ГРАФИЧЕСКИЙ СПОСОБ ПРЕДСТАВЛЕНИЯ СВЕРТОЧНЫХ КОДОВ

Дяченко О.Н.

Донецкий национальный технический университет
do@cs.dgtu.donetsk.ua

Abstract

Dyachenko O.N. Graphical method for convolutional codes representation. *Graphical method for convolutional codes description is proposed giving pictorial presentation of construction principles for these codes, allowing to construct codes without determination of generator polynomial and to perform comparative analysis with orchard codes.*

Введение

На современном этапе развития вычислительной техники все большую актуальность приобретают средства, обеспечивающие достоверность передачи, хранения и обработки информации. Основными способами обеспечения помехоустойчивости информационных систем является применение кодов, обнаруживающих и исправляющих ошибки, возникающих при работе системы и ее элементов.

Сверточные коды – один из самых популярных классов кодов, исправляющих ошибки. Эти коды нашли широкое применение на практике: радиосвязь (IMT-2000, GSM, IS-95), цифровая наземная и спутниковая связь, радиовещательные системы и т.д. [1]. В данной работе предлагается альтернативный (графический) способ представления кодов Ивадари – сверточных кодов, исправляющих пакеты ошибок.

1. Коды Ивадари

Сверточные коды принадлежат к классу древовидных. В отличие от блоковых кодов, кодовое слово древовидного кода n_0 (кадр кодового слова) формируется в зависимости не только от самого информационного блока k_0 (информационного кадра), но также от m уже переданных информационных кадров. Величина $k=(m+1)k_0$ называется информационной длиной слова. Кодовая длина блока $n=(m+1)n_0$ – это длина кодового слова, на которой сохраняется влияние одного кадра информационных символов [2].

Пусть λ и n_0 – любые положительные числа. По определению кодом Ивадари называется исправляющий пакеты ошибок двоичный

систематический сверточный код со следующей порождающей $((n_0-1)xn_0)$-матрицей из полиномов:

$$\mathbf{G}(x)=\begin{bmatrix} 1 & & & g_1(x) \\ & 1 & & g_2(x) \\ & & \ddots & \vdots \\ & & 1 & g_{(n_0-1)}(x) \end{bmatrix}$$

где для матричных элементов $g_{in_0}(x)$ использовалось сокращенное обозначение $g_i(x)$ и $g_i(x)=x^{(\lambda+1)(2n_0-i)+i-3}+x^{(\lambda+1)(n_0-i)-1}$, i = 1, … , $n_0 - 1$.

Наибольшую степень, равную $(\lambda+1)(2n_0-1)-2$ имеет полином $g_1(x)$. Таким образом, коды Ивадари являются сверточными $((m+1)n_0,(m+1)(n_0-1))$ кодами, исправляющими пакеты ошибок длины не более λn_0, с числом кадров

$m=(\lambda+1)(2n_0-1)-2$.

Поскольку $n_0-k_0=1$, для кодов Ивадари получаем только один синдромный полином

$$s(x)=\sum_{i=0}^{n_0-1} g_i(x)e_i(x)+e_{n_0}(x)=$$

$$=e_{n_0}(x)+\sum_{i=0}^{n_0-1}\left[x^{(\lambda+1)(n_0-i)-1}+x^{(\lambda+1)(2n_0-i)+i-3}\right]e_i(x).$$

Построение кодирующих устройств для кодов Ивадари, как правило, не вызывает затруднений. Декодирование лучше всего пояснить на примере, поскольку структура декодеров этих кодов одинакова.

Пример. Пусть $n_0=3,\ \lambda=3$.

В этом случае $m+1=19$, получаем (57, 38)-код Ивадари, исправляющий пакеты ошибок длины 9.

Кодер кода реализуется на основе регистров сдвига, каждый из которых выполняет операцию умножения на соответствующий порождающий полином: $g_1(x)=x^{18}+x^7$ и $g_2(x)=x^{15}+x^3$.

Для построения схемы декодера необходимо вначале выполнить анализ возможных синдромов для (57, 38)-кода Ивадари.

Если обозначить через $c_3(x)$ полином проверочных символов, через $c_1(x)$ и $c_2(x)$ полиномы информационных символов, тогда соответствующими полиномами ошибок будут полиномы $e_1(x)$, $e_2(x)$ и $e_3(x)$, а синдромный полином равен

$$s(x)=e_3(x)+(x^3+x^{15})e_2(x)+(x^7+x^{18})e_1(x).$$

Для пакета ошибок, начинающегося в первом кадре,

$$e_3(x)=e_{30}+e_{31}x+e_{32}x^2,$$

$$e_2(x)=e_{20}+e_{21}x+e_{22}x^2+e_{23}x^3,$$

$$e_1(x)=e_{10}+e_{11}x+e_{12}x^2+e_{13}x^3.$$

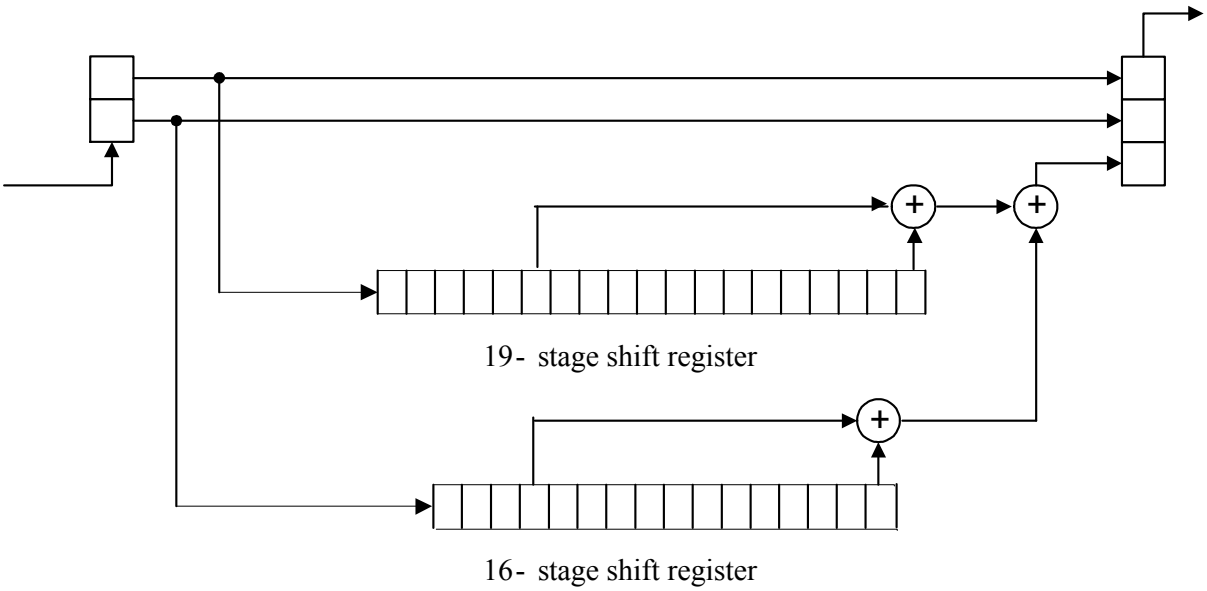

Рисунок 1 – Кодер (57, 38)-кода Ивадари

Исходя из того, что длина исправляемого пакета ошибок не больше 9, можно утверждать следующее. Если пакет начинается в первом кадре, то e_{33} всегда равно 0. Если e_{23} не равно нулю, то e_{20} и e_{10} равны нулю. Если e_{13} отличен от нуля, то $e_{10} = 0$.

На рисунке 2 представлены коэффициенты $s(x)$ для трех случаев, в которых пакет ошибок начинается в $e_{10}(x)$, $e_{20}(x)$ и $e_{30}(x)$ соответственно. При формировании синдрома полиномы перемежены и располагаются в порядке $e_1(x)$, $e_2(x)$ и $e_3(x)$, над их эхом затем производится обратное перемежение и они появляются в обратном порядке $e_3(x)$, $e_2(x)$ и $e_1(x)$.

Каждый бит полинома $e_2(x)$ воздействует на синдром дважды: за его первым появлением следует отклик с задержкой на 12 битов. Аналогично каждый бит полинома $e_1(x)$ воздействует на синдром дважды: за его первым появлением следует отклик с задержкой на 11 битов. Так как e_{23} и e_{10} не могут одновременно быть ненулевыми, эти пары ошибок никогда не перекрываются. Каждый бит полинома $e_3(x)$ воздействует на синдром один раз: за его первым появлением следуют только нули.

Если в действительности пакет ошибок начинается в l-*м* кадре, то синдром сдвигается вправо на l битов и перед ним вставляется l нулей. Процедура декодирования построена таким образом, что эта конфигурация не будет восприниматься как пакет ошибок, начинающийся в первом кадре.

Перейдем к описанию декодера, изображенного на рисунке 3. После того как в него поступили первые 16 кадров, в регистре синдрома хранятся первые 16 битов синдрома, причем s_0 располагается справа.

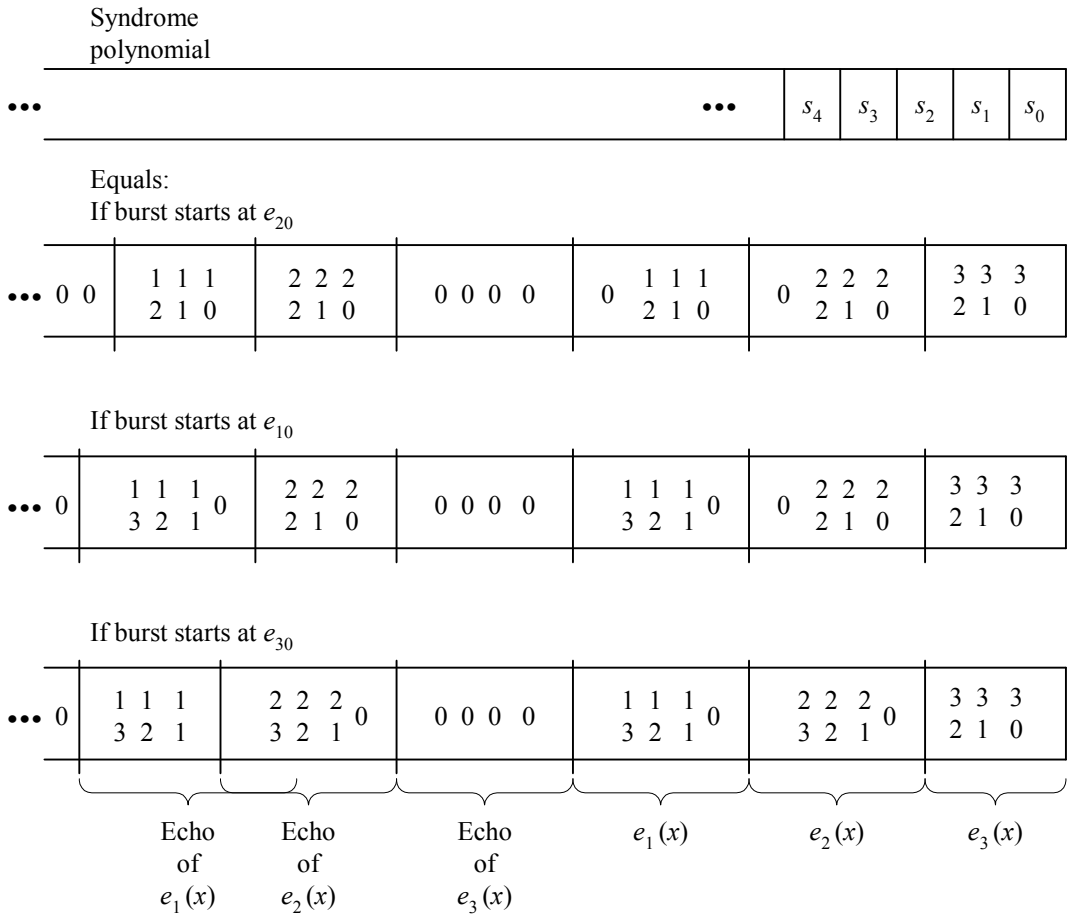

Рисунок 2 - Коэффициенты синдрома

Аналогично первый принятый бит полинома $v_2(x)$ находится в крайнем правом разряде v_2(x)-регистра. В течение еще трех последующих тактов первый принятый бит $v_1(x)$ не достигнет крайнего правого разряда v_1(x)-регистра. Описание работы декодера начнем с этого момента времени. Первые три бита синдрома зависят лишь от ошибок в принятых проверочных битах и нас не интересуют. Следовательно, та часть рисунка, которая нанесена штриховыми линиями, может быть опущена. Заштрихованные разряды регистра сдвига также могут быть опущены, а s_{15} может вводиться непосредственно из предшествующего сумматора по модулю 2.

Сначала декодер исправляет второй бит каждого кадра пакета ошибок, а затем возвращается к началу пакета и исправляет первый бит каждого кадра. Третий бит каждого кадра является проверочным; он не исправляется.

Для исправления ошибки в $v_2(x)$ декодер проверяет компоненты s_3 и s_{15} синдрома. Если обе компоненты равны единице, то правый бит $v_2(x)$ является ошибочным и в потоке данных, а также в регистре синдрома делаются соответствующие исправления. Пакет ошибок $e_3(x)$, начинающийся в более позднем кадре, не может вызвать сбоя, так как

отклик $e_3(x)$ равен нулю. Параллельно с этим для нахождения ошибки в $v_1(x)$ декодер четырьмя кадрами позднее проверяет компоненты s_4 и s_{15} синдрома. Если они обе равны единице, а s_3 равен нулю, то правый бит $v_1(x)$-регистра ошибочен. Такая проверка $v_1(x)$ не приводит к нахождению ошибок в течение первых четырех тактов, но затем начинает исправлять ошибки в $v_1(x)$. Пакет $e_3(x)$, начинающийся в более позднем кадре, не может вызвать сбоя, так как отклик $e_3(x)$ равен нулю. В силу выбора способа проверки еще не исправленный бит $e_2(x)$ также не может вызвать сбоя, и поэтому s_3 равен нулю.

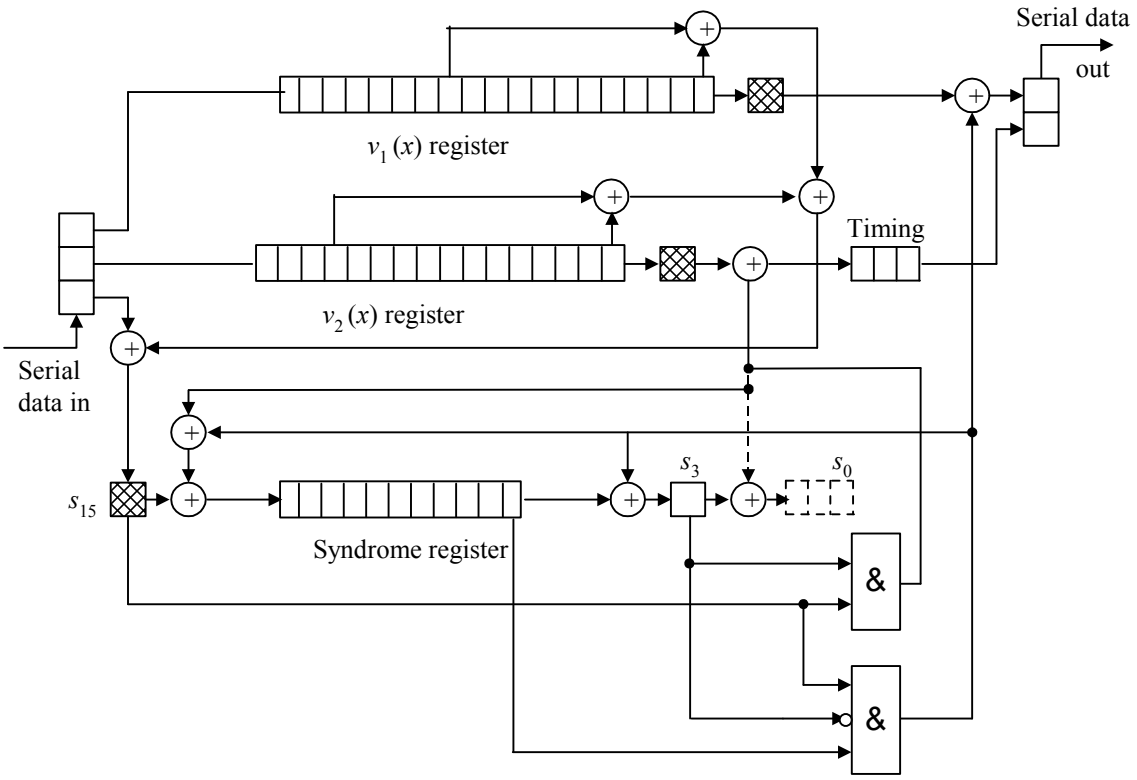

Рисунок 3 – Декодер (57, 38)-кода Ивадари

После появления пакета ошибок длины 9 принятое слово не должно содержать больше ошибок, чтобы синдрому ничто не мешало во время исправления этого пакета. Для этого между пакетами ошибок должно быть свободно от ошибок не менее 19 кадров (57 хороших битов). При выполнении этого правила декодер будет успешно исправлять следующие друг за другом пакеты ошибок длины не больше 9 каждый.

2. Векторные коды

Векторные коды (orchard code – код фруктового сада) – это класс кодов с контролем по четности [3]. Эти коды позволяют повысить

надежность обнаружения и исправления ошибок при минимальном повышении избыточности. Векторный метод предусматривает использование нескольких векторов двоичных символов для обработки данных, представленных в виде параллельного потока. Он гораздо более эффективен, чем обычные способы контроля четности.

Концепция векторного кода основана на том, что место нахождения любого ошибочного бита можно однозначно определить на пересечении двух не только перпендикулярных векторов, но также выбранных специальным способом. При этом для описания контрольных векторов кода используется графическое представление, которое поясняет принцип работы и на основании которого строится кодирующее и декодирующее устройства.

В работе [3] рассматриваются несколько вариаций векторного кода, из которых можно выделить следующие.

Код по паритету или код с четным числом единиц, является блоковым кодом, содержит один проверочный символ и позволяет обнаруживать все ошибки нечетной кратности. Формирование проверочного символа в параллельном потоке двоичных символов (рис. 4) можно представить следующим образом (сеткой обозначен проверочный символ, который формируется суммой по модулю два информационных, отмеченных диагональными линиями).

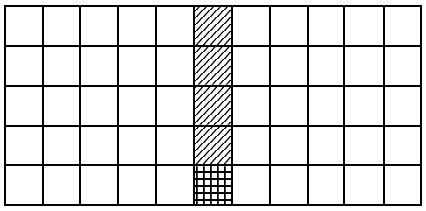

Рисунок 4

Если проверочный символ формировать с помощью суммы по модулю два двух векторов, то появляется возможность не только обнаруживать, но и исправлять одиночные ошибки (рис.5).

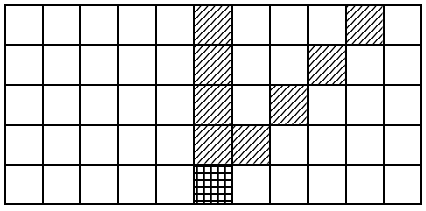

Рисунок 5

Возможен также и другой вариант формирования контрольного символа. Этот вариант для исправления одиночных ошибок по сравнению с предыдущим обладает излишней избыточностью (рис.6).

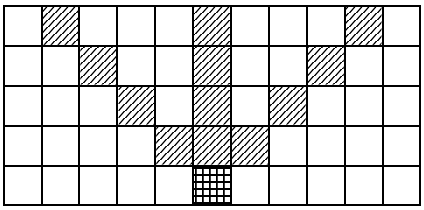

Рисунок 6

И, наконец, основной вариант формирования проверочного символа, рассматриваемый в качестве примера в работе [3], имеет следующий вид (рис 7):

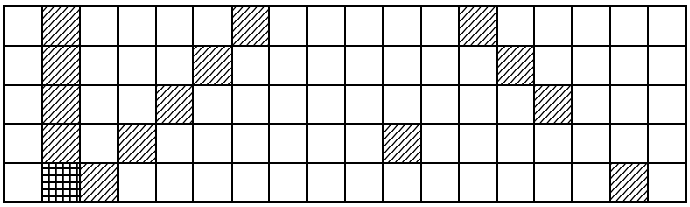

Рисунок 7

В типичной трехвекторной схеме контроля по методу фруктового сада устройство кодирования состоит из параллельной регистровой матрицы и генератора сигналов четности. Выход генератора сигналов четности по обратной связи поступает на вход регистровой матрицы для формирования проверочного бита. Декодер выполнен аналогично кодеру с тем исключением, что, поскольку поступающей на его вход параллельный поток данных уже содержит поверочный бит, выход генератора четности поступает на вход синдромного последовательного регистра. Первый корректирующий каскад состоит из параллельной регистровой матрицы, синдромного регистра и блока распознавания кода. Схема распознавания кода является основным элементом, которым различаются последующие корректирующие каскады. Для трехвекторного кода может быть два таких каскада.

Контроль и исправление ошибок по векторному методу можно производить с использованием более трех векторов. Для случая n векторов необходимо иметь (n-1) корректирующих каскадов. Первый корректирующий каскад будет исправлять все ошибки, для которых в синдромный регистр будет поступать n признаков ошибок. Каждый последующий каскад будет исправлять ошибки, для которых в синдромный регистр будут поступать признаки ошибок, число которых на единицу меньше, чем для предыдущего каскада. Последний

корректирующий каскад будет исправлять все ошибки, для которых будет по два признака ошибок.

К несомненному достоинству векторных кодов является их наглядность, которую обеспечивает их графическое представление. Вместе с тем, хотя по своей природе они являются сверточными, нет никакого упоминания о существовании таких кодов. Кроме того, нет никакого алгоритма выбора векторов и, в том числе, расстояния между ними. Для характеристики корректирующих возможностей кодов используются такие выражения, как “обычно” обнаруживают или исправляют ошибки или “много” обнаруживаемых и исправляемых ошибок.

3. Графическое представление кодов

Рассмотрим графическое представление нескольких вариантов кодов Ивадари.

1) Пусть конструктивный параметр $n_0 = 3$.

Для $\lambda = 2$ получаем два порождающих полинома: $g_1(x) = x^{13} + x^5$, $g_2(x) = x^{11} + x^2$ и исправляемый пакет ошибок $\lambda n_0 = 6$.

В графическом виде:

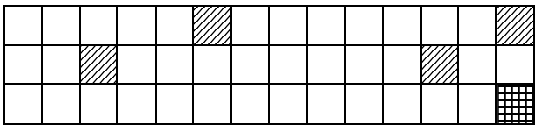

Рисунок 8

Для $\lambda = 3$ получаем два порождающих полинома: $g_1(x) = x^{18} + x^7$, $g_2(x) = x^{15} + x^3$ и исправляемый пакет ошибок $\lambda n_0 = 9$.

В графическом виде:

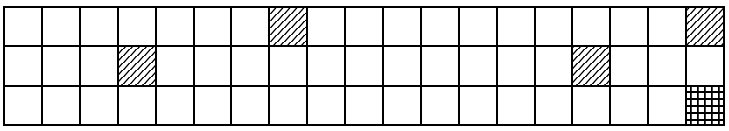

Рисунок 9

Для $\lambda = 4$ получаем два порождающих полинома: $g_1(x) = x^{23} + x^9$, $g_2(x) = x^{19} + x^4$

и исправляемый пакет ошибок $\lambda n_0 = 12$.

В графическом виде:

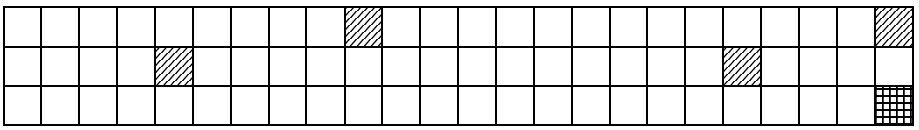

Рисунок 10

Итак, при $n_0 = 3$ проверочный бит формируется по 4-м (в общем случае – по $2(n_0 - 1)$) контрольным битам, которые можно разбить на два вектора. Расстояние между кадрами, в которых находятся контрольные биты, зависит от параметра λ. В первом векторе оно равно λ, во втором - λ-1.

2) Пусть конструктивный параметр $\lambda = 2$.

Для $n_0 = 3$ получаем два порождающих полинома: $g_1(x) = x^{13} + x^5$, $g_2(x) = x^{11} + x^2$ и исправляемый пакет ошибок $\lambda n_0 = 6$.

В графическом виде:

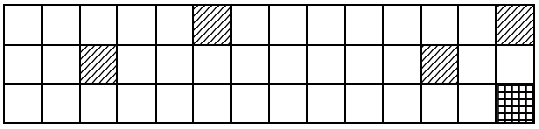

Рисунок 11

Для $n_0 = 4$ получаем три порождающих полинома: $g_1(x) = x^{19} + x^8$, $g_2(x) = x^{17} + x^5$, $g_3(x) = x^{15} + x^2$ и исправляемый пакет ошибок $\lambda n_0 = 8$.

В графическом виде:

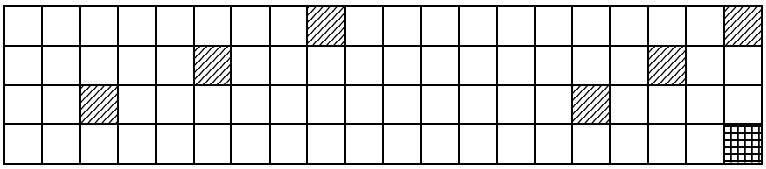

Рисунок 12

Для $n_0 = 5$ получаем четыре порождающих полинома:
$g_1(x) = x^{25} + x^{11}$, $g_2(x) = x^{23} + x^8$,
$g_3(x) = x^{21} + x^5$, $g_4(x) = x^{19} + x^2$
и исправляемый пакет ошибок $\lambda n_0 = 8$.

В графическом виде:

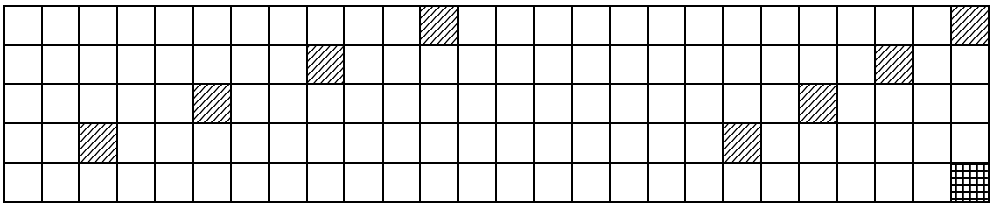

Рисунок 13

Итак, рассмотренные примеры показывают зависимость расстояния между кадром с последним контрольным битом первого вектора и кадром с первым контрольным битом второго вектора равно $(n_0 + 2(\lambda - 1))$.

Выводы

Графический способ описания кодов Ивадари дает наглядное представление о принципах построения этих кодов, позволяет строить эти коды, не используя порождающие полиномы, понятие которых требует определенных знаний из алгебры полей Галуа. Кроме того, этот способ позволяет выполнить простой сравнительный анализ с векторными кодами.

В отличие от векторных коды Ивадари обладают следующими преимуществами: формализованный алгоритм построения контрольных векторов для формирования проверочного символа, гарантированное исправление всех пакетов ошибок длины λn_0, декодирование не требует нескольких каскадов исправления ошибок.

Литература

1. Robert H. Morelos-Zaragoza. The Art of Error Correcting Coding. First Edition, John Wiley & Sons, 2002. – 221p.
2. R.E.Blahut. Theory and Practice of Error Control Codes. Addison-Wesley Publishing Company, Massachusetts, 1984. – 576p.
3. Скотт Э., Гётшель Д. Исправление многобитовых ошибок при помощи одного контрольного бита на слово// Электроника, 1981, № 9. – С.40-47.