М. Ф. Бондаренко, В. И. Хаханов, Е. И. Литвинова, Структура логического ассоциативного мультипроцессора, *Автомат. и телемех.*, 2012, выпуск 10, 71–92

Автоматика и телемеханика, № 10, 2012

Системный анализ и исследование операций

© 2012 г. М.Ф. БОНДАРЕНКО, д-р. техн. наук,
В.И. ХАХАНОВ, д-р. техн. наук,
Е.И. ЛИТВИНОВА, д-р. техн. наук
(Харьковский национальный университет радиоэлектроники)

СТРУКТУРА ЛОГИЧЕСКОГО АССОЦИАТИВНОГО МУЛЬТИПРОЦЕССОРА

Предлагается инфраструктура обеспечения параллельного анализа табличных или матричных форм ассоциативных отношений для поиска, распознавания и принятия решений в n-мерном векторном логическом пространстве на основе использования предложенной архитектуры мультипроцессора. Рассматриваются векторно-логические процесс-модели актуальных прикладных задач, в том числе встроенное диагностирование и восстановление работоспособности компонентов цифровых систем на кристаллах, где качество решения оценивается введенной неарифметической метрикой взаимодействия булевых векторов.

1. Введение

Идея исследования – убрать из компьютера "тяжеловесную" арифметику и трансформировать освободившиеся ресурсы для создания инфраструктуры векторно-логических вычислений для быстрого поиска, распознавания и принятия решений при анализе информационного пространства с помощью примитивных операций: *and*, *or*, *not*, *xor*. Специализация компьютерного изделия, ориентированная на использование только логических операций, дает возможность существенно (x100) повысить быстродействие решения неарифметических задач. Исключение арифметических операций, использование параллелизма алгебры векторной логики, мультипроцессорность архитектуры создают эффективную инфраструктуру, которая объединяет математическую и технологическую культуру для решения прикладных задач. Рыночная привлекательность логического ассоциативного мультипроцессора (Logical Associative MultiProcessor – LAMP) определяется тысячами старых и новых логических по своей природе задач, которые в настоящее время решаются неэффективно на избыточных универсальных компьютерах с мощным арифметическим процессором. Вот некоторые актуальные для рынка информационных технологий проблемы: 1. Анализ, синтез и коррекция синтаксических и семантических языковых конструкций (реферирование, исправление ошибок, оценивание качества текстов). 2. Распознавание видео- и аудио-образов

на основе их представления векторными моделями существенных параметров в дискретном пространстве. 3. Сервисное обслуживание сложных технических изделий и восстановление работоспособности в процессе их функционирования. 4. Тестирование знаний и экспертное обслуживание объектов или субъектов для определения их валидности. 5. Идентификация объекта или процесса для принятия решения в условиях неопределенности. 6. Точный поиск заданной вектором параметров информации в Internet. 7. Целеуказание в истребителе или в автоматической системе посадки лайнера, работающей в реальном микросекундном диапазоне времени. 8. Разведение объектов во времени и в пространстве для диспетчерской службы аэропорта или оптимизация инфраструктуры городского транспорта в целях исключения коллизий. Практически все упомянутые задачи решаются в реальном масштабе времени, являются изоморфными по логической структуре процесс-моделей, использующих совокупность взаимосвязанных ассоциативных таблиц. Для их решения необходима специализированная аппаратная платформа (LAMP), ориентированная на параллельное выполнение процедур поиска, распознавания и принятия решений, оцениваемых путем использования неарифметического критерия качества.

Цель исследования – существенное повышение быстродействия процедур поиска, распознавания и принятия решений путем мультипроцессорной и параллельной реализации ассоциативно-логических векторных операций для анализа графовых и табличных структур данных в векторном логическом пространстве без использования арифметических операций.

Для достижения поставленной цели необходимо разработать: 1) неарифметическую метрику оценивания векторно-логических решений в кибернетическом пространстве, 2) структуры данных и процесс-модели решения актуальных задач, 3) архитектуру логического ассоциативного мультипроцессора и показать пути его практического использования.

Объект исследования – инфраструктура процессов поиска, распознавания и принятия решений в векторно-логическом пространстве на основе использования алгебры векторной логики, вычислительной архитектуры анализа ассоциативно-логических структур данных и неарифметического интегрального критерия качества.

В процессе исследований использованы источники научно-технической информации: ассоциативно-логические структуры данных для решения информационных задач [1–5]; аппаратная платформа векторно-логического анализа информации [6–9]; модели и методы дискретного анализа и синтеза объектов киберпространства [10–15].

2. Метрика киберпространства для оценивания решения

Дискретное векторно-логическое пространство (киберпространство) – совокупность взаимодействующих по соответствующей метрике информационных процессов и явлений, описываемых векторами (кортежами) логических переменных и использующих в качестве носителя компьютерные системы и сети.

Метрика – способ измерения расстояния в пространстве между компонентами процессов или явлений, описанных векторами логических переменных. Расстояние в киберпространстве – это хог-отношение между парой векторов, обозначающих компоненты процесса или явления, что отличает его от кодового расстояния по Хэммингу. Расстояние, производная (булева), степень изменения, различия или близости есть изоморфные понятия, связанные с определением отношения двух компонентов процесса или явления. Понятие близости (расстояния) компонентов в киберпространстве есть мера их различия. Процедуры сравнения, измерения, оценивания, распознавания, тестирования, диагностирования, идентифицирования есть способ определения отношения при наличии не менее чем двух объектов.

Компонент пространства представлен k-мерным (двоичным) вектором $a = (a_1, \ldots, a_j, \ldots, a_k)$, $a_j \in \{0, 1\}$, где каждая его координата определена в двоичном алфавите, 0 – "ложь", 1 – "истина". Нуль-вектор есть k-мерный кортеж, все координаты которого равны нулю: $a_j = 0, j = \overline{1, k}$.

Метрика β кибернетического пространства определяется единственным равенством (1), которое формирует нуль-вектор для хог-суммы расстояний d_i между ненулевым и конечным числом точек (объектов), замкнутых в цикл:

$$\beta = \bigoplus_{i=1}^{n} d_i = 0, \tag{1}$$

где n – количество (целое число) расстояний между компонентами (векторами) пространства, составляющими цикл $D = (d_1, \ldots, d_i, \ldots, d_n)$, d_i – есть вектор расстояния, соответствующий ребру цикла, соединяющему два компонента (вектора) a, b пространства, который далее обозначается без индекса как $d(a, b)$. Расстояние между двумя объектами (векторами) a и b есть производный вектор: $d(a, b) = (a_j \oplus b_j)_1^k$. Векторному значению расстояния соответствует норма – скалярное расстояние по Хэммингу между двумя векторами – как число единиц вектора $d(a, b)$. Иначе: метрика β векторного логического двоичного пространства есть равная нуль-вектору *xor*-сумма расстояний между конечным числом точек (вершин) графа, образующих цикл. Сумма n-мерных двоичных векторов, задающих координаты точек циклической фигуры, равна нуль-вектору. Свертка пространства в нуль-вектор представляет интерес для многих практических задач, включая: диагностирование и исправление ошибок при передаче информации по каналам связи; поиск дефектов в цифровых изделиях на основе двузначных и многозначных таблиц неисправностей. Кроме того, на основе введенной метрики можно дать более формальное определение киберпространства, которое является векторно-логическим, нормируемым β-метрикой, где *xor*-сумма расстояний между конечным числом точек цикла равна нуль-вектору. Определение метрики ставит во главу угла не элементы множества, но отношения, что позволяет сократить систему аксиом (тождественности, симметрии и транзитивного треугольного замыкания) с трех до одной и распространить ее действие на сколь угодно сложные конструкции n-мерного логического пространства. Классическое задание метрики для определения взаимодействия одной, двух и трех точек в векторном логическом пространстве является частным случа-

ем β-метрики при $i = 1, 2, 3$ соответственно:

$$M = \begin{cases} d_1 = 0 \leftrightarrow a = b; \\ d_1 \oplus d_2 = 0 \leftrightarrow d(a,b) = d(b,a); \\ d_1 \oplus d_2 \oplus d_3 = 0 \leftrightarrow d(a,b) \oplus d(b,c) = d(a,c). \end{cases} \tag{2}$$

Векторно-логический транзитивный треугольник (2) имеет полную аналогию численному измерению расстояния в метрическом M-пространстве, которое задается системой аксиом, определяющей взаимодействие одной, двух и трех точек в любом пространстве:

$$M = \begin{cases} d(a,b) = 0 \leftrightarrow a = b; \\ d(a,b) = d(b,a); \\ d(a,b) + d(b,c) \geq d(a,c). \end{cases} \tag{3}$$

Специфика аксиомы треугольника (3) метрического пространства заключается в численном (скалярном) сравнении расстояний трех объектов, когда интервальная неопределенность ответа – две стороны треугольника могут быть больше либо равны третьей – малопригодна для определения точной длины последней стороны. Векторно-логическое пространство устраняет данный недостаток, полностью исключает степень неопределенности в бинарном отношении детерминированных состояний процессов или явлений. В этом случае численная неопределенность третьей стороны треугольника в векторном логическом пространстве приобретает форму точного двоичного вектора, который характеризует расстояние между двумя объектами и вычисляется на основе знания расстояний двух других сторон треугольника: $d(a,b) \oplus d(b,c) = = d(a,c) \rightarrow d(a,b) \oplus d(b,c) \oplus d(a,c) = 0$.

Пример. Имеется пять точек в векторном пространстве: (000111, 111000, 101010, 010101, 110011). Замыкание этих точек в цикл дает следующие стороны-расстояния в пятиугольнике: (111111, 010010, 111111, 100110, 110100). Покоординатное сложение всех векторов дает результат (000000). Практическая значимость данного факта заключается в возможности восстановления любого расстояния в замкнутом цикле, если известны $(n-1)$ сторон фигуры. Для треугольника это означает восстановление третьей стороны по известным двум. Если же создать из треугольников замкнутое логическое пространство, то можно сэкономить 66% от объема данных, который формирует все расстояния в логическом пространстве.

Метрика β кибернетического многозначного векторно-логического пространства есть вектор, равный значению $\varnothing$ по всем координатам, полученный путем применения симметрической разности расстояний между конечным числом точек, образующих цикл:

$$\beta = \overset{n}{\underset{i=1}{\Delta}} d_i = \varnothing, \tag{4}$$

где каждая координата вектора, соответствующего объекту, определена в алфавите, составляющем булеан на универсуме примитивов мощностью p: $a_j = = \{\alpha_1, \ldots, \alpha_r, \ldots, \alpha_m\}, m = 2^p$.

Равенство пустому вектору симметрической разности покоординатного теоретико-множественного взаимодействия (4) подчеркивает равнозначность компонентов (расстояний), участвующих в формировании уравнения, где единственная координатная операция $d_{i,j}\Delta d_{i+1,j}$, используемая, например, в четырехзначной модели Кантора $A = \{0, 1, x, \varnothing\}$, $x = \{0, 1\}$, определяется соответствующей таблицей:

(5)

Δ	0	1	x	$\varnothing$
0	$\varnothing$	x	1	0
1	x	$\varnothing$	0	1
x	1	0	$\varnothing$	x
$\varnothing$	0	1	x	$\varnothing$

$\cap$	0	1	x	$\varnothing$
0	0	$\varnothing$	0	$\varnothing$
1	$\varnothing$	1	1	$\varnothing$
x	0	1	x	$\varnothing$
$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$

$\cup$	0	1	x	$\varnothing$
0	0	x	x	0
1	x	1	x	1
x	x	x	x	x
$\varnothing$	0	1	x	$\varnothing$

a	0	1	x	$\varnothing$
$\tilde{a}$	1	0	$\varnothing$	x

.

Здесь также приведены таблицы истинности для других базовых теоретико-множественных операций (пересечение, объединение, дополнение), далее используемых по тексту. Число примитивных символов, образующих замкнутый относительно теоретико-множественных координатных операций алфавит, может быть увеличено. При этом мощность алфавита (булеана) определяется выражением $m = 2^p$, где p – число примитивов. Для практического использования введенной метрики киберпространства далее предлагается доказательный переход от численной характеристики бинарного отношения объектов, объединяющей три скалярные оценки их взаимодействия, к чисто векторно-логическому критерию качества отношения двух объектов.

Для понимания последующего материала необходимо ввести некоторые допущения и определения. Входной вектор $m = (m_1, \ldots, m_j, \ldots, m_k)$, $m_j \in \in \{0, 1, x\}$ и анализируемый объект $A = (A_1, \ldots, A_j, \ldots, A_k)$, $A_j \in \{0, 1, x\}$, который также представлен вектором, имеют одинаковую размерность k. Степень принадлежности m-вектора к A обозначается как $\mu(m \in A)$. Существует 5 типов координатного теоретико-множественного Δ-взаимодействия двух векторов $m\Delta A$, определенных на рис. 1. Они формируют все примитивные варианты реакции обобщенной системы поиска, распознавания и принятия решения на входной вектор-запрос. В технологической отрасли знаний – тех-

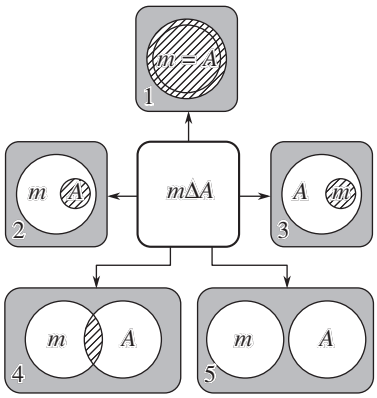


Рис. 1. Результаты взаимодействия двух векторов.

нической диагностике – упомянутая последовательность действий изоморфна маршруту: поиск дефектов, их распознавание, принятие решения на восстановление работоспособности. Данные стадии технологического маршрута нуждаются в метрике оценивания решений для выбора оптимального варианта.

Определение 1. Интегральная теоретико-множественная метрика для оценивания качества запроса есть функция взаимодействия многозначных по координатам векторов $m\Delta A$, которая определяется средней суммой трех параметров: кодовое расстояние $d(m, A)$, функция принадлежности $\mu(m \in A)$ и функция принадлежности $\mu(A \in m)$:

$$
\begin{gathered}
Q = \frac{1}{3}[d(m,A) + \mu(m \in A) + \mu(A \in m)], \\
d(m,A) = \frac{1}{n}[n - card[(i : m_i \cap A_i = \varnothing, i = 1, \ldots, k)]]; \\
\mu(m \in A) = 2^{c-a}; \mu(A \in m) = 2^{c-b}; \\
a = card[(i : A_i = x, i = 1, \ldots, k)]; \\
b = card[(i : m_i = x, i = 1, \ldots, k)]; \\
c = card[(i : m_i \cap A_i = x, i = 1, \ldots, k)].
\end{gathered} \tag{6}
$$

Пояснения. Пересечение (объединение) векторов – есть векторная операция, основанная на соответствующих координатных теоретико-множественных операциях. Операции координатного пересечения и объединения (6) определены в алфавите Кантора $A = \{0, 1, x = \{0, 1\}, \varnothing\}$. Нормирование параметров позволяет оценить уровень взаимодействия векторов в численном интервале $[0, 1]$. Если зафиксировано предельное максимальное значение каждого параметра, равное 1, то векторы равны между собой. Минимальная оценка, $Q = 0$, фиксируется в случае полного несовпадения векторов по всем n координатам. Если $m \cap A = m$ и мощность покоординатного пересечения равна половине мощности пространства вектора A, то функции принадлежности и качества равны:

$$
\mu(m \in A) = \frac{1}{2}; \quad \mu(A \in m) = 1; \quad d(m,A) = 1; \quad Q(m,A) = \frac{5}{2 \times 3} = \frac{5}{6}.
$$

Аналогичное значение будет иметь параметр Q, если $m \cap A = A$ и мощность покоординатного пересечения равна половине мощности пространства вектора m. Здесь пространство вектора есть функция от числа координат ω, равных x: $q = 2^{\omega}$. Если $card(m \cap A)$ – мощность покоординатного пересечения равна половине мощностей пространств векторов A и m, то функции принадлежности имеют значения:

$$
\begin{gathered}
\mu(m \in A) = \frac{1}{2}; \quad \mu(A \in m) = \frac{1}{2}; \quad d(m,A) = 1; \\
Q(m,A) = \frac{1}{3} \times \left(\frac{1}{2} + \frac{1}{2} + 1 \right) = \frac{2}{3}.
\end{gathered}
$$

Например, даны два вектора: $A = (\mathrm{XXX10})$ и $m = (\mathrm{XX0X0})$. Их пересечение равно $(\mathrm{XX010}) = \{00010,\ 01010,\ 10010,\ 11010\}$. Иначе, мощность результирующего пространства равна четырем двоичным векторам или половине мощностей исходных двоичных векторов.

Следует заметить, если пересечение двух векторов равно пустому множеству $\exists i(m_i \cap A_i) = \varnothing$, то количество общих точек (двоичных векторов) при пересечении двух пространств, формируемых двумя векторами, равно нулю.

Цель введения векторно-логического критерия качества решения заключается в существенном повышении быстродействия при подсчете оценки Q взаимодействия компонентов (векторов) m и A при анализе ассоциативных структур данных путем использования только векторных операций, выполняемых одновременно (параллельно) над всеми разрядами. С учетом изоморфизма теоретико-множественных и логических операций арифметический критерий (6) без усреднения функций принадлежности и кодового расстояния можно трансформировать к виду:

$$
\begin{gathered}
Q = d(m, A) + \mu(m \in A) + \mu(A \in m), \\
d(m, A) = card\left(\{i : m_i \oplus A_i = U, i = 1, \ldots, k\}\right); \\
\mu(m \in A) = card(\{i : A_i = U, i = 1, \ldots, k\}) - \\
- card\left(\{i : m_i \oplus A_i = U, i = 1, \ldots, k\}\right); \\
\mu(A \in m) = card(\{i : m_i = U, i = 1, \ldots, k\}) - \\
- card\left(\{i : m_i \oplus A_i = U, i = 1, \ldots, k\}\right); \\
U = \begin{cases} 1 \leftarrow \{m_i, A_i\} \in \{0, 1\}; \\ x \leftarrow \{m_i, A_i\} \in \{0, 1, x\}. \end{cases}
\end{gathered}
\tag{7}
$$

Если векторы m и A – двоичные по всем координатам, то переменная $U = 1$ и вычисления проводятся по правилам двоичной $\oplus$ – операции. Если векторы m и A определены в троичном алфавите, то переменная $U = x$ инициирует вычисления на основе использования теоретико-множественной операции симметрической разности Δ (7). Введение переменной U дает возможность уйти от написания двух формул критерия в зависимости от значности алфавита описания координат взаимодействующих векторов. Представленные в (7) векторные логические операции $(\wedge, \vee, \oplus, \neg)$ изоморфны теоретико-множественным $(\cap, \cup, \Delta, \sim)$. При этом теоретико-множественные координатные операции, соответствующие данным логическим, были определены ранее на многозначном алфавите Кантора в выражении (5). Первый компонент (7), составляющий критерий, формирует степень несовпадения k-мерных векторов – кодовое расстояние, путем выполнения операции xor, второй и третий определяют степени непринадлежности результата конъюнкции к числу единиц каждого из двух взаимодействующих векторов. Понятия принадлежности и непринадлежности являются взаимодополняющими, но в данном случае технологичнее вычислять непринадлежность. Следовательно, необходимый критерий качества равен нулю (по всем координатам), когда два вектора равны между собой. Оценка качества взаимодействия двух двоичных векторов ухудшается по мере возрастания критерия от нуля до единицы. Для того, чтобы окончательно исключить арифметические операции при подсчете

векторно-логического критерия качества, необходимо логически объединить три оценки (7) в одну:

$$Q = d(m, A) \vee \mu(m \in A) \vee \mu(A \in m) =$$
$$= (m \oplus A) \vee (A \wedge \overline{m \wedge A}) \vee (m \wedge \overline{m \wedge A}) =$$
$$= (m \oplus A) \vee [A \wedge (\bar{m} \vee \bar{A})] \vee [m \wedge (\bar{m} \vee \bar{A})] =$$
$$= (m \oplus A) \vee [(A \wedge \bar{m}) \vee (A \wedge \bar{A}) \vee (m \wedge \bar{m}) \vee (m \wedge \bar{A})] =$$
$$= [(A \wedge \bar{m}) \vee (m \wedge \bar{A})] \vee [(A \wedge \bar{m}) \vee (A \wedge \bar{A}) \vee (m \wedge \bar{m}) \vee (m \wedge \bar{A})] =$$
$$= (A \wedge \bar{m}) \vee (m \wedge \bar{A}) \vee (A \wedge \bar{m}) \vee (A \wedge \bar{A}) \vee (m \wedge \bar{m}) \vee (m \wedge \bar{A}) =$$
$$= m \oplus A.$$

Процедура вычисления векторного критерия качества зависит от значности алфавита:

$$Q' = \begin{cases} m \oplus A \leftarrow \{m_i, A_i\} \in \{0, 1\}; \\ m \Delta A \leftarrow \{m_i, A_i\} \in \{0, 1, x\}. \end{cases} \tag{8}$$

Для двоичного алфавита таблица истинности координатной *xor*-операции имеет вид:

$\oplus$	0	1
0	0	1
1	1	0

.

Во втором случае, когда алфавит описания координат имеет три значения, вычисление симметрической разности осуществляется в соответствии с Δ-операцией, представленной в (5).

Критерий качества Q однозначно определяет три формы взаимодействия двух любых объектов в n-мерном векторном логическом пространстве: расстояние и две функции принадлежности. При ненулевом расстоянии по Хэммингу функции принадлежности равны нулю, поскольку пространства двух векторов в данном случае не пересекаются. В противном случае – кодовое расстояние, равное нулю – взаимодействие объектов оценивается по функциям принадлежности. Увеличение числа нулей повышает критерий качества, а увеличение количества единиц обусловливает ухудшение качества взаимодействия по соответствующим булевым переменным. Критерий качества $Q = m \oplus A$ согласуется с введенной выше метрикой оценивания расстояния или взаимодействия объектов в векторно-логическом пространстве, а также имеет тривиальную вычислительную процедуру для оценивания решений, связанных с анализом и синтезом информационных объектов. В самом деле, векторное логическое пространство не должно иметь метрического расстояния и численных критериев качества, включающих арифметические операции на скалярных величинах.

Для сравнения критериев качества необходимо определять число единиц в каждом векторе без выполнения операций суммирования. Это можно сделать с помощью регистра сдвига [6] (рис. 2), который позволяет за один такт выполнить процедуру slc (shift left bit crowding) – сдвиг влево с одновременным уплотнением всех единичных координат n-разрядного двоичного вектора.

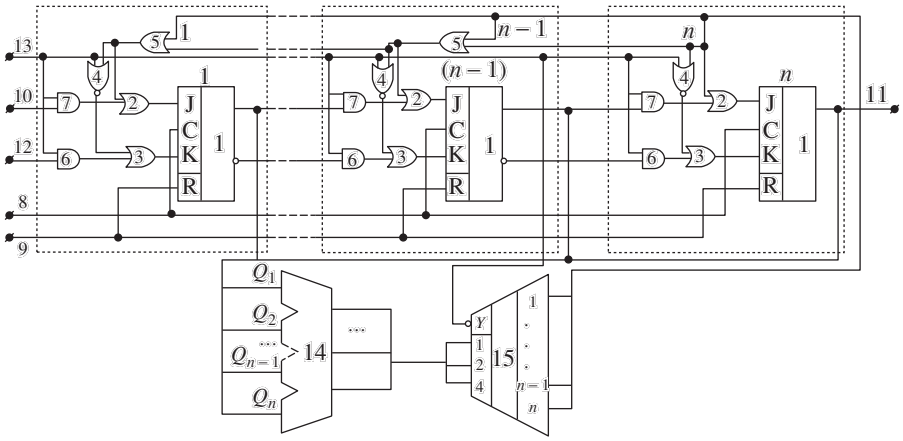


Рис. 2. Регистр сдвига и уплотнения единиц.

После процедуры сжатия номер правого единичного бита уплотненной серии единиц формирует значение критерия качества взаимодействия векторов. Но в данном случае такое число есть дань скалярной оценке бинарного отношения, которая нужна лишь человеку как информация для сравнения предлагаемой инфраструктуры относительно существующих технологий. В практических задачах такая оценка теряет свой смысл при выборе (квази-) оптимальной альтернативы. Поскольку векторная оценка удобней для вычислителя, который определяет лучшее решение без участия в данном процессе пользователя. Для наборов $m = (110011001100)$, $A = (000011110101)$ определение качества их взаимодействия по формулам (8) представлено в виде следующих процедур (нулевые координаты отмечены точками):

m	1	1	.	.	1	1	.	.	1	1	.	.
A	.	.	.	.	1	1	1	1	.	1	.	1
$Q^* = m \oplus A$	1	1	.	.	.	.	1	1	1	.	.	1
$Q(m, A)$	1	1	1	1	1	1						

.

Здесь сформирована оценка взаимодействия векторов и, что самое главное, единичные координаты строки $Q^* = d(m, A) \vee \mu(m \in A) \vee \mu(A \in m)$ идентифицируют все существенные переменные, по которым взаимодействие векторов не соответствует критерию качества. Процедура сжатия для получения $Q(m, A)$ не означает потерю информативной векторной оценки $Q^* = m \oplus A$. Результат сжатия позволяет лишь сделать выбор лучшего из двух или более решений путем сравнения суммарного числа единиц, формирующих скалярные оценки критериев. Где их меньше, там решение лучше.

Как выбрать лучшее решение, если их несколько?

Процесс-модель поиска оценки лучшего решения с минимальным числом единичных координат из более, чем двух альтернатив, представлена на рис. 3. Она включает следующие операции: 1. Первоначально в вектор-результат Q, в котором будет сохранено лучшее решение, заносятся единичные значения

$$Q = Q\overline{(\vee((Q \wedge Q_i) \oplus Q))} \vee Q_i(\vee((Q \wedge Q_i) \oplus Q));$$
$$Y = \vee((Q \wedge Q_i) \oplus Q);$$
$$Q = Q\bar{Y} \vee Q_i Y.$$

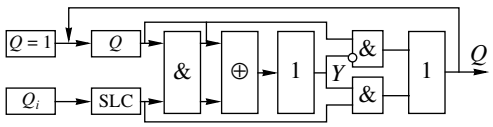


Рис. 3. Процесс-модель выбора решения.

во все координаты (худшее решение) и одновременно осуществляется операция slc сдвига влево с уплотнением единиц текущего вектора Q_i. 2. Выполняется сравнение двух векторов: Q и очередной оценки Q_i из списка решений. 3. Реализуется векторная операция and $(Q \wedge Q_i)$, а результат сравнивается с вектором Q, что дает возможность изменить его, если вектор Q_i имеет меньшее число единичных значений. 4. Процедура поиска оценки лучшего решения повторяется n раз.

Здесь or-оператор редукции (после xor элемента) формирует двоичное однобитовое решение на основе логической операции or над n разрядами критерия качества. Для двух двоичных векторов – критериев качества – процедура выбора лучшего из них представлена ниже:

$Q_1(m, A) = (6, 12)$	1 1 1 1 1 1 . . . . .
$Q_2(m, A) = (8, 12)$	1 1 1 1 1 1 1 1 . . . .
$Q_1(m, A) \wedge Q_2(m, A)$	1 1 1 1 1 1 . . . . .
$Q_1(m, A) \oplus Q_1(m, A) \wedge Q_2(m, A)$	. . . . .
$Q(m, A) = Q_1(m, A)$	1 1 1 1 1 1 . . . . .

.

Здесь численные оценки представлены как пояснительная информация для пользователя, который технологичнее воспринимает числа и символы. Таким образом, векторный логический критерий качества взаимодействия объектов в киберпространстве позволяет получать оценку поиска, распознавания и принятия решения с высоким быстродействием логических параллельных операций, что особенно существенно для критических систем реального времени.

3. Структуры данных для поиска, распознавания и принятия решения

Метрика качества, представленная в (8), дает возможность оценивать близость пространственных объектов друг к другу или их взаимодействие. Практическим примером полезности векторно-логического критерия качества может служить стрельба по цели, которая иллюстрируется ранее приведенными диаграммами (см. рис. 1) взаимодействия векторов: 1. Снаряд попал точно в цель и поразил ее полностью. 2. Мишень поражена необоснованно большим калибром снаряда. 3. Калибр снаряда недостаточен для поражения крупной цели. 4. Неэффективный и неточный выстрел снарядом большого калибра. 5. Снаряд пролетел мимо мишени. Процесс-модель (9) взаимодействия $P(m, A)$ сопровождается критерием качества, который оценивает не только попадание или промах, но и эффективность использования калибра оружия.

Аналитическая форма записи обобщенной процесс-модели для выбора лучшего взаимодействия входного запроса m с системой логических ассоциативных отношений имеет вид:

$$
\begin{gathered}
P(m,A)=\min Q_i\left(m\overset{n}{\underset{i=1}{\Delta}}A_i\right)=\vee\left[\left(Q_i\overset{j\neq i}{\underset{j=1,n}{\wedge}}Q_j\right)\oplus Q_i\right]=0;\\
Q(m,A)=(Q_1,Q_2,\ldots,Q_i,\ldots,Q_n);\quad A=(A_1,A_2,\ldots,A_i,\ldots,A_n);\\
\Delta=\{and,or,xor,not,slc,nop\};\quad A_i=(A_{i1},A_{i2},\ldots,A_{ij},\ldots,A_{is});\\
A_{ij}=(A_{ij1},A_{ij2},\ldots,A_{ijr},\ldots,A_{msq});\ m=(m_1,m_2,\ldots,m_r,\ldots,m_q);\\
Q_i=d(m,A_i)\vee\mu(m\in A_i)\vee\mu(A_i\in m);\quad d(m,A_i)=m\oplus A_i;\\
\mu(m\in A_i)=A_i\wedge\overline{m\wedge A_i};\quad \mu(A_i\in m)=m\wedge\overline{m\wedge A_i}.
\end{gathered}
\tag{9}
$$

Здесь выражение, определяющее функциональность, $P(m,A)$, можно представить как аналитическую модель вычислительного процесса в виде высказывания, минимизирующего критерий качества; структуры данных представлены в виде совокупности таблиц $A=(A_1,\ldots,A_i,\ldots,A_m)$, логически взаимосвязанных между собой; каждая таблица задается упорядоченной совокупностью вектор-строк ассоциативной таблицы $A_i=(A_{i1},\ldots,A_{ij},\ldots,A_{is})$ явных решений, а строка $A_{ij}=(A_{ij1},\ldots,A_{ijr},\ldots,A_{msq})$ представляет собой истинное высказывание. Функционал, представленный в виде таблицы, не имеет постоянных во времени входных и выходных переменных. Равнозначность всех переменных в векторе $A_{ij}=(A_{ij1},\ldots,A_{ijr},\ldots,A_{msq})$ создает одинаковые условия их существования. Это означает инвариантность решения задач прямой и обратной импликации в пространстве $A_i\in A$. Ассоциативный вектор A_{ij} определяет собой явное решение, где каждая переменная задается в конечном, многозначном и дискретном алфавите $A_{ijr}\in$ $\in\{\alpha_1,\ldots,\alpha_i,\ldots,\alpha_k\}=\beta$. Взаимодействие $P(m,A)$, входного вектора-запроса $m=(m_1,\ldots,m_r,\ldots,m_q)$ с множеством $A=(A_1,\ldots,A_i,\ldots,A_m)$, формирует решения с выбором лучшего из них по минимальному критерию качества:

$$P(m,A)=\min Q_i[m\wedge(A_1\vee A_2\vee\ldots\vee A_i\vee\ldots\vee A_m)].$$

Конкретное взаимодействие вершин (таблиц) графа между собой создает функциональность $A=(A_1,\ldots,A_i,\ldots,A_m)$, которая может быть оформлена в следующие структуры: 1. Единственная ассоциативная таблица, содержащая все решения логической задачи в явном виде. Преимущество – максимальное быстродействие параллельного ассоциативного поиска решения по таблице. Недостаток – максимально высокая аппаратурная сложность размещения таблицы большой размерности. 2. Древовидная (графовая) структура бинарных отношений между функциональными примитивами, каждый из которых формирует таблицу истинности для незначительного количества переменных. Преимущество – максимально низкая аппаратурная сложность решения задачи. Недостаток – минимальное быстродействие последовательного ассоциативного поиска решения по дереву. 3. Компромиссная графовая структура логически понятных для пользователя отношений между примитивами, каждый из которых формирует таблицу истинности для логически

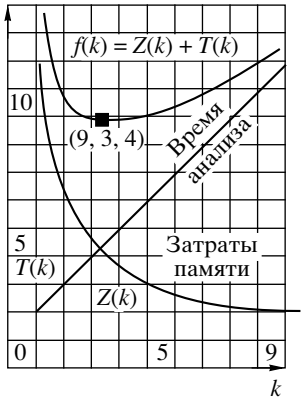


Рис. 4. Функции аппаратуры и времени от числа разбиений.

сильно взаимосвязанных переменных. Преимущество – высокое быстродействие параллельного ассоциативного поиска решений по минимальному числу таблиц, составляющих граф, а также сравнительно невысокая аппаратурная сложность решения задачи. Недостаток – снижение быстродействия из-за последовательной логической обработки графовой структуры явных решений, найденных в таблицах и необходимость введения дополнительной переменной для логического связывания таблиц (вершин) в граф отношений. Разбиение одной таблицы (ассоциативной памяти) на k частей приводит к уменьшению аппаратных затрат, выраженных в компонентах (лутах) (LUT – Look Up Table) программируемой логической матрицы [9, 13]. Каждая ячейка памяти создается с помощью четырех лутов. Учитывая, что ассоциативную матрицу можно представить квадратом со стороной n, то суммарные аппаратные затраты $Z(k)$ памяти для хранения данных и время $T(k)$ анализа логического ассоциативного графа функционально зависят от числа k разбиений таблицы или числа вершин графа:

$$
\begin{aligned}
Z(k) &= k \times \frac{1}{4} \times \left(\frac{n}{k}\right)^2 + h = \frac{n^2}{4 \times k} + h \quad (h = \{n, \text{const}\}); \\
T(k) &= \frac{4 \times k}{t_{clk}} + \frac{4}{t_{clk}} = \frac{4}{t_{clk}}(k+1), \quad (t_{clk} = \text{const}).
\end{aligned} \tag{10}
$$

Здесь h – затраты на общую схему управления системой ассоциативных памятей. Платой за уменьшение аппаратуры является снижение быстродействия обработки структуры памятей или увеличение времени анализа компонентов графа. Период обработки одной ассоциативной памяти представлен циклом из четырех синхроимпульсов. Число разбиений k пропорционально увеличивает количество тактов в худшем варианте последовательного соединения памятей. Слагаемое $\frac{4}{t_{clk}}$ в (10) задает время, необходимое для подготовки данных на входе вычислительной структуры, а также для их декодирования на выходе. Функциональные зависимости аппаратных затрат и времени анализа графа ассоциативных памятей от числа вершин или разбиений представлены на рис. 4.

Обобщенная функция эффективности графовой структуры от числа вершин

$$f[Z(k), T(k)] = Z(k) + T(k) = \left(\frac{n^2}{4 \times k} + h\right) + \left(\frac{4}{t_{clk}}(k+1)\right) \tag{11}$$

позволяет определить оптимальное разбиение совокупного и наперед заданного объема ассоциативной памяти [11]. В случае, представленном на рис. 4, лучшее разбиение есть минимум аддитивной функции, который определяется значением k, обращающим производную функции в нуль: $n \times n = 600 \times 600$, $h = 200$, $t_{clk} = 4$, $k = 4$. Предложенная структура ассоциативных таблиц и введенный критерий качества получаемых решений являются основой для разработки специализированной мультипроцессорной архитектуры, ориентированной на параллельное выполнение векторных логических операций.

4. Архитектура логического ассоциативного мультипроцессора

Для анализа больших информационных объемов логических данных существует несколько практически ориентированных технологий: 1. Использование рабочей станции для последовательного программирования задачи, где стоимость ее решения, а также временные затраты очень высоки. 2. Разработка специализированного параллельного процессора на основе PLD. Высокий параллелизм обработки информации компенсирует сравнительно низкую по сравнению с CPU тактовую частоту. Такое схемотехническое решение с возможностью перепрограммирования является по производительности выигрышным вариантом. Недостаток – отсутствие гибкости, характерной программным методам решения логических задач и высокая стоимость реализации системы на кристалле PLD при больших объемах промышленного выпуска изделия. 3. Лучшее решение может быть получено в результате объединения достоинств центрального процессорного устройства, программируемой логической матрицы и заказной СБИС [8, 13], таких как гибкость программирования, возможность корректирования исходных кодов; минимальное число команд и простые схемотехнические решения аппаратной реализации мультипроцессора; распараллеливание логических процедур на структуре однобитовых процессоров. Реализация вычислителя в кристалле заказной СБИС дает возможность получать максимальную тактовую частоту, минимальную стоимость микросхемы при больших объемах выпуска изделия, низкое энергопотребление. Объединение преимуществ перечисленных технологий определяет базовую конфигурацию вычислителя, имеющего сферическую структуру (рис. 5), состоящую из 16 векторных секвенсоров – устройств последовательного управления (УПУ), каждый из которых, включая граничные элементы, соединен с восьмью соседними. Прототипом данного вычислителя является процессор PRUS, описанный в работе [7].

Занесение информации в процессор выполняется по классической схеме процесса проектирования, за исключением стадии размещения и трассировки, которая заменяется фазой распределения программ и данных между всеми логическими бит-процессорами, работающими параллельно. Компилятор

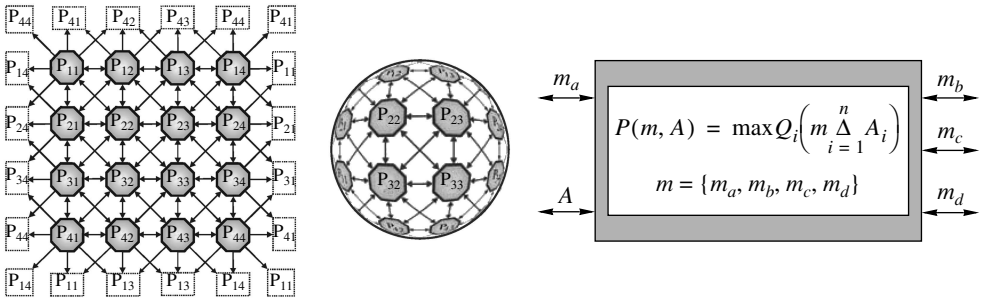


Рис. 5. Макроархитектура мультипроцессора и интерфейс.

обеспечивает распределение данных по процессорам, задает время формирования решения на выходе каждого из них, а также планирует передачу полученных результатов другому процессору.

Логический ассоциативный мультипроцессор (ЛАМП) – это эффективная сеть процессоров, которая обрабатывает данные и обеспечивает обмен информацией между компонентами сети в процессе их решения. Простая схемотехника каждого процессора позволяет эффективно обрабатывать сверхбольшие массивы, насчитывающие миллионы бит информации, затрачивая на это в сотни раз меньше времени по сравнению с универсальным процессором.

Базовая ячейка – векторный процессор для вычислителя может быть синтезирован на 200 вентилях, что дает возможность сеть, содержащую 4096 вычислителей, легко реализовать в кристалле заказной СБИС, используя современную кремниевую технологию. Поскольку затраты памяти для хранения данных весьма незначительны, вычислитель может быть использован при проектировании систем управления в таких областях человеческой деятельности, как промышленное производство, защита информации, медицина, искусственный интеллект, космонавтика, геология, метеорология. Особый интерес вычислитель представляет для цифровой обработки данных, распознавания образов и криптоанализа. Однако основное назначение ЛАМП – получение квазиоптимального решения в задаче поиска и (или) распознавания с использованием компонентов архитектуры, ориентированных на выполнение векторных логических операций:

$$P(m, A) = \min Q_i \left(m \overset{n}{\underset{i=1}{\Delta}} A_i \right), \quad m = \{m_a, m_b, m_c, m_d\}. \tag{12}$$

Интерфейс системы, соответствующий данному функционалу (12), представлен на рис. 5. Все компоненты $\{A, m_a, m_b, m_c, m_d\}$ могут быть как входными, так и выходными. Двунаправленная детализация интерфейса связана с инвариантностью отношения всех переменных, векторов, A-матрицы и компонентов к входам и (или) выходам архитектуры. Поэтому структурная модель системы вычислителя может быть использована для решения любых задач прямой и обратной импликации в дискретном логическом пространстве, в чем и заключается ее отличие от концепции автоматной модели вычислительного устройства с выраженными входами и выходами. Компоненты или регистры

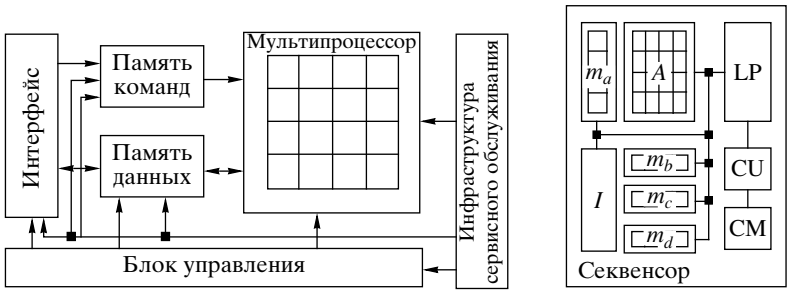


Рис. 6. Архитектура ЛАМП и структура УПУ.

$m = (m_a, m_b, m_c, m_d)$ используются для получения решения в виде буферных, входных и выходных векторов, а также для идентификации оценки качества удовлетворения входного запроса.

Один из возможных вариантов архитектуры ЛАМП представлен на рис. 6. Основным компонентом является матрица $P = [P_{ij}], (i.j = \overline{1,4})$, содержащая 16 вектор-процессоров, каждый из которых предназначен для выполнения пяти логических векторных операций над памятью данных, представленной в виде таблицы A, размерностью $(m \times n)$.

В блоке интерфейса происходит обмен данными и загрузка программы обработки данных в соответствующую память команд. Блок управления инициирует выполнение команд логической обработки данных и синхронизирует функционирование всех компонентов мультипроцессора. Блок *IP* [8] предназначен для сервисного обслуживания всех модулей, диагностирования дефектов и восстановления работоспособности компонентов и устройства в целом. Элементарный логический ассоциативный процессор или УПУ (см. рис. 5), входящий в состав вычислителя, содержит логический процессор LP, ассоциативную (память) A-матрицу для параллельного выполнения базовых операций, блок векторов m, предназначенный для параллельного обслуживания строк и столбцов A-матрицы, а также обмена данными в процессе вычислений, память прямого доступа CM, сохраняющую команды программы обработки информации, автомат управления CU выполнением логических операций, интерфейс I связи УПУ с другими элементами и устройствами ЛАМП.

LP (рис. 7) выполняет пять операций (*and*, *or not*, *xor*, *s1c*), являющихся базовыми для создания алгоритмов и процедур информационного поиска и оценивания решения. Модуль LP имеет мультиплексор, коммутирующий один из пяти операндов с выбранным логическим векторным оператором. Сформированный результат через мультиплексор (элемент or) заносится в один из четырех операндов, выбираемый соответствующим адресом.

Особенности реализации логического процессора заключаются в наличии трех бинарных (*and*, *or*, *xor*) и двух унарных (*not*, *slc*) операций. Последние можно присоединять к такту обработки регистровых данных, выбрав одну из трех операций *not*, *slc*, *nop* (нет операции). Для повышения эффективности работы логического устройства вводятся два элемента с пустой операцией. Если необходимо выполнить только унарную операцию, то на уровне бинарных команд следует выбрать nop, что практически означает передачу дан-

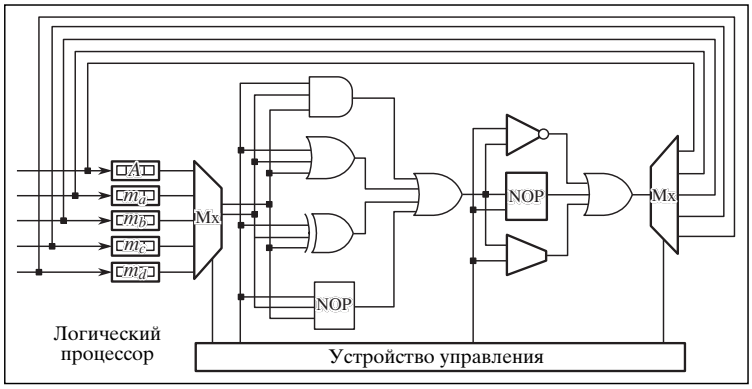


Рис. 7. Структура блока логических вычислений.

ных через повторитель ко второму уровню унарных операций. Все операции в LP регистровые или регистрово-матричные. Последние предназначены для анализа вектор-строк таблицы при использовании входного m-вектора как запроса для точного поиска информации. В блоке логических вычислений допустимо следующее сочетание операций и операндов:

$$C = \begin{cases} \{m_a, m_b,\ m_c, m_d\}\Delta A_i; \\ \{m_a, m_b, m_c, m_d\}\Delta\{m_a, m_b, m_c, m_d\}; \\ \{not, nop, slc\}\{m_a, m_b,\ m_c, m_d, A_i\}. \\ \Delta = \{and, or, xor\}. \end{cases}$$

Реализация всех векторных операций блока логических вычислений, выполняемых с тактовой частотой 100 МГц, для одного УПУ в среде Verilog с последующей послесинтезной реализацией в кристалле программируемой логики Virtex 4, Xilinx содержит 2400 эквивалентных вентилей.

5. Инфраструктура векторно-логического анализа

Инфраструктура – совокупность моделей, методов и средств описания, анализа и синтеза структур данных для решения функциональных задач. Модель (системная) – совокупность взаимосвязанных, определенных в пространстве и времени компонентов с заданной адекватностью описывающая процесс или явление и используемая для достижения поставленной цели при наличии ограничений и метрики оценивания качества решения. Здесь ограничения есть аппаратурные затраты, время разработки и производства до появления изделия на рынке (time-to-market), подлежащие минимизации. Метрика оценивания решения при использовании модели определена двоичным логическим вектором в дискретном булевом пространстве. Концептуальная модель вычислительного изделия представлена совокупностью управляющего и операционного автоматов. В модели функциональности использована иерархическая технология создания цифровых систем с локальной синхронизацией отдельных модулей и одновременно глобальной асинхронностью функционирования всего устройства [13].

$$m_{ai}^{m} = \bar{\vee}\left[\left(m_b \overset{n}{\underset{i=1}{\wedge}} A_i\right) \oplus m_b\right];$$

$$A_i = \left(m_b \overset{n}{\underset{i=1}{\wedge}} A_i\right).$$

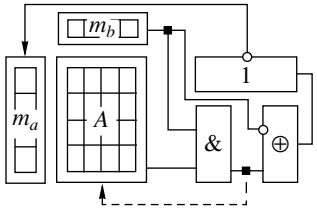


Рис. 8. Поиск всех допустимых решений.

$$m_b^{s} = \left(\underset{\forall m_{ai}=1}{\wedge} A_i\right) \wedge \left(\overline{\underset{\forall m_{ai}=0}{\vee} A_i}\right)$$

$$m_b^{m} = \left(\underset{\forall m_{ai}=1}{\vee} A_i\right) \wedge \left(\overline{\underset{\forall m_{ai}=0}{\vee} A_i}\right)$$

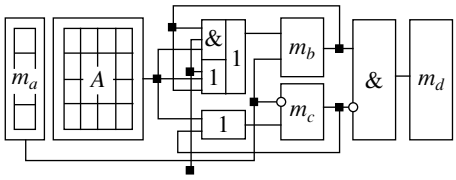


Рис. 9. Выбор оптимального решения.

Для детализации структуры векторного процессора и УПУ далее рассмотрены аналитические и структурные процесс-модели, выполняющие анализ A-матрицы по столбцам или строкам. Первая из них представлена на рис. 8 и предназначена для определения множества допустимых решений относительно входного запроса m_b, вторая (рис. 9) осуществляет поиск оптимального решения на множестве строк, найденных с помощью первой модели в результате их анализа.

Возможно и самостоятельное применение второй модели, ориентированное на определение однозначного и многозначного решения при поиске дефектов в цифровой системе.

Все операции, выполняемые двумя процесс-моделями, векторные. Модель анализа строк (см. рис. 8) формирует вектор m_a идентификации допустимых $m_{ai} = 1$ или противоречивых $m_{ai} = 0$ решений относительно входного условия m_b за n тактов обработки всех m-разрядных векторов таблицы $A = = card(m \times n)$. Качество (допустимость) решения определяется для каждого взаимодействия входного вектора m_b и строки $A_i \in A$ на блоке (редукции) дизъюнкции. Матрица A может быть модифицирована ее пересечением с входным вектором на основе использования операции $A_i = \left(m_b \overset{n}{\underset{i=1}{\wedge}} A_i\right)$, если необходимо исключить из $A-$таблицы все незначимые для решения координаты и векторы, отмеченные единичными значениями вектора m_a.

Решение задач диагностирования посредством анализа строк таблицы (см. рис. 9) осуществляется так. После выполнения диагностического эксперимента формируется двоичный вектор экспериментальной проверки m_a, маскирующий A-таблицу неисправностей для поиска одиночных или кратных дефектов. Векторы m_b и m_c используются для накопления результатов выполнения операций конъюнкции и дизъюнкции. Затем выполняется логическое вычитание (xor-операция) из первого регистра m_b содержимого второго вектора m_c с последующей записью результата в регистр m_d. Для реализации

$$\begin{cases} m_b = (m_b \vee A_i); \\ m_{ai} = \overset{n}{\underset{i=1}{\vee}} [(m_b \vee A_i) \wedge \bar{m}_b]. \end{cases}$$

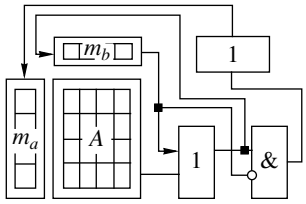
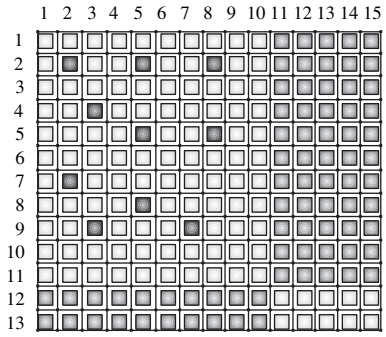


Рис. 10. Процесс-модель поиска квазиоптимального покрытия.

второго уравнения, которое формирует множественное решение, элемент and заменяется функцией or. В схеме используется также переменная выбора режима поиска решения: single или multiple. В качестве входного условия в модели использован вектор m_a, управляющий выбором векторной операции and, or для обработки единичных $A_i(m_{ai} = 1) \in A$ или нулевых $A_i(m_{ai} = 0) \in$ $\in A$ строк A-таблицы. В результате выполнения n тактов осуществляется накопление единичных и нулевых относительно значений координат вектора m_a решений в регистрах A_1, A_0. Априори в указанные регистры заносится вектор единиц и нулей: $A_1 = 1$, $A_0 = 0$. После обработки всех n строк $A-$таблицы за n тактов выполняется векторная конъюнкция содержимого регистра A_1 с инверсией регистра A_0, которая формирует результат в виде вектора m_b, где единичные значения координат определяют решение. В таблице неисправностей цифрового изделия единичным координатам вектора m_b соответствуют столбцы, отождествляемые с номерами дефектов или неисправных блоков, подлежащих восстановлению или ремонту.

При сервисном обслуживании функциональных модулей можно на универсальной структуре системы векторного логического анализа решить оптимизационную задачу восстановления работоспособности. С помощью минимального числа ремонтных запасных строк и (или) столбцов, например, памяти необходимо обеспечить квазиоптимальное покрытие всех обнаруженных в ячейках неисправностей. Технологическая и математическая составляющие векторной логики в данном случае обусловливают простое схемотехническое решение для получения квазиоптимального покрытия (рис. 10), преимущества которого заключаются в следующем: 1. Вычислительная сложность процедуры: число векторных операций, равное числу строк таблицы, $Z = n$. 2. Минимум аппаратных затрат: таблица и два вектора (m_b, m_a) для хранения промежуточных покрытий и накопления результата в виде единичных координат, соответствующих строкам таблицы, которые составляют квазиоптимальное покрытие. 3. Отсутствие классического деления задачи покрытия на поиск ядра покрытия и дополнения. 4. Отсутствие сложных процедур манипулирования ячейками строк и столбцов. Получение не всегда оптимального покрытия – недостаток, который компенсируется технологичностью векторной процедуры, представленной на рис. 10.

Операция редукции, которая на последнем этапе превращает векторный результат в бит m_{ai} вектора m_a по функции or $m_{ai} = \vee[(m_b \vee A_i) \wedge \bar{m}_b]$. В общем случае операция редукции в алгебре векторных операций записывается в виде <бинарная операция><вектор>: $\vee A_i$, $\wedge m$, $\bar{\wedge}(m \vee A_i)$. Обратная процедура (векторизация) есть конкатенация булевых переменных:



1	.	.	.	.	.	1	.	.	.
.	.	.	1	.	.	.	.	1	.
.	1	.	.	1	.	.	1	.	.
.	.	.	.	.	.	.	.	.	1
.	.	1	.	.	1	.	.	.	.
1	1	1	.	.	.	.	.	.	.
.	.	.	1	.	.	.	.	.	.
.	.	.	.	1	1	.	.	.	.
.	.	.	.	.	.	1	.	.	.
.	.	.	.	.	.	.	1	.	.
.	.	.	.	.	.	.	.	1	1

Рис. 11. Модуль памяти с резервом и таблица покрытия.

$m_a(a,b,c,d,e,f,g,h)$. В процедуре поиска покрытия априори векторы $m_b = = 0$, $m_a = 0$ становятся равными. Квазиоптимальное покрытие накапливается за n тактов в векторе m_a последовательным сдвигом. Биты, заносимые в регистр m_a, формируются схемой or, которая выполняет редукцию после анализа полученного результата $[(m_b \vee A_i) \wedge \bar{m}_b]$ на наличие единиц.

Представляет интерес функциональная законченность цикла диагностирования, когда после получения квазиоптимального покрытия данная информация используется для восстановления работоспособности дефектных ячеек памяти [8]. Размерность модуля памяти (13x15 ячеек) не влияет на вычислительную сложность получения покрытия десяти дефектных ячеек с помощью резервных строк и столбцов (рис. 11).

Для решения оптимизационной задачи выполняется построение таблицы покрытия (см. рис. 11) неисправных ячеек, в которой строки – резервные ресурсы для покрытия дефектов $(C_2, C_3, C_5, C_7, C_8, C_2, R_2, R_4, R_5, R_7, R_8, R_9)$, а столбцы – дефекты ячеек $(F_{2,2}, F_{2,5}, F_{2,8}, F_{4,3}, F_{5,5}, F_{5,8}, F_{7,2}, F_{8,5}, F_{9,3}, F_{9,7})$, подлежащие ремонту. При этом столбцы соответствуют координатам дефектных ячеек, а строки идентифицируют резервные компоненты (строки и столбцы), которые могут восстановить работоспособность неисправных координат. Модель вычислительного процесса, представленная на рис. 9, дает возможность получить оптимальное решение в виде $m_a = = \boxed{1\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0}$, которому соответствует покрытие $R = \{C_2, C_3, C_5, C_7, C_8\}$, как одно из трех возможных минимальных решений $R = C_2, C_3, C_5, C_7, C_8 \vee C_2, C_3, C_5, C_8, R_9 \vee C_2, C_5, C_8, R_4, R_9$ для таблицы неисправностей. Технологическая модель встроенного диагностирования и ремонта памяти (рис. 12) имеет четыре компонента: 1. Тестирование модуля (Unit Under Test (UUT)) с использованием эталонной модели (Model Under Test (MUT)) для формирования вектора экспериментальной проверки m_a, размерность которого соответствует числу тестовых наборов. 2. Поиск дефектов на основе анализа таблицы неисправностей A. 3. Оптимизация покрытия дефектных ячеек ремонтными строками и столбцами на основе анализа таблицы A. 4. Восстановление работоспособности памяти посредством замены адресов (Address Decoder (AD)) неисправных строк и столбцов, представленных вектором m_a, на адреса компонентов из ремонтного запаса (Spare Memory (SM)) [8].

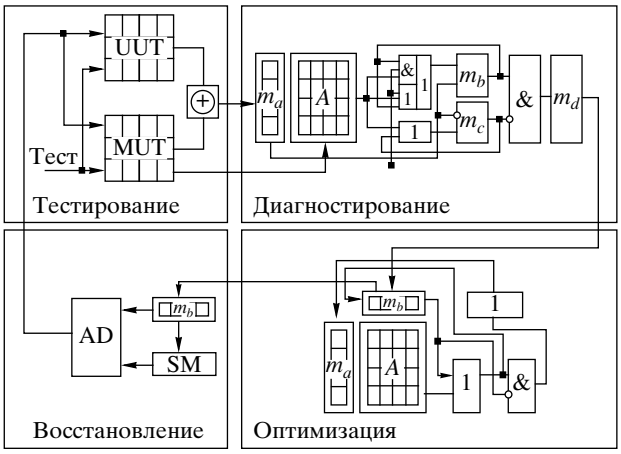


Рис. 12. Модель встроенного тестирования и восстановления памяти.

$$E = F(L,T,H) = \min\left[\frac{1}{3}(L+T+H)\right],$$

$$Y = (1-P)^n;$$

$$L = 1 - Y^{(1-k)} = 1 - (1-P)^{n(1-k)};$$

$$T = \frac{(1-k)\times H^s}{H^s + H^a}; \quad H = \frac{H^a}{H^s + H^a}.$$

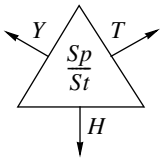


Рис. 13. Оценка эффективности процесс-модели.

Процесс-модель встроенного сервисного обслуживания работает в реальном масштабе времени и позволяет поддерживать в работоспособном состоянии, без вмешательства человека, цифровую систему на кристалле, что является целесообразным решением в случае применения технологий, связанных с дистанционной эксплуатацией изделия. Предложенные процесс-модели анализа ассоциативных таблиц, а также введенные критерии качества логических решений позволяют решать задачи квазиоптимального покрытия, диагностирования дефектов программных и (или) аппаратных блоков. Модель векторных вычислений стала основой для разработки специализированной мультипроцессорной архитектуры, ориентированной на поиск, распознавание и принятие решений об использовании структур ассоциативных таблиц.

Аналитическая оценка эффективности проектного решения, направленного на выполнение условий специализации Sp и стандартизации St (рис. 13) определяется минимумом среднего значения следующих трех взаимно противоречивых относительных и безразмерных параметров: уровень ошибок проекта L, время верификации и (или) тестирования T, программно-аппаратная избыточность, определяемая механизмами ассерций и (или) граничного сканирования H.

Параметр L, как дополнение к Y, характеризующему выход годной продукции, зависит от тестопригодности проекта k, вероятности P существования неисправных компонентов и числа необнаруженных ошибок n. Время верификации определяется тестопригодностью проекта k, умноженной на структурную сложность аппаратно-программной функциональности, отнесенной к общей сложности проекта в строках кода или эквивалентных вентилях. Программно-аппаратная избыточность находится в функциональной зависимости от сложности ассерционного кода или механизма граничного сканирования, отнесенной к общей сложности проекта. При этом ассерционная, или сканирующая, избыточность должна обеспечивать заданную глубину диагностирования ошибок функциональности за время выхода изделия на рынок, определенное заказчиком.

6. Выводы

Существующие программные аналоги практически не предлагают ассоциативно-логических технологий поиска, распознавания и принятия решений в дискретном информационном пространстве [4, 5, 15]. Практически все они используют универсальную систему команд современного дорогостоящего процессора с математическим сопроцессором. В то же время, аппаратные специализированные средства логического анализа, являющиеся их прототипами [1, 6, 7], как правило, ориентированы на побитовую или невекторную обработку информации. Предложенный новый подход векторно-логической обработки ассоциативных данных с полным исключением арифметических операций, влияющих на быстродействие и аппаратную сложность, может быть эффективно реализован на основе использования современной микроэлектронной аппаратуры в виде мультипроцессорной цифровой системы на кристалле. Фактическая реализация подхода основана на предложении моделей и методов, использующих общую идею векторно-логической метрики киберпространства:

1. Процесс-модели анализа ассоциативных таблиц ориентирован на достижение высокого быстродействия анализа информационных объектов и подсчета критериев качества их взаимодействия на основе векторных логических операций для поиска, распознавания образов, принятия и оценивания решений в киберпространстве.

2. Метод параллельного решения ассоциативно-логических задач с минимальным числом векторных логических операций и полным исключением арифметических команд, что обеспечивает высокое быстродействие, минимальную стоимость и незначительное энергопотребление вычислителя, реализованного на кристалле программируемой логики.

3. Новые векторно-логические процесс-модели встроенного диагностирования цифровых систем на кристаллах, поиска квазиоптимального покрытия, использующие средства логического ассоциативного мультипроцессора, параллельные операции вычислительных процессов и векторно-логический критерий качества.

Практическая значимость полученных результатов подтверждена созданием встроенного компонента для диагностирования и восстановления рабо-

тоспособности памяти в цифровой системе на кристалле. Дальнейшие исследования направлены на разработку прототипа логического ассоциативного мультипроцессора для решения актуальных задач поиска, распознавания и принятия решений с помощью векторного логического анализа.

СПИСОК ЛИТЕРАТУРЫ

1. *Бондаренко М.Ф., Дударь З.В., Ефимова И.А. и др.* О мозгоподобных ЭВМ // Радиоэлектроника и информатика. Харьков: ХНУРЭ 2004. № 2. С. 89–105.
2. *Cohen A.A.* Addressing architecture for Brain-like Massively Parallel Computers // Euromicro Sympos. Digital Syst. Design (DSD'04). 2004. P. 594–597.
3. *Кузнецов О.П.* Быстрые процессы мозга и обработка образов // Новости искусственного интеллекта. 1998. № 2. С. 88–92.
4. *Васильев С.Н., Жерлов А.К., Федосов Е.А., Федунов Б.Е.* Интеллектуальное управление динамическими системами. М.: Физматлит, 2000.
5. *Липаев В.В.* Программная инженерия. Методологические основы. Учебник. М.: Теис, 2006.
6. А.С. №1439682. 22.07.88. Регистр сдвига / Какурин Н.Я., Хаханов В.И., Лобода В.Г., Какурина А.Н. 4с.
7. *Гайдук С.М., Хаханов В.И., Обризан В.И. и др.* Сферический мультипроцессор PRUS для решения булевых уравнений // Радиоэлектроника и информатика. Харьков. 2004. № 4(29). С. 107–116.
8. *Хаханов В.И., Литвинова Е.И., Гузь О.А.* Проектирование и тестирование цифровых систем на кристаллах. Харьков: Новое слово, 2009.
9. *Хаханов В.И., Хаханова И.В., Литвинова Е.И. и др.* Проектирование и верификация цифровых систем на кристаллах. Харьков: Новое слово, 2010.
10. *Акритас А.* Основы компьютерной алгебры с приложениями: Пер. с англ. / А. Акритас. М.: Мир, 1994.
11. *Аттетков А.В, Галкин С.В., Зарубин В.С.* Методы оптимизации. М: Издво МГТУ им. Н.Э. Баумана, 2003.
12. *Abramovici M., Breuer M.A., Friedman A.D.* Digital System Testing and Testable Design Comp. Sc. Press. 1998.
13. *Densmore D., Passerone R., Sangiovanni–Vincentelli A.* A Platform–Based taxonomy for ESL Design // Design & Test Comput. 2006. P. 359–373.
14. Автоматизация диагностирования электронных устройств / Ю.В. Малышенко и др. / Под ред. В.П. Чипулиса. М.: Энергоатомиздат, 1986.
15. *Трахтенгерц Э.А.* Компьютерные методы реализации экономических и информационных управленческих решений. М.: СИНТЕГ, 2009.

Статья представлена к публикации членом редколлегии П.Ю. Чеботаревым.

Поступила в редакцию 13.06.2010