УДК 303.732.4

К ВОПРОСУ РАЗРАБОТКИ ВИЗУАЛИЗИРУЮЩЕГО АЛГОРИТМА ДЛЯ КОДА ХЕММИНГА

В.А. АТРОЩЕНКО, Н.Д. ЧИГЛИКОВА, М.В. СЕРИКОВА, А.Б. ХИСМАТУЛИН

Кубанский государственный технологический университет,
350072, Российская Федерация, г. Краснодар, ул. Московская, 2;
электронная почта : marinella04@list.ru

Современные запоминающие устройства состоят из огромного количества запоминающих элементов, каждый из которых хранит бинарное значение – 0 или 1. Так, оперативная память ёмкостью всего в 1 Мбайт содержит 8 388 698 ЗЭ. При работе подобных устройств могут возникать ошибки, обусловленные воздействием различных факторов. В этом случае для повышения надёжности работы запоминающих устройств используют специальные обнаруживающие и корректирующие коды, позволяющие обнаруживать место, где произошла ошибка, и исправлять её. В данной статье будет рассмотрен корректирующий код Хемминга. Простейшим способом обнаружения ошибок является проверка на чётность. В этом случае к битам передаваемого или хранимого М-разрядного слова добавляется ещё один бит – бит чётности, значение которого подбирается таким образом, чтобы среди получившихся N разрядов (N=M+1) обязательно было чётное число единиц. Такой избыточный код позволяет лишь констатировать факт наличия ошибки в слове даже без указания места, где она произошла. Для повышения надёжности работы запоминающих устройств используют корректирующие коды, в частности, код Хемминга. Принцип построения корректирующих кодов заключается в том, что к каждому хранимому или передаваемому М-разрядному слову добавляют К битов с соответствующим их расположением среди битов М-разрядного слова. Подобные N-разрядные коды (N=M+K) были впервые рассмотрены в 1948 г. Р. Хеммингом и с тех пор обычно называются кодами Хемминга (N, M). В данной статье рассмотрим принцип построения кода Хемминга для 16-разрядного слова данных (M=16), исправляющего все одиночные ошибки, а также визуализирующий алгоритм для данного кода.

Ключевые слова: код Хемминга, кодирование, компьютерная сеть, биты чётности, информационные биты, реализация алгоритма.

Применение кода Хемминга студентами института ИКСИБ в процессе обучения, проведение расчетов, как показала практика, затруднительно. В следствие чего, возникла необходимость в создании обучающего визуализирующего алгоритма. Далее рассмотрим основные понятия кода и их визуализацию по средствам программно реализованного алгоритма.

Рассмотрим Принцип построения кода Хемминга для 16-разрядного слова данных (M=16), исправляющего все одиночные ошибки.

Определяется необходимое число проверочных разрядов К (K=N-M) из соотношения:

$$2^{N-M}-1 \geq N.$$

Откуда для M=16 находим N=21 и K=5.

Для различных K существуют свои верхние границы N и M.

Посмотрим табл.1, в которой представлены верхние границы M и N для различных K представлены в таблице 1.

Таблица 1 - Верхние границы M и N для различных K

K	1	2	3	4	5	6	7	8	9	10
M	0	1	4	11	26	57	120	247	502	1013
N	1	3	7	15	31	63	127	255	511	1023

Далее все биты N-разрядного слова нумеруются слева направо, начиная с 1, при этом все биты, номера которых равны степени числа 2, являются битами чётности, а остальные – информационными. Полученный таким образом код Хемминга для 16-разрядного слова данных представлен на рисунке 1.

Нумерация битов кода Хемминга для 16-разрядного слова данных

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
0	0	1	1	1	0	1	0	1	0	1	0	0		1	1	1	1	0	1	1	0
		15			14	13	12			11	10	9	8	7			6	5		4	3	2	1	0

Рисунок 1 - Код Хемминга для 16-разрядного слова данных

В 1, 2, 4, 8 и 16 разрядах данного кода располагаются биты чётности, а в разрядах 3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17, 18, 19, 20 и 21 биты данных слова 16-разрядного исходного слова. Для каждого бита чётности существуют свои контролируемые разряды.

Каждый бит чётности используется для контроля лишь определённых разрядов N-разрядного слова. Первый контрольный разряд контролирует разряды кода Хемминга с номерами $a_jx2^j+a_{j-1}x2^{j-1}+\ldots+a_2x2^2+a_jx2^1+1x2^0$, где $j=\log_2 N$, a_j – произвольное значение. Таким образом, первый контрольный разряд контролирует все нечётные номера.

Второй контрольный разряд контролирует разряды кода Хемминга с номерами $a_jx2^j+a_{j-1}x2^{j-1}+\ldots+a_2x2^2+1x2^1+a_0x2^0$.

Соответственно третий контрольный разряд контролирует разряды кода Хемминга с номерами $a_jx2^j+a_{j-1}x2^{j-1}+\ldots+1x2^2+a_1x2^1+\ldots+a_0x2^0$ и т.д.

Номера битов чётности и контролируемых им разрядов слов в коде Хемминга. В число контролируемых разрядов включается и тот разряд, где расположен сам бит чётности. При этом содержимое бита чётности устанавливается так. Если суммарное число единиц в контролируемых им разрядах чётное - содержимое бита четности равно 0.

Таким образом в рассматриваемом примере:

1. Первый контрольный разряд в коде Хемминга равен 0, так как биты 3, 5, 7, 9, 11, 13, 15, 17, 19 и 21 содержит восемь единиц;

2. Второй контрольный разряд в коде Хемминга равен 0, так как биты 3, 6, 7, 10, 11, 14, 15, 18 и 19 содержат шесть единиц;

3. Четвёртый контрольный разряд в коде Хемминга равен 1, так как биты 5, 6, 7, 12, 13, 14, 15, 20 и 21 содержат пять единиц;

4. Восьмой контрольный разряд в коде Хемминга равен 0, так как биты 9, 10, 11, 12, 13, 14 и 15 содержат четыре единицы;

5. Шестнадцатый контрольный разряд в коде Хемминга равен 1, так как биты 17, 18, 19, 20 и 21 содержат три единицы.

Код, образованный значениями контрольных разрядов, называют дополнительным кодом. То есть для 16-разрядного кода данных 1101101001110110 дополнительный код равен 10100. Записывается и читается, начиная со старших разрядов кода Хемминга.

Дополнительный код можно также получить путём инвертирования результата поразрядного сложения (т.е. сложения по модулю 2) номеров тех разрядов кода данных, значения которых равны 1.

Нумерация битов кода Хемминга для 16-разрядного слова данных

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
0	0	1	1	1	0	1	0	0	0	1	0	0	1	1	1	1	0	1	1	0
		15		14	13	12		11	10	9	8	7	6	5		4	3	2	1	0

Рисунок 2 - Код Хемминга для 16-разрядного слова данных с искажённым девятым разрядом (11-информационным)

Нумерация информационных битов 16-разрядного слова данных

Проверка образовавшегося кода даёт следующий результат: бит чётности 2, 4 и 16 правильны, а биты чётности 1 и 8 – неправильны.

Получение неправильного значения бита чётности 1 указывает на то, что ошибка должна быть в одном из контролируемых им битов: 1, 3, 5, 7, 9, 11, 13, 15, 17, 19 или 21. Поскольку бит чётности 2 правилен, то правильны и контролируемые им нечётные биты 3, 7, 11, 15 и 19, так что ошибка произошла не в них. Правильность контрольного бита 4 исключает возникновение ошибки в битах 5, 13 и 21, а правильность контрольного бита 16 – в бите 17. Следовательно, под подозрением остаются биты 1 и 9. Так как неправилен и бит чётности 8, который не контролирует бита с номером 1, но контролирует бит с номером 9, то можно сделать вывод об ошибочности бита 9. Инвертирование этого бита (изменение его значения с 0 на 1) исправляет положение – все биты чётности становятся правильными.

Таким образом, номер искажённого разряда определяется суммой номеров неправильных битов чётности. В нашем примере биты 1 и 8 неправильны, следовательно, искажённый разряд – 9 (9=1+8).

Средой разработки программной реализации переводов чисел для кода Хемминга был выбран язык высокого уровня C#. На данном языке был разработан графический пользовательский интерфейс и реализован функционал. Начальное окно работы программы приведено на рисунке 1.

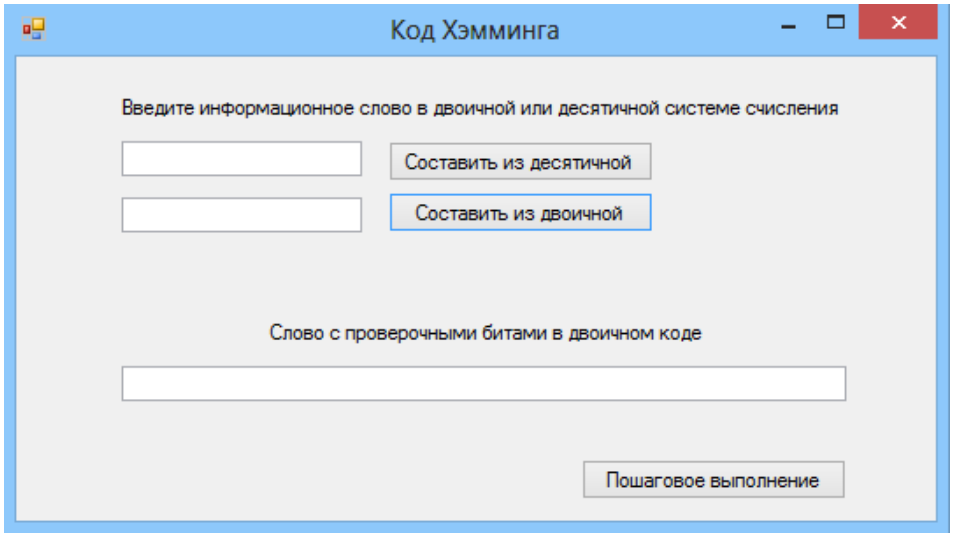

Рисунок 3 – Начальное окно программы для кода Хемминга

На первом шаге работы программа предлагает ввести любое число, затем выводит результат при нажатии на кнопку «Составить из десятичной/Составить из двоичной». А также выводит слово с проверочными битами в двоичном коде. Студент может либо воспользоваться результатом вычисления, либо, если он хочет детально разобрать работу алгоритма – посмотреть пошаговое выполнение алгоритма с пояснениями.

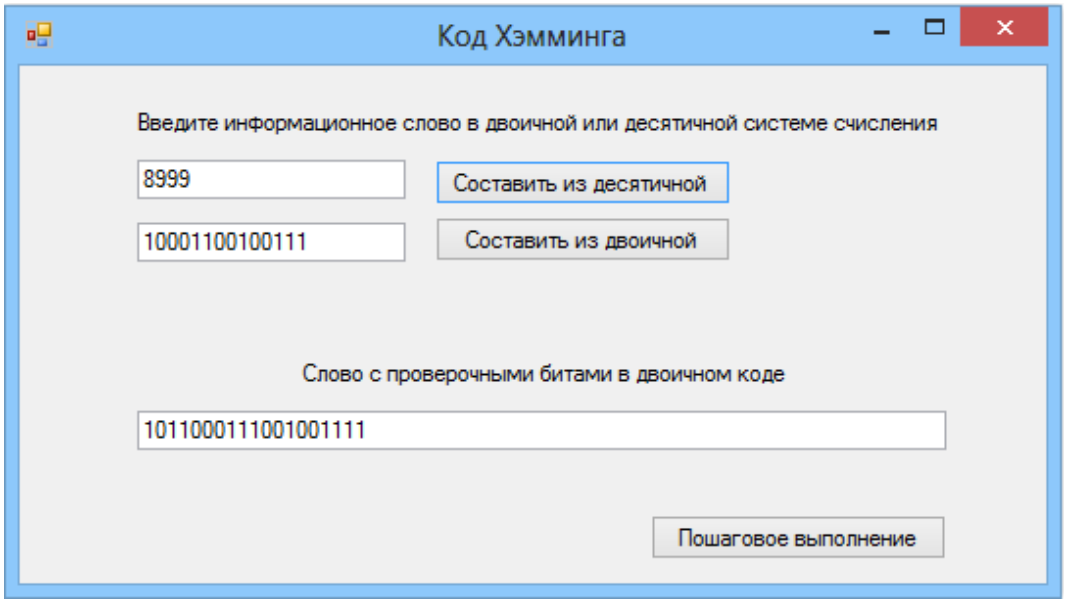

Рисунок 4 – Окно ввода информационного слова

Рассмотрим окно пошагового выполнения. На первом шаге программа выводит количество проверочных бит.

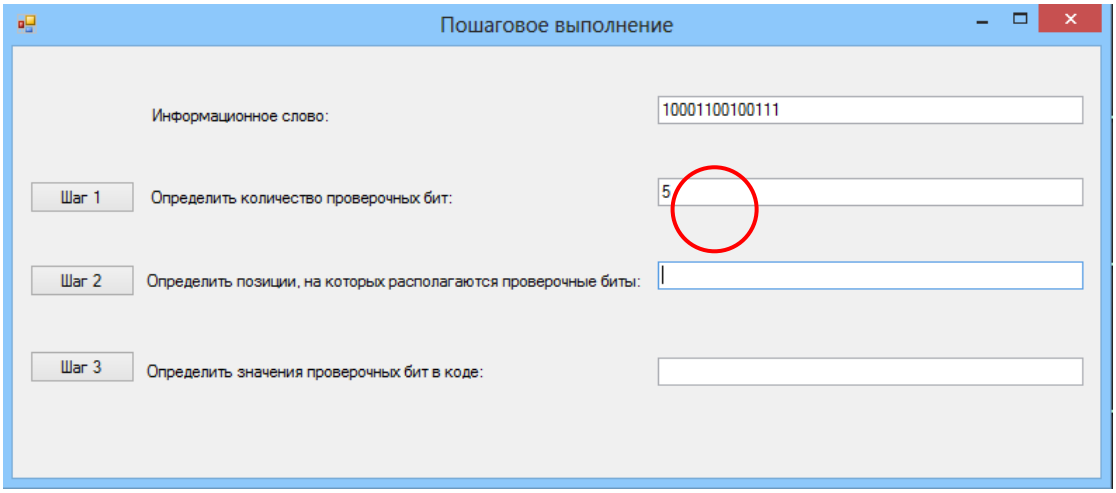

Рисунок 5 – «Шаг 1» вычисление количества проверочных бит

В программе отсутствует подсказка о том, как именно было найдено число проверочных бит, поэтому студенту для решения этого вопроса следует обратиться к теоретическим основам кода Хемминга.

Далее программа определяет позиции проверочных бит.

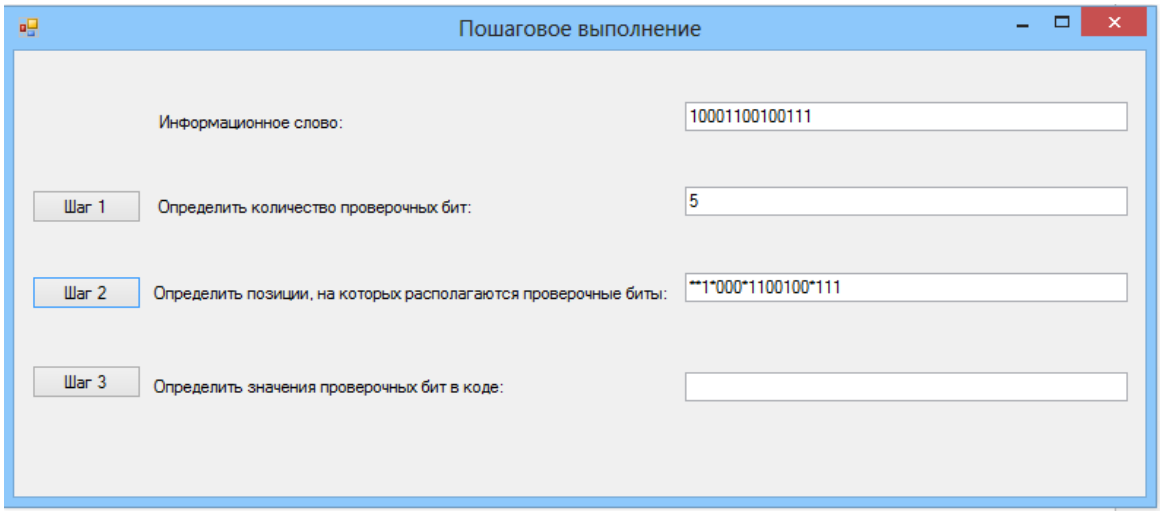

Рисунок 6 – «Шаг 2» определение позиций проверочных бит

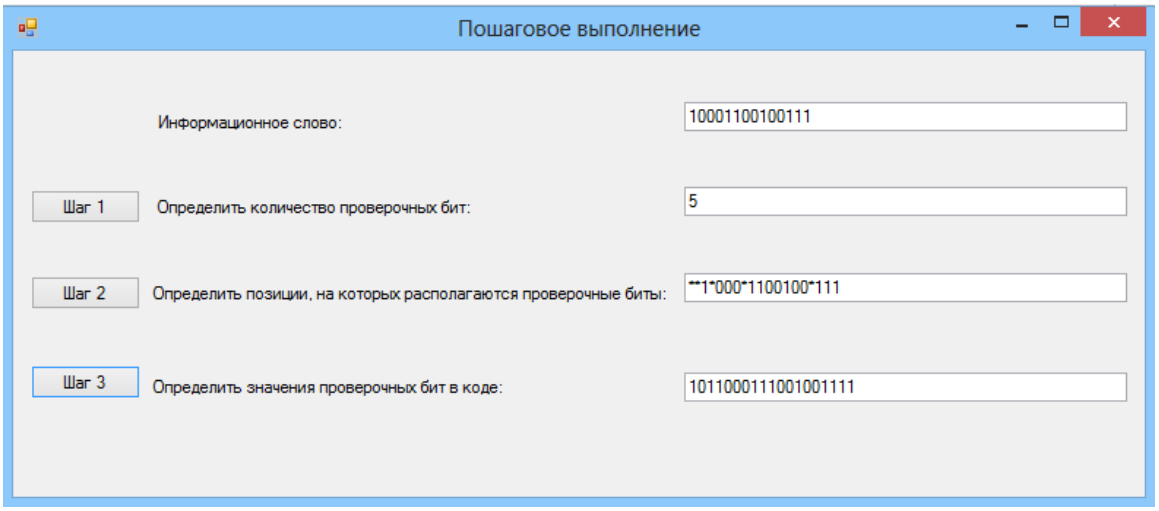

Рисунок 7 – «Шаг 3» определение значений проверочных бит

Данное программное обеспечение и визуализирующий алгоритм может быть использован в учебном процессе студентами института ИКСИБ КубГТУ.

ЛИТЕРАТУРА

1. Бройдо В.Л. Вычислительные системы, сети и телекоммуникации. - СПб.: Питер 2002.

2. Кирмайер М. Информационные технологии. - СПб.: Питер, 2003.

3. Мэтьюз Дж. Web - сервер. - СПб.: Символ, 1998.

4. Олифер В. Г., ОлиферН.А. Компьютерные сети. - СПб.: Питер, 2005.

5. Олифер В. Г. Сетевые операционные системы. - СПб.: Питер, 2003.

6. Коннолли Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика - М.: Вильямс, 2000.

7. Математические модели систем управления. Учеб.пособие.// Под общ.ред.В.Ф.Демьянова. – СПб, Изд-во СПб ун-та, 2000.

REFERENCES

1. Broydo V.L. Vychislitelnye sistemy, seti i telekommunikatsii. - SPb.: Piter 2002.

2. Kirmayer M. Informatsionnye tekhnologii. - SPb.: Piter, 2003.

3. Metyuz Dzh. Web - server. - SPb.: Simvol, 1998.

4. Olifer V. G., OliferN.A. Kompyuternye seti. - SPb.: Piter, 2005.

5. Olifer V. G. Setevye operatsionnye sistemy. - SPb.: Piter, 2003.

6. Konnolli T. Bazy dannykh. Proektirovanie, realizatsiya i soprovozhdenie. Teoriya i praktika - M.: Vilyams, 2000.

7. Matematicheskie modeli sistem upravleniya. Ucheb.posobie.// Pod obshch.red.V.F.Demyanova. – SPb, Izd-vo SPb un-ta, 2000.

ON THE ISSUE OF DEVELOPING IMAGING ALGORITHM FOR THE HAMMING CODE

V.A. ATROSCHENKO, N.D. CHIGLIKOVA, M.V. SERIKOVA, A.B. HISMATULIN

Kuban State Technological University,
2, Moskovskaya st., Krasnodar, Russian Federation, 350072,
e-mail: marinella04@list.ru

Modern storage devices are composed of a huge number of storage elements, each of which stores a binary value 0 or 1. So, the RAM capacity of just 1 MB contains 8 388 698 GE. When operating such devices, errors can occur, due to the influence of various factors. In this case, to improve the reliability of storage devices use a special detecting and error-correcting code that can detect the place where the error occurred, and fix it. In this article we will consider correcting Hamming code. The simplest method of error detection is parity. In this case, the bits transmitted or stored M-bit word is added to another bit – parity bit whose value is selected so that the resulting N bits (N=M+1) is necessarily an even number of units. Such redundant code allows only state the fact of presence of errors in word even without specifying the place where it happened. To improve the reliability of storage devices use error-correcting codes, in particular, the Hamming code. The principle of construction of error-correcting codes is that each stored or transmitted M-bit word add K bits, with the appropriate arrangement of bits among M bit word. Such N-bit codes (N=M+K) were first examined in 1948, R. Hemming and since then, usually referred to as Hamming codes (N, M). In this paper we consider the principle of construction of the Hamming code for the 16-bit data word (M=16) which corrects all single errors, as well as visualizing the algorithm for the code.

Key words: Hamming code, coding, computer network, parity bits, data bits, an implementation of the algorithm.