УДК 004.052.32+681.518.5 DOI: 10.31114/2078-7707-2020-1-119-125

Троичные коды с суммированием и их модификации

Д. В. Ефанов, д-р техн. наук

Российский университет транспорта, г. Москва

TrES-4b@yandex.ru

***Аннотация* — Автором статьи освещаются вопросы построения кодов с суммированием, предназначенных для кодирования данных, представленных в троичной логике. Приводится краткий обзор в области построения цифровых устройств троичной логики. Сформированы алгоритмы построения троичного кода с суммированием, модульного троичного кода с суммированием и модульно модифицированного троичного кода с суммированием. Освещены некоторые особенности, присущие рассматриваемым типам троичных кодов с суммированием. Использование представленных способов построения кодов с суммированием может оказаться полезным в перспективе, при реализации и использовании троичных цифровых устройств и систем.**

***Ключевые слова* — троичные цифровые устройства; троичный код с суммированием; обнаружение ошибок в информационных векторах; модульный код с суммированием; модульно модифицированный код с суммированием.**

I. Введение

Цифровые устройства троичной логики в настоящее время не имеют распространения и полностью остаются за пределами использования при построении систем управления в промышленности и в бытовой сфере [1]. Однако исследователями всего мира ведутся изыскания в области реализации троичной цифровой техники, методов построения элементарных электронных приборов, имеющих по три устойчивых состояния [2] – [4], методов построения устройств, функционирующих в троичной логике, на традиционных элементах с двумя устойчивыми состояниями [5] – [8], методов защиты троичных данных [9] – [11] и т. д.

Первая вычислительная машина, реализуемая на принципах троичной логики, была разработана в 1840 году Томасом Фаулером [12]. Спустя более чем век под руководством Николая Петровича Брусенцова была создана опытная отечественная электронная вычислительная машина «Сетунь», функционирующая в троичной сбалансированной системе счисления (МГУ, 1958 год) [13]. Спустя год под руководством Брусенцова была реализована первая серийная электронная вычислительная машина «Сетунь». Несколько позже, в 1970 году, под его же руководством была реализована вторая машина «Сетунь-70» [14]. Известны и более современные реализации вычислительных устройств, функционирующих в троичной логике. К примеру, система TCA2 (версия v2.0), построенная в 2008 году Джозеффом Коннели, Кирагом Пателем и Антонио Чавезом при поддержке Филлипа Нико в Калифорнийском государственном политехническом университете [15]. Следует также отметить, что ряд специалистов высказывается в пользу использования троичной логики вместо двоичной логики при реализации квантовых компьютеров [16].

Несмотря на малую известность разработок в области троичной цифровой техники до сих пор исследователи всего мира не потеряли интереса к ней, к методам построения подобных устройств, обнаружения неисправностей в них и защиты передаваемых данных. Целым направлением исследований является разработка методов кодирования троичных данных [9] – [11], [17] – [19].

В данной работе автором освещаются некоторые способы построения простых троичных кодов, основанных на принципе суммирования значений информационных разрядов, а также способы их модификации. Подобные коды с суммированием, как и известные коды с суммированием, реализуемые в бинарной логике, могут впоследствии найти применение при построении узлов троичных цифровых систем, средств их технического диагностирования, а также при передаче данных между ними.

II. Способы построения троичных кодов с суммированием

A. *Троичные коды с суммированием*

В троичной логике используются три различных сигнала. Введем следующую систему обозначений: $a \in \{0,1,2\}$, где a – разряд троичного вектора. В данном случае речь идет не о конкретных числовых значениях, а только о разделении сигналов. Другими словами, в дальнейших рассуждениях нет привязки к конкретному виду троичной системы счисления, а использованы общие принципы. Таким образом, каждый разряд x_i, $i = \overline{1,m}$ информационного вектора длиной m, может принимать три различных значения.

Вопросам построения троичных кодов посвящена ни одна публикация. В ряде работ, например, в [20], [21], обсуждаются особенности простых кодов, каждый из кодовых векторов которых имеет одинаковое количество разрядов, равных 1 и 2. Такой код называется *кодом с постоянной композицией значений (constant-composition code)*. Такой код может эффективно использоваться при построении троичных цифровых систем, наделенных свойством обнаружения неисправностей. Рассмотрим еще один способ

построения простейшего кода, основанный на использовании операции суммирования разрядов, равных 1 и 2.

Алгоритм 1. *Правила определения значений разрядов контрольных векторов троичных кодов с суммированием:*

1. В информационном векторе длиной m определяется число разрядов, равных 1, и число разрядов, равных 2, – числа r_1 и r_2 соответственно.
2. Число r_1 представляется в троичном виде и записывается в $k_1 = \lceil \log_3(m+1) \rceil$ старших разрядах контрольного вектора (запись $\lceil ... \rceil$ обозначает целое сверху от вычисляемого значения).
3. Число r_2 представляется в троичном виде и записывается в $k_2 = \lceil \log_3(m+1) \rceil$ младших разрядах контрольного вектора.

Код, строящийся по алгоритму 1, обозначим через $\Sigma(m,k)$ код. Данный код будет аналогом известного бинарного кода с суммированием (кода Бергера) [22].

Количество разрядов в контрольных векторах $\Sigma(m,k)$ кода определяется как $k = k_1 + k_2 = 2\lceil \log_3(m+1) \rceil$. Это следует их того факта, что каждое из чисел $r_1, r_2 \in \{1,2,...,m\}$ представляется в троичной форме с использованием $\lceil \log_3(m+1) \rceil$ разрядов.

К примеру, получим значения разрядов контрольного вектора $\Sigma(m,k)$ кода для информационного вектора < 0121 2101 2111 1100 >. Длина информационного вектора m=16. Отсюда следует, что числа $k_1 = k_2 = \lceil \log_3 16 \rceil = 3$, а k=6. Числу r_1=9 соответствует троичный вектор < 100 >. Числу r_2=3 соответствует троичный вектор < 010 >. Таким образом, контрольный вектор $\Sigma(16,6)$ кода будет иметь вид < 100 010 >.

Строящийся по алгоритму 1 троичный код будет обладать следующей особенностью. Любому контрольному вектору данного кода будут соответствовать все информационные векторы, имеющие одинаковое число разрядов, равных 1 и 2. То есть одной композиции разрядов будет соответствовать один контрольный вектор. В табл. 1, например, приводятся представители 729 кодовых векторов $\Sigma(6,4)$ кода. Они приводятся в порядке возрастания значений десятичных эквивалентов контрольных векторов, разбитых на пары, соответствующие числам r_1 и r_2 (пары r_1-r_2).

Таблица 1

Представители кодовых векторов $\Sigma(6,4)$ кода

r_1-r_2	$x_6x_5x_4x_3$ x_2x_1	Количество информационных векторов	Количество возможных необнаруживаемых ошибок
00-00	000000	1	0
00-01	000002	6	30
00-02	000022	15	210
00-10	000222	20	380
00-11	002222	15	210
00-12	022222	6	30
00-20	222222	1	0
01-00	000001	6	30
01-01	000012	30	870
01-02	000122	60	3540
01-10	001222	60	3540
01-11	012222	30	870
01-12	122222	6	30
02-00	000011	15	210
02-01	000112	60	3540
02-02	001122	90	8010
02-10	011222	60	3540
02-11	112222	15	210
10-00	000111	20	380
10-01	001112	60	3540
10-02	011122	60	3540
10-10	111222	20	380
11-00	001111	15	210
11-01	011112	30	870
11-02	111122	15	210
12-00	011111	6	30
12-01	111112	6	30
20-00	111111	1	0
Общее количество		729	34440

Все информационные векторы $\Sigma(6,4)$ кода распределяются по 28 контрольным группам, соответствующим различным контрольным векторам, определяемым парой чисел (r_1-r_2). При этом количество информационных векторов, соответствующих различным контрольным группам, существенно отличается. Количество векторов, соответствующих паре (r_1-r_2) зависит от числа разрядов, равных 1 и 2 (рис. 1). Так как любая ошибка, возникающая в информационном векторе конкретной группы и переводящая его в информационный вектор этой же группы, не будет обнаружена $\Sigma(m,k)$ кодом, то и число таких ошибок будет определяться только количеством информационных векторов в каждой контрольной группе. Чем больше информационных векторов принадлежит одной и той же группе, тем большее возможное количество необнаруживаемых ошибок она

будет давать. Следует также отметить, что часть возможных контрольных векторов для $\Sigma(m,k)$ кода может никогда не формироваться (проанализируйте первый столбец таблицы 1). Указанная неравномерность распределения информационных векторов между всеми контрольными векторами определяет высокое общее число необнаруживаемых $\Sigma(m,k)$ кодом ошибок в информационных векторах. К примеру, рассматриваемым $\Sigma(6,4)$ кодом не обнаруживается 34440 ошибок из 530712 возможных, что составляет 6,489%. Если бы все информационные векторы были распределены равномерно между всеми возможными контрольными векторами, то это бы давало 5832 необнаруживаемые ошибки, что составляет 1,099% от общего возможного числа (то есть, в шесть раз меньше).

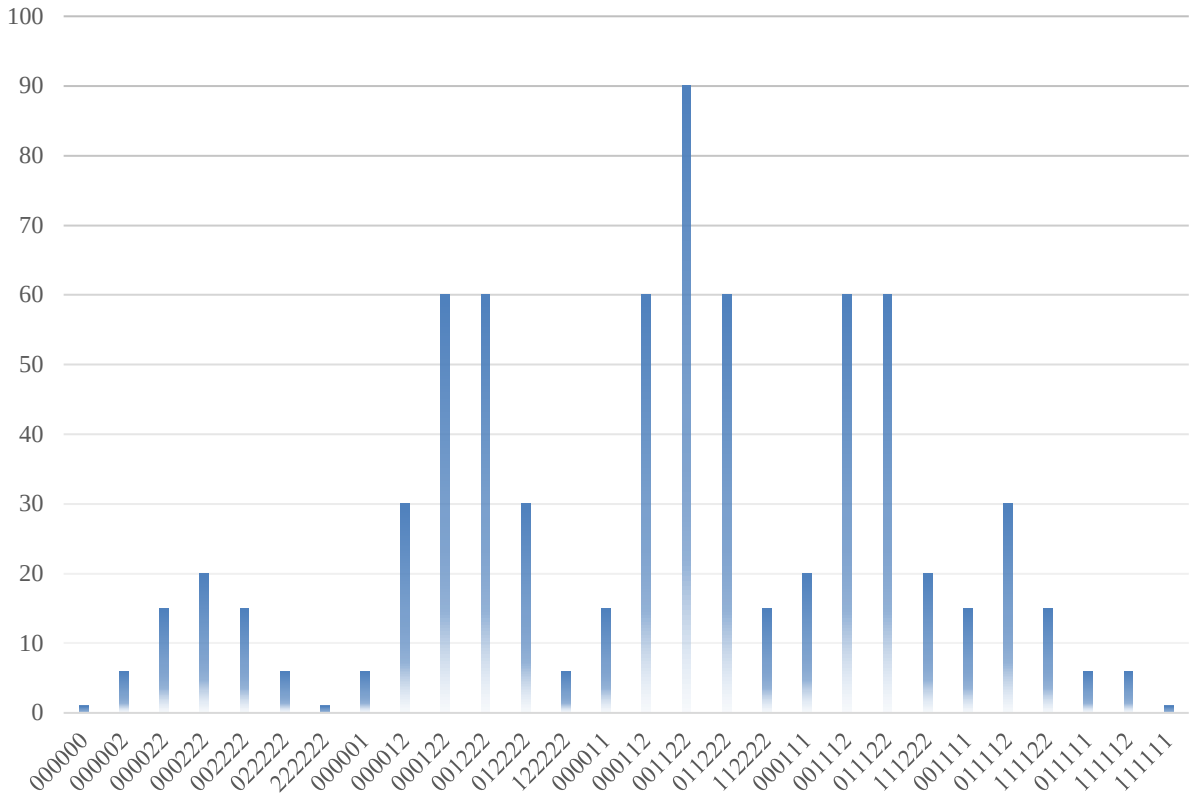

Рис. 1. Распределение информационных векторов на контрольные группы для $\Sigma(6,4)$ кода

B. Модульный троичный код с суммированием

Невозможность формирования всех вариантов контрольных векторов для $\Sigma(m,k)$ кодов влияет на особенности реализации их самопроверяемых кодеров [23]. С другой стороны, неравномерность распределения информационных векторов между всеми контрольными векторами для $\Sigma(m,k)$ кодов приводит к большому общему числу необнаруживаемых кодом ошибок. Заполнить все разряды контрольных векторов различными значениями при построении кода с суммированием по алгоритму 1 невозможно. Рассмотрим следующую модификацию $\Sigma(m,k)$ кода, для которой указанный недостаток ликвидирован.

Алгоритм 2. *Правила определения значений разрядов контрольных векторов модульных троичных кодов с суммированием:*

1. Устанавливаются значения модулей M_1 и M_2: $M_1, M_2 \in \{3^1, 3^2, \ldots, 3^{\lceil \log_3(m+1) \rceil - 1}\}$.
2. В информационном векторе длиной m определяется число разрядов, равных 1, и число разрядов, равных 2, – числа r_1 и r_2 соответственно.
3. Определяются значения наименьших неотрицательных вычетов чисел r_1 и r_2 по соответствующим модулям M_1 и M_2: числа $r_{M_1} = r_1 (\mathrm{mod}\, M_1)$ и $r_{M_2} = r_2 (\mathrm{mod}\, M_2)$.
4. Число r_{M_1} представляется в троичном виде и записывается в $k_1 = \lceil \log_3 M_1 \rceil$ старших разрядах контрольного вектора.
5. Число r_{M_2} представляется в троичном виде и записывается в $k_2 = \lceil \log_3 M_2 \rceil$ младших разрядах контрольного вектора.

Коды, строящиеся по алгоритму 2, будут являться аналогами известных в двоичной логике модульных (остаточных) кодов с суммированием [24]. Строго

говоря, значения обоих модулей могут быть выбраны из множества $M_1, M_2 \in \{3,4,...,m\}$. Однако выбор в качестве модуля числа, равного степени числа «три» позволяет строить коды с суммированием, у которых наиболее рационально используются разряды контрольных векторов (используются все варианты контрольных векторов, а не только их часть).

Обозначим модульные коды с суммированием как $\Sigma^{M_1/M_2}(m,k)$ коды. Количество контрольных разрядов в таких кодах определяется величиной $k = k_1 + k_2 = \lceil \log_3 M_1 \rceil + \lceil \log_3 M_2 \rceil$. Следует отметить, что если модуль является степенью числа 3, то полученный логарифм всегда будет целым числом, и определение целого сверху не потребуется.

Введем в рассмотрение отдельный класс модульных троичных кодов с суммированием – кодов с одинаковыми значениями модулей. Рассмотрение такого класса кодов целесообразно, так как значащие числа 1 и 2 в информационных векторах равноправны. Обозначим такие модульные коды отдельно как $\Sigma^{M}(m,k)$ коды. Количество контрольных разрядов в $\Sigma^{M}(m,k)$ коде равно $k = k_1 + k_2 = 2\lceil \log_3 M \rceil$.

Для рассмотренного ранее примера с информационным вектором < 0121 2101 2111 1100 > получим значения разрядов контрольного вектора $\Sigma^{9}(16,4)$ кода. Числа r_1=9 и r_2=3. Определяем числа $r_{M_1} = 9 \pmod 9 = 0$ и $r_{M_2} = 3 \pmod 9 = 3$. Первому вычету соответствует вектор < 00 >, а второму – < 10 >. Таким образом, контрольный вектор $\Sigma^{9}(16,4)$ кода будет иметь вид < 00 10 >.

Модульные коды имеют более равномерное распределение информационных векторов между всеми контрольными векторами, чем $\Sigma(m,k)$ коды. При этом, однако, контрольных векторов у $\Sigma^{M_1/M_2}(m,k)$ кодов меньше, чем у $\Sigma(m,k)$ кодов и, соответственно, общее число необнаруживаемых ошибок больше. Тем не менее, данный класс кодов с суммированием представляется весьма перспективным для использования при построении троичных цифровых устройств.

C. Модульно модифицированный троичный код с суммированием

Второй недостаток $\Sigma(m,k)$ кодов, связанный с большим общим числом необнаруживаемых в информационных векторах ошибок, ликвидируется при использовании следующего алгоритма построения кода с суммированием. Он является трансляцией принципов модификации классических двоичных кодов с суммированием в коды с улучшенными показателями обнаружения ошибок, описанные в [23].

Алгоритм 3. *Правила определения значений разрядов контрольных векторов модульно модифицированных троичных кодов с суммированием:*

1. Устанавливаются значения модулей M_1 и M_2: $M_1, M_2 \in \{3^1, 3^2, ..., 3^{\lceil \log_3 (m+1) \rceil - 1}\}$.
2. В информационном векторе длиной m определяется число разрядов, равных 1, и число разрядов, равных 2, – числа r_1 и r_2 соответственно.
3. Определяются значения наименьших неотрицательных вычетов чисел r_1 и r_2 по соответствующим модулям M_1 и M_2: числа $r_{M_1} = r_1 \pmod{M_1}$ и $r_{M_2} = r_2 \pmod{M_2}$.
4. Определяется значение поправочного коэффициента δ, равного сумме по модулю M=3 (операция далее обозначена символом $\oplus$) заранее установленных разрядов информационного вектора.
5. Вычисляется значение модифицированного веса информационного вектора:

$$W = r_{M_2} + r_{M_1} M_2 + \delta(M_1 + M_2). \qquad (1)$$

6. Число W представляется в троичном виде и записывается в разрядах контрольного вектора.

Обозначим модульно модифицированный троичный код с суммированием как $\Sigma_{\mu}^{M_1/M_2}(m,k)$ код.

Поясним смысл формулы (1). В ней слагаемое r_{M_2} определяет значения $k_2 = \lceil \log_3 M_2 \rceil$ младших разрядов контрольного вектора; слагаемое r_{M_1} – соответствует двоичному числу, равному r_{M_1} , но сдвинутому в контрольном векторе на M_2 разрядов влево относительно младших разрядов; $\delta(M_1 + M_2)$ – значение старшего контрольного разряда (он сдвинут на величину $(M_1 + M_2)$ относительно остальных разрядов). Строго говоря, алгоритм 3 может вместо пунктов 5 и 6 включать в себя пункт, предписывающий заполнить $k_2 = \lceil \log_3 M_2 \rceil$ младших разрядов контрольного вектора числом r_{M_2} , следующие $k_1 = \lceil \log_3 M_1 \rceil$ разрядов – числом r_{M_1} и старший разряд – числом δ.

Как и в случае с модульными троичными кодами с суммированием введем в рассмотрение такой класс модульно модифицированных кодов, для которых M_1=M_2, и обозначим их как $\Sigma_{\mu}^{M}(m,k)$ коды.

Количество разрядов в контрольных векторах $\Sigma_{\mu}^{M_1/M_2}(m,k)$ кода определяется как $k = 1 + k_1 + k_2 = 1 + \lceil \log_3 M_1 \rceil + \lceil \log_3 M_2 \rceil$. Если речь идет о $\Sigma_{\mu}^{M}(m,k)$ коде, то количество контрольных разрядов в нем $k = 1 + k_1 + k_2 = 1 + 2\lceil \log_3 M \rceil$.

Для рассмотренного ранее примера с информационным вектором < 0121 2101 2111 1100 > получим значения разрядов контрольного вектора $\Sigma_{\mu}^{9}(16,5)$ кода, для которого коэффициент δ определяется как сумма по мо-

дулю M=3 старших восьми разрядов. Числа r_1=9 и r_2=3. Определяем числа $r_{M_1} = 9 (\mathrm{mod}\, 9) = 0$ и $r_{M_2} = 3 (\mathrm{mod}\, 9) = 3$. Значение коэффициента $\delta = 1 \oplus 0 \oplus 1 \oplus 2 \oplus 1 \oplus 2 \oplus 1 \oplus 0 = 2$. Первому вычету соответствует вектор < 00 >, а второму – < 10 >. Таким образом, контрольный вектор $\Sigma_{\mu}^{9}(16,5)$ кода будет иметь вид < 2 00 10 >.

Отметим, что можно предложить несколько альтернативных алгоритмов построения модульно модифицированных троичных кодов с суммированием. Например, можно подсчитывать два поправочных коэффициента δ_1 и δ_2, определяющих суммы по модулям M=3 неравных подмножеств разрядов информационного вектора.

В табл. 2 приведены данные для числа контрольных разрядов для некоторых из рассмотренных кодов с суммированием, использование которых целесообразно при построении цифровых троичных устройств, наделенных свойством обнаружения неисправностей.

Таблица 2

Количество контрольных разрядов в троичных кодах с суммированием

m	$\Sigma(m,k)$	$\Sigma^{3^{\lceil \log_3(m+1) \rceil - 1}}(m,k)$	$\Sigma_{\mu}^{3^{\lceil \log_3(m+1) \rceil - 1}}(m,k)$	$\Sigma^{9}(m,k)$	$\Sigma_{\mu}^{9}(m,k)$
6	4	4	3	4	5
7	4	4	3	4	5
8	4	4	3	4	5
9	4	4	3	4	5
10	4	4	3	4	5
11	6	4	5	4	5
12	6	4	5	4	5
13	6	4	5	4	5
14	6	4	5	4	5
15	6	4	5	4	5
m	$2\lceil \log_3(m+1) \rceil$	$2\lceil \log_3(m+1) \rceil - 2$	$2\lceil \log_3(m+1) \rceil - 1$	4	5

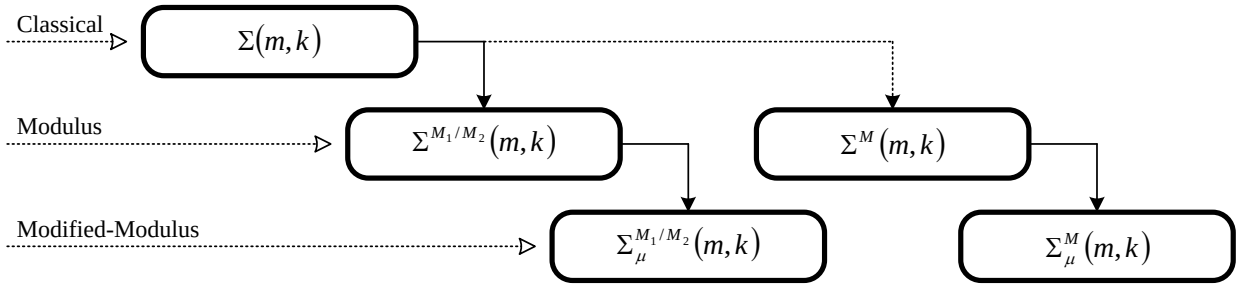

Рис. 2. Эволюция троичных кодов с суммированием

D. Многообразие троичных кодов с суммированием

На рис. 2 приведены этапы «эволюции» троичных кодов с суммированием с учетом ликвидации недостатков описанных выше свойств $\Sigma(m,k)$ кодов. Фактически при построении любого $\Sigma_{\mu}^{M_1/M_2}(m,k)$ кода первоначально получается вес, соответствующий модульному коду $\Sigma^{M_1/M_2}(m,k)$, а затем уже осуществляется корректировка значений суммарного веса информационного вектора.

III. Заключение

Представленные в данной работе способы построения троичных кодов с суммированием позволяют получать коды, имеющие относительно низкую избыточность, что определяет и возможности их эффективного применения при реализации контролепригодных троичных цифровых устройств и их диагностического обеспечения.

Описанные $\Sigma(m,k)$ коды обладают возможностью обнаружения любых ошибок, возникающих в информационных векторах кодовых слов, которые приводят к нарушению чисел r_1 и r_2 (композиции значений). Другими словами, данный код принадлежит к кодам с обнаружением любых некомпозиционных ошибок (нарушающих числа r_1 и r_2) в информационных векторах. Несмотря на отмеченное преимущество, $\Sigma(m,k)$ коды используют не все возможные контрольные векторы, а также имеют высокое общее количество необнаруживаемых ошибок (все эти ошибки принадлежат к типу композиционных ошибок). Первый недостаток ликвидируется при построении модульных кодов с суммированием ($\Sigma^{M_1/M_2}(m,k)$ кодов),

а второй – при построении модульно модифицированных кодов с суммированием ($\Sigma_{\mu}^{M_1/M_2}(m,k)$ кодов).

Троичные коды с суммированием – перспективный класс избыточных кодов, пригодных для их применения при синтезе цифровых устройств, функционирующих в троичной логике и наделенных особыми диагностическими свойствами: контролепригодными и самопроверяемыми структурами.

Литература

[1] Roy D., Merril Jr. Ternary Logic in Digital Computers // Proceedings of the SHARE design automation project (DAC '65), ACM New York, NY, USA, pp. 6.1-6.17, doi: 10.1145/800266.810759.

[2] Кушнеров А. Троичная цифровая техника. Ретроспектива и современность // Университет им. Бен-Гуриона, Беэр-Шева, Израиль. 28.10.05.

[3] Vudadha C., Katragadda S., Phaneendra P.S. 2:1 Multiplexer Based Design for Ternary Logic Circuits // IEEE Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics (PrimeAsia), 19-21 December 2013, Visakhapatnam, India, pp. 46-51, doi: 10.1109/PrimeAsia.2013.6731176.

[4] Kim S., Lim T., Kang S. An Optimal Gate Design for the Synthesis of Ternary Logic Circuits // 23rd Asia and South Pacific Design Automation Conference (ASP-DAC), 22-25 January 2018, Jeju, South Korea, pp. 476-481, doi: 10.1109/ASPDAC.2018.8297369.

[5] Куликов А. С. Троичные триггеры. – Электронный ресурс. Режим доступа: http://andserkul.narod.ru/troichnye_triggery.html, дата обращения: 05.04.2020 г.

[6] Hu M., Smith K.C. Self-Checking Binary Logic Systems Using Ternary Logic Circuits // Canadian Electrical Engineering Journal, 1984, Vol. 9, Issue 3, pp. 100-104, DOI: 10.1109/CEEJ.1984.6593793.

[7] Wu J. Ternary Logic Circuit for Error Detection and Error Correction // Proceedings of 19th International Symposium on Multiple-Valued Logic, 29-31 May 1989, Guangzhou, China, pp. 94-99, doi: 10.1109/ISMVL.1989.37766.

[8] Zhuang N., Wu H. Novel Ternary JKL Flip-Flop // Electronics Letters, 1990, Vol. 26, Issue 15, pp. 1145-1146, doi: 10.1049/el:19900741.

[9] Brouwer A.E., Hamalainen H.O., Ostergard P.R.J., Sloane N.J.A. Bounds on Mixed Binary/Ternary Codes // IEEE Transactions on Information Theory, 1988, vol. 44, Issue 1 pp. 140-161, doi: 10.1109/18.651001.

[10] Gulliver T.A., Ostergard P.R.J. Improved Bounds for Ternary Linear Codes of Dimension 7 // IEEE Transactions on Information Theory, 1997, Vol 43, Issue 4, pp. 1377-1381, doi: 10.1109/18.605613.

[11] Mirzaee R.F., Daliri M.S., Navi K., Bagherzadeh N. A Single Parity-Check Digit for One Trit Error Detection in Ternary Communication Systems: Gate-Level and Transistor-Level Designs // Journal of Multiple-Valued Logic and Soft Computing, 2017, 29 (3-4), pp. 303-326.

[12] The Thomas Fowler story by John McKay. – Электронный ресурс. Режим доступа: http://www.thomasfowler.org.uk/, дата обращения: 05.04.2020 г.

[13] Брусенцов Н.П., Маслов С.П., Розин В.П., Тишулина А.М. Малая цифровая вычислительная машина «Сетунь». – М.: Издательство МГУ, 1962, 140 с.

[14] Брусенцов Н.П., Маслов С.П., Рамиль Альварес Х. Микрокомпьютерная система обучения «Наставник». – М.: Наука, 1990, 223 с.

[15] Connely J. Ternary Computing Testbed 3-Trit Computer Architecture. – California Polytechnic State University of San Luis Obispo, August 29th, 2008, 184 p.

[16] Lanyon B.P., Barbieri M., Almeida M.P., Jennewein T., Ralph T.C., Resch K.J., Pryde G.J., O'Brien J.L., Gilchrist A., White A.G. Simplifying Quantum Logic Using Higher-Dimensional Hilbert Spaces // Nature Physics, 2009, Vol. 5, Issue 2, pp. 134-140, doi: 10.1038/nphys1150.

[17] Bitouze N., Graell i Amat A., Rosnes E. Error Correcting Coding for a Nonsymmetric Ternary Channel // IEEE Transactions on Information Theory, 2010, Vol. 56, Issue 11, pp. 5715-5729, doi: 10.1109/TIT.2010.2069211.

[18] Laaksonen A., Östergård P.R.J. New Lower Bounds on Error-Correcting Ternary, Quaternary and Quinary Codes // Lecture Notes in Computer Science 10495, Springer: Coding Theory and Applications, 5th International Castle Meeting, ICMCTA 2017, Vihula, Estonia, August 28-31, 2017, pp. 228-237.

[19] Efanov D.V. Ternary Parity Codes: Features // Proceedings of 17th IEEE East-West Design & Test Symposium (EWDTS`2019), Batumi, Georgia, September 13-16, 2019, pp. 315-319, doi: 10.1109/EWDTS.2019.8884414.

[20] Svanström M. A Lower Bound for Ternary Constant Weight Codes // IEEE Transactions on Information Theory, 1997, vol. 43, pp. 1630-1632.

[21] Svanström M., Östergård P.R.J., Bogdanova G.T. Bounds and Constructions for Ternary Constant-Composition Codes // IEEE Transactions on Information Theory, 2002, vol. 48, pp. 101-111.

[22] Berger J.M. A Note on Error Detecting Codes for Asymmetric Channels // Information and Control, 1961, vol. 4, issue 1, pp. 68-73, doi: 10.1016/S0019-9958(61)80037-5.

[23] Сапожников В.В., Сапожников Вл.В., Ефанов Д.В. Коды с суммированием для систем технического диагностирования. Том 1: Классические коды Бергера и их модификации. – М.: Наука, 2020, 383 с.

[24] Согомонян Е.С., Слабаков Е.В. Самопроверяемые устройства и отказоустойчивые системы. М.: Радио и связь, 1989, 208 с.

Ternary Sum Codes and their Modifications

D. V. Efanov, D. Sc.

Russian University of Transport, Moscow, TrES-4b@yandex.ru

***Abstract* — This article is devoted to the construction of ternary redundant codes designed to detect errors in the data bits of code words. The codes with this property can be used in the development of digital devices and systems with the checkable and self-checking structures. In the future, the use of ternary digital devices can create a new round in the development of engineering and technology, despite the fact that at present, ternary digital devices have not found application and are completely losing out to binary devices. The article provides a brief overview in the field of the**

construction of ternary digital devices and computing systems. The main results of the development of the ternary computers are listed, and the possibility of using ternary logic in the development of quantum computers is also noted. The article focuses on the development of the coding procedures and the protection of ternary data. The author offers the methods for constructing a range of sum codes with low redundancy and the ability to detect errors in data vectors. The article provides the descriptions of codes, the examples of their code words, and also notes some of their key features. The article presents the algorithms of sum codes modification, focused on a more efficient use of bits of the control vectors and on minimizing the number of errors that are not detected by codes. Using the presented methods for sum codes constructing can be useful in the future when implementing and using ternary digital devices and systems.

***Keywords* — ternary digital devices; ternary sum code; error detection in the information vectors; modular sum code; modified modulus sum code.**

REFERENCES

[1] Roy D., Merril Jr. Ternary Logic in Digital Computers // Proceedings of the SHARE design automation project (DAC '65), ACM New York, NY, USA, pp. 6.1-6.17, doi: 10.1145/800266.810759.

[2] Kushnerov A. Troichnaya cifrovaya tekhnika. Retrospektiva i sovremennost' (Ternary digital technology. Retrospective and modern) // University named after Ben gurion, Beer-Sheva, Israel, 28.10.05.

[3] Vudadha C., Katragadda S., Phaneendra P.S. 2:1 Multiplexer Based Design for Ternary Logic Circuits // IEEE Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics (PrimeAsia), 19-21 December 2013, Visakhapatnam, India, pp. 46-51, doi: 10.1109/PrimeAsia.2013.6731176.

[4] Kim S., Lim T., Kang S. An Optimal Gate Design for the Synthesis of Ternary Logic Circuits // 23rd Asia and South Pacific Design Automation Conference (ASP-DAC), 22-25 January 2018, Jeju, South Korea, pp. 476-481, doi: 10.1109/ASPDAC.2018.8297369.

[5] Kulikov A.S. Troichnye triggery (Ternary Triggers). – URL: http://andserkul.narod.ru/troichnye_triggery.html (access date: 05.04.2020).

[6] Hu M., Smith K.C. Self-Checking Binary Logic Systems Using Ternary Logic Circuits // Canadian Electrical Engineering Journal. – 1984. – Vol. 9. – Issue 3. – Pp. 100-104. – DOI: 10.1109/CEEJ.1984.6593793.

[7] Wu J. Ternary Logic Circuit for Error Detection and Error Correction // Proceedings of 19th International Symposium on Multiple-Valued Logic, 29-31 May 1989, Guangzhou, China, pp. 94-99, doi: 10.1109/ISMVL.1989.37766.

[8] Zhuang N., Wu H. Novel Ternary JKL Flip-Flop // Electronics Letters, 1990, Vol. 26, Issue 15, pp. 1145-1146, doi: 10.1049/el:19900741.

[9] Brouwer A.E., Hamalainen H.O., Ostergard P.R.J., Sloane N.J.A. Bounds on Mixed Binary/Ternary Codes // IEEE Transactions on Information Theory, 1988, vol. 44, Issue 1 pp. 140-161, doi: 10.1109/18.651001.

[10] Gulliver T.A., Ostergard P.R.J. Improved Bounds for Ternary Linear Codes of Dimension 7 // IEEE Transactions on Information Theory, 1997, Vol 43, Issue 4, pp. 1377-1381, doi: 10.1109/18.605613.

[11] Mirzaee R.F., Daliri M.S., Navi K., Bagherzadeh N. A Single Parity-Check Digit for One Trit Error Detection in Ternary Communication Systems: Gate-Level and Transistor-Level Designs // Journal of Multiple-Valued Logic and Soft Computing, 2017, 29 (3-4), pp. 303-326.

[12] The Thomas Fowler story by John McKay. – URL: http://www.thomasfowler.org.uk/ (access date: 05.04.2020).

[13] Brusencov N.P., Maslov S.P., Rozin V.P., Tishulina A.M. Malaya cifrovaya vychislitel'naya mashina «Setun'» (Setun Small Digital Computing Machine). – M.: Izdatel'stvo MGU, 1962, 140 p.

[14] Brusencov N.P., Maslov S.P., Ramil' Al'vares H. Mikrokomp'yuternaya sistema obucheniya «Nastavnik» (Microcomputer training system "Mentor). – M.: Nauka, 1990, 223 p.

[15] Connely J. Ternary Computing Testbed 3-Trit Computer Architecture. – California Polytechnic State University of San Luis Obispo, August 29th, 2008, 184 p.

[16] Lanyon B.P., Barbieri M., Almeida M.P., Jennewein T., Ralph T.C., Resch K.J., Pryde G.J., O'Brien J.L., Gilchrist A., White A.G. Simplifying Quantum Logic Using Higher-Dimensional Hilbert Spaces // Nature Physics, 2009, Vol. 5, Issue 2, pp. 134-140, doi: 10.1038/nphys1150.

[17] Bitouze N., Graell i Amat A., Rosnes E. Error Correcting Coding for a Nonsymmetric Ternary Channel // IEEE Transactions on Information Theory, 2010, Vol. 56, Issue 11, pp. 5715-5729, doi: 10.1109/TIT.2010.2069211.

[18] Laaksonen A., Östergård P.R.J. New Lower Bounds on Error-Correcting Ternary, Quaternary and Quinary Codes // Lecture Notes in Computer Science 10495, Springer: Coding Theory and Applications, 5th International Castle Meeting, ICMCTA 2017, Vihula, Estonia, August 28-31, 2017, pp. 228-237.

[19] Efanov D.V. Ternary Parity Codes: Features // Proceedings of 17th IEEE East-West Design & Test Symposium (EWDTS`2019), Batumi, Georgia, September 13-16, 2019, pp. 315-319, doi: 10.1109/EWDTS.2019.8884414.

[20] Svanström M. A Lower Bound for Ternary Constant Weight Codes // IEEE Transactions on Information Theory, 1997, vol. 43, pp. 1630-1632.

[21] Svanström M., Östergård P.R.J., Bogdanova G.T. Bounds and Constructions for Ternary Constant-Composition Codes // IEEE Transactions on Information Theory, 2002, vol. 48, pp. 101-111.

[22] Berger J.M. A Note on Error Detecting Codes for Asymmetric Channels // Information and Control, 1961, vol. 4, issue 1, pp. 68-73, doi: 10.1016/S0019-9958(61)80037-5.

[23] Sapozhnikov V.V., Sapozhnikov Vl.V., Efanov D.V. Kody s summirovaniem dlya sistem tekhnicheskogo diagnostirovaniya. Tom 1: Klassicheskie kody Bergera i ih modifikacii (Codes with summation for technical diagnostic systems. Volume 1: Classic Berger Codes and Their Modifications). – M.: Nauka, 2020, 383 p.

[24] Sogomonyan E.S., Slabakov E.V. Samoproveryaemye ustrojstva i otkazoustojchivye sistemy (Self-checking devices and fault-tolerant systems). – M.: Radio i svyaz', 1989, 208 p.