DOI 10.37882/2223–2966.2022.01–2.17

ИССЛЕДОВАНИЕ ВЫЧИСЛИТЕЛЬНОЙ СЛОЖНОСТИ ФОРМИРОВАНИЯ КОНТРОЛЬНЫХ СУММ ДЛЯ АЛГОРИТМА CRC В ЗАВИСИМОСТИ ОТ РАЗРЯДНОСТИ ПОРОЖДАЮЩЕГО ПОЛИНОМА

INVESTIGATION OF THE COMPUTATIONAL COMPLEXITY OF GENERATING CHECKSUMS FOR THE CRC ALGORITHM DEPENDING ON THE WIDTH OF THE GENERATING POLYNOMIAL

O. Turdiev

Summary. Statement of the problem: The need to ensure the integrity of data transmitted in communication networks makes the issue of ensuring the formation of checksums actual. At the same time, it is advisable to reduce the complexity of algorithms for generating checksums to increase data integrity. The well-known CRC (Cyclic Redundancy Code) checksum generation algorithm has high computational complexity. The aim of the work is to perform search studies to substantiate the fundamental possibility of reducing the computational complexity of the algorithm for generating CRC checksums and searching for possible ways of practical implementation. The scientific novelty of the research lies in the fact that the first considers the computational complexity of the CRC algorithm depending on the generating polynomials and their bit width.

Keywords: computational complexity, generator polynomial, cyclic redundancy code, batch errors, erroneous bits, integrity.

Турдиев Одилжан Акрамович
Аспирант, ФГБОУ ВО «Петербургский Государственный Университет Путей Сообщения Александра I», г. Санкт-Петербург
odiljan.turdiev@mail.ru

Аннотация. Постановка задачи: Необходимость обеспечения целостности данных, передаваемых в сетях связи, актуализирует вопрос обеспечения формирования контрольных сумм. При этом целесообразно снижение сложности алгоритмов формирования контрольных сумм для повышения целостности данных. Известный алгоритм формирования контрольных сумм CRC (Cyclic Redundancy Code) обладает высокой вычислительной сложностью. Целью работы является выполнение поисковых исследований для обоснования принципиальной возможности снижении вычислительной сложности алгоритма формирования контрольных сумм CRC и поиска возможных путей практической реализации. Научная новизна исследования заключается в том, что в первые рассматривается вычислительная сложность алгоритма CRC в зависимости от порождающих полиномов и их разрядности.

Ключевые слова: вычислительная сложность, порождающий полином, циклический избыточный код, пакетные ошибки, ошибочные биты, целостность.

Введение

Одной из важных задач в современных сетях связи является обеспечение целостности данных. Наиболее распространенным алгоритмом определения целостности передаваемых данных является алгоритм вычисления циклического избыточного кода CRC (Cyclic Redundancy Check).

Алгоритм CRC основан на теории циклических кодов с исправлением ошибок. Этот алгоритм впервые предложен В.В. Петерсон и Д.Т. Бровн, в работе [1]. Алгоритм CRC вычисляет короткую двоичную последовательность, имеющую определенную неизменную длину, известную как контрольное значение или код алгоритма CRC. Контрольное значение CRC вычисляется для каждого отдельного блока данных, который должен быть передан по сети, и добавляется к блоку, образуя кодовое слово. Когда кодовое слово принимается на приемной стороне выполняется одно из двух действий. Либо сравнивается принятое контрольное значение CRC со значением того CRC, которое формируется заново для передаваемых данных на приемной стороне, либо заново на приемной стороне формируется CRC для всего принятого кодового слова и сравнивается результирующее контрольное значение CRC с ожидаемой константой остатка. Если контрольные значения не совпадают, то передаваемые данные содержат ошибку и приемное устройство может предпринять действия по их коррекции, такие как повторное считывание блока или его повторная отправка.

Теоретические положения функционирования алгоритма CRC приведены в работах [2–5]. Предполагается, что когда блок данных и его контрольное CRC получены правильно, то для этого блока данных обеспечивается

Таблица 1. Популярные стандартизованные порождающие полиномы [9]

Название	Порождающие полиномы
CRC-4-TU	x^4+x+1 (ITU G.704)
CRC-5-EPC	x^5+x^3+1 (Gen 2 RFID)
CRC-5-ITU	$x^5+x^4+x^2+1$ (ITU G.704)
CRC-5-USB	x^5+x^2+1 (USB token packets)
CRC-6-ITU	x^6+x+1 (ITU G.704)
CRC-7	x^7+x^3+1 (ITU-T G.707, ITU-T G.832, MMC, SD)
CRC-8-CCITT	x^8+x^2+x+1 (ATM HEC), ISDN Header Error Control and Cell Delineation ITU-T I.432.1 (02/99)
CRC-8-Dallas/Maxim	$x^8+x^5+x^4+1$ (1-Wire bus)
CRC-8	$x^8+x^7+x^6+x^4+x^2+1$ (ETSI EN302 307, 5.1.4)
CRC-8-SAE J1850	$x^8+x^4+x^3+x^2+1$
CRC-10	$x^{10}+x^9+x^5+x^4+x+1$
CRC-11	$x^{11}+x^9+x^8+x^7+x^2+1$ (FlexRay)
CRC-12	$x^{12}+x^{11}+x^3+x^2+x+1$ (системы телекоммуникации)
CRC-15-CAN	$x^{15}+x^{14}+x^{10}+x^8+x^7+x^4+x^3+1$
CRC-16-IBM	$x^{16}+x^{15}+x^2+1$ (Bisync, Modbus, USB, ANSI X3.28,)
CRC-16-CCITT	$x^{16}+x^{12}+x^5+1$ (X.25, HDLC, XMODEM, Bluetooth, SD и др.)
CRC-16-T10-DIF	$x^{16}+x^{15}+x^{11}+x^9+x^8+x^7+x^5+x^4+x^2+1$ (SCSI DIF)
CRC-16-DNP	$x^{16}+x^{13}+x^{12}+x^{11}+x^{10}+x^8+x^6+x^5+x^2+1$ (DNP, IEC870, M-Bus)
CRC-24	$x^{24}+x^{22}+x^{20}+x^{19}+x^{18}+x^{16}+x^{14}+x^{13}+x^{11}+x^{10}+x^8+x^6+$ x^3+x+1 (FlexRay)
CRC-24-Radix-64	$x^{24}+x^{23}+x^{18}+x^{17}+x^{14}+x^{11}+x^{10}+x^7+x^6+x^5+x^4+x^3+x+1$ (OpenPGP)
CRC-30	$x^{30}+x^{29}+x^{21}+x^{15}+x^{13}+x^{12}+x^{11}+x^8+x^7+x^6+x^2+x+1$ (CDMA)
CRC-32-IEEE802.3	$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$ (V.42, MPEG-2, PNG, POSIX cksum)
CRC-32C (Castagnoli)	$x^{32}+x^{28}+x^{27}+x^{26}+x^{25}+x^{23}+x^{22}+x^{20}+x^{19}+x^{18}+x^{14}+$ $x^{13}+x^{11}+x^{10}+x^9+x^8+x^6+1$
CRC-32K (Koopman)	$x^{32}+x^{28}+x^{27}+x^{26}+x^{25}+x^{23}+x^{22}+x^{20}+x^{19}+x^{18}+x^{14}+x^{13}+$ $x^{11}+x^{10}+x^9+x^8+x^6+1$
CRC-32Q	$x^{32}+x^{31}+x^{24}+x^{22}+x^{16}+x^{14}+x^8+x^7+x^5+x^3+x+1$

Таблица 1 (продолжение). Популярные стандартизованные порождающие полиномы [9]

Название	Порождающие полиномы
CRC-64-ISO	$x^{64}+x^4+x^3+x+1$ (HDLC — ISO 3309)
CRC-64-ECMA	$x^{64}+x^{62}+x^{57}+x^{55}+x^{54}+x^{53}+x^{52}+x^{47}+x^{46}+x^{45}+x^{40}+x^{39}+$ $x^{38}+x^{37}+x^{35}+x^{33}+x^{32}+x^{31}+x^{29}+x^{27}+x^{24}+x^{23}+x^{22}+x^{21}+$ $x^{19}+x^{17}+x^{13}+x^{12}+x^{10}+x^9+x^7+x^4+x+1$

```
  10011011      00110011      11110000      01010101
+ 11001010    + 11001101    + 10100110    + 10101111
----------    ----------    ----------    ----------
  01010001      11111110      01010110      11111010
```

Рис. 1. Деление полинов выполняется в двоичной системе, с той разницей, что вычитание выполняется по модулю 2.

целостность. Процесс формирования и проверки CRC может быть достаточно трудоемким, при использовании сетевых устройств с низким быстродействием или высокой интенсивностью переданы данных. В связи с этим снижение вычислительной сложности алгоритма CRC является актуальной научно-практической задачей.

Кроме того, в реальных условиях передачи, на канал связи могут воздействовать различного рода помехи, проявляющиеся в исследуемом процессе в виде ошибочных бит, которые приводят к нарушению целостности данных [6]. В работе В.В. Яковлева [7] предложен выбор порождающего полинома для увеличения вероятности распознавание ошибок при формировании контрольных сумм в передаваемых данных.

Обобщая вышесказанное, можно отметать что целью работы является выполнение поисковых исследований для обоснования принципиальной возможности и возможных путей снижения вычислительной сложности алгоритма формирования контрольных сумм CRC, используемого для контроля целостности передаваемых данных.

Для поиска путей снижения вычислительной сложности в работе впервые проведено исследование вычислительной сложности формирования контрольных сумм CRC в зависимости от различных порождающих полиномов и их разрядности. Для оценки вычислительной сложности CRC в работе выполнено моделирование процесса передачи двоичных данных по симметричному каналу.

Основные понятия и характеристики CRC-кодов

Циклические избыточные коды CRC являются подклассом блочных кодов и применяются в протоколах HDLC, Token Ring, Token Bus, в семействах протоколов Ethernet и других протоколах канального уровня [8]. Под вычислительными ресурсами понимается память, мощность процессора, а также количество регистров сдвига. Одним из способов представления циклического кода является его представление в виде порождающего полинома — множества всех полиномов степени (r–код-1), содержащих в качестве общего множителя некоторый фиксированный полином $G(x)$. Полином $G(x)$ называется порождающим полиномом кода. Например, x^4+x+1, здесь r–код = 5, поскольку двоичная последовательность выглядит как 10011. Стандартизованные и рекомендованные порождающие полиномы для алгоритма CRC приведены в таблице 1, где показаны название стандарта и порождающий полином: например, запись x^4+x+1 эквивалентна $1\cdot x^4+0\cdot x^3+0\cdot x^2+1\cdot x+1\cdot x^0=10011$ (в двоичном виде).

Значение результата реализации алгоритма CRC есть остаток от деления двоичной последовательность $M(x)$, соответствующей входным данным, на порождающую двоичную последовательность $G(x)$ степени r.

Каждой конечной последовательности битов взаимно однозначно ставится в соответствие двоичный полином, последовательность коэффициентов которого представляет собой исходную последовательность. На-

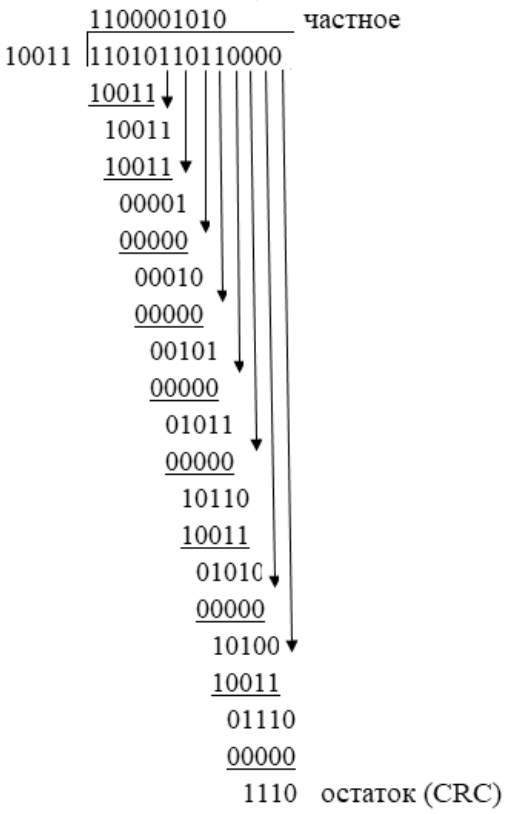

Рис. 2. CRC кодовое слова (передаваемый кадр) — 11010110111110 [13–15].

пример, последовательности битов 1011010 соответствует двоичный полином,

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

$$= x^6 + x^4 + x^3 + x.$$

Рассмотрим примеры, в которых сложение и вычитание выполняются без переноса разрядов в соответствии с операцией «исключающее ИЛИ», что соответствует сложению по модулю 2 двоичной арифметики, пример показано на рис. 1.

Использование полиномиальных кодов при передаче заключается в следующем. Отправитель и получатель заранее выбирают одинаковый генераторный полином $G(x)$, у которого коэффициенты при старшем члене и при младшем члене должны быть равны 1. При вычислении контрольных сумм блока размером m бит, должно соблюдаться следующий условия $m > r$. Далее, реализуя алгоритм вычисления CRC, отправитель прибавляет контрольную сумму к передаваемому блоку, рассматриваемому как полином $M(x)$, так чтобы передаваемый блок с контрольной суммой был кратен $G(x)$. Когда получатель получает блок с контрольной суммой, он делит его на $G(x)$. Если образуется ненулевой остаток, то это свидетельствует о возникновении ошибки при передаче [10,11].

Алгоритм вычисления контрольной суммы:

1. Добавить r нулей в конец блока так, чтобы он содержал $m + r$ разрядов, в результате этого получится полином $x^r M(x)$.
2. Разделить по модулю 2 полином $x^r M(x)$ на $G(x)$, частное игнорируется.
3. Вычесть по модулю 2 остаток (его длина не превышает r разрядов) из строки, соответствующей $x^r M(x)$. Полученный результат и есть блок с контрольной суммой.

В настоящее время в большинстве сетевых технологии наиболее часто используются три следующих вида порождающих полиномов $G(x)$ [12]:

$$\text{CRC-12} = x^{12} + x^{11} + x^3 + x^2 + x + 1$$

$$\text{CRC-16-IBM} = x^{16} + x^{15} + x^2 + 1$$

$$\text{CRC-16-CCITT} = x^{16} + x^{12} + x^5 + 1$$

CRC-12 используется для передачи символов из 6 разрядов. Два остальных — для 8 разрядных. CRC-16

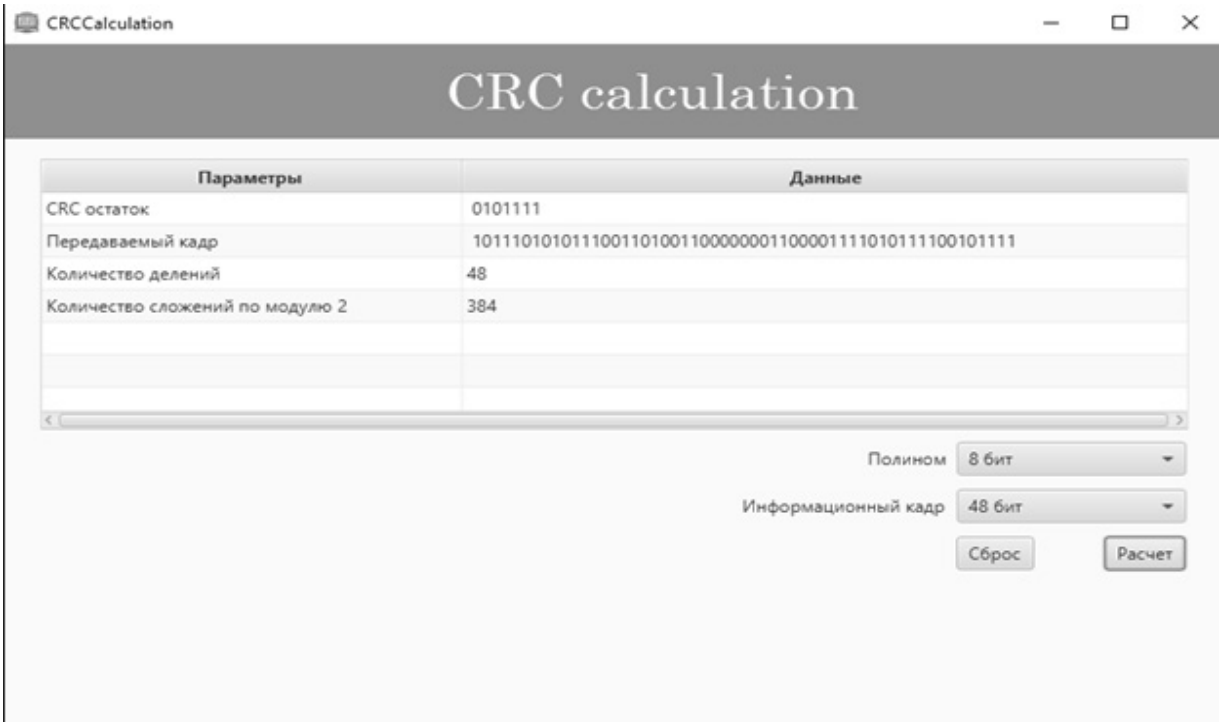

Рис. 3. CRC calculation

и CRC–CCITT обнаруживают все одиночные и все двойные ошибки, нечетное число изолированных ошибок, одиночные пакеты ошибок длиной не более 16 и многие пакетные ошибки длиной более 16 с вероятностью 99,97%.

После ознакомления с основными понятиями и характеристиками CRC кодов рассмотрим пример вычислений остатка циклических избыточных кодов и оценим вычислительную сложность этого процесса.

Пример вычисления остатка для построения CRC кодового слова и оценки вычислительной сложности

Вычисление CRC, детально показано на рис. 2, где:
Информационный кадр — 1101011011
Генераторный полином — 10011
Кадр с дополнительными нулями — 11010110110000

Таким образам, пример показывает, что при генераторном полиноме CRC-4-TU (10011) и битовой длине информационного кадра, равной 10 бит, требуется 10 делений и 50 сложений по модулю 2. В общем случае неочевидна зависимость между вычислительной сложностью (число сложений и делений), размером порождающего полинома и размером информационного полинома.

Для наилучшего представления о числе делений и сложений в алгоритме CRC с различными генераторными полиномами и информационными кадрами, создано программное обеспечение для оценки вычислительной сложности алгоритма CRC.

Программа для оценки вычислительной сложности алгоритма CRC

Программа предназначена для оценки сложности алгоритма CRC по разрядностям полинома с использованием метода контрольной суммы CRC, который приведен в ***примере циклических избыточных кодов.*** С помощью оценки сложности реализации алгоритма CRC путем выполнения сложений по разрядностям полинома получаем примерное количество сложений (сдвиг в ячейки). Это исследование открывает возможные пути снижения вычислительной сложности алгоритма формирования контрольных сумм CRC, используемого для контроля целостности передаваемых данных [16,17].

Для рассматриваемого способа генерации CRC написано приложение «CRC calculation» на языке Java, представленное в приложении 1, позволяющее для задаваемого пользователем размера кадра и порождающего полинома оценить требуемое число операций сложений и делений для получения итоговой контрольной CRC-суммы. Для написания приложения использована библиотека javaFX. Из состава этой библиотеки ис-

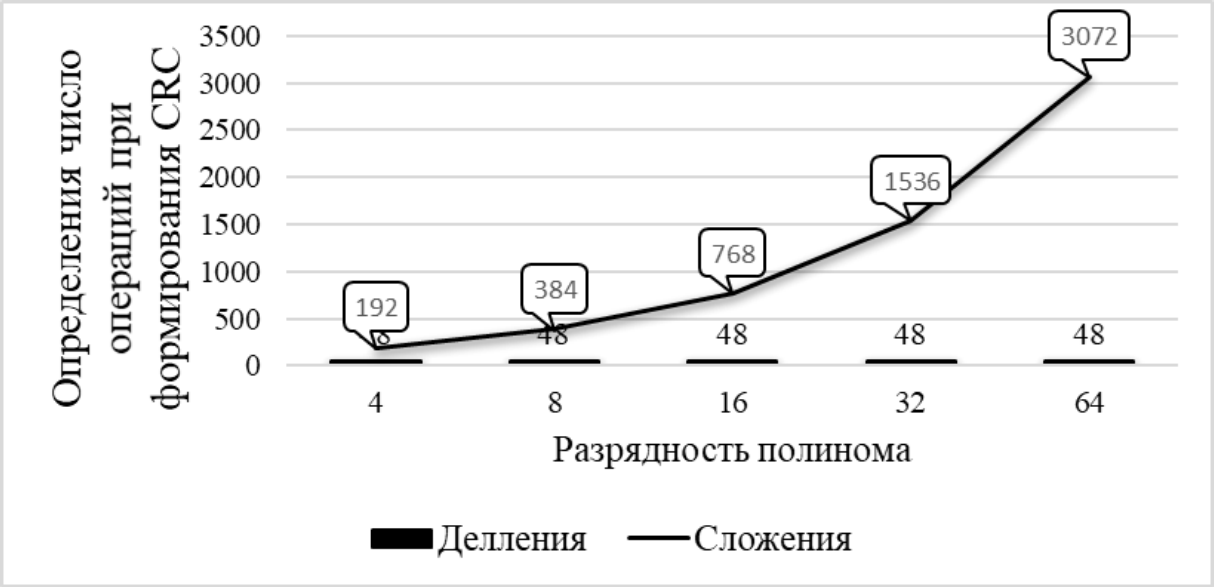

Рис. 4. Результаты оценки вычислительной сложности алгоритма формирования CRC

пользованы классы CRC и TableData. В классе CRC есть три операции:

1. Операция (calculation) — в нем выполняется вычисление кода CRC.
2. Операция (initialize).
3. Операция (xor).

Операция initialize обрабатывает массив битовой последовательности и передает его таблице, операция xor эмулирует операции логическое сложение разрядов входного кода и CRC кода на вход и подаёт два параметра X и Y –двоичные код, 1 и 1, или 1 и 0, или 0 и 1, или 0 и 0.

Далее генерируется полином определенной разрядности и в методе calculation вычисляется CRC код.

Программа имитирует процесс передачи данных между источником и приемником (рис. 3), а также обеспечивает выполнение следующих функций:

1. Задание исходных данных в указанной таблице («полином» и «информационный кадр», рис. 3) создает генератор чисел, использующий уникальное начальное число.
2. Расчет исходных данных отображается в программном окошке (кнопка «Расчет», рис. 3).
3. В результате исходных данных реализуется подсчет количества сложений и формирования контрольных сумм CRC циклических избыточных кодов (поле «параметры и данные», рис. 3).
4. Дополнительно на усмотрение пользователя можно произвести сброс и начать расчет заново (кнопка «Сброс», рис. 3).

После выполнения имитационного моделирования процесса расчета данных заново производится расчет числа сложения алгоритма контрольных сумм CRC циклических избыточных кодов, которые впоследствии сравниваются с ранее рассчитанными для выявления факта расчёта сложения алгоритма контрольных сумм, передаваемых данных (рис. 3); поле «CRC calculation».

Анализ результатов оценки вычислительной сложности алгоритма CRC

Расчет числа сложений, являющийся одной из важных составляющих алгоритма CRC, производится в соответствии с формулой 1. Отметим, что значение, полученное в результате вычислений, не должно быть малым, так как это приведет к необходимости повторной передачи данных. В этом случае алгоритм передачи выполняется повторно, что приводит к вторичным расчетам. Полиномы выбираются так, чтобы не возникало простоев в каналах передачи данных.

Расчет числа сложений имеет вида проводится по формуле:

$$R = (P \cdot N) \bmod 2, \quad (1)$$

здесь R — число операций сложений по модулю 2;
P — размер пакета;
N — разрядность полинома.

Рассмотрим пример, при котором разрядность 8 полинома имеет 48 делений, поэтому число сложений

по формуле (1) $R = 48 \cdot 8 = 384$. Результаты оценки вычислительной сложности алгоритма формирования CRC, выраженная в числе операций показаны с помощью графика на рис. 4 и подтверждены имитационной моделью.

Заключение

В статье исследован вычислительный алгоритм CRC. Показано, что при увеличении количества разрядов порождающих полиномов увеличивается число операций, при этом число операций деления остаётся неизмененным. При увеличении количества разрядов и операций деления порождающие полиномы остаются неизменными, однако увеличивается число сложений.

Показано, что возможным путем снижения вычислительной сложности алгоритма формирования контрольных сумм CRC, используемого для контроля целостности передаваемых данных, является уменьшение числа операций сложения при сохранении количества операций деления в рассматриваемом алгоритме.

В результате проведенной работы на основании исследований порождающих полиномов и числа операций деления получено выражение для расчёта числа операций сложения. Результаты исследования открывают путь для дальнейшего исследования по снижению вычислительной сложности операции циклически избыточный кодов.

ЛИТЕРАТУРА

1. Peterson, W.W. and Brown, D.T. (January 1961). "Cyclic Codes for Error Detection". Proceedings of the IRE49: 228.
2. Ritter, Terry (February 1986). "The Great CRC Mystery". Dr. Dobb's Journal 11 (2): 26–34, 76–83. http://www.ciphersbyritter.com/ARTS/CRCMYST.HTM. Retrieved 21 May 2009.
3. N. Cam-Winget, Nancy; R. Housley, Russ; D. Wagner, David; J. Walker, Jesse (May 2003). "Security Flaws in 802.11 Data Link Protocols". Communications of the ACM 46 (5): 35–39.
4. Stigge, Martin; Plötz, Henryk; Müller, Wolf; Redlich, Jens-Peter (May 2006). Reversing CRC — Theory and Practice (http://sar.informatik.hu-berlin.de/research/publications/SAR-PR-2006–05/SAR-PR-2006–05_.pdf). Berlin: Humboldt University Berlin. p. 17. Retrieved 4 February 2011
5. Anachriz (30 April 1999). "CRC and how to Reverse it" Retrieved 21 January 2010. Online essay with example x86 assembly code.
6. "Eurocontrol — FAQ: Technologies" (http://www.eurocontrol.int/aim/public/faq/chain faq3.html). European Organisation for the Safety of Air Navigation. 29 April 2009.
7. Яковлев В.В., Федоров Р.Ф. Стохастические вычислительные машины. Л., «Машиностроение», 1974,
8. Ross N. Williams Элементарное руководство по CRC — алгоритмам обнаружения ошибок. Copyright (C) Ross Williams, 1993
9. Циклический избыточный код [Электронный ресурс]. — Страница в интернете. — Режим доступа: https://ru.wikipedia.org/wiki/Циклический_избыточный_код, свободный
10. Яковлев В.В. Оценка влияния помех на производительность протоколов канального уровня / В.В. Яковлев, Ф.И. Кушназаров // Изв. Петерб. гос. ун-та путей сообщения. — СПб.: ПГУПС, 2015. — Вып. 1 (42). — С. 133–138.
11. Halsall F. Fifth edition, computer networks and the Internet / F. Halsall. — Addison-Wesley: Pearson Education, 2005. — 803 p.
12. Lin S. and Costello D.J. Jr. Error Control Coding: Fundamentals and Applications. Prentice-Hall, Inc., EnglewoodCliffs, N. J., 1983.
13. Halsall F. Data communications, computer networks and open systems / F. Halsall. — Addison-Wesley: Pearson Education, 1996. — 907 p.
14. Олифер В.Г. Компьютерные сети. Принципы, технологии, протоколы / В.Г. Олифер, Н.А. Олифер. — СПб.: Питер, 2008. — 958 с.
15. А. Ромащенко, А. Румянцев, А. Шень. Заметки по теории кодирования. |2-е изд., испр. и доп. | М.: МЦНМО, 2017. |88 с. ISBN978–5–4439–0689–8
16. Турдиев О.А., Яковлев В.В., Клименко С.В., Болтаев А.Х. Исследование формирования блоковой контрольный суммы (BCC) передаваемых данных. Известия СПбГЭТУ «ЛЭТИ» Выпуск № 6 2019 года.
17. Турдиев О.А., Клименко С.В., Тухтаходжаев А.Б. Оценки эффективности обнаружения ошибок контрольного суммирования (CRC) передаваемых данных Известия СПбГЭТУ «ЛЭТИ» Выпуск № 8 2019 года.

Журнал «Современная наука: актуальные проблемы теории и практики»