Скремблирование

Скремблирование заключается в побитном вычислении результирующего кода на основании битов исходного кода и полученных в предыдущих тактах битов результирующего кода. Например, скрэмблер может реализовывать следующее соотношение:

$$B_i = A_i \, B_{i-3} \, B_{i-5}$$

Здесь B_i — двоичная цифра результирующего кода, полученная на i-м такте работы скрэмблера, A_i — двоичная цифра исходного кода, поступающая на i-м такте на вход скрэмблера, B_{i-3} и B_{i-5} — двоичные цифры результирующего кода, полученные на предыдущих тактах работы скрэмблера (соответственно на 3 и на 5 тактов ранее текущего такта) и объединенные операцией исключающего ИЛИ (сложение по модулю 2).

Например, для исходной последовательности 110110000001 скрэмблер даст следующий результирующий код (первые три цифры результирующего кода будут совпадать с исходным кодом, так как еще нет нужных предыдущих цифр):

$B_1 = A_1 = 1$

$B_2 = A_2 = 1$

$B_3 = A_3 = 0$

$B_4 = A_4 \, B_1 = 1 \; 1 = 0$

$B_5 = A_5 \, B_2 = 1 \; 1 = 0$

$B_6 = A_6 \, B_3 \, B_1 = 0 \; 0 \; 1 = 1$

$B_7 = A_7 \, B_4 \, B_2 = 0 \; 0 \; 1 = 1$

$B_8 = A_8 \, B_5 \, B_3 = 0 \; 0 \; 0 = 0$

$B_9 = A_9 \, B_6 \, B_4 = 0 \; 1 \; 0 = 1$

$B_{10} = A_{10} \, B_7 \, B_5 = 0 \; 1 \; 0 = 1$

$B_{11} = A_{11} \, B_8 \, B_6 = 0 \; 0 \; 1 = 1$

$B_{12} = A_{12} \, B_9 \, B_7 = 1 \; 1 \; 1 = 1$

Таким образом, на выходе скрэмблера появится код 110001101111, в котором нет последовательности из шести нулей, присутствовавшей в исходном коде.

После получения результирующей последовательности приемник передает ее дескрэмблеру, который восстанавливает исходную последовательность на основании обратного соотношения:

$$C_i = B_i \, B_{i-3} \, B_{i-5} = (A_i \, B_{i-3} \, B_{i-5}) \, B_{i-3} \, B_{i-5} = A_i$$

Различные алгоритмы скрэмблирования отличаются количеством слагаемых, дающих цифру результирующего кода, и сдвигом между слагаемыми. Так, в сетях ISDN при передаче данных от сети к абоненту используется преобразование со сдвигами на 5 и 23 позиции, а при передаче данных от абонента в сеть — со сдвигами на 18 и 23 позиции.

Существуют и более простые методы борьбы с последовательностями единиц, также относимые к классу скрэмблирования. Для улучшения биполярного кода AMI используются два метода, основанные на искусственном искажении последовательности нулей *запрещенными символами*.

Рисунок 9.8 иллюстрирует использование методов **B8ZS** (Bipolar with 8-Zeros Substitution) и **HDB3** (High-Density Bipolar 3-Zeros) для корректировки кода AMI. Исходный код состоит из двух длинных последовательностей нулей: в первом случае — из 8, а во втором — из 5.

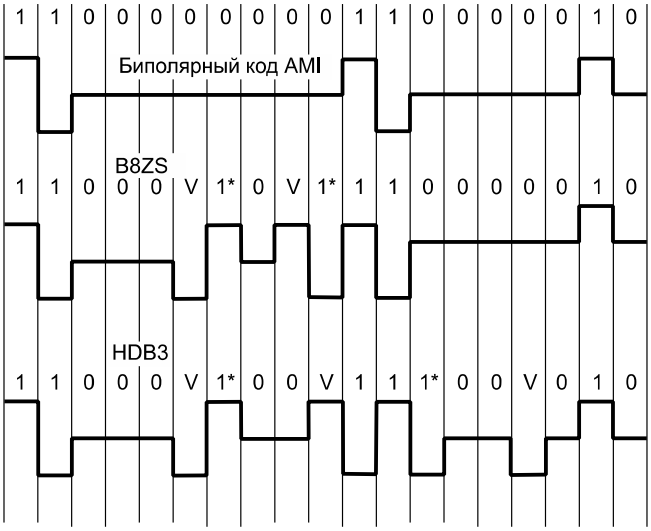

Рис. 9.8. Коды B8ZS и HDB3

Код B8ZS исправляет только последовательности, состоящие из 8 нулей. Для этого он после первых трех нулей вместо оставшихся пяти нулей вставляет пять цифр: *V*-1*-0-*V*-1*. Здесь *V* обозначает сигнал единицы, запрещенной (*V*iolations) для данного такта полярности, то есть сигнал, не изменяющий полярность предыдущей единицы, 1* — сигнал единицы корректной полярности (знак звездочки отмечает тот факт, что в исходном коде в этом такте была не единица, а ноль). В результате на 8 тактах приемник наблюдает 2 искажения — очень маловероятно, что это случается из-за шума на линии или других сбоев передачи. Поэтому приемник считает такие нарушения кодировкой 8 последовательных нулей и после приема заменяет их исходными 8 нулями. Код B8ZS построен так, что его постоянная составляющая равна нулю при любых последовательностях двоичных цифр.

Код HDB3 исправляет любые четыре смежных нуля в исходной последовательности. Правила формирования кода HDB3 более сложные, чем кода B8ZS. Каждые четыре нуля заменяются четырьмя сигналами, в которых имеется один сигнал *V*. Для подавления постоянной составляющей полярность сигнала *V* чередуется при последовательных заменах. Кроме того, для замены используются два образца четырехтактовых кодов. Если перед заменой исходный код содержал нечетное число единиц, задействуется последовательность 000*V*, а если число единиц было четным — последовательность 1*00*V*.

Улучшенные потенциальные коды обладают достаточно узкой полосой пропускания для любых последовательностей единиц и нулей, которые встречаются в передаваемых данных. На рис. 9.9 приведены спектры сигналов разных кодов, полученные при передаче произвольных данных, в которых различные сочетания нулей и единиц в исходном коде равновероятны. При построении графиков спектр усреднялся по всем возможным наборам исходных последовательностей. Естественно, что результирующие коды могут иметь и другое распределение нулей и единиц. Из рисунка видно, что потенциальный код NRZ обладает хорошим спектром с одним недостатком — у него имеется постоянная составляющая. Коды, полученные из потенциального кода путем логического кодирования, обладают более узким спектром, чем манчестерский код, даже при повышенной тактовой частоте (на рисунке спектр кода 4B/5B должен был бы примерно совпадать с кодом B8ZS, но он сдвинут в область более высоких частот, так как его тактовая частота повышена на 1/4 по сравнению с другими кодами). Этим объясняется преимущественное применение в современных технологиях, подобных FDDI, Fast Ethernet, Gigabit Ethernet, ISDN и т. п., потенциальных избыточных и скрэмблированных кодов вместо манчестерского и биполярного импульсного кода.

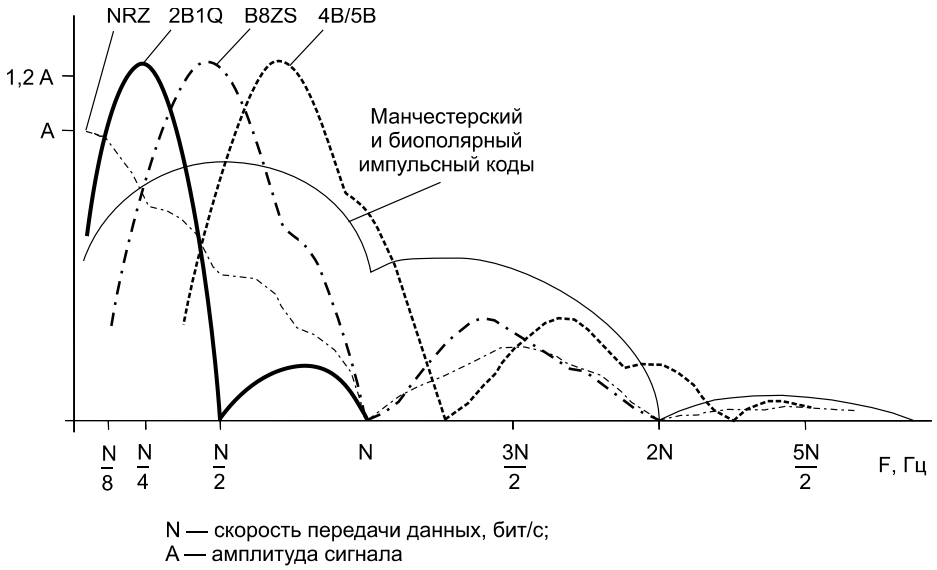

Рис. 9.9. Спектры потенциальных и импульсных кодов

Компрессия данных

Компрессия, или **сжатие**, данных применяется для сокращения времени их передачи. Так как на компрессию данных передающая сторона тратит дополнительное время, к которому нужно еще прибавить аналогичные затраты времени на декомпрессию этих данных принимающей стороной, то выгоды от сокращения времени на передачу сжатых данных обычно бывают заметны только на низкоскоростных каналах. Соответствующий порог скорости для современной аппаратуры составляет около 64 Кбит/с. Многие программные и аппаратные средства сети способны выполнять *динамическую компрессию* данных в отличие от *статической*, когда данные сначала сжимаются (например, с помощью популярных архиваторов типа WinZip), а уже затем отсылаются в сеть.

На практике может использоваться ряд алгоритмов компрессии, каждый из которых применим к определенному типу данных. Некоторые модемы (называемые интеллектуальными) предлагают **адаптивную компрессию**, при которой в зависимости от передаваемых данных выбирается определенный алгоритм компрессии. Рассмотрим некоторые из общих алгоритмов компрессии данных.

Когда данные состоят только из чисел, значительную экономию можно получить путем уменьшения количества используемых на цифру битов с 7 до 4, просто заменяя десятичные цифры кода ASCII двоичными. Просмотр таблицы кодов ASCII показывает, что старшие три бита всех кодов десятичных цифр содержат комбинацию 011. Если все данные в кадре информации состоят из десятичных цифр, то поместив в заголовок кадра соответствующий управляющий символ, можно существенно сократить длину кадра. Этот метод носит название **десятичной упаковки**.

Альтернативой десятичной упаковке при передаче числовых данных с небольшими отклонениями между последовательными цифрами является передача только этих отклонений вместе с известным опорным значением. Такой метод называется **относительным кодированием** и используется, в частности, при цифровом кодировании голоса с помощью кода ADPCM, когда в каждом такте передается только разница между соседними замерами голоса.

Часто передаваемые данные содержат большое количество повторяющихся байтов. Например, при передаче черно-белого изображения черные поверхности будут порождать большое количество нулевых значений, а максимально освещенные участки изображения — большое количество байтов, состоящих из всех единиц. Передатчик сканирует последовательность передаваемых байтов, и если обнаруживает последовательность из трех или более одинаковых байтов, заменяет ее специальной трехбайтовой последовательностью, в которой указывает значение байта, количество его повторений, а также отмечает начало этой последовательности специальным управляющим символом. Этот метод носит название **символьного подавления**.

Метод кодирования с помощью **кодов переменной длины** опирается тот факт, что не все

символы в передаваемом кадре встречаются с одинаковой частотой. Поэтому во многих схемах кодирования коды часто встречающихся символов заменяют кодами меньшей длины, а редко встречающихся — кодами большей длины. Такое кодирование называется также **статистическим кодированием**. Из-за того что символы имеют разную длину, для передачи кадра возможна только бит-ориентированная передача. При статистическом кодировании коды выбираются таким образом, чтобы при анализе последовательности битов можно было бы однозначно определить соответствие определенной порции битов тому или иному символу или же запрещенной комбинации битов. Если данная последовательность битов представляет собой запрещенную комбинацию, то необходимо к ней добавить еще один бит и повторить анализ. Например, если при неравномерном кодировании для наиболее часто встречающегося символа «Р» выбран код 1, состоящий из одного бита, то значение 0 однобитного кода будет запрещенным. Иначе мы сможем закодировать только два символа. Для другого часто встречающегося символа «О» можно использовать код 01, а код 00 оставить как запрещенный. Тогда для символа «А» можно выбрать код 001, для символа «П» — код 0001 и т. п.

Неравномерное кодирование наиболее эффективно, когда неравномерность распределения частот передаваемых символов велика, как при передаче длинных текстовых строк. Напротив, при передаче двоичных данных, например кодов программ, оно малоэффективно, так как 8-битные коды при этом распределены почти равномерно.

Одним из наиболее распространенных алгоритмов, на основе которых строятся неравномерные коды, является **алгоритм Хафмана**, позволяющий строить коды автоматически на основании известных частот появления символов. Существуют адаптивные модификации метода Хафмана, которые позволяют строить дерево кодов «на ходу», по мере поступления данных от источника.

Многие модели коммуникационного оборудования, такие как модемы, мосты, коммутаторы и маршрутизаторы, поддерживают протоколы динамической компрессии, позволяющие сократить объем передаваемой информации в 4, а иногда и в 8 раз. В таких случаях говорят, что протокол обеспечивает коэффициент сжатия 1:4 или 1:8. Существуют стандартные протоколы компрессии, например V.42bis, а также большое количество нестандартных фирменных протоколов. Реальный коэффициент компрессии зависит от типа передаваемых данных. Так, графические и текстовые данные обычно сжимаются хорошо, а коды программ — хуже.