждая восьмибитная последовательность заменяется на десятибитную. При этом в исходной последовательности содержится 256 различных комбинаций нулей и единиц, а в результирующей 1024. Таким образом, 768 комбинаций являются запрещенными.

Все рассмотренные избыточные коды находят применение в сетях Ethernet. Так, код 4B/5B используется в стандарте 100Base-TX, а код 8B/6T - в стандарте 100Base-4T, который в настоящее время практически уже не используется. Код 8B/10T используется в стандарте 1000Base-X (когда в качестве среды передачи данных используется оптоволокно).

Кроме использования избыточного кодирования широкое применение находит и другой способ улучшения исходных свойств кодов - это так называемое скрэмблирование.

XX. Скремблирование

Скрэмблирование (scramble - перемешивание) заключается в перемешивании исходной последовательности нулей и единиц с целью улучшения спектральных характеристик и самосинхронизирующих свойств результирующей последовательности битов. Осуществляется скремблирование путем побитовой операции исключающего ИЛИ (XOR) исходной последовательности с псевдослучайной последовательностью. В результате получается «зашифрованный» поток, который восстанавливается на стороне приемника с помощью дескрэмблера.

С аппаратной точки зрения скрэмблер состоит из нескольких логических элементов XOR и регистров сдвига. Напомним, что логический элемент XOR (исключающее ИЛИ) совершает над двумя булевыми операндами x и y, которые могут принимать значение 0 или 1, логическую операцию на основе таблицы истинности (табл. 2).

Из данной таблицы непосредственно следует основное свойство операции исключающего ИЛИ:

Кроме того, нетрудно заметить, что к операции исключающего ИЛИ применим сочетательный закон:

На схемах логический элемент XOR принято обозначать так, как показано на рис. 8 .

Как уже отмечалось, другим составным элементом скрэмблера является регистр сдвига. Регистр сдвига состоит из нескольких последовательно связанных друг с другом элементарных запоминающих ячеек, выполненных на основе триггерных схем и передающих информационный сигнал с входа на выход по управляющему сигналу - тактирующему импульсу. Регистры сдвига могут реагировать как на положительный фронт тактирующего импульса (то есть при переходе управляющего сигнала из состояния 0 в состояние 1), так и на отрицательный фронт.

Рассмотрим простейшую запоминающую ячейку регистра сдвига, управляемую по положительному фронту тактирующего импульса C (рис. 9).

В момент изменения тактирующего импульса из состояния 0 в состояние 1 на выход ячейки передается тот сигнал, который был на его входе в предыдущий момент времени, то есть когда управляющий сигнал C был равен 0. После этого состояние выхода не изменяется (ячейка заперта) вплоть до прихода следующего положительного фронта тактирующего импульса.

Используя цепочку, состоящую из нескольких последовательно связанных запоминающих ячеек с одним и тем же управляющим сигналом, можно составить регистр сдвига (рис. 10), в котором информационные биты будут последовательно передаваться от одной ячейки к другой синхронно по положительному фронту тактирующего импульса.

Составным элементом любого скрэмблера является генератор псевдослучайной последовательности. Такой генератор образуется из регистра сдвига при создании обратной связи между входом и выходами запоминающих ячеек регистра сдвига посредством логических элементов XOR.

Рассмотрим генератор псевдослучайной последовательности, изображенный на рис. 11 . Пусть в начальный момент времени все четыре запоминающие ячейки хранят некоторое предустановленное состояние. К примеру, можно предположить, что Q1=1, Q2=0, Q3=0 и Q4=1, а на входе первой ячейки D=0. После прихода тактирующего импульса все разряды сдвинутся на один бит, а на вход D поступит сигнал, значение которого определится по формуле:

Пользуясь данной формулой, нетрудно определить значения выходов запоминающих ячеек на каждом такте работы генератора. В табл. 3 показано состояние выходов запоминающих ячеек генератора псевдослучайной последовательности на каждом такте работы. При этом нетрудно заметить, что в начальный момент времени и через 15 тактов состояние генератора полностью повторяется, то есть 15 тактов работы - это период повторения нашей псевдослучайной последовательности (именно из наличия периода повторения последовательность и называется псевдослучайной). В общем случае, если генератор состоит из n-ячеек, период повторения равен:

Рассмотренный нами генератор использовал некоторое произвольное начальное состояние ячеек, то есть имел предустановку. Однако вместо такой предустановки в скрэмблерах часто используют саму исходную последовательность, подвергающуюся скрэмблированию. Такие скрэмблеры называются самосинхронизирующими. Пример подобного скрэмблера изображен на рис. 12 .

Если обозначить двоичную цифру исходного кода, поступающую на i-м такте работы на вход скрэмблера, через A i , а двоичную цифру результирующего кода, полученную на i-м такте работы, через B i , то нетрудно заметить, что рассматриваемый скрэмблер осуществляет следующую логическую операцию: , где B i -3 и B i -4 - двоичные цифры результирующего кода, полученные на предыдущих тактах работы скрэмблера, соответственно на 3 и на 4 такта ранее текущего момента.

После раскодирования полученной таким образом последовательности на стороне приемника используется дескрэмблер. Самое удивительное, что схема дескрэмблера полностью идентична схеме скрэмблера. В том, что это действительно так, нетрудно убедиться путем простых рассуждений. Если обозначить через B i двоичную цифру исходного кода, поступающую на i-м такте работы на вход дескрэмблера, а двоичную цифру результирующего кода, полученную на i-м такте работы, через C i , то дескрэмблер, работая по той же схеме, что и скрэмблер, должен реализовывать следующий алгоритм:

Следовательно, если схема дескрэмблера совпадает со схемой скрэмблера, то дескреблер полностью восстанавливает исходную последовательность информационных бит.

Рассмотренная четырехразрядная схема скрэмблера является одной из простейших. В технологии 1000Base-T используется значительно более сложный скрэмблер на 33 разряда, что увеличивает период повторения до 8 589 934 591 бит (2 33 –1), то есть формируемые псевдослучайные последовательности повторяются через 68,72 с.

XXI. Кодирование PAM-5

Разобравшись с тем, какие коды используются для представления данных, и рассмотрев методы улучшения самосинхронизирующих и спектральных свойств этих

кодов, попробуем выяснить, достаточно ли этих мер, чтобы обеспечить передачу данных на скорости 1000 Мбит/с с использованием четырехпарного кабеля 5-й категории.

Как уже отмечалось, манчестерское кодирование обладает хорошими самосинхронизирующими свойствами и в этом смысле не требует каких-либо доработок, однако максимальная частота основной гармоники численно равна скорости передачи данных, то есть количеству переданных бит в секунду. Этого достаточно для передачи данных со скоростью 10 Мбит/с, так как кабель 3-й категории (а в стандарте 10Base-T может использоваться такой кабель) ограничен частотами в 16 МГц. Однако манчестерское кодирование не годится для передачи данных со скоростью 100 Мбит/с и выше.

Использование кода NRZI после дополнительной доработки с помощью избыточного блочного кода 4B/5B и скрэмблирования, а также трехпозиционного кода MLT-3 (с целью уменьшения максимальной частоты основной гармоники) позволяет передавать данные со скоростью 100 Мбит/с по кабелю 5-й категории. Действительно, при использовании кода MLT-3 максимальная частота основной гармоники численно равна одной четвертой от скорости передачи данных, то есть при скорости передачи 100 Мбит/с частота основной гармоники не превосходит 25 МГц, что вполне достаточно для кабеля 5-й категории. Однако такой способ не годится для передачи данных на скорости 1000 Мбит/с.

Поэтому в стандарте 1000Base-T используется принципиально иной способ кодирования. Для уменьшения тактовой частоты до величин, позволяющих передавать данные по витым парам категории 5, данные в линии представляются в так называемом коде PAM-5 (рис. 13). В нем передаваемый сигнал имеет набор из пяти фиксированных уровней {–2, –1, 0, +1, +2}. Четыре из них используются для кодирования информационных битов, а пятый предназначен для коррекции ошибок. На наборе из четырех фиксированных уровней одним дискретным состоянием сигнала можно закодировать сразу два информационных бита, поскольку комбинация из двух бит имеет четыре возможные комбинации (так называемые дибиты) - 00, 01, 10 и 11.

Переход к дибитам позволяет в два раза повысить битовую скорость. Чтобы различать битовую, или информационную, скорость и скорость различных дискретных состояний сигнала, вводят понятие бодовой скорости. Бод - это количество различных дискретных состояний сигнала в единицу времени. Поэтому, если в одном дискретном

состоянии кодируется два бита, битовая скорость в два раза больше бодовой, то есть 1 Бод = 2 бит/с.

Если учесть, что кабель 5-й категории рассчитан на частоту 125 МГц, то есть способен работать с бодовой скоростью 125 МБод, то информационная скорость по одной витой паре составит 250 Мбит/с. Вспомним, что в кабеле имеется четыре витые пары, поэтому если задействовать все четыре пары (рис. 14), то можно повысить скорость передачи до 250 Мбит/сх4=1000 Мбит/с, то есть достичь желаемой скорости.

Как уже отмечалось, в кодировании PAM-5 имеется пять дискретных уровней, однако для передачи дибитов используется только четыре уровня. Пятый избыточный уровень кода (Forward Error Correction, FEC) используется для механизма построения коррекции ошибок. Он реализуется кодером Треллиса и декодером Витерби. Применение механизма коррекции ошибок позволяет увеличить помехоустойчивость приемника на 6 дБ.

XXII. Треллис-кодирование

Рассмотрим принципы треллис-кодирования на основе простейшего кодера, состоящего из двух запоминающих ячеек и элементов XOR (рис. 15). Пусть на вход такого кодера поступает со скоростью k бит/с последовательность бит 0101110010. Если на выходе кодера установить считывающую ячейку, работающую с вдвое большей частотой, чем скорость поступления бит на вход кодера, то скорость выходного потока будет в два раза выше скорости входного потока. При этом считывающая ячейка за первую половину такта работы кодера считывает данные сначала с логического элемента XOR 2, а вторую половину такта - с логического элемента XOR 3. В результате каждому входному биту ставится в соответствие два выходных бита, то есть дибит, первый бит которого формируется элементом XOR 2, а второй - элементом XOR 3. По временной диаграмме состояния кодера нетрудно проследить, что при входной последовательности бит 0101110010 выходная последовательность будет 00 11 10 00 01 10 01 11 11 10.

Отметим одну важную особенность принципа формирования дибитов. Значение каждого формируемого дибита зависит не только от входящего информационного бита, но и от двух предыдущих бит, значения которых хранятся в двух запоминающих

ячейках. Действительно, если принято, что A i - входящий бит, то значение элемента XOR 2 определится выражением , а значение элемента XOR 3 - выражением . Таким образом, дибит формируется из пары битов, значение первого из которых равно , а второго - . Следовательно, значение дибита зависит от трех состояний: значения входного бита, значения первой запоминающей ячейки и значения второй запоминающей ячейки. Такие кодеры получили название сверточных кодеров на три состояния (K = 3) с выходной скоростью ½.

Работу кодера удобно рассматривать на основе не временных диаграмм, а так называемой диаграммы состояния. Состояние кодера будем указывать с помощью двух значений - значения первой и второй запоминающих ячеек. К примеру, если первая ячейка хранит значение 1 (Q1=1), а вторая - 0 (Q2=0), то состояние кодера описывается значением 10. Всего возможно четыре различных состояния кодера: 00, 01, 10 и 11.

Пусть в некоторый момент времени состояние кодера равно 00. Нас интересует, каким станет состояние кодера в следующий момент времени и какой дибит будет при этом сформирован. Возможны два исхода в зависимости от того, какой бит поступит на вход кодера. Если на вход кодера поступит 0, то следующее состояние кодера также будет 00, если же поступит 1, то следующее состояние (то есть после сдвига) будет 10. Значение формируемых при этом дибитов рассчитывается по формулам и . Если на вход кодера поступает 0, то будет сформирован дибит 00 (), если же на вход поступает 1, то формируется дибит 11 (). Приведенные рассуждения удобно представить наглядно с помощью диаграммы состояний (рис. 16), где в кружках обозначаются состояния кодера, а входящий бит и формируемый дибит пишутся через косую черту. Например, если входящий бит 1, а формируемый дибит 11, то записываем: 1/11.

Продолжая аналогичные рассуждения для всех остальных возможных состояний кодера, легко построить полную диаграмму состояний, на основе которой легко вычисляется значение формируемого кодером дибита.

Используя диаграмму состояний кодера, несложно построить временную диаграмму переходов для уже рассмотренной нами входной последовательности бит 0101110010. Для этого строится таблица, в столбцах которой отмечаются возможные состояния кодера, а в строках - моменты времени. Возможные переходы между различными состояниями кодера отображаются стрелками (на основе полной диаграммы состояний

кодера - рис. 17), над которыми обозначаются входной бит, соответствующий данному переходу, и соответствующий дибит. Например, для двух первых моментов времени диаграмма состояния кодера выглядит так, как показано на рис. 18 . Красной стрелкой отображен переход, соответствующий рассматриваемой последовательности бит.

Продолжая отображать возможные и реальные переходы между различными состояниями кодера, соответствующие различным моментам времени (рис. 19 , ,), получим полную временную диаграмму состояний кодера (рис. 22).

Основным достоинством изложенного выше метода треллис-кодирования является его помехоустойчивость. Как будет показано в дальнейшем, благодаря избыточности кодирования (вспомним, что каждому информационному биту ставится в соответствие дибит, то есть избыточность кода равна 2) даже в случае возникновения ошибок приема (к примеру, вместо дибита 11 ошибочно принят дибит 10) исходная последовательность бит может быть безошибочно восстановлена.

Для восстановления исходной последовательности бит на стороне приемника используется декодер Витерби.

XXIII. Декодер Витерби

Декодер Витерби в случае безошибочного приема всей последовательности дибитов 00 11 10 00 01 10 01 11 11 10 будет обладать информацией об этой последовательности, а также о строении кодера (то есть о его диаграмме состояний) и о его начальном состоянии (00). Исходя из этой информации он должен восстановить исходную последовательность бит. Рассмотрим, каким образом происходит восстановление исходной информации.

Зная начальное состояние кодера (00), а также возможные изменения этого состояния (00 и 10), построим временную диаграмму для первых двух моментов времени (рис. 22). На этой диаграмме из состояния 00 существует только два возможных пути, соответствующих различным входным дибитам. Поскольку входным дибитом декодера является 00, то, пользуясь диаграммой состояний кодера Треллиса, устанавливаем, что следующим состоянием кодера будет 00, что соответствует исходному биту 0.

Однако у нас нет 100% гарантии того, что принятый дибит 00 является правильным, поэтому не стоит пока отметать и второй возможный путь из состояния 00 в состояние 10, соответствующий дибиту 11 и исходному биту 1. Два пути, показанные на диаграмме, отличаются друг от друга так называемой метрикой ошибок, которая для каждого пути рассчитывается следующим образом. Для перехода, соответствующего принятому дибиту (то есть для перехода, который считается верным), метрика ошибок принимается равной нулю, а для остальных переходов она рассчитывается по количеству отличающихся битов в принятом дибите и дибите, отвечающем рассматриваемому переходу. Например, если принятый дибит 00, а дибит, отвечающий рассматриваемому переходу, равен 11, то метрика ошибок для этого перехода равна 2.

Для следующего момента времени, соответствующего принятому дибиту 11, возможными будут два начальных состояния кодера: 00 и 10, а конечных состояния будет четыре: 00, 01, 10 и 11 (рис. 23). Соответственно для этих конечных состояний существует несколько возможных путей, отличающихся друг от друга метрикой ошибок. При расчете метрики ошибок необходимо учитывать метрику предыдущего состояния, то есть если для предыдущего момента времени метрика для состояния 10 была равной 2, то при переходе из этого состояния в состояние 01 метрика ошибок нового состояния (метрика всего пути) станет равной 2 + 1 = 3.

Для следующего момента времени, соответствующего принятому дибиту 10, отметим, что в состояния 00, 01 и 11 ведут по два пути (рис. 24). В этом случае необходимо оставить только те переходы, которым отвечает меньшая метрика ошибок. Кроме того, поскольку переходы из состояния 11 в состояние 11 и в состояние 01 отбрасываются, переход из состояния 10 в состояние 11, отвечающий предыдущему моменту времени, не имеет продолжения, поэтому тоже может быть отброшен. Аналогично отбрасывается переход, отвечающий предыдущему моменту времени из состояния 00 в 00.

Продолжая подобные рассуждения, можно вычислить метрику всех возможных путей и изобразить все возможные пути.

При этом количество самих возможных путей оказывается не так велико, как может показаться, поскольку большинство из них отбрасываются в процессе построения, как не имеющие продолжения (рис. 25). К примеру, на шестом такте работы декодера по описанному алгоритму остается всего четыре возможных пути.

Аналогично и на последнем такте работы декодера имеется всего четыре возможных пути (рис. 26), причем истинный путь, однозначно восстанавливающий исходную последовательность битов 0101110010, соответствует метрике ошибок, равной 0.

При построении рассмотренных временных диаграмм удобно отображать метрику накопленных ошибок для различных состояний кодера в виде таблицы. Именно эта таблица и является источником той информации, на основе которой возможно восстановить исходную последовательность бит (табл. 4).

В описанном выше случае мы предполагали, что все принятые декодером дибиты не содержат ошибок. Рассмотрим далее ситуацию, когда в принятой последовательности дибитов содержатся две ошибки. Пусть вместо правильной последовательности 00 11 10 00 01 10 01 11 11 10 декодер принимает последовательность 00 11 11 00 11 10 01 11 11 10, в которой третий и пятый дебит являются сбойными. Попробуем применить рассмотренный выше алгоритм Витерби, основанный на выборе пути с наименьшей метрикой ошибок, к данной последовательности и выясним, сможем ли мы восстановить в правильном виде исходную последовательность битов, то есть исправить сбойные ошибки.

Вплоть до получения третьего (сбойного) дибита алгоритм вычисления метрики ошибок для всех возможных переходов не отличается от рассмотренного ранее случая. До этого момента наименьшей метрикой накопленных ошибок обладал путь, отмеченный на рис. 27 красным цветом. После получения такого дибита уже не существует пути с метрикой накопленных ошибок, равной 0. Однако при этом возникнут два альтернативных пути с метрикой, равной 1. Поэтому выяснить на данном этапе, какой бит исходной последовательности соответствует полученному дибиту, невозможно.

Аналогичная ситуация возникнет и при получении пятого (также сбойного) дибита (рис. 28). В этом случае будет существовать уже три пути с равной метрикой накопленных ошибок, а установить истинный путь возможно только при получении следующих дибитов.

После получения десятого дибита количество возможных путей с различной метрикой накопленных ошибок станет достаточно большим (рис. 29), однако на приведенной диаграмме (с использованием табл. 5 , где представлена метрика накопленных

ошибок для различных путей) нетрудно выбрать единственный путь с наименьшей метрикой (на рис. 29

Рассмотренный пример сверточного кодера имел всего четыре различных состояния: 00, 01, 10 и 11. В технологии 1000Base-T используется сверточный кодер уже на восемь различных состояний (с тремя элементами задержки), поэтому он называется восьмипозиционным. Кроме того, поскольку символы передаются по всем четырем витым парам кабеля одновременно с использованием пятиуровневого кодирования PAM-5, такое кодирование получило название четырехмерного 4D/PAM-5.

Другим существенным отличием кодера Треллиса, используемого в технологии 1000Base-T, является алгоритм перехода между различными состояниями кодера. В рассмотренном нами простейшем примере состояние кодера в следующий момент времени определялось исключительно текущим состоянием и входным битом. Так, если текущее состояние 00, а входной бит 1, то следующее состояние, то есть поле сдвига битов по запоминающим ячейкам, будет соответствовать 10. В реальном восьмипозиционном кодере Треллиса, управляющих (входных) битов два, а переходы между различными состояниями определяются по алгоритму наибольшего расстояния между точками сигнального созвездия. Как следует из рис. 30 , кодер Треллиса реализует соотношение:

где d_6 , d_7 и d_8 - соответственно биты данных на линиях 6, 7 и 8.

Поясним это на конкретном примере.

Вспомним, что в коде PAM-5 используется пять уровней для передачи сигналов: –2, –1, 0, +1, +2. При этом уровням +2/–2 соответствует напряжение +1/–1 В, а уровням +1/–1 - напряжение +0,5/–0,5 В. Учитывая, что по четырем витым парам одновременно передается четыре уровня сигнала и каждый из этих уровней может принимать одно из пяти значений, всего получаем 625 (5x5x5x5) разных комбинаций сигналов. Различные возможные состояния сигнала удобно изображать на так называемой сигнальной плоскости. На этой плоскости каждое возможное состояние сигнала изображается сигнальной точкой, а совокупность всех сигнальных точек называют сигнальным созвездием. Естественно, что изобразить четырехмерное пространство не представляется возможным, поэтому рассмотрим для наглядности двухмерное сигнальное созвездие 5x5. Такое созвездие формально может соответствовать двум витым парам. Изобразим вдоль оси X точки, отвечающие одной

витой паре, а вдоль оси Y - другой. Тогда наше 2D-созвездие будет выглядеть так, как показано на рис. 31 .

Обратим внимание, что минимальное расстояние между двумя точками такого созвездия равно 1.

Под воздействием шума и затухания сигнала сигнальное созвездие претерпевает искажения (рис. 32), в результате которых положение каждой сигнальной точки расплывается, а расстояние между ними сокращается. Вследствие этого точки в сигнальном созвездии становятся трудноразличимыми и велика вероятность их перепутывания.

Поэтому одной из задач кодера Треллиса является такое формирование сигнального созвездия, которое обеспечивало бы максимальное расстояние между различными сигнальными точками. Для того чтобы понять, как это делается, обозначим уровни сигналов –1 и +1 через X, а уровни –2, 0, +2 через Y. Тогда исходное созвездие можно изобразить в виде, показанном на рис. 33 .

Разделив это созвездие на два подсозвездия, одно из которых сформировано из точек XX и YY, а другое - из точек XY и YX, можно увеличить расстояние между сигнальными точками до (рис. 34).

При использовании двух витых пар задача кодера Треллиса заключается в том, чтобы по одной витой паре посылать только символы, принадлежащие какому-либо одному из сигнальных созвездий, например D0=XX+YY, а по второй витой паре - символы, принадлежащие другому созвездию, например D1=XY+YX. Тогда расстояние между посылаемыми символами станет в два раза больше, чем было в исходном созвездии. В результате улучшается распознаваемость точек в сигнальном созвездии, то есть возрастает помехозащищенность.

Приблизительно по такой же схеме работает и реальный треллис-кодер, формирующий символы, посылаемые по четырем витым парам, однако, поскольку каждой точке созвездия отвечают четыре координаты (по одной на каждую пару) и каждая точка может принимать значение X или Y, то всего существует 16 различных комбинаций, из которых можно сформировать восемь подсозвездий:

В полученных подсозвездиях минимальное расстояние между точками в два раза больше, чем в исходном созвездии. Кроме того, минимальное расстояние между точками двух разных подсозвездий также равно 2. Именно эти восемь сигнальных созвездий формируют диаграмму состояний треллис-кодера. К примеру, состоянию кодера 000 соответствует комбинация точек из созвездий D0D2D4D6 в том смысле, что по первой паре передаются точки из созвездия D0, по второй паре - из созвездия D2 и т.д. Следующему возможному состоянию кодера будет соответствовать такая комбинация, при которой минимальное расстояние между посылаемыми символами по каждой паре равно 2.

Использование треллис-кодирования по описанной схеме позволяет снизить соотношение «сигнал/шум» (SNR) на 6 дБ, то есть значительно увеличить помехоустойчивость при передаче данных.

КомпьютерПресс 2"2002

Каждый вид компьютеров имеет свой внутренний вид кодирования для представления данных – символьной и текстовой информации. Наиболее часто используются коды ASCII (American Standard Code for Information Interchange, американский стандартный код для обмена информацией) и EBCDIC (Extended Binary Coded Decimal Interchange Code, расширенный двоично-десятичный код обмена информацией).

ASCII представляет собой 8-битную кодировку для представления десятичных цифр, латинского и национального алфавитов, знаков препинания и управляющих символов. Первую половину кодовой таблицы (0 - 127) занимают символы US-ASCII, которые включают 95 печатаемых и 33 управляющих символа (разработана ANSI – Американским институтом национальных стандартов). Вторая половина таблицы (128 - 255) содержит национальные шрифты (кириллица) и символы псевдографики. Этот код используется в персональных компьютерах и в несовместимых с IBM больших машинах. На больших компьютерах (мейнфреймах) используется 8-битовый код EBCDIC, разработанный компанией IBM. При передаче данных от одного компьютера к другому может потребоваться перекодировка символов, которая осуществляется системным ПО передающего или принимающего компьютеров. Эти действия являются функциями уровня представления модели OSI. Далее рассмотрим наиболее часто применяемые методы кодирования на физическом уровне.

Большинство компьютеров для представления «0» и «1» оперирует стандартными уровнями сигналов (логическими уровнями), которые определяются видом микросхем. TTL-логика представляет 0,5В как «0», и 5В как «1». ECL и CMOS-логики представляют -1,75В как «0», и -0,9В как «1». Для передачи данных, например, в оптоволоконных системах, в трансивере (приемопередатчике) устанавливается специальный чип, обрабатывающий любую логику и выдающий управляющий сигнал источнику света с конвертацией 0,5В и 5В TTL в 0 мА и 50 мА соответственно (включи свет, выключи свет).

В большинстве компьютерных сетей цифровые данные передаются при помощи цифрового сигнала, т.е. последовательностью импульсов. Для передачи данных может использоваться более двух уровней сигнала, при этом единичный импульс сигнала может представлять не один бит, а группу бит. Возможна обратная ситуация, когда для передачи одного бита может использоваться два импульса сигнала.

При цифровой передаче используют потенциальные и импульсные коды. В потенциальных кодах для представления логических единиц и нулей используется только значение сигнала в течение битового интервала, а фронты сигнала, формирующие законченные импульсы, во внимание не принимаются. Импульсные коды представляют логический ноль и логическую единицу перепадом потенциала определенного направления. В значение импульсного кода включается весь импульс вместе с его фронтами.

Сигнал в виде импульсной последовательности имеет бесконечный спектр. Основная энергия сигнала сосредоточена в диапазоне частот от нуля до частоты $f=1/t_o$ (первый лепесток энергетического спектра сигнала), где t_o – бодовый интервал, то есть длительность единичного импульса линейного сигнала.

Теоретически в соответствии с пределом Найквиста максимально допустимая скорость изменения значений дискретного сигнала ($B=1/t_o$, скорость передачи в Бодах) при передаче последовательности прямоугольных импульсов по каналу связи, эквивалентом которого является идеальный ФНЧ с прямоугольной АЧХ и линейной ФЧХ и с частотой среза $f_{гр}$, равна $B_{max}=2f_{гр}$. Указанное ограничение связано с наличием переходных процессов на выходе ФНЧ, при этом время нарастания/спада фронта сигнала определяется как

При максимально допустимой скорости передачи сигнала $t_o=t_н$. Если интервал t_o

Используемый метод физического цифрового кодирования должен достигать несколько целей:

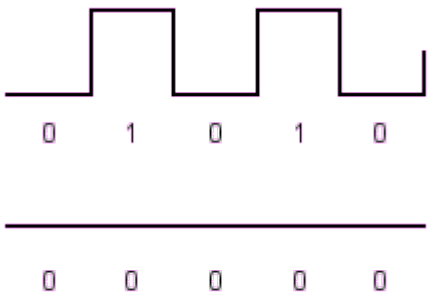
- Обеспечивать наименьшую ширину спектра сигнала при заданной скорости передачи информации N (Бит/с). Минимизировать величину постоянной составляющей в спектре линейного сигнала.
- Обеспечивать приемнику возможность тактовой синхронизации (clocking). Так называемые самосинхронизирующиеся коды позволяют приемнику выделять из принимаемого цифрового потока колебание тактовой частоты и затем формировать из него тактовые импульсы при любой статистике битового потока на входе передатчика.
- Обладать способностью распознавать ошибки
- Обладать низкой стоимостью реализации.

Рассмотрим методы физического цифрового кодирования сигналов.

XXIV. Потенциальный код без возвращения к нулю NRZ (Non Return to Zero)

Сигнал в линии имеет два уровня: нулю соответствует нижний уровень, единице - верхний. Переходы происходят на границе битового интервала. При передаче последовательности единиц сигнал не возвращается к нулю в течение битового интервала.

Рассмотрим частные случаи передачи данных кодом NRZ (рис.4.1): чередующаяся последовательность нулей и единиц, последовательность нулей и последовательность единиц. Определим частоту основной гармоники спектра сигнала в каждом из этих случаев.



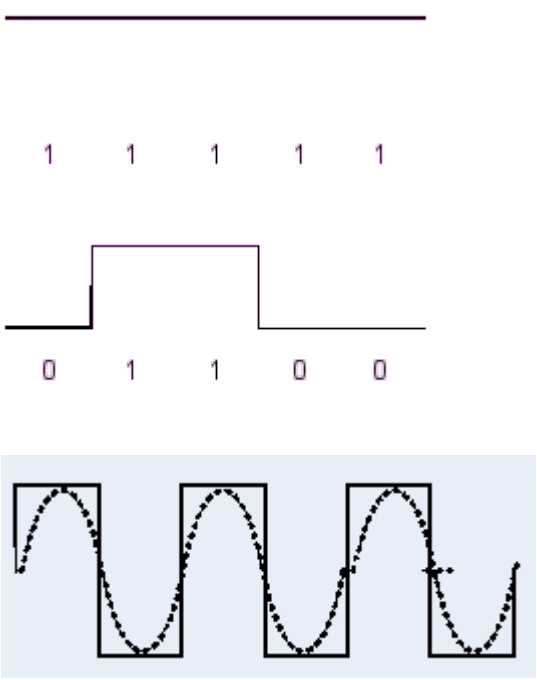


Рис.4.1. Кодирование по методу NRZ

При чередовании единиц и нулей и скорости передачи N (Бит/с) период основной гармоники в спектре сигнала равен T=2t 0 =2/N (с). Частота основной гармоники f 0 равна f o =N/2 (Гц).

При передаче только единиц, или только нулей сигнал в линии представляет собой постоянный ток.

Спектр реального сигнала постоянно меняется в зависимости от того, какова структура данных, передаваемых по линии связи. При передаче длинных последовательностей нулей или единиц, спектр сигнала сдвигается в сторону низких частот. Линейный сигнал NRZ обычно содержит постоянную составляющую и не всегда обеспечивает приемнику возможность синхронизироваться с поступающим сигналом. С другой стороны код NRZ прост в реализации, обладает хорошей помехоустойчивостью (из-за двух резко отличающихся уровней сигнала). Основная энергия сигнала в коде NRZ сосредоточена на частотах от 0 до N/2 (Гц).

В чистом виде код NRZ в сетях не используется. Тем не менее, используются его различные модификации, в которых с успехом устраняют как плохую самосинхронизацию, так и наличие постоянной составляющей.

Потенциальный код с инверсией при единице NRZI (Non Return to Zero with ones Inverted, NRZI)

При этом методе кодирования передаче нуля соответствует уровень сигнала, который был установлен в предыдущем битовом интервале (уровень сигнала не меняется), а при передаче единицы – уровень изменяется на противоположный.

Код используется при передаче по оптоволоконным кабелям, где приемник устойчиво распознает два состояния сигнала - свет и темнота.

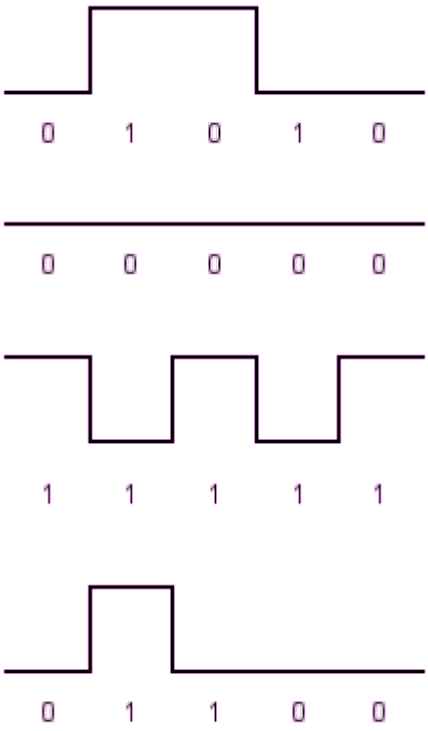


Рис.4.2. Кодирование по методу NRZI

Определим частоты основных гармоник линейного сигнала для частных случаев битовых последовательностей.

Для последовательности чередующихся единиц и нулей период сигнала равен T=4t 0 (с), основная частота сигнала f o =N/4 (Гц), при последовательности единиц – f o =N/2 (Гц), при передаче последовательности нулей f o =0 - постоянный ток в линии (или отсутствие света).

Код NRZI использует только два уровня сигнала и поэтому обладает хорошей помехоустойчивостью. Максимальную энергию имеют спектральные составляющие сигнала около частоты N/4 (Гц). Следует отметить, что код NRZI стал основным при разработке улучшенных методов кодирования для систем передачи данных.

XXV. Метод биполярного кодирования с альтернативной инверсией (Bipolar Alternate Mark Inversion, AMI, квазитроичный код)

В этом методе используются три значения сигнала –«-1», «0» и «+1». Для различения трех уровней необходимо лучшее соотношение сигнал/шум на входе приемника. Дополнительный уровень требует увеличение мощности передатчика примерно на 3 дБ для обеспечения той же достоверности приема бит на линии. Это общий недостаток кодов с несколькими состояниями сигнала по сравнению с двухуровневыми кодами.

Для кодирования логического нуля используется нулевой потенциал, логическая единица кодируется попеременно либо положительным потенциалом, либо отрицательным.

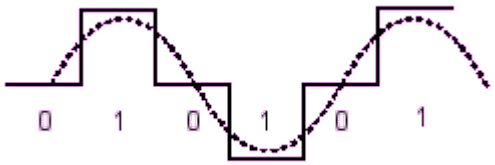


Рис.4.3. Квазитроичное кодирование (AMI)

При передаче любой битовой последовательности сигнал в линии не содержит постоянную составляющую. При передаче единиц основная гармоника сигнала находится на частоте f o =N/2 (Гц). В случае чередующегося набора единиц и нулей основная гармоника находится на частоте f o =N/4 (Гц), что в два раза меньше чем у кода NRZ. Но остается проблема синхронизации при передаче последовательности нулей.

В целом, для различных комбинаций бит использование кода AMI приводит к более узкому спектру сигнала, чем для кода NRZ. Код AMI предоставляет также некоторые

возможности по распознаванию ошибочных сигналов. Так, нарушение строгого чередования полярности сигналов говорит о ложном импульсе или исчезновении корректного импульса.

XXVI. Манчестерский код (Manchester)

Манчестерский код относится к самосинхронизирующимся кодам и имеет два уровня, что обеспечивает хорошую помехозащищенность. Каждый битовый интервал делится на две части. Информация кодируется перепадом уровня, происходящим в середине каждого интервала.

Единица кодируется перепадом от высокого уровня сигнала к низкому, а ноль - обратным перепадом. В начале каждого битового интервала может происходить служебный перепад сигнала (при передаче несколько единиц или нулей подряд).

Рис.4.4. Манчестерское кодирование

При передаче любой битовой последовательности сигнал не содержит постоянную составляющую. Длительность единичного импульса линейного сигнала t 0 равна половине битового интервала, то есть B=2N. Частота основной гармоники сигнала зависит от характера битовой последовательности и находится в диапазоне f o =N/2 – N (Гц).

Манчестерский код используется в сетях Ethernet со скоростью передачи 10 Мбит/с (спецификация 10Base-T).

В настоящее время разработчики пришли к выводу, что во многих случаях рациональнее применять потенциальное кодирование, ликвидируя его недостатки с помощью так называемого *логического кодирования* (см. ниже в этом разделе).

XXVII. Потенциальный код 2B1Q

Это потенциальный код с четырьмя уровнями сигнала для кодирования данных. Название отражает суть кодирования – каждые два бита (2B) передаются за один такт сигналом определенного уровня (1Q). Линейный сигнал имеет четыре состояния.

Дибиту «00» соответствует потенциал -2,5 В (-3), «01» - потенциал -0,833 В (-1), «11» - потенциал +0,833 В (+1), «10» - потенциал +2,5 В (+3). Скорость передачи сигнала В при таком кодировании в 2 раза меньше скорости передачи информации N. На рис.4.6 изображен сигнал, соответствующий последовательности бит: 01 01 10 00.

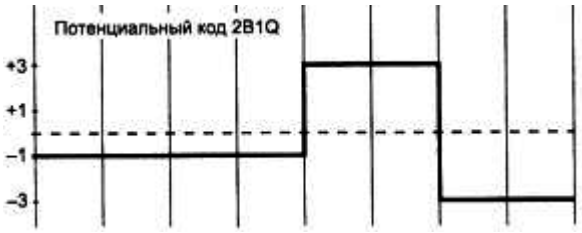


Рис.4.6. Сигнал в коде 2B1Q

Основная частота сигнала не превышает f о =N/4 Гц. Однако для реализации этого метода кодирования мощность передатчика должна быть выше, чтобы четыре значения потенциала четко различались приемником на фоне помех.

XXVIII. Код MLT3 (Multi Level Transmission - 3).

Используются три уровня линейного сигнала: «-1», «0», «+1». Логической единице соответствует обязательный переход с одного уровня сигнала на другой. При передаче логического нуля изменение уровня линейного сигнала не происходит.

При передаче последовательности единиц период изменения уровня сигнала включает четыре бита. В этом случае f о =N/4 (Гц). Это максимальная основная частота сигнала в коде MLT-3. В случае чередующейся последовательности нулей и единиц основная гармоника сигнала находится на частоте f о =N/8 (Гц).

Рис.4.7. Сигнал в коде MLT-3

Логическое кодирование выполняется передатчиком до физического кодирования, рассмотренного выше, обычно средствами физического уровня. На этапе логического кодирования борются с недостатками методов физического цифрового кодирования - отсутствие синхронизации, наличие постоянной составляющей. Таким образом, сначала с помощью средств логического кодирования формируются исправленные последовательности данных, которые потом с помощью методов физического кодирования передаются по линиям связи.

Логическое кодирование подразумевает замену бит исходной информационной последовательности новой последовательностью бит, несущей ту же информацию, но обладающей, кроме этого, дополнительными свойствами.

Различают два метода логического кодирования:

- избыточные коды;
- скремблирование.

Избыточные коды основаны на разбиении исходной последовательности бит на группы и замене каждой исходной группы в соответствии с заданной таблицей кодовым словом, которое содержит большее количество бит.

Логический код 4B/5B заменяет исходные группы (слова) длиной 4 бита словами длиной 5 бит. В результате, общее количество возможных битовых комбинаций 25=32 больше, чем для исходных групп 24=16. В кодовую таблицу включают 16 кодовых слов, которые не содержат более двух нулей подряд, и используют их для передачи данных. Код гарантирует, что при любом сочетании кодовых слов на линии не могут встретиться более трех нулей подряд.

Остальные комбинации кода используются для передачи служебных сигналов (синхронизация передачи, начало блока данных, конец блока данных, управление передачей). Неиспользуемые кодовые слова могут быть задействованы приемником для обнаружения ошибок в потоке данных. Цена за полученные достоинства при таком способе кодирования данных - снижение скорости передачи полезной информации на 25%.

Имеются также коды и с тремя состояниями сигнала, например, в коде 8B/6T для кодирования 8 бит исходной информации используются кодовые слова из 6 элементов, каждый из которых может принимать одно из трех значений. Избыточность кода 8B/6T выше, чем кода 4B/5B, так как на 28=256 исходных комбинаций приходится 36=729 результирующих комбинаций.

В *коде 8B/10B* каждые 8 бит исходной последовательности заменяются десятью битами кодового слова. При этом на 256 исходных комбинаций приходится 1024 результирующих комбинаций. При замене в соответствии с кодовой таблицей соблюдаются следующие правила:

- ни одна результирующая комбинация не должна иметь более 4-х одинаковых бит подряд;
- ни одна результирующая комбинация не должна содержать более 6 нулей или 6 единиц;

Все рассмотренные избыточные коды применяются в сетях Ethernet, которые нашли самое широкое распространение. Так, код 4B/5B используется в стандартах 100Base-TX/FX, а код 8B/6T - в стандарте 100Base-T4, который в настоящее время практически уже не используется. Код 8B/10B используется в стандарте 1000Base-X, код 64/66 в стандарте 10 GbE (когда в качестве среды передачи данных используется оптоволокно).

Осуществляют логическое кодирование сетевые адаптеры. Поскольку, использование таблицы перекодировки является очень простой операцией, метод логического кодирования избыточными кодами не усложняет функциональные требования к этому оборудованию.

Для обеспечения заданной пропускной способности линии передатчик, использующий избыточный код, должен работать с повышенной скоростью (тактовой частотой). Так, для обеспечения скорости передачи информации 100 Мбит/с с использованием кодирования 4B/5B + NRZI передатчик должен работать на скорости 125 МБод. При этом спектр линейного сигнала расширяется. Тем не менее, спектр сигнала избыточного потенциального кода уже спектра сигнала в манчестерском коде, что оправдывает дополнительный этап логического кодирования, а также работу приемника и передатчика на повышенной скорости.

Скремблирование представляет собой "перемешивание" исходной последовательности данных таким образом, чтобы вероятность появления единиц и нулей на линии становилась близкой 0,5. Устройства (или программные модули), выполняющие такую операцию, называются скремблерами (scramble - свалка, беспорядочная сборка).

Скремблер в передатчике выполняет преобразование структуры исходного цифрового потока. Дескремблер в приемнике восстанавливает исходную последовательность бит. Практически единственной операцией, используемой в скремблерах и дескремблерах, является XOR - "побитное исключающее ИЛИ" (сложение по модулю 2).

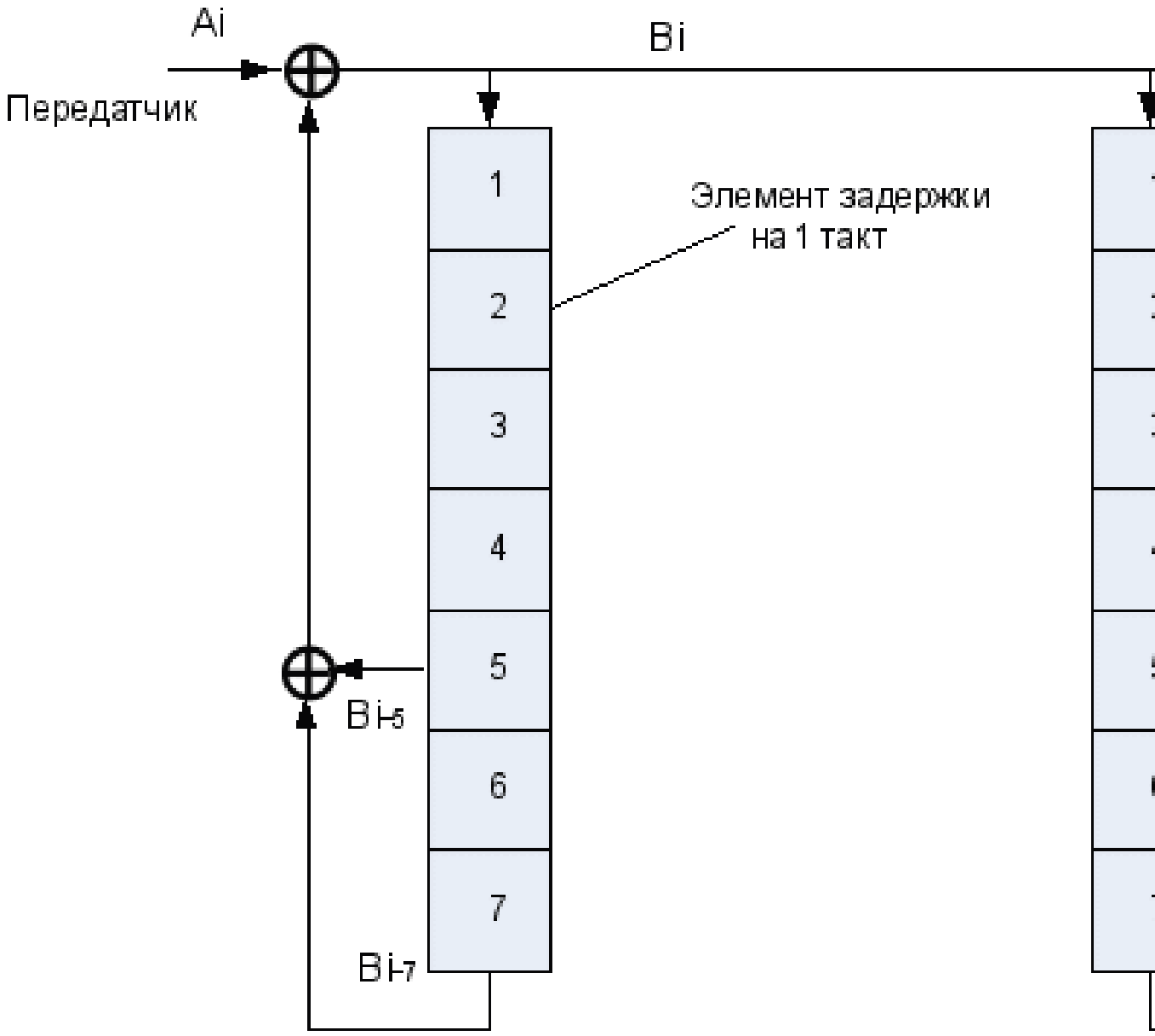


Рис.4.8. Вариант реализации скремблирования

Пусть, например, скремблер реализует соотношение Bi=Ai+Bi-5+Bi-7.

Здесь Bi – бит результирующего кода, полученный на i-м такте работы скремблера; Ai – бит исходного кода, поступающий в передатчике на вход скремблера на i-м такте; Bi-5 и Bi-7 – биты результирующего кода, полученные на предыдущих тактах работы скремблера, соответственно на «i-5» и «i-7» тактах.

Дескремблер в приемнике восстанавливает исходную последовательность, используя соотношение Ci=Bi+Bi-5+Bi-7=(Ai+Bi-5+Bi-7)+Bi-5+Bi-7=Ai

XXIX. Выводы

В сетях высокоскоростной передачи данных используются различные виды логического и физического кодирования данных. Логическое кодирование выполняется передатчиком до физического кодирования средствами физического уровня. На этапе логического кодирования борются с недостатками методов физического цифрового кодирования - отсутствие синхронизации, наличие постоянной составляющей. Сформированные исправленные последовательности данных затем с помощью методов физического кодирования передаются по линиям связи.