

## Лекция 3. Контроль достоверности передачи

(Механизм контрольных сумм  
Коды ЕСС. Код Хемминга)

# Механизм контрольных сумм

**Контрольная сумма** — некоторое значение, рассчитанное по набору данных путём применения определённого алгоритма и используемое для проверки целостности данных при их передаче или хранении. Под контрольную сумму может быть выделена некоторая совокупность  $n$  бит (один бит, полубайт, байт, несколько байт).

Контрольной суммой некоторого массива будет являться величина, полученная путем деления с остатком суммы всех элементов массива на максимально возможное числовое значение контрольной суммы, увеличенное на единицу ( $2^n$ ). Функция, реализующая алгоритм и выполняющая преобразование, называется «хеш-функцией» или «функцией свёртки». Значение контрольной суммы добавляется в конец блока данных непосредственно перед началом передачи. Впоследствии оно проверяется для подтверждения целостности данных - приемное устройство, используя тот же самый алгоритм, рассчитывает контрольную сумму принятого сообщения и сравнивает ее с переданным значением. Вероятность ошибки равна  $1/2^n$ .

Сообщение : 6 23 4. Сообщение с контрольной суммой : 6 23 4 33

(используем простое сложение байтов сообщения по модулю 256)

Сообщение после передачи : 8 23 4 33 ( $8+23+4=35 \neq 33$  - Ошибка передачи)

# Механизм четности

*Механизм четности (parity) – Частный случай механизма контрольных сумм ( $n=1$ ).*

Для передаваемых данных формируется специальный сигнал – **признак нечетного количества единиц на линиях**. В случае обнаружения нарушения четности в фазе данных сигнал ошибки вырабатывает приемник, для фазы адреса проверку четности выполняет целевое устройство, выставляется соответствующий бит в регистре состояния.

Сообщение + **Бит четности**

01110110**1**

00010100**0**

Сообщение после передачи 01010110**1** - **Ошибка передачи**

00010100**0** - **Ошибки нет**

# Недостаток механизма контрольных сумм

Этот метод ограничен определением изменения состояния одного байта или одиночного бита в байте (в parity).

1) Например, если для расчета контрольной суммы используем простое сложение байтов сообщения по модулю 256, то может возникнуть примерно следующая ситуация.

Сообщение : 6 23 4    Сообщение с контрольной суммой : 6 23 4 **33**

Сообщение после передачи : 128 157 4 **33**.    **Ошибка не определяется.**

2) Например, в случае изменения состояния двух битов (**parity**) , возможна ситуация, когда вычисление паритетного бита совпадет с записанным. В этом случае система также не определит ошибку.

Важная отличительная особенность хороших алгоритмов хэширования заключается в том, что генерируемые с его помощью значения настолько уникальны и трудноповторимы, что задача нахождения коллизий, т.е. ситуаций когда разным наборам входных блоков данных соответствует один хэш.

Не решается вообще либо является чрезвычайно тяжелой как по ресурсоемкости так и по производительности (практически не решается за приемлемое время)

# Требования для формирования надежной контрольной суммы

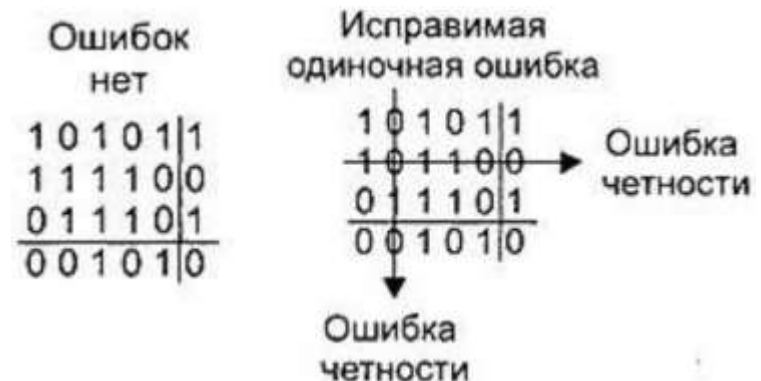
Недостаток этих способов контроля достоверности в том, что алгоритмы суммирования в них слишком просты. Для формирования надежной контрольной суммы следует

- увеличивать количество бит под контрольную сумму (изменение размера регистра с 8 битного на 16 битный, то есть суммировать по модулю 65536 вместо модуля 256, что скорее всего снизит вероятность ошибки с  $1/256$  до  $1/65536$ ) (ширина)

Сообщение : 6 23 4    Сообщение с контрольной суммой : 6 23 4 **33**

Сообщение после передачи : 128 157 4 **289**.    **Ошибка определяется.**

- необходим такой алгоритм расчета, когда каждый новый байт может оказать влияние на любые биты регистра (случайность)
- выполнять проверку по нескольким битам. В этом случае определяется достоверность передачи целого **блока** символов.



# Циклический избыточный код

Если сложение, очевидно, не пригодно для формирования эффективной контрольной суммы, то таким действием вполне может оказаться **деление** при условии, что делитель имеет ширину регистра контрольной суммы.

(*cyclic redundancy check* - CRC). Основная идея алгоритма CRC состоит в представлении сообщения в виде одного двоичного числа, делении его на другое фиксированное двоичное число и использовании остатка этого деления в качестве контрольной суммы. Получив сообщение, приемник может выполнить аналогичное действие и сравнить полученный остаток с "контрольной суммой" (переданным остатком).

Так же, как и для контрольных сумм, контрольный код не занимает много места (обычно 16/32 бита), однако вероятность обнаружения ошибки существенно выше. Например, в отличие от контрольных сумм метод CRC сможет **обнаружить перестановку двух байт** либо **добавление единицы к одному и вычитание единицы из другого**.

# Циклический избыточный код

CRC (Cyclic Redundancy Check - циклический избыточный контроль) является разновидностью так называемых хеш-функций. Хеш-функция является, по сути, формирует остаток от деления *двоичного многочлена (полинома)*  $A(x)$ , соответствующего информационным данным, на некий фиксированный *порождающий многочлен*  $G(x)$ .

Каждый бит некоторого блока данных соответствует одному из коэффициентов двоичного полинома. Например, полином двоичного числа 10110110 будет выглядеть следующим образом:

$$A(x) = 1 \cdot x^7 + 0 \cdot x^6 + 1 \cdot x^5 + 1 \cdot x^4 + 0 \cdot x^3 + 1 \cdot x^2 + 1 \cdot x^1 + 0 \cdot x^0 = x^7 + x^5 + x^4 + x^2 + x.$$

Для каждой реализации алгоритма контроля CRC порождающий полином выбирается заранее. Например, полином 11010, для него  $G(x) = 1 \cdot x^4 + 1 \cdot x^3 + 0 \cdot x^2 + 1 \cdot x^1 + 0 \cdot x^0 = x^4 + x^3 + x$ .

**Теорема, на которой основано деление.** Для любых двух многочленов  $Pn(x)$  (делимое) и  $Gm(x)$  (делитель) можно найти такие многочлены  $Qp(x)$  (частное) и  $Rk(x)$  (остаток), что будет выполнено равенство  $Pn(x) = Gm(x) \cdot Qp(x) + Rk(x)$ ,

$n, m, p, k$  – степени многочленов, причём  $k < m$ .

все наборы, образованные циклической перестановкой кодовой комбинации, также являются кодовыми комбинациями. Так, например, циклические перестановки комбинации 1000101 будут также кодовыми комбинациями 0001011, 0010110, 0101100 и т.д. Это свойство позволяет в значительной степени упростить кодирующее и декодирующее устройства, особенно при обнаружении ошибок и исправлении одиночной ошибки. Внимание к циклическим кодам обусловлено тем, что присущие им высокие корректирующие свойства реализуют на основе сравнительно простых алгебраических методов.

Циклическим кодом называется линейный блочный  $(n, k)$ -код, который характеризуется свойством цикличности, т.е. сдвиг влево на один шаг любого разрешенного кодового слова дает также разрешенное кодовое слово, принадлежащее этому же коду, и у которого множество кодовых слов представляется совокупностью многочленов степени  $(n - 1)$  и менее, делящихся на порождающий многочлен  $g(x)$  степени  $r = n - k$ , являющийся сомножителем двучлена  $x^{n+1}$



# Пример ы кодов

Если  $n = 10$  и  $X = 2$ , то  $X^n - 1 = 1023 = 11 \cdot 93$ , и если  $g(X) = 11$  или в двоичном коде 1011, то примеры циклических кодов  $A_i g(X)$  чисел  $A_i$  в кодовой группе при этом образующем полиноме можно видеть в следующей таблице:

Число	Циклический код	Число	Циклический код
0	0000000000	13	0010001111
1	0000001011	14	0010011010
2	0000010110	15	0010100101
3	0000100001	16	0011000110
5	0000110111	18	0011000110

Упражнение :: найти нод числа 5, код числа 17 :

# Деление многочленов «СТОЛБИКОМ»

$$\begin{array}{r|l}
 2x^6 - x^5 + 0 \cdot x^4 + 12x^3 - 72x^2 + 0 \cdot x + 3 & x^3 + 2x^2 - 1 \\
 \hline
 2x^6 + 4x^5 + 0 \cdot x^4 - 2x^3 & \\
 \hline
 -5x^5 + 0 \cdot x^4 + 14x^3 - 72x^2 + 0 \cdot x + 3 & \\
 - & \\
 -5x^5 + 10x^4 + 0 \cdot x^3 + 5x^2 & \\
 \hline
 -10x^4 + 14x^3 - 77x^2 + 0 \cdot x + 3 & \\
 - & \\
 -10x^4 + 20x^3 + 0 \cdot x^2 - 10x & \\
 \hline
 -6x^3 - 77x^2 + 10x + 3 & \\
 - & \\
 -6x^3 - 12x^2 + 0 \cdot x + 6 & \\
 \hline
 -65x^2 + 10x - 3 &
 \end{array}$$

**Упражнение 1 :** Разделить  $A(x) = x^7 + x^5 + x^4 + x^2 + x$  на  $G(x) = x^4 + x^3 + x$ .  $A(x)$  и  $G(x)$  – целочисленные многочлены. Ответ – частное:  $x^3 - x^2 + 2x - 2$ , остаток:  $3x^3 - x^2 + 3x$

# Полиномиальная арифметика

В CRC алгоритме вместо представления делителя, делимого (сообщения), частного и остатка в виде **положительных целых чисел** (как ранее при рассмотрении механизма контрольных сумм), используются **полиномы с двоичными коэффициентами** или **строки бит**, каждый из которых является коэффициентом полинома.

Выполняя различные арифметические действия, условились использовать так называемую полиномиальную арифметику - двоичную арифметику по модулю 2 без учета переносов.

$$0+0=0 \quad 0+1=1 \quad 1+0=1 \quad 1+1=0$$

$$0-0=0 \quad 0-1=1 \quad 1-0=1 \quad 1-1=0$$

*Упражнение 1:* Разделить  $A(x) = x^7 + x^5 + x^4 + x^2 + x$  на  $G(x) = x^4 + x^3 + x$ .  $A(x)$  и  $G(x)$  – **двоичные многочлены**.

*Упражнение 2 :* Разделить  $A(x) = x^7 + x^5 + x^4 + x^2 + x$  на  $G(x) = x^4 + x^3 + x$ , используя операцию "Исключающее ИЛИ»

*Упражнение 3:* Разделить 10110110 на 11010.

# Алгоритм CRC

Полином  $R(x)$  называется контрольным кодом полинома  $A(x)$  при порождающем полиноме  $G(x)$ , если  $R(x)$  является остатком от деления полинома  $A(x) \cdot x^r$  на  $G(x)$ , где  $r$  – степень полинома  $G(x)$ :

$$R(x) = (A(x) \cdot x^r) \bmod G(x)$$

## Алгоритм

1. Выбрать полином  $G$  (степени  $r$ ).
2. Добавить к сообщению  $r$  нулевых битов. Назовем полученную строку  $M'$ .
3. Поделим  $M'$  на  $G$  с использованием правил CRC арифметики. Полученный остаток и будет контрольной суммой.

Контрольная сумма добавляется к сообщению и вместе с ним передается по каналам связи.

На другом конце канала приемник должен вычислить контрольную сумму для всего переданного сообщения (без добавления нулей), и посмотреть, получится ли в результате нулевой остаток.

# Выбор полинома

16 битные: (16,12,5,0) [стандарт "X25"]

(16,15,2,0) ["CRC 16"]

32 битные: (32,26,23,22,16,12,11,10,8,7,5,4,2,1,0) [Ethernet]

используется полином 32-й степени  $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ , двоичное представление которого **100000100110000010001110110110111** (всего 33 бита)

*Упражнение 4:* Применить CRC алгоритм к 01110000110, делитель 10110.

*Упражнение 5:* Найти остаток от деления 01110000110 0110 на 10110.

*Упражнение 6:* Найти остаток от деления 01110000110 0100 на 10110.

# Передача блоками с количественной характеристикой

*Prarity, CRC* - это передача блоками с количественной характеристикой (предполагает передачу данных блоками с количественной характеристикой блока:

- число единиц или нулей в блоке,
- контрольная сумма передаваемых символов в блоке,
- остаток от деления контрольной суммы на постоянную величину и др.

На приемном пункте эта характеристика вновь подсчитывается и сравнивается с переданной по каналу связи. Если характеристики совпадают, считается, что блок не содержит ошибок. В противном случае на передающую сторону поступает сигнал с требованием повторной передачи блока. В современных телекоммуникационных вычислительных сетях такой метод получил самое широкое распространение.

# Повышение достоверности передачи

Для повышения достоверности и качества передачи применяются:

- групповые методы защиты от ошибок,
- избыточное кодирование ,
- системы с обратной связью.

Из групповых методов получили широкое применение:

- мажоритарный метод, реализующий принцип Вердана,
- метод передач информационными блоками с количественной характеристикой блока.

Суть *мажоритарного метода* состоит в том, что каждое сообщение ограниченной длины передается несколько раз (чаще всего три раза), принимаемые сообщения запоминаются, а потом производится их поразрядное сравнение. Суждение о правильности передачи выносится по совпадению большинства из принятой информации методом «два из трех».

Например, кодовая комбинация 01101 при трехразовой передаче была частично искажена помехами, поэтому приемник принял такие комбинации: 10101, 01110, 01001.

В результате проверки по отдельности каждой позиции правильной считается комбинация 01101.

# Способы определения ошибок

Существует три наиболее распространенных способа борьбы с ошибками в процессе передачи данных:

- коды обнаружения ошибок (ECC);
- **FEC** (Forward Error Correction - *прямая коррекция ошибок*) - заблаговременное исправление ошибок, коррекция ошибок путем передачи избыточной информации, без требования повторной передачи.
- протоколы с автоматическим запросом повторной передачи (Automatic Repeat Request - ARQ).

Рассмотрим основные механизмы обеспечения целостности передаваемой информации с помощью введения избыточности.



# Показатели корректирующих кодов

*Помехоустойчивое (избыточное) кодирование* предполагает разработку и использование корректирующих (помехоустойчивых) кодов. Его основное назначение заключается в обеспечении малой вероятности искажений передаваемой информации, несмотря на присутствие помех или сбоев в работе сети.

К числу наиболее важных показателей корректирующих кодов относятся:

- **значность кода  $n$** , или **длина кодовой комбинации**, включающей информационные символы ( $m$ ) и проверочные, или контрольные, символы ( $k$ ):  $n = m + k$  (значения контрольных символов при кодировании определяются путем контроля на четность количества единиц в информационных разрядах кодовой комбинации (0, если количество единиц будет четным, 1 - при нечетном количестве));
- **избыточность кода ( $K_{изб}$ )**, выражаемая отношением числа контрольных символов в кодовой комбинации к значности кода:  $K_{изб} = k / n$ ;
- **корректирующая способность кода ( $K_{кс}$ )** это отношение числа кодовых комбинаций  $L$ , в которых ошибки были обнаружены и исправлены, к общему числу переданных кодовых комбинаций  $M$  в фиксированном объеме информации:  $K_{кс} = L / M$ .

# Выбор корректирующего кода

Выбор корректирующего кода для его использования в данной компьютерной сети зависит от требований по достоверности передачи информации. Для правильного выбора кода необходимы статистические данные о

закономерностях появления ошибок,

их характере,

численности и

распределении во времени.

Например, корректирующий код, обнаруживающий и исправляющий одиночные ошибки, эффективен лишь при условии, что ошибки статистически независимы, а вероятность их появления не превышает некоторой величины. Он оказывается непригодным, если ошибки появляются группами. При выборе кода надо стремиться, чтобы он имел меньшую избыточность. Чем больше коэффициент  $K_{\text{изб}}$ , тем менее эффективно используется пропускная способность канала связи и больше затрачивается времени на передачу информации, но зато выше помехоустойчивость системы.

# Коррекция ошибок (код исправления ошибок)

*(Error Correcting Code – ECC)*

Этот метод включает определение ошибки не только в одиночном разряде, но и двух, трех и четырех разрядах.

Кроме того, ECC может также исправлять ошибку в одиночном разряде.

Во время считывания результат ECC-слова сравнивается с рассчитанным, подобно тому, как происходит в описанных выше методах проверки контрольных сумм. Основное различие состоит в том, что в проверке четности каждый бит связан с одним байтом, в то время как **ЕСС-слово связано с совокупностью байт.**

Если четность корректна, то для всех групп, то это свидетельствует об отсутствии ошибок.

Если одно или более значение четности для переданного блока не верно, генерируется уникальный код, называемый **синдромом**, по которому можно идентифицировать переданный с ошибкой бит.

# Основные понятия корректирующих кодов

Введем основные понятия из области корректирующих кодов.

**Кодовый вес** - Количество единиц в двоичной кодовой комбинации называется ее кодовым весом  $W$ . Например, если  $X = 1,1010_2$ , то  $W = 3$ .

**Расстояние** - Количество разрядов  $d$ , в которых не совпадают двоичные цифры в двух кодовых комбинациях, называется расстоянием между этими комбинациями. Оно обычно определяется методом поразрядного сложения по модулю два (исключающее ИЛИ) с последующим вычислением веса суммы:

$$X1 = 1010, \quad X2 = 0011 \quad C = X1 \text{ XOR } X2 = 1001 \quad W = 2 \quad d = 2$$

**Минимальным кодовым расстоянием**  $d_{min}$  называется самое малое кодовое расстояние, возможное между двумя любыми кодовыми комбинациями из рассматриваемых множеств. В обычном двоичном коде  $n$ -разрядных чисел возможны кодовые расстояния от 1 до  $n$ , т. е. здесь  $d_{min} = 1$ .

# Обнаружение кратных ошибок

В общем случае, когда в кодовой комбинации могут появляться ошибки любой кратности  $i \leq n$  (одиночная, двойная, тройная и т. д.), для обнаружения некоторой ошибки требуется корректирующий код с минимальным расстоянием  $d_{min} = i + 1$ .

Следовательно, обнаружение одиночной ошибки можно обеспечить ценой минимальной избыточности – добавлением к слову всего одного контрольного разряда (контроль четности). Возникновение одиночной ошибки (или же ошибки с нечетным изменением количества разрядов) приводит к нарушению нечетности веса всей комбинации.

Таким образом, избыточность кодовой комбинации на один разряд позволяет обнаруживать все нечетные групповые ошибки и, как частный случай, одиночную ошибку.

# Исправление одиночной ошибки

Идея (двоичного) кода с исправлением одиночной ошибки состоит в следующем. Пусть исходный (незащищенный) код имеет  $m$  двоичных разрядов. Выберем  $n > m$  так, чтобы


$$\frac{2^n}{n+1} \geq 2^m$$

Полученный защищенный код будет иметь  $k = n - m$  контрольных разрядов, причем

$$2^k \geq m + k + 1 = n + 1$$

Каждому из  $n$  разрядов присваивается номер от 1 до  $n$ .

Далее для любого значения защищенного ( $n$ -разрядного) кода составляется  $k$  контрольных сумм  $S_1, S_2, \dots, S_k$ , по модулю 2 значений специально выбранных разрядов этого кода. Для  $S_i$  выбираются разряды, для которых двоичные коды номеров имеют в  $i$ -м разряде единицу. Для суммы  $S_1$  это будут, очевидно, разряды с номерами 1, 3, 5, 7, ..., для суммы  $S_2$  – разряды с номерами 2, 3, 6, 7, 10, 11, ... и т. д.



При любом исходном коде контрольные разряды могут быть выбраны так, чтобы все контрольные суммы были равны нулю. Проверочный (двоичный) код  $S = S_k \dots S_1$  при этом будет нулевым. При таком условии защищенный код называется *правильным*.

При возникновении одиночной ошибки в этом коде в любом (безразлично, основном или контрольном) разряде проверочный код будет отличен от нуля. При этом в силу способа выбора контрольных сумм значение проверочного кода  $S$  будет представлять собой двоичный код номера разряда защищенного кода, в котором возникла ошибка. Для исправления этой ошибки достаточно изменить значение указанного разряда на противоположное.

Примером корректирующих кодов являются код Хемминга и код Рида-Соломона.

# Код Хемминга

Код, в котором в качестве контрольных берут разряды  $1, 2, 4, \dots, 2^n$ , называется *кодом Хемминга*.

Код Хемминга может либо исправлять одиночные ошибки (при гипотезе, что ошибки более высокой кратности невозможны), либо обнаруживать любые ошибки, не кратные трем.

Тройные ошибки считаются маловероятными, и поэтому иногда говорят, что *код Хэмминга позволяет обнаруживать одиночные и двойные ошибки*.

Покажем на примере, как ликвидируется одиночная ошибка с помощью кода Хемминга при передаче четырехразрядного двоичного слова

$$X = 0110 \ (m = 4).$$



Построение кода выполняется в три этапа:

- 1 Вычислим необходимый размер ( $n$ ) разрядной сетки защищенного слова и количество контрольных разрядов ( $k$ ).  
Минимальное значение  $n$ , при котором выполняется неравенство

$$\frac{2^n}{n+1} \geq 2^m = 2^8$$

равно 7,  $k = 7 - 4 = 3$ .

- 2 Определим номера контрольных разрядов в защищенном слове:

$$2^0 = 1, 2^1 = 2, 2^2 = 4.$$

- 3 Построим защищенное слово в 7-разрядной сетке. Для этого запишем исходное слово 0110 в порядке следования его разрядов, пропуская места контрольных разрядов:

1	2	3	4	5	6	7
001	010	011	100	101	110	100
		0		1	1	0

# Построение кода Хемминга

Определим значения защищенных разрядов исходя из равенства сумм  $S_i$  нулю. В сумму  $S_1$  должны войти значения разрядов, расположенных на нечетных местах  $(1, 3, 5, 7) = (001, 011, 101, 111)$ , Чтобы сумма  $S_1$  была равна нулю, входящий в нее 1-й контрольный разряд должен быть равен 1,

1	2	3	4	5	6	7
001	010	011	100	101	110	100
		0		1	1	0

Аналогично отыскиваются и другие контрольные разряды.

В  $S_2$  войдут разряды  $(2, 3, 6, 7) = (010, 011, 110, 111)$ ,

В  $S_3$  войдут разряды  $(4, 5, 6, 7) = (100, 101, 110, 111)$ ,

По результатам вычисления контрольных разрядов и значениям разрядов передаваемого слова строим защищенное слово:

1	1	0	0	1	1	0
---	---	---	---	---	---	---

# Исправление ошибки

В месте приема переданного слова снова вычисляются контрольные суммы  $S_1$ ,  $S_2$ ,  $S_3$ , но уже при известных значениях заштрихованных контрольных разрядов. При равенстве всех сумм нулю считается, что передача произошла правильно.

Предположим, что в процессе передачи произошло искажение только одного разряда (не важно какого: контрольного или основного). Например, предположим, что неправильно передан 6-й разряд. Это означает, что вместо 1 в 6-й разряде появился 0.

1	1	0	0	1	0	0
---	---	---	---	---	---	---

Тогда после вычисления контрольных сумм получим:

$$S_1 = 0, S_2 = 1, S_3 = 1.$$

Располагаем эти суммы в порядке убывания их номеров и получаем

$$S = S_1 S_2 S_3 = 011. \text{ Это двоичное число соответствует номеру 6.}$$

Тогда можно автоматически исправить ошибку в 6-м разряде, заменив пришедший 0 на 1. После исправления ошибки контрольные биты отбрасываются.

# Возможности исправления ошибок

В коде Хемминга возможности исправления ошибок связаны с минимальным кодовым расстоянием  $d_{min}$ . Исправляются ошибки кратности  $r = \lfloor (d_{min}-1)/2 \rfloor$  и обнаруживаются ошибки кратности  $d_{min}-1$  (здесь  $\lfloor \rfloor$  означает “целая часть”).

Так, при контроле на нечетность  $d_{min} = 2$  и обнаруживаются одиночные ошибки. В коде Хемминга  $d_{min} = 3$ . Дополнительно к информационным разрядам вводится  $L = \log_2 K$  избыточных контролирующих разрядов, где  $K$  - число информационных разрядов,  $L$  округляется до ближайшего большего целого значения.  $L$ -разрядный контролирующий код есть инвертированный результат поразрядного сложения (т.е. сложения по модулю 2) номеров тех информационных разрядов, значения которых равны 1.

Определяется четность или нечетность целого блока символов. Каждый бит четности вычисляется для определенной комбинации бит данных.

# Примеры коррекции ошибок

Пример 1. Пусть имеем основной код 100110, т.е.  $m = 6$ . Следовательно,

$N=10$  ( $k=4$ )

$K=4$   $n = 10$

и дополнительный код равен 1011

Пример 2 Пусть имеем основной код 100110, т.е.  $m = 6$ .

$k = \log_2 m$

$k$ -разрядный контролирующий код есть инвертированный результат поразрядного сложения (сложения по модулю 2) номеров тех информационных разрядов, значения которых равны единице.

$010 \# 011 \# 110 = 111$ ,

Передается 100110 000.

Принят 100010 (ошибка в 3-м разряде). КАК Определяется ОШИБКА?\

# Заключение

**Выполнить упражнения**

**(эти и такого типа будут на рейтинговом  
тестировании)**