НОВІ ЗАСОБИ КІБЕРНЕТИКИ, ІНФОРМАТИКИ, ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ ТА СИСТЕМНОГО АНАЛІЗУ

С.Л. КРЫВЫЙ, В.Н. ОПАНАСЕНКО, С.Б. ЗАВЬЯЛОВ

УДК 516.813

РАЗБИЕНИЕ МНОЖЕСТВА ВЕКТОРОВ С ЦЕЛЫМИ КООРДИНАТАМИ ЛОГИЧЕСКИМИ АППАРАТНЫМИ СРЕДСТВАМИ

Аннотация. Рассмотрена задача разбиения множества векторов с целыми координатами относительно покоординатного и лексикографического порядка на векторах с использованием автоматной интерпретации. Предложена аппаратная реализация операций трехзначной логики на базе кристаллов FPGA для проверки выполнимости формул этой логики.

Ключевые слова: целочисленные векторы, пороговые значения, конечные автоматы, трехзначная логика.

ВВЕДЕНИЕ

Синтез аппаратных средств для решения задачи разбиения множества булевых векторов и векторов с целочисленными неотрицательными координатами и порогового значения рассматривался в [1] (с использованием алгоритмов из работ [2, 3]). Такой подход известен как «технология реконфигурируемого компьютинга» [4, 5], и его воплощение в реальные проекты (см. [6–13]) стало возможным благодаря появлению программируемых логических интегральных схем (ПЛИС).

В настоящей статье рассматривается обобщение метода решения задачи разбиения множества векторов, который рассматривался в работе [1], и использование этого метода для проверки выполнимости формул трехзначной логики.

ПОСТАНОВКА ЗАДАЧИ

Даны конечное множество векторов $V = \{v_1, v_2, \ldots, v_m\}$ размерности $n \in N$ и фиксированный пороговый вектор $a = (c_1, c_2, \ldots, c_n)$, где v_i, $a \in Z^n$, Z — множество целых чисел. На множестве векторов задано отношение порядка. Наиболее часто таким отношением является покоординатный (частичный) порядок или лексикографический (полный) порядок.

Определение 1. Пусть $x = (x_1, \ldots, x_n) \in Z^n$, $y = (y_1, \ldots, y_n) \in Z^n$. Бинарное отношение $x \le y \Leftrightarrow \forall i = 1, \ldots, n,\ x_i \le y_i$ называется покоординатным порядком.

Бинарное отношение $x < y \Leftrightarrow (\forall i < j\ x_i = y_i) \wedge (x_j < y_j)$, $i, j = 1, \ldots, n$, называется лексикографическим порядком.

Считаем, что относительно покоординатного порядка $x < y$ хотя бы для одного $i = 1, \ldots, n$ имеем $x_i < y_i$.

Задача разбиения. Необходимо разбить множество V относительно порогового вектора a на подмножества $V_1 = \{v \in V \mid v < a\}$, $V_2 = \{v \in V \mid v > a\}$, $V_3 = \{v \in V \mid v = a\}$, $V_4 = \{v \in V \mid v \propto a\}$, где символ $\propto$ означает, что векторы не сравнимы

(для лексикографического порядка таких подмножеств будет три: V_1, V_2, V_3).

В настоящей статье рассматривается покоординатный порядок, поскольку решение задачи разбиения для лексикографического порядка очевидным образом следует из решения задачи разбиения для покоординатного порядка.

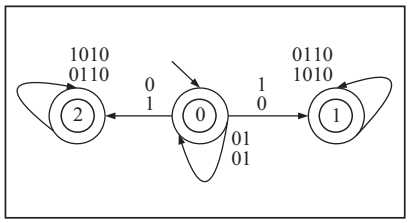

Рис. 1. Автомат A_1

АВТОМАТНАЯ ИНТЕРПРЕТАЦИЯ ЗАДАЧИ РАЗБИЕНИЯ

Случай целых неотрицательных координат векторов. Автоматный подход к решению задачи разбиения состоит в построении однородной сети из простых автоматов, по финальным состояниям которых определяется принадлежность вектора $v \in V$ к одному из подмножеств: V_1, V_2, V_3, V_4. Преимущество автоматного подхода состоит в том, что он позволяет выполнить разбиение множества V на подмножества V_1, V_2, V_3, V_4 одновременно для всех отношений $R \in \{<, >, =, \propto\}$. Автоматы, из которых строится сеть, являются автоматами без выходов [14, 15], а формальное определение сети автоматов имеет следующий вид.

Определение 2. Сетью автоматов называется n-ка $\boldsymbol{A} = (A_1, \ldots, A_n)$, состоящая из автоматов, представленных в виде $A_i = (S_i, X_i, f_i, a_0^i, F_i)$, $i = 1, 2, \ldots, n$. Состоянием сети $\boldsymbol{A}$ называется n-ка состояний $(a_1, \ldots, a_n)$, где $a_i \in S_i$ для каждого $i = 1, 2, \ldots, n$. Состояние сети $(a_1, \ldots, a_n)$ называется начальным, если $a_i = a_0^i$, и финальным или заключительным, если $a_i \in F_i$ для всех $i = 1, 2, \ldots, n$. Тогда n-ка $x = (x_1, \ldots, x_n)$, где $x_i \in X_i$, $i = 1, 2, \ldots, n$, называется действием в сети $\boldsymbol{A}$, которое переводит сеть из состояния $(a_1, \ldots, a_n)$ в состояние $(b_1, \ldots, b_n)$ так, что $(b_1, \ldots, b_n) = (f_1(a_1, x_1), \ldots, f_n(a_n, x_n))$.

Сеть назовем однородной, если все ее автоматы имеют один вид.

Для решения задачи классификации используется однородная сеть автоматов [14, 15], вид которых показан на рис. 1. Начальное состояние автомата нулевое, все состояния автомата заключительные, а входным алфавитом являются вектор-столбцы в двоичном алфавите, которые представляют соответствующие пары координат вектора v из V и порогового вектора a.

Поясним это представление на примере.

Пример 1. Пусть $v = (3, 4)$ и $a = (3, 2)$. Эти векторы представляются двоичными словами $v^{\mathrm{T}} = \begin{pmatrix} 3 \\ 4 \end{pmatrix} = \begin{bmatrix} 011 \\ 100 \end{bmatrix} = v_3 v_2 v_1$, где $v_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $v_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $v_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$; $a^{\mathrm{T}} = \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{bmatrix} 011 \\ 010 \end{bmatrix} = a_3 a_2 a_1$, где $a_3 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $a_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $a_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$. Таким образом, алфавит автомата A_1 составляют символы $x_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $x_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $x_3 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $x_4 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$. На вход автомата A_1 старшими разрядами подаются слова $p_1 = \begin{bmatrix} 011 \\ 011 \end{bmatrix} = x_1 x_4 x_4$, $p_2 = \begin{bmatrix} 100 \\ 010 \end{bmatrix} = x_3 x_2 x_1$, представляющие первые и вторые координаты векторов v и a соответственно.

Конец примера 1.

Определение множества, которому принадлежит вектор v, зависит от того, в каком финальном состоянии остановится сеть автоматов $\boldsymbol{A}$. В приведенном примере имеем вектор $v > a$ относительно покоординатного и лексикографического порядков и его следует отнести к подмножеству V_2.

Предложение 1. Автомат A_1 правильно вычисляет отношение $x \le y$ между двумя целыми положительными числами: x и y, представленными в двоичной системе счисления.

Доказательство очевидным образом следует из того факта, что отношение порядка $x \le y$ на векторах переносится на разряды их двоичного представления координат. Действительно, пусть $p_x = x_1 x_2 \dots x_k$ и $p_y = y_1 y_2 \dots y_k$ — двоичные слова, представляющие числа x и y соответственно. Поскольку двоичные слова подаются на вход автомата A_1 старшими разрядами, то $x < y$, если $x_j < y_j$ для некоторого $1 \le j \le k$, и $x_i = y_i$ для всех $i < j$. Это означает, что j-й разряд числа x меньше j-го разряда числа y, и тогда автомат из состояния 0 перейдет в состояние 2 (см. рис. 1). А это значит, что $x < y$. Остальные случаи аналогичны.

Конец доказательства.

Общее решение задачи разбиения множества V выполняется сетью автоматов $\boldsymbol{A} = (A_1, \dots, A_n)$ (случай n-мерных векторов из V). Настройка сети на решение задачи разбиения состоит в следующем: все автоматы сети одинаковые (n экземпляров автомата A_1 на рис. 1); сеть находится в начальном состоянии; слова подаются на вход автоматов сети старшими разрядами. На вход первого автомата сети подается двоичное слово, представляющее первые координаты векторов v и a, на вход второго автомата сети — слово, представляющее вторые координаты векторов и т.д. Принадлежность вектора v к подмножеству V_i, как отмечалось выше, зависит от финального состояния сети, в котором она оказалась после прочтения входных слов. Если отношение порядка покоординатное и финальным состоянием сети являются

а) $(0, 0, 0)$, то $v = a$, поэтому $v \in V_3$;

б) (p, q, r), где $p, q, r \in \{0, 1\}$ и не все значения p, q, r равны нулю, то $v > a$, поэтому $v \in V_2$;

в) (p, q, r), где $p, q, r \in \{0, 2\}$ и не все значения p, q, r равны нулю, то $v < a$, поэтому $v \in V_1$;

г) (1, 2, 0), (0, 1, 2), (2, 1, 0), (2, 1, 2). (2, 2, 1), …, то вектор v не сравним с a, поэтому $v \in V_4$.

Если отношение порядка лексикографическое и финальным состоянием сети является

а′) $(0, 0, 0)$, то $v = a$, поэтому $v \in V_3$;

б′) (p, q, r) и $p < q$ или $p = q$ и $q < r$, то $v > a$, поэтому $v \in V_2$;

в′) (p, q, r) и $p > q$ или $p = q$ и $q > r$, то $v < a$, поэтому $v \in V_1$.

Пример 2. Пусть $V = \{v_1 = (4, 5, 0), v_2 = (3, 4, 2), v_3 = (0, 1, 1), v_4 = (1, 1, 2), v_5 = (4, 3, 1)\}$ и пороговый вектор $a = (1, 1, 2)$. Тогда пара (v_1, a) имеет представление

$$v_1 = \begin{pmatrix} 4 \\ 5 \\ 0 \end{pmatrix} = \begin{bmatrix} 100 \\ 101 \\ 000 \end{bmatrix}, \quad a = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix} = \begin{bmatrix} 001 \\ 001 \\ 010 \end{bmatrix}$$

и слова, соответствующие первым координатам, имеют вид $p_1 = \begin{bmatrix} 100 \\ 001 \end{bmatrix}$, соответствующие вторым координатам — $p_2 = \begin{bmatrix} 101 \\ 001 \end{bmatrix}$, третьим координатам — $p_3 = \begin{bmatrix} 000 \\ 010 \end{bmatrix}$.

Слово p_1 подается на вход первого автомата сети, слово p_2 — на вход второго автомата сети и p_3 подается на вход третьего автомата. В результате получаем состояния (1, 1, 2), в которых остановились автоматы. Это значит, что векторы v_1 и a не сравнимы относительно покоординатного порядка, а относительно лексикографического порядка $v_1 > a$. Далее для покоординатного порядка получаем координаты

$$v_2 = \begin{bmatrix} 011 \\ 100 \\ 010 \end{bmatrix} \Rightarrow p_1 = \begin{bmatrix} 011 \\ 001 \end{bmatrix},\ p_2 = \begin{bmatrix} 100 \\ 001 \end{bmatrix},\ p_3 = \begin{bmatrix} 010 \\ 010 \end{bmatrix},$$

которые переводят сеть в состояния (1, 1, 0), что означает $v_2 > a$;

координаты

$$v_3 = \begin{bmatrix} 000 \\ 001 \\ 001 \end{bmatrix} \Rightarrow p_1 = \begin{bmatrix} 000 \\ 001 \end{bmatrix},\ p_2 = \begin{bmatrix} 001 \\ 001 \end{bmatrix},\ p_3 = \begin{bmatrix} 001 \\ 010 \end{bmatrix},$$

которые переводят сеть в состояния (2, 0, 2), что означает $v_3 < a$;

координаты

$$v_4 = \begin{bmatrix} 100 \\ 100 \\ 010 \end{bmatrix} \Rightarrow p_1 = \begin{bmatrix} 001 \\ 001 \end{bmatrix},\ p_2 = \begin{bmatrix} 001 \\ 001 \end{bmatrix},\ p_3 = \begin{bmatrix} 010 \\ 010 \end{bmatrix},$$

которые переводят сеть в состояние (0, 0, 0), что означает $v_4 = a$;

координаты

$$v_5 = \begin{bmatrix} 100 \\ 011 \\ 001 \end{bmatrix} \Rightarrow p_1 = \begin{bmatrix} 100 \\ 001 \end{bmatrix},\ p_2 = \begin{bmatrix} 011 \\ 001 \end{bmatrix},\ p_3 = \begin{bmatrix} 001 \\ 010 \end{bmatrix},$$

которые переводят сеть в состояние (1,1,2), это означает, что вектор v_5 не сравним с a.

Таким образом, $V = V_1 \cup V_2 \cup V_3 \cup V_4$, где $V_1 = \{v_3\}$, $V_2 = \{v_2\}$, $V_3 = \{v_4\}$, $V_4 = \{v_1, v_5\}$ — разбиение множества A относительно покоординатного порядка.

Конец примера 2.

Предложение 2. Однородная сеть автоматов $\boldsymbol{A} = (A_1, \ldots, A_n)$ правильно вычисляет отношение $x \le y$ между двумя векторами x и y с целыми положительными координатами, которые представлены в двоичной системе счисления.

Доказательство следует непосредственно из предложения 1 и пп. а) – г) для покоординатного отношения порядка, а для лексикографического отношения порядка — из пп. а′) – в′). Действительно, пусть $x < y$, тогда один из автоматов сети $\boldsymbol{A}$ согласно предложению 1 переходит в заключительное состояние 2 (см. рис. 1), из которого уже не выходит, а остальные автоматы либо остаются в состоянии 0 (равенство координат), либо переходят в состояние 2, где остаются до конца работы. Согласно п. в) для покоординатного порядка $v < a$. Для лексикографического отношения порядка доказательство следует из п. в′). Остальные случаи аналогичны.

Конец доказательства.

Случай целочисленных координат векторов. Рассмотрим приведенную выше постановку задачи, но координатами векторов из V здесь являются не натуральные, а целые числа. В этом случае задача будет решаться рассмотренным выше способом, если отрицательные координаты векторов представить в виде 2-дополнения. Отметим, что 2-дополнение числа x, двоичное представление которого $x_n x_{n-1} \ldots x_1 x_0$, определяется таким образом:

если $x \ge 0$, то $x_n = 0$ и $(x)_2 = 0 x_{n-1} \ldots x_1 x_0$,

если $x < 0$, то $x_n = 1$ и $(x)_2 = 1\,\overline{x_{n-1}} \ldots \overline{x_0} + 1$, где бит x_n является знаковым, а $\overline{x_i} = 1 - x_i$ для $0 \le i \le n-1$.

Автомат для вычисления 2-дополнения представляется композицией двух автоматов, первый из которых строит дополнение, а второй реализует операцию

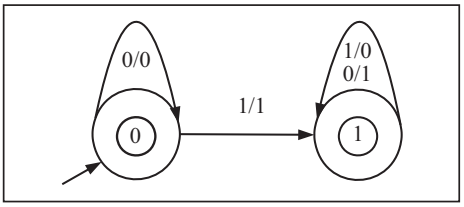

Рис. 2. Автомат C для вычисления 2-дополнения

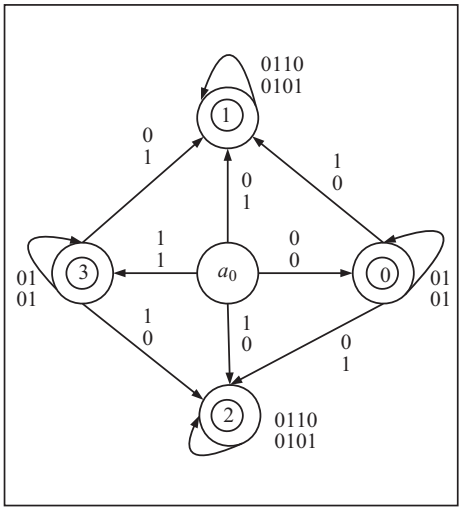

Рис. 3. Автомат B_1

прибавления единицы. Композиция этих автоматов представляется автоматом C, показанным на рис. 2.

На рис. 3 представлен автомат B_1; из таких автоматов строим однородную сеть $\boldsymbol{B}$. Состояние a_0 введено для распознавания знаков двоичных чисел, представляющих координаты векторов из V. Все состояния автомата B_1, кроме состояния a_0, заключительные. Если знаковые биты различны, то сразу можно определить, какой из векторов больше. Если знаковые биты одинаковые, то сравнение выполняется в случае положительных координат, как показано выше, а если биты отрицательные, то сравнение выполняется частью автомата, которая соответствует состоянию 3 на рис. 3.

Пример 3. Пусть $V = \{v_1 = (-4, 5, 0), v_2 = (3, 4, -2), v_3 = (0, -1, 1), v_4 = (1, 1, 2)\}$ и пороговый вектор $a = (-1, 1, 2)$. Тогда пара (v_1, a) имеет представление

$$v_1 = \begin{pmatrix} -4 \\ 5 \\ 0 \end{pmatrix} = \begin{bmatrix} 1100 \\ 0101 \\ 0000 \end{bmatrix}, \quad a = \begin{pmatrix} -1 \\ 1 \\ 2 \end{pmatrix} = \begin{bmatrix} 1111 \\ 0001 \\ 0010 \end{bmatrix}$$

и соответствующие координатам слова

$$p_1 = \begin{bmatrix} 1100 \\ 1111 \end{bmatrix}, \quad p_2 = \begin{bmatrix} 0101 \\ 0001 \end{bmatrix}, \quad p_3 = \begin{bmatrix} 0000 \\ 0010 \end{bmatrix},$$

а результат представляется состоянием (1, 1, 2). Это значит, что векторы v_1 и a не сравнимы.

Для v_2 и a имеем

$$v_2 = \begin{bmatrix} 0011 \\ 0100 \\ 1110 \end{bmatrix} \Rightarrow p_1 = \begin{bmatrix} 1100 \\ 1111 \end{bmatrix}, \quad p_2 = \begin{bmatrix} 0100 \\ 0001 \end{bmatrix}, \quad p_3 = \begin{bmatrix} 1110 \\ 0010 \end{bmatrix}$$

и результат — состояние (1, 1, 2); следовательно, v_2 и a также не сравнимы.

Для v_3 и a имеем

$$v_3 = \begin{bmatrix} 0000 \\ 1111 \\ 0001 \end{bmatrix} \Rightarrow p_1 = \begin{bmatrix} 0000 \\ 1111 \end{bmatrix}, \quad p_2 = \begin{bmatrix} 1111 \\ 0001 \end{bmatrix}, \quad p_3 = \begin{bmatrix} 0001 \\ 0010 \end{bmatrix}$$

и результат — состояние (1, 2, 1); это значит, что v_3 и a не сравнимы.

Для v_4 и a имеем

$$v_4 = \begin{bmatrix} 0001 \\ 0001 \\ 0010 \end{bmatrix} \Rightarrow p_1 = \begin{bmatrix} 0001 \\ 1111 \end{bmatrix}, \quad p_2 = \begin{bmatrix} 0001 \\ 0001 \end{bmatrix}, \quad p_3 = \begin{bmatrix} 0010 \\ 0010 \end{bmatrix}$$

и результат — это (1, 0, 0), что означает $a < v_4$.

Конец примера 3.

Предложение 3. Автомат B_1 правильно вычисляет отношение $x \le y$ между двумя целыми числами x и y, представленными в двоичной системе счисления в виде 2-дополнения.

Доказательство очевидным образом следует из предложения 1. Действительно, автомат B_1 при переходе из начального состояния a_0 распознает знак и переходит в состояние 0, если числа положительные, и далее работает как автомат A_1. Если числа отрицательные, то автомат B_1 переходит в состояние 3 и далее работает как автомат A_1, только переходы из состояния 3 в состояние 1 или 2 асимметричны переходам автомата A_1 из состояния 0 в состояние 1 или 2. Если знаки разные, то автомат B_1 переходит сразу либо в состояние 1 (случай $x > y$), либо в состояние 2 (случай $x < y$). Автомат B_1 в процессе сравнения, попадая в состояние 1 (или 2), из него уже не выходят. Это значит, что сравнение выполняется однозначно.

Конец доказательства.

Сравнение векторов размерности n выполняется однородной сетью из автоматов B_1.

Теорема 1. Сети ***A*** и ***B*** правильно выполняют разбиение множества векторов относительно покоординатного и лексикографического порядков и порогового вектора.

Доказательство следует из предложений 1–3.

Корректность функционирования однородной сети ***B*** для лексикографического порядка иллюстрируется также результатами тестирования, приведенными в табл. 1.

Таблица 1. Результаты сравнения векторов A и B, представленных в прямом и дополнительном кодах

Прямой двоичный код			**Дополнительный двоичный код**		
Двоичный вектор A_j	Двоичный вектор B_i	Результат сравнения	Двоичный вектор A_j	Двоичный вектор B_i	Результат сравнения
$A_1 = 000$ (+0)	$B_1 = 000$ (+0)	$A = B$ (Equal)	$A_1 = 000$ (+0)	$B_1 = 000$ (+0)	$A = B$ (Equal)
	$B_2 = 010$ (+2)	$B > A$ (More)		$B_2 = 010$ (+2)	$B > A$ (More)
	$B_3 = 111$ (–3)	$B < A$ (Less)		$B_3 = 101$ (–3)	$B < A$ (Less)
	$B_4 = 110$ (–2)	$B < A$ (Less)		$B_4 = 110$ (–2)	$B < A$ (Less)
	$B_5 = 101$ (–1)	$B < A$ (Less)		$B_5 = 111$ (–1)	$B < A$ (Less)
$A_2 = 001$ (+1)	$B_1 = 000$ (+0)	$B < A$ (Less)	$A_2 = 001$ (+1)	$B_1 = 000$ (+0)	$B < A$ (Less)
	$B_2 = 010$ (+2)	$B > A$ (More)		$B_2 = 010$ (+2)	$B > A$ (More)
	$B_3 = 111$ (–3)	$B < A$ (Less)		$B_3 = 101$ (–3)	$B < A$ (Less)
	$B_4 = 110$ (–2)	$B < A$ (Less)		$B_4 = 110$ (–2)	$B < A$ (Less)
	$B_5 = 101$ (–1)	$B < A$ (Less)		$B_5 = 111$ (–1)	$B < A$ (Less)
$A_3 = 011$ (+3)	$B_1 = 000$ (+0)	$B < A$ (Less)	$A_3 = 011$ (+3)	$B_1 = 000$ (+0)	$B < A$ (Less)
	$B_2 = 010$ (+2)	$B < A$ (Less)		$B_2 = 010$ (+2)	$B < A$ (Less)
	$B_3 = 111$ (–3)	$B < A$ (Less)		$B_3 = 101$ (–3)	$B < A$ (Less)
	$B_4 = 110$ (–2)	$B < A$ (Less)		$B_4 = 110$ (–2)	$B < A$ (Less)
	$B_5 = 101$ (–1)	$B < A$ (Less)		$B_5 = 111$ (–1)	$B < A$ (Less)
$A_4 = 110$ (–2)	$B_1 = 000$ (+0)	$B > A$ (More)	$A_4 = 110$ (–2)	$B_1 = 000$ (+0)	$B > A$ (More)
	$B_2 = 010$ (+2)	$B > A$ (More)		$B_2 = 010$ (+2)	$B > A$ (More)
	$B_3 = 111$ (–3)	$B < A$ (Less)		$B_3 = 101$ (–3)	$B < A$ (Less)
	$B_4 = 110$ (–2)	$A = B$ (Equal)		$B_4 = 110$ (–2)	$A = B$ (Equal)
	$B_5 = 101$ (–1)	$B > A$ (More)		$B_5 = 111$ (–1)	$B > A$ (More)

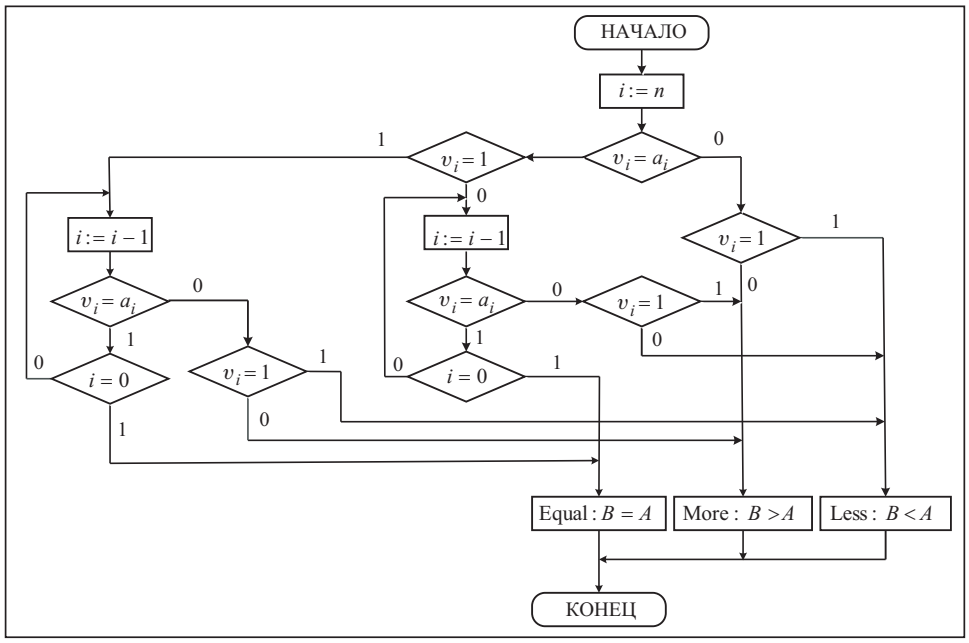

Рис. 4. Блок-схема алгоритма сравнения двоичных чисел со знаком

РЕАЛИЗАЦИЯ СРАВНЕНИЯ ДВУХ ВЕКТОРОВ НА ОСНОВЕ ЛОГИЧЕСКИХ СТРУКТУР

Поскольку решение задачи классификации выполняется однородной сетью автоматов, достаточно реализовать на основе логической структуры только операции сравнения двух векторов, т.е. только автоматов A_1 и B_1. Для произвольного вектора $v \in V$ размерности $n \in N$ и фиксированного порогового вектора a, где $v, a \in Z^n$, а Z — множество целых чисел, необходимо сформировать результаты сравнения: $v < a$ (Less); $v > a$ (More) и $v = a$ (Equal).

Пусть A и B ($A = a$, $B = v$) — двоичные векторы размерности n, представленные следующим образом: $A = a_n a_{n-1} a_{n-2} \ldots a_0$, $B = b_n b_{n-1} b_{n-2} \ldots b_0$, где a_n, b_n — знаковые разряды A и B. Блок-схема алгоритма сравнения двоичных векторов (в прямых кодах) представлена на рис. 4. Отсюда можно видеть результат сравнения двух векторов.

Пример 4. Пусть A и B — множества векторов, имеющих двоичное представление своих координат, т.е. $a = a_n a_{n-1} a_{n-2} \ldots a_0 \in A$, $b = b_n b_{n-1} b_{n-2} \ldots b_0 \in B$, где a_n, b_n — знаковые разряды a и b. Для случая $n = 3$ пусть

$$A = \{A_1 = 000;\ A_2 = 001;\ A_3 = 011;\ A_4 = 110\},$$

$$B = \{B_1 = 000;\ B_2 = 010;\ B_3 = 111;\ B_4 = 110;\ B_5 = 101\}.$$

Найти разбиение векторов из множества B относительно лексикографического порядка и соответствующих пороговых векторов из множества A для отношения $R \in \{<, >, =\}$.

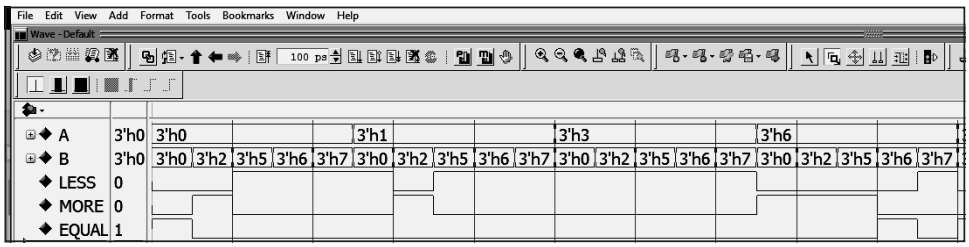

Рис. 5. Временная диаграмма алгоритма сравнения двоичных чисел со знаком в прямом коде

Рассмотрим теперь реализацию алгоритма сравнения (числа представлены в прямых кодах) на основе кристаллов FPGA с использованием САПР с последующим моделированием в среде ModelSim. Результаты моделирования (временная диаграмма, представленная на рис. 5) подтверждают правильность функционирования структуры сравнения чисел со знаком и имеют следующие обозначения:

$$A_1 = 000 = 3'h0,\ A_2 = 001 = 3'h1,\ A_3 = 011 = 3'h3,\ A_4 = 110 = 3'h6;$$

$$B_1 = 000 = 3'h0,\ B_2 = 010 = 3'h2,\ B_3 = 111 = 3'h7,\ B_4 = 110 = 3'h6,$$
$$B_5 = 101 = 3'h5.$$

Рассмотренный пример реализации алгоритма сравнения двухразрядных чисел со знаком по результатам моделирования выполняется в кристалле FPGA за единицы наносекунд. При расширении разрядности чисел время выполнения операции практически не изменяется, поскольку реализация предполагает параллельное поразрядное сравнение.

Конец примера 4.

РЕАЛИЗАЦИЯ ОПЕРАЦИЙ В ТРЕХЗНАЧНОЙ ЛОГИКЕ

Рассмотрим метод проверки выполнимости формул трехзначной логики Лукасевича [16], который применим также для трехзначной логики Клини и для трехзначной логики Бочвара, а также для любой k-значной логики. Возможность при-

Таблица 2. Значения истинности логических функций для троичной логики

$A = x$	$B = y$	**min** (x, y) **соответствует** $A \wedge B$	**max** (x, y) **соответствует** $A \vee B$	**min** $(1, 1 - x + y)$ **соответствует** $A \to B$	$(1 - x)$ **соответствует** $\overline{A}$
1	1	1	1	1	0
1	1/2	1/2	1	1/2	0
1	0	0	1	0	0
1/2	1	1/2	1	1	1/2
1/2	1/2	1/2	1/2	1	1/2
1/2	0	0	1/2	1/2	1/2
0	1	0	1	1	1
0	1/2	0	1/2	1	1
0	0	0	0	1	1

Таблица 3. Моделирование значений истинности логических функций для троичной логики

$A = x$	$B = y$	**min** (x, y) **соответствует** $A \wedge B$	**max** (x, y) **соответствует** $A \vee B$	**min** $(1, 1 - x + y)$ **соответствует** $A \to B$	$(1 - x)$ **соответствует** $\overline{A}$
10	10	10	10	10	00
10	01	01	10	01	00
10	00	00	10	00	00
01	10	01	10	10	01
01	01	01	01	10	01
01	00	00	01	01	01
00	10	00	10	10	10
00	01	00	01	10	10
00	00	00	00	10	10

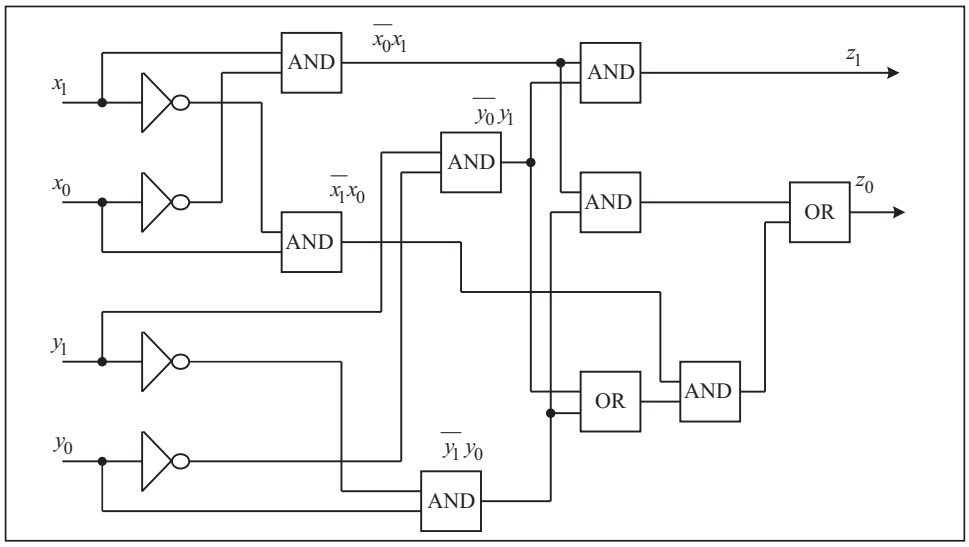

Рис. 6. Схема реализации операции $A \wedge B$

менения описанного метода строится на моделировании операций трехзначной логики с помощью двузначной логики. Представим таблицу операций трехзначной логики Лукасевича, в которой используются значения 1, 1/2, 0 (табл. 2), и таблицу моделирования этих операций двузначной логикой (табл. 3).

Для соответствия с двузначной логикой применяется такое кодирование значений трехзначной логики: $1 \to 10$, $1/2 \to 01$ и $0 \to 00$.

Поскольку для кодирования значений трехзначной логики используются два бита, то выходные значения также будут двухбитовыми. Пусть $x = x_1 x_0$, $y = y_1 y_0$, $z = z_1 z_0$ — битовое представление входных и выходных значений переменных. Построим алгебраические выражения для представленных логических операций с учетом принятой кодировки:

1) операция $A \wedge B$:

$$z_0 = \overline{x_1} x_0 (y_1 \overline{y_0} + \overline{y_1} y_0) + \overline{x_0} x_1 \overline{y_1} y_0, \quad z_1 = \overline{x_0} x_1 y_1 \overline{y_0},$$

схема реализации этой операции представлена на рис. 6;

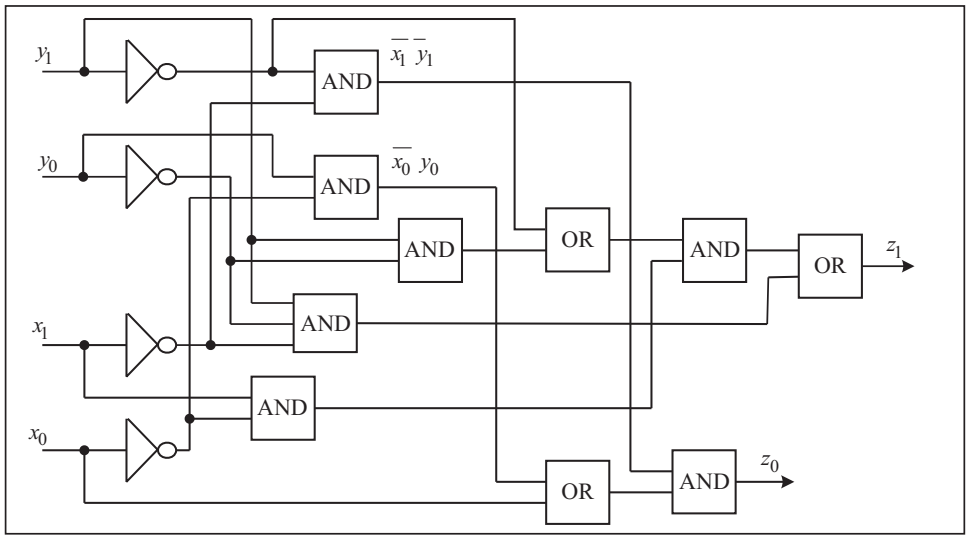

Рис. 7. Схема реализации операции $A \vee B$

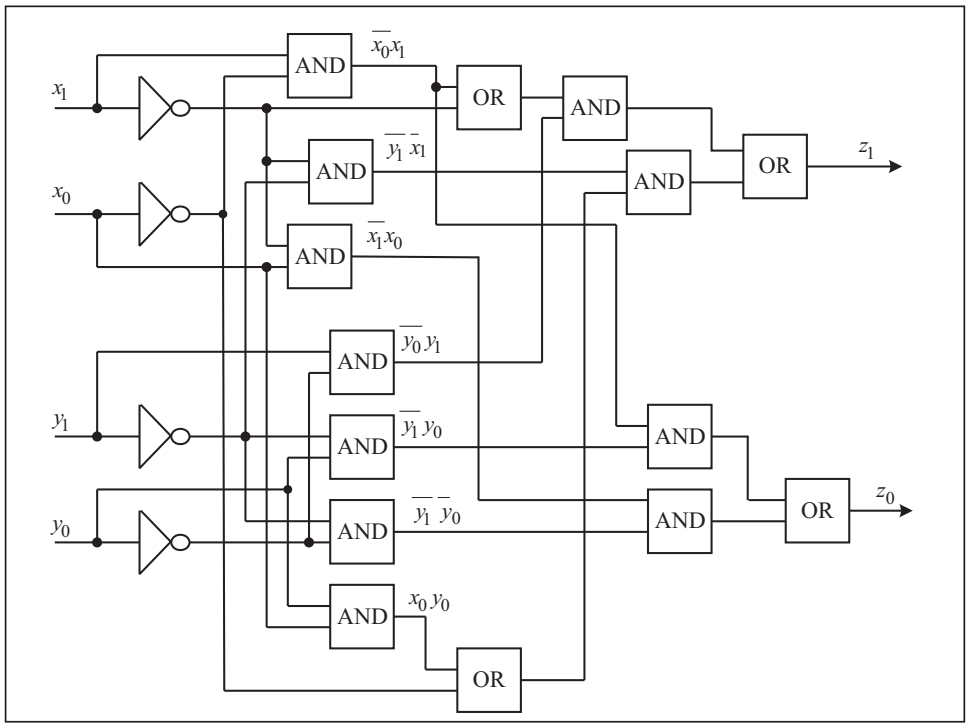

Рис. 8. Схема реализации операции $A \to B$

2) операция $A \vee B$:

$$z_0 = \overline{x_1}\,\overline{y_1}(x_0 + \overline{x_0}\,\overline{y_0}),\ z_1 = \overline{x_1}\,y_1\,\overline{y_0} + \overline{x_0}x_1(y_1\,\overline{y_0} + \overline{y_1}),$$

схема реализации этой операции представлена на рис. 7;

3) операция $A \to B$:

$$z_0 = \overline{y_1}(x_1\overline{x_0}\,y_0 + \overline{x_1}x_0\,\overline{y_0}),\ z_1 = y_1\,\overline{y_0}(\overline{x_1} + \overline{x_0}x_1) + \overline{x_1}\,\overline{y_1}(\overline{x_0} + x_0\,y_0)$$

схема реализации этой операции представлена на рис. 8;

4) операция $\overline{A}$:

$$z_0 = \overline{x_1}x_0,\ z_1 = \overline{x_0}\,\overline{x_1};$$

схема реализации этой операции представлена на рис. 9.

Рассмотрим теперь реализацию операций трехзначной логики (соответствует табл. 3) на основе кристаллов FPGA с использованием САПР с последующим моделированием в среде ModelSim. Результаты моделирования (временная диаграмма, представленная на рис. 10) подтверждают правильность функционирования структур трехзначной логики. Использованы следующие обозначения: Zand, Zor, AsB, invA соответствуют логическим операциям $A \wedge B$, $A \vee B$, $A \to B$, $\overline{A}$, а состояния $2'h0, 2'h1, 2'h2$ соответствуют значениям трехзначной логики в кодировке $0 \to 00,\ 1/2 \to 01,\ 1 \to 10$; $2'h3$ — запрещенное состояние.

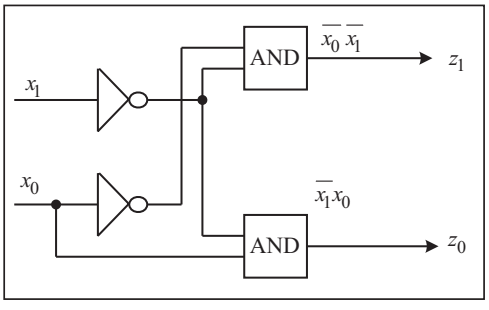

Рис. 9. Схема реализации операции $\overline{A}$

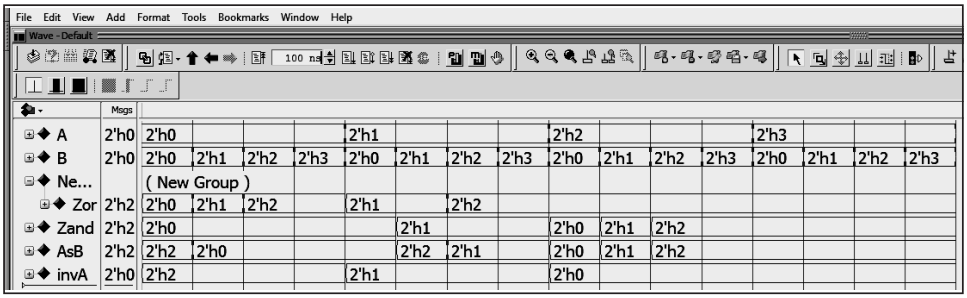

Рис. 10. Временная диаграмма функционирования операций трехзначной логики

Теорема 2. Приведенная реализация логических операций для трехзначной логики корректна, т.е. правильно вычисляются значения этих операций.

Доказательство сводится к рассмотрению правильности реализации каждой операции. Приведем доказательство только для случая операции конъюнкции и дизъюнкции, поскольку для остальных операций оно выполняется аналогично.

Операции $A \wedge B$ согласно таблице кодирования соответствует ДНФ вида $z_0 = \overline{x}_0 x_1 \overline{y}_1 y_0 + x_1 x_0 \overline{y}_1 y_0 + x_1 \overline{x}_0 \overline{y}_1 y_0$, которая преобразовывается к такому окончательному выражению: $z_0 = x_1 x_0 (\overline{y}_1 \overline{y}_0 + \overline{y}_1 y_0) + \overline{x}_0 x_1 \overline{y}_1 y_0$, а для $z_1 = \overline{x}_0 x_1 \overline{y}_1 \overline{y}_0$ никаких преобразований не требуется. Именно эти выражения реализуются схемой для $A \wedge B$.

Операции $A \vee B$ согласно таблице кодирования соответствует ДНФ вида $z_0 = \overline{x}_1 x_0 \overline{y}_0 \overline{y}_1 + x_1 \overline{x}_0 \overline{y}_0 \overline{y}_1 + \overline{y}_1 \overline{y}_0 \overline{x}_0 \overline{x}_1$, которая преобразовывается к такому окончательному выражению: $z_0 = \overline{x}_1 \overline{y}_1 (x_0 + \overline{x}_0 \overline{y}_0)$, а для z_1 ДНФ имеем вид $z_1 = \overline{x}_0 x_1 \overline{y}_1 \overline{y}_0 + x_0 x_1 \overline{y}_1 \overline{y}_0 + x_0 \overline{x}_1 \overline{y}_0 \overline{y}_1 + \overline{y}_1 \overline{y}_0 x_1 x_0 + \overline{y}_1 \overline{y}_0 \overline{x}_1 \overline{x}_0$, которая преобразовывается к такому окончательному выражению: $z_1 = \overline{x}_1 \overline{y}_1 \overline{y}_0 + \overline{x}_0 x_1 (\overline{y}_1 \overline{y}_0 + \overline{y}_1)$. Именно эти выражения реализуются схемой для $A \vee B$.

Конец доказательства.

Пример 5. Вычислим значение формулы трехзначной логики Лукасевича вида

$$F = ((\mathrm{A} \to \mathrm{B}) \vee (\mathrm{C} \wedge \mathrm{D})) \wedge \neg \mathrm{B}.$$

Для вычисления значения этой формулы синтезируется сеть по структуре формулы F. Синтезированная сеть принимает вид

$$S_1(S_2(S_3(\mathrm{A}, \mathrm{B}), S_1(\mathrm{C}, \mathrm{D})), S_4(\mathrm{B})),$$

где S_1, S_2, S_3, S_4 — схемы реализации функций конъюнкции, дизъюнкции, импликации и отрицания соответственно. Так, пусть заданы входные значения переменных: $\mathrm{A} = 1$, $\mathrm{B} = 0$, $\mathrm{C} = 1/2$, $\mathrm{D} = 0$, тогда на выходе получаем такое значение формулы F:

$$S_3(10, 00) = 00, \; S_1(01, 00) = 00, \; S_4(00) = 10,$$

$$S_1(S_2(S_3(10, 00), S_1(01, 00)), S_4(00)) = S_1(S_2(00, 00), S_4(00)) = S_1(00, 10) = 00.$$

Таким образом, значение формулы равно нулю, что соответствует значению, вычисленному по табл. 2. Для значений $\mathrm{A} = 1$, $\mathrm{B} = 1$, $\mathrm{C} = 1/2$, $\mathrm{D} = 1/2$ получаем следующее значение формулы F:

$$S_3(10, 10) = 10, \; S_1(01, 01) = 01, \; S_4(10) = 00,$$

$$S_1(S_2(S_3(10, 10), S_1(01, 01)), S_4(10)) = S_1(S_2(10, 01), S_4(10)) = S_1(10, 00) = 00.$$

Конец примера 5.

Из примера 5 следует, что данный подход позволяет по структуре заданной формулы трехзначной логики синтезировать сеть, с помощью которой можно проверять выполнимость данной формулы для значений входных переменных.

ЗАКЛЮЧЕНИЕ

В настоящей статье рассмотрено решение задачи разбиения множества векторов с целыми координатами относительно покоординатного и лексикографического порядков на векторах. При этом использовалась автоматная интерпретация задачи разбиения, которая решает задачу в общем виде. Приведенное решение не зависит от разрядности координат векторов и применимо к произвольным величинам координат векторов и пороговым значениям. Таким образом, обобщаются способы решения задач разбиения, которые рассматривались в работе [1]. Корректность предложенного решения доказана и подтверждена аппаратной реализацией на кристаллах FPGA и соответствующим моделированием с получением временных диаграмм. Эти методы применяются для проверки выполнимости формул трехзначной логики. Соответствующая логическая сеть синтезируется по структуре формулы.

СПИСОК ЛИТЕРАТУРЫ

1. Kryvyi S.L., Opanasenko V.N. Partitioning a set of vectors with nonnegative integer coordinates using logical hardware. *Cybernetics and Systems Analysis*. 2018, Vol. 54, N 2. P. 310–319.
2. Krivoi S. A criteria of compatibility systems of linear diophantine constraints. *Lecture Notes in Computer Science.* 2002. Vol. 2328. P. 264–271.
3. Kryvyi S.L. Algorithms for solving systems of linear Diophantine equations in residue fields. C*ybernetics and Systems Analysis*. 2007. Vol. 43, N 2. P. 3–17.
4. Palagin A.V., Opanasenko V.N. Reconfigurable computing technology. *Cybernetics and Systems Analysis.* 2007. Vol. 43, N 5. P. 675–686.
5. Palagin A.V., Opanasenko V.N. Design and application of the PLD-based reconfigurable devices. In: *Design of Digital Systems and Devices. Lecture Notes in Electrical Engineering.* Adamski M., Barkalov A., Wegrzyn M. (Eds.). Berlin; Heidelberg: Springer-Verlag, 2011. Vol. 79. P. 59–91.
6. Opanasenko V., Kryvyi S. Synthesis of multilevel structures with multiple outputs. *CEUR Workshop Proceeding of 10th International Conference of Programming, UkrPROG 2016*, Kyiv, Ukraine. 2016. Vol. 1631, Code 122904. P. 32–37.
7. Kondratenko Y.P., Gordienko E. Implementation of the neural networks for adaptive control system on FPGA. In: *Annals of DAAAM for 2012 & Proceeding of 23rd DAAAM International Symposium on Intelligent Manufacturing and Automation.* Katalinic B. (Ed.). Vienna, Austria, 2012. Vol. 23, N 1. P. 0389–0392.
8. Kondratenko Y., Kondratenko V. Soft computing algorithm for arithmetic multiplication of fuzzy sets based on universal analytic models. In: *Information and Communication Technologies in Education, Research, and Industrial Application*. *Ser. Communications in Computer and Information Science.* 2014. Vol. 469. P. 49–77.
9. Palagin A.V., Opanasenko V.N., Kryvyi S.L. Resource and energy optimization oriented development of FPGA-based adaptive logical networks for classification problem. In: *Green IT Engineering: Components, Networks and Systems Implementation*. Kharchenko V., Kondratenko Y., Kacprzyk J. (Eds.). 2017. Vol. 105. P. 195–218. DOI: DOI 10.1007 978-3-319-55595-9_10.
10. Opanasenko V.N., Kryvyi S.L. Synthesis of neural-like networks on the basis of conversion of cyclic hamming codes. *Cybernetics and Systems Analysis*. 2017. Vol. 53, N 4. P. 627–635.

11. Palagin A., Opanasenko V. The implementation of extended arithmetic's on FPGA-based structures. *Proceedings of the 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS'2017*. (21–23 September 2017, Bucharest, Romania). 2017. Vol. 2. P. 1014–1019.
12. Drozd J., Drozd A., Antoshchuk S., Kushnerov A., Nikul V. Effectiveness of matrix and pipeline FPGA-based arithmetic components of safety-related systems. *Proceedings of the 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*. Warsaw, Poland, 2015. P. 785–789.
13. Woods R., McAllister J., Lightbody G., Ying Yi. FPGA-based implementation of signal processing systems. Chichester: John Wiley and Sons, 2008. 362 p.
14. Глушков В.М., Летичевский А.А., Годлевский А.Б. Методы синтеза дискретных моделей биологических систем. Киев: Вища шк., 1983. 262 с.
15. Anderson J. Automata theory with modern applications. Cambridge: Cambridge University Press, 2006. 255 p.
16. Лукасевич Я. Аристотелевская силлогистика с точки зрения современной формальной логики. Москва: Иностранная литература, 1959. 312 с.

Надійшла до редакції 12.09.2018

С.Л. Кривий, В.М. Опанасенко, С.Б. Зав'ялов
РОЗБИТТЯ МНОЖИНИ ВЕКТОРІВ З ЦІЛИМИ КООРДИНАТАМИ ЛОГІКОВИМИ АПАРАТНИМИ ЗАСОБАМИ

Анотація. Розглянуто задачу розбиття множини векторів з цілими координатами відносно покоординатного і лексикографічного порядку на векторах із використанням автоматної інтерпретації. Запропоновано апаратну реалізацію операцій тризначної логіки на базі кристалів FPGA для перевірки виконуваності формул цієї логіки.

Ключові слова: цілочислові вектори, порогові значення, скінченні автомати, тризначна логіка.

S.L. Kryvyi, V.M. Opanasenko, S.B. Zavyalov
PARTITIONING OF A SET OF VECTORS WITH INTEGER COORDINATES BY MEANS OF THE LOGICAL HARDWARE

Abstract. The problem of partitioning a set of vectors with integer coordinates with respect to the coordinate-wise and lexicographic order on vectors by using an automatic interpretation is considered. The FPGA-based hardware implementation of three-valued logic operations for feasibility verification of the formulas of this logic is proposed.

Keywords: vectors with integer items, threshold value, finite automata, three-valued logic.

Крывый Сергей Лукьянович,
доктор физ.-мат. наук, профессор, профессор Киевского национального университета имени Тараса Шевченко, Киев, e-mail: sl.krivoi@gmail.com.

Опанасенко Владимир Николаевич,
доктор техн. наук, профессор, ведущий научный сотрудник Института кибернетики им. В.М. Глушкова НАН Украины, Киев, e-mail: OpanasenkoVM@nas.gov.ua.

Завьялов Станислав Борисович,
кандидат техн. наук, директор ООО «Радионикс», Киев, e-mail: radionix13@gmail.com.