УДК 004.032.5:004.312.26 DOI: 10.31673/2412-4338.2025.018183

Yaroslav Toroshanko
State University of Information and Communication Technologies, Kyiv
ORCID 0000-0002-9053-7156

Oleksandr Toroshanko
Taras Shevchenko Kyiv National University, Kyiv
ORCID 0000-0002-2354-0187

Svitlana Kufterina
State University of Information and Communication Technologies, Kyiv
ORCID 0009-0004-1696-5699

Vladyslav Tkachuk
State University of Information and Communication Technologies, Kyiv
ORCID 0009-0007-0759-375X

SCALABLE MULTI-LEVEL CODER

Abstract: *The construction of a multi-level coder for converting a 2^n-bit unitary code into an n-bit positional binary code is considered. The coder contains $(n\text{-}1)$ levels of conversion. At each i-th level the corresponding i-th bit of the output code and the inputs for the next $(i\text{-}1)$-th level are formed, and the number of inputs is halved compared to the i-th level. At the 1-st level two lower bits of the output code are formed. The coder inputs are the inputs of the senior $(n\text{-}1)$-th level of conversion, at which an $(2^{n\text{-}1})$-bit unitary code for the next $(n\text{-}2)$-th level of conversion is formed. Thus, levels with numbers from 1-st to $(i\text{-}1)$-th represent an i-bit coder, the inputs of which are formed at the senior i-th level.*

An example of the coder constructing with a bitness of the output code equal to 4 and a bitness of the input unitary code equal to 16 using the proposed method is given. For this example the scaling coder scheme is given – constructing an coder with a bitness of the source code equal to 5 and a bitness of the input unitary code equal to 32. The proposed method of the scaling coder does not require changes in the scheme of the existing coder. The scheme that implements the scaling procedure is an additional n-th level of transformation and is connected only to the inputs of the existing coder. In the same way, scaling is provided for any bit depth of the coder output.

A comparative assessment of the complexity of the proposed solution for any bitness of the output code is performed. The purpose of the work is to reduce the hardware costs of constructing the coder, as well as to provide a simple acceptable method of scaling the existing coder.

Keywords: *multi-level coder, unitary code, positional code, scaling, Quine price, hardware costs, device complexity.*

Торошанко Ярослав Іванович
Державний університет інформаційно-комунікаційних технологій, Київ
ORCID 0000-0002-9053-7156

Торошанко Олександр Станіславович
Київський національний університет імені Тараса Шевченка, Київ
ORCID 0000-0002-2354-0187

Куфтеріна Світлана Ростиславівна
Державний університет інформаційно-комунікаційних технологій, Київ
ORCID 0009-0004-1696-5699

Ткачук Владислав Олександрович
Державний університет інформаційно-комунікаційних технологій, Київ
ORCID 0009-0007-0759-375X

МАСШТАБОВАНИЙ БАГАТОСТУПЕНЕВИЙ ШИФРАТОР

Анотація: *Розглянута побудова багатоступеневого шифратора для перетворення* 2^n *-розрядного унітарного коду в* n*-розрядний позиційний двійковий код. Шифратор містить* $(n\text{-}1)$ *ступінь перетворення, на кожному* i*-му ступені формується відповідний* i*-й розряд вихідного коду і входи для наступного (*i*-1)-го ступеня, причому кількість входів зменшується вдвоє у порівняння із* i*-м ступенем. На* 1*-му ступені формуються два молодших розряди вихідного коду. Входи шифратора являються входами старшого* $(n\text{-}1)$*-го ступеня перетворення, на якому формується* (2^{n-1})*-розрядний унітарний код для наступного* $(n\text{-}2)$*-го ступеня перетворення. Таким чином ступені з номерами від* 1*-го до (*i*-1)-го представляють собою* i*-розрядний шифратор, входи якого формуються на більш старшому* i*-му ступені.*

Наведений приклад побудови запропонованим способом шифратора з розрядністю вихідного коду рівною 4 і розрядністю вхідного унітарного коду рівною 16. Для цього прикладу наведена схема масштабування шифратора – побудова шифратора з розрядністю вихідного коду рівною 5 і розрядністю вхідного унітарного коду рівною 32. Запропонований спосіб масштабування шифратора не потребує змін у схемі існуючого шифратора. Схема, яка реалізує процедуру масштабування, представляє собою додатковий n*-й ступінь перетворення і підключається тільки до входів існуючого шифратора. Таким же чином забезпечується масштабування для будь-якої кількості розрядів вихідного коду шифратора.*

Виконана порівняльна оцінка складності запропонованого рішення для будь-якої кількості розрядів вихідного коду шифратора. Метою роботи є зменшення апаратних витрат на побудову шифратора, а також забезпечення простого прийнятного способу масштабування існуючого шифратора.

Ключові слова: *багатоступеневий шифратор, унітарний код, позиційний код, масштабування, ціна по Квайну, апаратні витрати, складність пристрою.*

1. Introduction. The coder function is to convert a unitary 2^n-bit binary code into an n-bit positional code. As a computing operational unit, the coder performs the decoder inverse function. Note that in a unitary code each number is represented by the code "1" in only one bit.

An example of the coders use is a computer keyboard, the lines from each key are the coder inputs. The unitary code from the keyboard keys is converted into a 7- or 8-bit code of the corresponding symbol.

Another example is the use of coders in priority interrupt schemes. The interrupt requests with the highest priority represented in the unitary code are converted into the positional code of the interrupt program number [1...4].

2. Problem statement. The large number of inputs and the significant redundancy of the unitary information coding used somewhat limited the possibilities of hardware implementation of coders (a significant number of inputs-outputs, overall dimensions, etc.). With the constant development of microelectronics technologies, the increasing degree of integration and microminiaturization of circuits and computer elements, such limitations do not play an important role in the use of hardware solutions of coders as part of large integrated circuits.

Therefore, the issues of improving such coder parameters as hardware costs, power consumption, reliability of operation, etc. are relevant and interest to specialists and developers of digital devices. Also important are requirements such as structure regularity, coders scalability (the possibility of the output bit number expansion, in terms of arguments number), etc.

The functional coder scheme has 2^n inputs $(x_{(0)}, x_{(1)}, \ldots, x_{(2^n\text{-}1)})$ and n outputs $(y_{(0)}, y_{(1)}, \ldots, y_{(n\text{-}1)})$, where 2^n is the input number bitness, n is the output number bitness, Fig. 1. Note that the numbering of the input and output bits starts with "0", the lowest bit is the bit with the index "0".

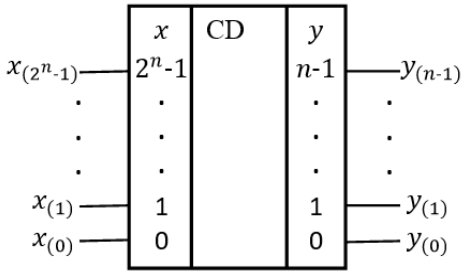

Fig. 1. Graphical coder representation

The coder inputs are a unitary binary code, that is, only one of the coder inputs $x_{(i)}$ is active. At the coder outputs y an n-bit binary number i is formed, which is equal to the number of the active the coder input. In scientific, technical and educational literature, as a rule, single-level coders based on logical elements "OR" are described [5-8]. Each j-th digit of the output binary number $y_{(j)}$ is formed as a disjunction of the coder inputs, the binary numbers of which in the corresponding j-th digit contain "1":

$$y_{(0)} = x_{(1)} \vee x_{(3)} \vee x_{(5)} \vee x_{(7)} \vee \ldots \vee x_{(2^n\text{-}1)}; \quad (1)$$

$$y_{(1)} = x_{(2)} \vee x_{(3)} \vee x_{(6)} \vee x_{(7)} \vee x_{(10)} \vee x_{(11)} \vee \ldots \vee x_{(2^n\text{-}2)} \vee x_{(2^n\text{-}1)}; \quad (2)$$

. . .

As follows from (1) - (3), when the output positional code bitness (number of bits) increases by "1", the number of inputs for each "OR" element doubles. If the coder bitness (bitness of the input and output number) grows, the hardware costs and, as a result, the power consumption of the device increase significantly.

3. Multi-level coder structure. The multi-level coder structure consists of ($(n\text{-}1)$ transformation levels [9]. Each i-th level $(i = \overline{1, n\text{-}1})$ implements two functions:

– formation of the i-th output bit;

– formation of inputs for the next (lower) $(i\text{-}1)$-th transformation level. The number of the $(i\text{-}1)$-th level inputs is halved compared to the previous i-th level. Note, at the 1-st level 1-st and 0-th coder outputs are formed.

The construction of a multi-level coder begins with the highest $(n\text{-}1)$-th level. On Fig. 2 a scheme of $(n\text{-}1)$-th and $(n\text{-}2)$-th (highest) of the coder levels are shown. The $(n\text{-}1)$-th level inputs are the coder inputs $x_{(0)}, \ldots, x_{(2^n\text{-}1)}$.

The formation of the highest $(n\text{-}1)$-th output bit is carried out by the block $A^{(n\text{-}1)}$, which is a $(2^{n\text{-}1})$-input logical element "OR". This logical element combines the highest half of the coder inputs, i.e. all inputs whose binary numbers contain "1" in the highest digit:

$$y_{(n\text{-}1)} = x_{(2^{n\text{-}1})} \vee x_{(2^{n\text{-}1}+1)} \vee \ldots \vee x_{(2^n\text{-}1)}.$$

The formation of input signals $x_{(0)}^{(n\text{-}2)}, \ldots, x_{(2^{n\text{-}1}\text{-}1)}^{(n\text{-}2)}$ for the next – $(n\text{-}2)$-th level is carried out by 2-input logical elements "OR" of block B: $B_{(0)}^{(n\text{-}1)}, \ldots, B_{(2^{n\text{-}1}\text{-}1)}^{(n\text{-}1)}$. Each such logical element "OR" combines the coder inputs, the binary numbers of which have different values only in the most significant $(n\text{-}1)$-th bit. Thus, the number of inputs of the next level is halved.

In the elements and signals designations of blocks A and B, the superscript indicates the number of the coder level, the subscript indicates the ordinal number of the element.

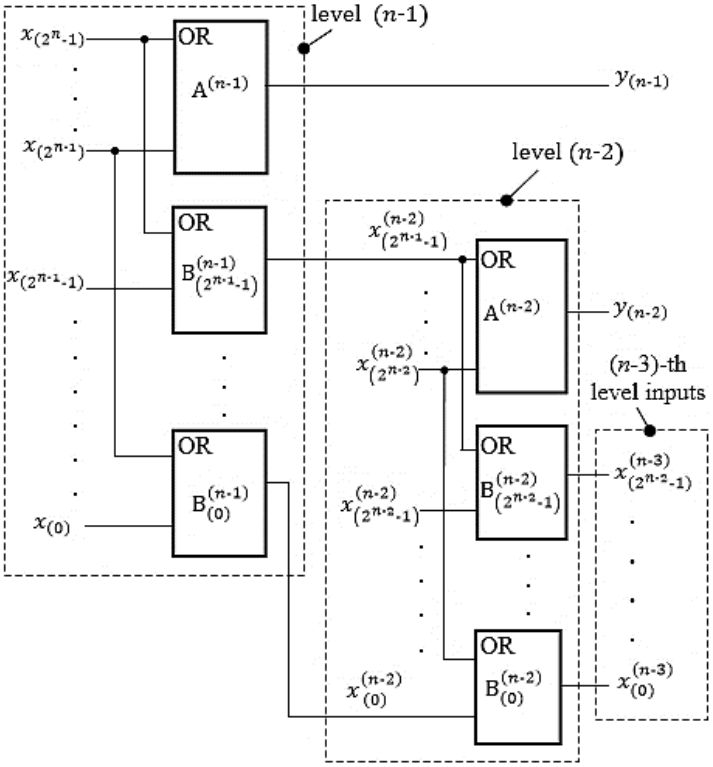

Fig. 2. Higher coder levels

Note: the superscript in parentheses indicates the coder level number; the subscript indicates the sequence number of the input, output, or circuit element

In other words, the lower $(n\text{-}1)$ digits of the $(n\text{-}1)$-th level represent the numbers of the inputs of the next – $(n\text{-}2)$-th level (inputs $x_{(0)}^{(n\text{-}2)},\ldots, x_{(2^{n\text{-}1}\text{-}1)}^{(n\text{-}2)}$).

The formation of the next $(n\text{-}2)$-th bit of the output code is carried out by the block $A^{(n\text{-}2)}$, which is a $(2^{n\text{-}2})$-input logical element "OR". This logical element combines the upper half of the inputs of the $(n\text{-}2)$-th coder level, that is, all the inputs of the $(n\text{-}2)$-th level, the binary numbers of which contain "1" in the upper bit:

$$y_{(n\text{-}2)} = x_{(2^{n\text{-}2})}^{(n\text{-}2)} \vee x_{(2^{n\text{-}2}+1)}^{(n\text{-}2)} \vee \ldots \vee x_{(2^{n\text{-}1}\text{-}1)}^{(n\text{-}2)}.$$

The formation of input signals $x_{(0)}^{(n\text{-}3)},\ldots, x_{(2^{n\text{-}2}\text{-}1)}^{(n\text{-}3)}$ for the next $(n\text{-}3)$-th level is carried out by 2-input logical elements "OR" of block B: $B_{(0)}^{(n\text{-}2)},\ldots, B_{(2^{n\text{-}2}\text{-}1)}^{(n\text{-}2)}$.

Fig. 3 shows the scheme of the coder 1-st level. The outputs of this stage x_0^0 and x_1^0 are designated by analogy with the previous levels. The output x_1^0 of the element $B_{(1)}^{(1)}$ is the 0-th bit of the coder output.

The logic element B_0^1 on Fig. 3, which is shown by analogy with the previous levels, is not used in the scheme of the 1-st stage. The value of the 0-th bit of the source code is formed by the logic element of the 1-st level B_0^1. Therefore, in each i-th level, the 2-input elements “OR”, which form the inputs with the number “0” for the younger $(i\text{-}1)$-th level, can be removed.

The generalized scheme of the i-th and $(i\text{-}1)$-th levels, taking into account the above, is shown in Fig. 4.

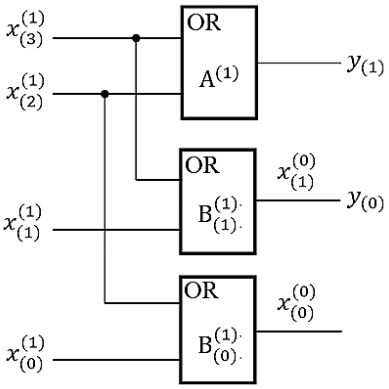

Fig. 3. The coder 1-st level

The inputs of the i-th level $x_{(1)}^{(i)}, \ldots, x_{(2^{i-1}-1)}^{(i)}$ are formed at the highest $(i+1)$-th level. The values of the i-th and $(i\text{-}1)$-th bits, the inputs for the $(i\text{-}1)$-th and $(i\text{-}2)$-th levels are formed in the same way as described when considering the highest $(n\text{-}1)$-th and $(n\text{-}2)$-th levels.

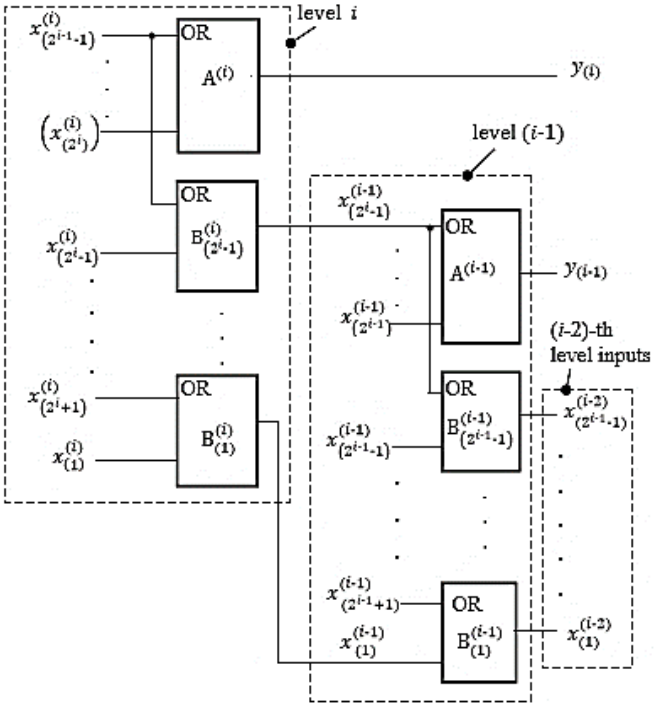

Fig. 4. The generalized scheme of the i-th and $(i\text{-}1)$-th coder levels

4. The example of a multi-level coder. Fig. 5 shows an example of the coder with 4 outputs – $y_{(0)}$, $y_{(1)}$, $y_{(2)}$, $y_{(3)}$, which has 3 levels of conversion. The coder inputs $x_{(1)}, x_{(2)}, \ldots, x_{(15)}$ are the inputs of the highest – 3-rd level, which is formed in the manner described above.

The formation of the highest 3-rd digit of the source code is carried out by the block $A^{(3)}$ – an 8-input logical OR element that implements the function

$$y_{(3)} = x_{(8)} \vee x_{(9)} \vee \ldots \vee x_{15)}.$$

The formation of input signals $x_{(1)}^{(2)}, x_{(2)}^{(2)}, \ldots, x_{(7)}^{(2)}$, …, x_((7))^((2)) for the next 2-nd level is carried out by 2-input logical elements "OR" $B_{(1)}^{(3)}, B_{(2)}^{(3)} \ldots, B_{(7)}^{(3)}$:

$$x_{(1)}^{(2)} = x_{(1)} \vee x_{(9)}; \quad x_{(2)}^{(2)} = x_{(2)} \vee x_{(10)}; \quad \ldots \quad x_{(7)}^{(2)} = x_{(7)} \vee x_{(15)}.$$

The 2-nd level contains a 4-input "OR" element – block $A^{(2)}$, to which the inputs $x_{(4)}^{(2)}$, $x_{(5)}^{(2)}$, $x_{(6)}^{(2)}$, $x_{(7)}^{(2)}$ are connected. The output of this element forms the 2-nd bit $y_{(2)}$of the coder output.

The inputs for the 1-st stage are formed by 2-input "OR" elements $B_{(1)}^{(2)}$, $B_{(2)}^{(2)}$, $B_{(3)}^{(2)}$:

$$x_{(1)}^{(1)} = x_{(1)}^{(2)} \vee x_{(5)}^{(21)}; \quad x_{(2)}^{(1)} = x_{(2)}^{(2)} \vee x_{(6)}^{(2)}; \quad x_{(3)}^{(1)} = x_{(3)}^{(2)} \vee x_{(7)}^{(2)}.$$

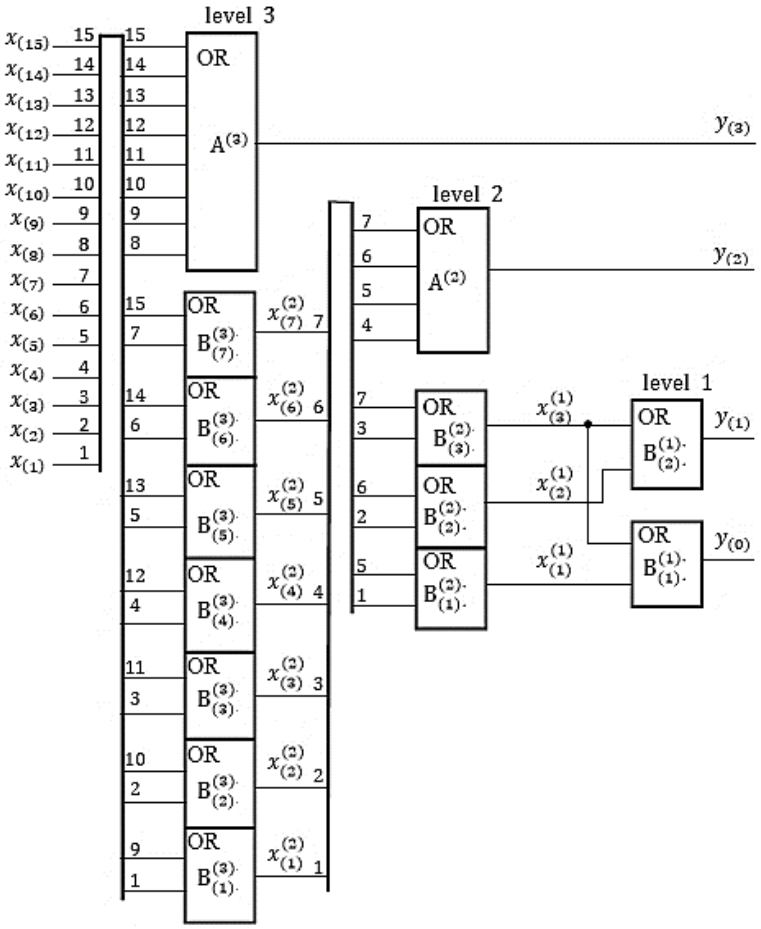

Fig. 5. Coder of 4-bit output number

In Fig. 3 the construction of the 1-st level is shown (excluding the element $B_{(0)}^{(1)}$).

5. The coders scaling. Scaling of technical systems is carried out in order to increase the productivity of the projects by introducing additional resources into the existing system. The main requirement for scaling is minimal execution time, preservation of functionality and the absence of significant changes in the existing system.

An example is a telecommunications network, in which scaling consists of connecting additional nodes and transmission channels without changing the previous topology, introducing new exchange protocols while maintaining or improving the quality of service and user interface, etc. [3, 10, 11].

Another example is the scaling of a storage device by the address and data bitness, which provides a significant increase in the speed of the computer system [2, 4, 12].

Scaling of the coder in order to increase the bit size of the input and output numbers is carried out by connecting additional level circuits to its inputs without any changes in the circuit of the already existing designed coder.

Thus, to construct a (2^{n+1})-input coder based on the existing 2^n-input coder (scaling by the bitness of the input and output numbers), an additional n-th level will be used, which is formed according to Fig. 2 (see section 3 of this article). This additional level contain a 2^n-input "OR"

element, at the output of which an additional output bit $y_{(n)}$ is formed, as well as 2^n 2-input "OR" elements, on which the inputs of the existing encoder are formed.

Fig. 6 shows a scheme of an additional 4-th level when scaling the coder of a 4-bit output number, which provides the construction of a 5-bit output coder.

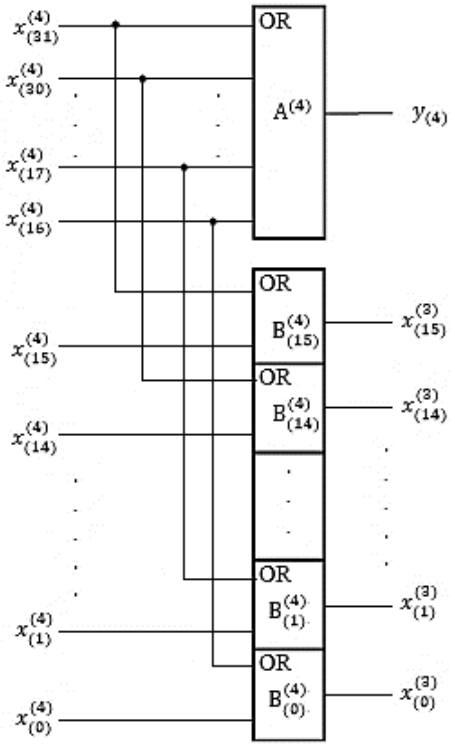

Fig. 6. Additional scaling level of the 4-bit coder

The additional bit $y_{(4)}$ is formed by the logical element "OR" of the block $A^{(4)}$. The inputs to the existing encoder are formed by 2-input elements "OR" $B^{(4)}_{(1)}, B^{(4)}_{(2)}, \dots, B^{(4)}_{(14)}, B^{(4)}_{(15)}$.

The proposed method is universal and can be used to scale coders regardless of the way they are construct.

However, the following should be noted. If the existing coder is constructed using the 0-th bit of the input unitary number, then the 2-input element "OR" $B^{(n)}_{(0)}$) must be taken into account in the additional level. For the given example of 4-bitness coder scaling it the 2-input element "OR" $B^{(4)}_{(0)}$. If the 0th bit of the input unitary number is not used, then the 2-input elements "OR" $B^{(n)}_{(0)}$and $B^{(4)}_{(0)}$ are not taken into account.

6. Comparative characteristics of the coder complexity. To assess the complexity of the considered coder schemes, we will use such a measurement parameter as the Quine price C – the total number of all logical elements inputs of a digital circuit built in the corresponding basis [4, 5, 13]. The considered in the article coders are constructed using logical elements "OR", that is, the basis is one logical element "OR". The Quine price C_1 of the described in the article multi-stage coder (see Fig. 2, 3, 4) is determined by the law of geometric progression [14], as the total price of all its levels, according to the formula

$$C_1 = {a_1(q^m\text{-}1)}/{q\text{-}1} = 4(2^{n\text{-}1}\text{-}1), \quad (4)$$

where $a_1 = 4$ is a first term of the progression (price of the 1st degree, see Fig. 3);

$m = n\text{-}1$ is a number of progression terms, transformation levels;

$q = 2$ is a denominator of a geometric progression.

Note that for the proposed coder scheme with 2-input elements $B_{(0)}^{(i)}$ removed in each stage (Fig. 4), the hardware costs (Quine price) is determined by the following expression:

$$C_2 = 4(2^{n\text{-}1}\text{-}1) - 2(n\text{-}1).$$

The second term in this expression $2(n\text{-}1)$ represents the Quine price of the 2-input elements $B_{(0)}^{(i)}$ removed in each level.

Let us make a comparative assessment of the hardware costs for constructing the known single-level and proposed multi-level coders. The Quine price of the known single-stage encoder constructed according to expressions (1)-(3) is defined as

$$C_3 = n \cdot 2^{n-1}. \quad (5)$$

From expressions (4) and (5) it follows that, in comparison with the single-level structure according to expression (1), (2), (3), the relative complexity of multi-level coders is significantly reduced when increasing the output bitness n:

$$\xi^{(n)} = {C_1^{(n)}}\Big/{C_3^{(n)}},$$

where $\xi^{(n)}$ is the relative complexity of the multi-level coder;

the superscript (n) indicates the coder source code bitness.

The comparative characteristics of both solutions are shown in the table below.

Table 1

The comparative characteristics of both solutions

(n)	$C_1^{(n)}$	$C_3^{(n)}$	$\xi^{(n)}$
2	4	4	1
3	12	12	1
4	28	32	0,875
5	60	80	0,75
6	124	192	0,65
7	252	448	0,56
8	508	1024	0,5
9	1020	2304	0,44
10	2044	5120	0,4
11	4092	11264	0,36
12	8188	24576	0,33

As shown in the table, the coefficient ξ of the reduction in hardware costs for building a multi-level coder decreases starting from the value of the source code bitness $n = 4$. With an increase in of the output code bitness the coefficient ξ constantly decreases.

7. Conclusions. The aim of the work is to reduce the hardware costs of coder construction, as well as to provide a simple and acceptable way to scale an existing, already designed coder. The proposed structure of a multi-level coder is characterized by a significant reduction in complexity (reduction in hardware costs for construction) compared to a single-level encoder. This in turn leads to a corresponding reduction in the power consumption of the device.

The coder contains $(n\text{-}1)$ transformation level, at each i-th level the corresponding i-th bit of the output code and inputs for the next $(i\text{-}1)$-th level are formed, and the number of inputs is halved compared to the i-th stage. Thus, the complexity of each subsequent level is halved compared to the previous one.

The levels with numbers from 1-st to (i-1)-th represent an (i)-bit coder, the inputs of which are formed at the higher (i)-th level. The proposed method of the coder scaling is to increase the inputs and outputs bitness. The circuit that implements the scaling procedure is an additional n-th level of conversion and is connected only to the existing coder inputs. The method does not require changes

in the circuit of the already designed existing coder. In the same way, scaling is provided for any input and output bits of the coder.

This ensures the structure regularity of the digital device, and also significantly simplifies the implementation of the coder expansion in terms of the number of input arguments and the bitness of the output word.

A comparative assessment of the complexity of the proposed solution for coders of any bitness is performed. It is shown that with an increase in the bitness, the indicators of reducing the coder complexity improve.

REFERENCES

1. Мельник А. О. Архітектура комп´ютера. – Луцьк: Волинська обласна друкарня, 2008. – 470 с.

2. Тарарака В. Д. Архітектура комп'ютерних систем: навчальний посібник. – Житомир: ЖДТУ, 2018. – 383 с. https://www.twirpx.com/file/2720671/.

3. Stallings William. Computer Organization and Architecture: 10th Ed. – Pearson Education, Inc., Hoboken, NJ, 2016. – 864 pp.

4. Tanenbaum A., Austin T. Structured Computer Organization, 6th ed. – Pearson Education, 2013. – 769 p.)

5. Дичка І. А., Легеза В. П., Онай М. В. Комп’ютерна логіка. Прикладна теорія цифрових автоматів: комп’ютерний практикум: навчальний посібник / – Київ : КПІ ім. Ігоря Сікорського, 2018. – 88 с.

6. Лахно В. А., Гусєв Б. С., Касаткін Д. Ю. Комп'ютерна логіка: навчальний посібник. – Київ, вид-во: КОМПРІНТ, 2018. – 422 с.

7. Матвієнко М. П. Комп’ютерна логіка. Навчальний посібник. – Київ: Видавництво Ліра-К, 2012. – 288 с.

8. Жабін В. І., Жуков І. А., Клименко І. А., Ткаченко В. В. Прикладна теорія цифрових автоматів : навчальний посібник. – Київ: НАУ-друк, 2009. – 360 с.

9. Авторське свідоцтво на винахід № 783786, МПК G 06 F 5/02. Шифратор. Автори: Бойчев О. Н., Корнійчук В. І., Сушко В. В., Тарасенко В. П., Торошанко Я. И. Заявник Київський політехнічний інститут. – 1980. – Бюл. № 44.

10. Bonaventure O. Computer Networking: Principles, Protocols and Practices. Release. – cnp3book, 2018. – 272 p.

11. Speidel J. Introduction to Digital Communications. Springer Nature Switzerland AG, 2019. – 329 p.

12. Карачка А. Ф., Струбицький П. Р., Дудко О. І. Архітектура комп’ютерів: навчальний посібник. – Тернопіль, 2006. – 152 с. https://www.twirpx.com/file/353857/.

13. Самофалов К. Г., Корнійчук В. І., Тарасенко В. П. Цифрові ЕВМ: Теорія і проектування: під заг. ред. К. Г. Самофалова. – Київ: Вища школа, 1989. – 424 с.

14. Ленюк О. М., Ленюк Ю. В. Короткий довідник з математики для підготовки до ЗНО та ДПА: навчально-практичний посібник. 2-е видання.– Чернівці, 2021. – 20 с.