2. Carlson R. E., Fritsch F. N. An algorithm for monotone piecewise bicubic interpolation // SIAM J. Numer. Anal.—1989.—26, N 1.—P. 230.
3. Shima T. et al. Three-dimensional table look-up MOSFET-model for precise circuit simulation // IEEE J. of Solid State Circuits.—1982.—17, N 3.—P. 449.
4. Hsu M. C. et al. Inverse-geometry dependence of MOS transistor electrical parameters // IEEE Trans. on Computer-Aided Design.—1987.—6, N 4.—P. 582.
5. Груданов Н. Б. и др. Модель МДП-транзистора со встроенным каналом // Автоматизация проектирования в электронике: Респ. межвед. науч.-техн. сб.—Киев, 1980.—Вып. 21.

Поступила в редакцию 28 мая 1991 г.

УДК 681.327

А. И. Мишин

(Новосибирск)

ПАМЯТЬ С АДРЕСАЦИЕЙ ПО СОДЕРЖАНИЮ И ЕЕ ПОТЕНЦИАЛЬНЫЕ ВОЗМОЖНОСТИ

Проводится сравнительный анализ параллельного и последовательного поисков на точное соответствие. Дается оценка трудоемкости поиска при использовании ассоциативного кодирования (хеширования), основанная на r-разрядных обращениях к запоминающему устройству с произвольной выборкой (ЗУПВ) ($1 < r < m$, m — число разрядов в слове). Показано, что минимальное среднее число обращений к памяти, равное 1, достигается при двухкратной избыточности памяти и $r = 4$. Показано, что минимальное среднее время поиска достигается не параллельным поиском в ассоциативной памяти (АЗУ), а при использовании метода хеширования, реализуемого памятью с адресацией по содержанию (ПАС), включающей ЗУПВ и процессор. Сравнение поиска в АЗУ и ПАС по комплексному критерию η, определяемому произведением площади П, занимаемой памятью емкостью M (слов), на время поиска T, дает $\eta = (\Pi T)_{\text{АЗУ}}/(\Pi T)_{\text{ЗУПВ}} = O(M^{3/4})$, причем время поиска в АЗУ в $O(M^{1/4})$ раз больше, чем в ЗУПВ.

Введение. Известно, что возможность адресации по содержанию существенно повышает производительность ЭВМ за счет более эффективной организации поиска данных. Однако несмотря на то что основные принципы адресации по содержанию были разработаны более 30 лет назад, ассоциативное запоминающее устройство используется лишь в качестве буферной памяти небольшой емкости. Основным недостатком АЗУ является неэффективное исполнение операции поиска на точное соответствие. При исполнении этой операции каждый элемент АЗУ выполняет проверку на точное соответствие предъявленного слова и слова, хранящегося в самом элементе, т. е. все элементы работают одновременно. Однако параллельный поиск на точное соответствие не ускоряет процесса поиска по сравнению с поиском в однопроцессорной ЭВМ с памятью произвольного доступа. Понятие «ассоциативный» отражает лишь факт наличия связи между данными и не имеет отношения к самому механизму поиска информации. Поэтому вычислительному устройству, ориентированному на решение задачи поиска информации, в большей степени подходит название память с адресацией по содержанию. В литературе также используется термин «память с адресацией по данным».

Выборка и запись информации в запоминающие ячейки ПАС проводятся в зависимости от содержащихся в них данных. Не имеет принципиального значения, каким способом ведется поиск: программным путем (как в ЭВМ) или с помощью параллельного опроса (как в АЗУ).

Используя ПАС, можно, в частности, получить ответы на следующих два вопроса: 1) принадлежит ли некоторое слово w множеству данных (проверка на точное соответствие); 2) для заданного слова w найти во множестве данных слово w', наиболее близкое к w (поиск на наилучшее соответствие).

Взаимосвязь затрат времени и емкости памяти при проверке точного соответствия рассмотрена в [1—4]. В [5] высказано предположение, что, используя ассоциативное кодирование (хеширование), можно за четыре поразрядных обращения к памяти с произвольной выборкой емкостью, в 2 раза большей, чем это необходимо для хранения всего множества данных, получить ответ на первый вопрос. При считывании за одно обращение к памяти всех m разрядов слова среднее число обращений T' к памяти равно $\frac{-\ln(1-z)}{z}$, где $z = N/M$ — коэффициент заполнения памяти, M (слов) — емкость памяти, а N — число слов, хранящихся в памяти [1—3]. Выражение получено в предположении, что ключевым словам (признакам) поставлены в соответствие адреса ячеек, распределенных в памяти ЭВМ по закону, близкому к равномерному. При $z = 0{,}5$ $T' \approx 1{,}4$. Поэтому использование параллельных вычислительных устройств (например, АЗУ) при поиске на точное соответствие не дает преимущества по сравнению с однопроцессорной ЭВМ.

В свою очередь, параллельный поиск на наилучшее соответствие, поиск максимального (минимального) элемента данных, параллельное выполнение многих практически важных алгоритмов (например, итерационных) позволяют получить максимально возможное ускорение (по сравнению с последовательным вычислением), пропорциональное корню квадратному из размера данных N. Такое ускорение достигается на одномерной системе, содержащей $\sqrt{N}$ процессорных элементов [6]. Таким образом, использование устройств с N параллельно вычисляющими элементами (как в АЗУ) снижает эффективность использования элемента (отношение ускорения поиска к числу процессорных элементов) до значения $1/\sqrt{N}$. При оптимальном распараллеливании (т. е. с $\sqrt{N}$ вычисляющими элементами) указанный показатель близок к 1, а временная сложность, например, алгоритма поиска на наилучшее соответствие равна $O(\sqrt{N}\log N)$ и не может быть уменьшена при использовании любых других архитектур систем с произвольным числом процессорных элементов.

В данной работе дается оценка трудоемкости поиска (числа обращений к памяти) при использовании ассоциативного кодирования, основанная на r-разрядных обращениях ($1 \le r \le m$, m — число разрядов в слове) к памяти с произвольной выборкой; показано, что минимальное время поиска на точное соответствие достигается не параллельным поиском в АЗУ, а при использовании хеширования, реализуемого памятью с адресацией по содержанию, включающей ЗУПВ и процессор.

1. **Методика ассоциативного кодирования.** Существуют два способа адресации по содержанию. Первый из них базируется на распределении памяти в зависимости от содержания данных и реализуется на ЭВМ с помощью программных средств, второй основывается на использовании специализированных процессоров, ориентированных на поиск данных.

«Ассоциативное кодирование» предусматривает занесение и поиск информации в соответствии с некоторой функцией расстановки $f(q(x))$, указывающей по значению признака (ключа) q элемента x из массива данных адрес участка памяти, где размещается этот элемент.

Временная сложность операции поиска информации в памяти большой емкости определяется временем обращения к памяти, так как время выполнения вычислительной операции связано логарифмической зависимостью с емкостью памяти, а время выборки элемента данных из памяти пропорционально корню квадратному из ее емкости.

Известно, что трудоемкость поиска элемента данных в линейно упорядоченном массиве из N элементов осуществляется за $\log_2 N$ обращений к памяти. Однако если размещение элементов массива в памяти производить независимо друг от друга, руководствуясь лишь значением признака, характеризующего каждый элемент, то среднее число обращений к памяти при поиске на точное соответствие может незначительно отличаться от 1. Экономия времени поиска в $\sim \log_2 N$ раз по сравнению с поиском в упорядоченном массиве достигается, если допустить многократные попытки адресования (при записи и считы-

вании) предъявленного слова за счет введения в функцию расстановки второго аргумента (j), принимающего значения натурального ряда чисел. При фиксированном значении аргумента j двум различным ключевым словам q_1 и q_2 функция расстановки может сопоставить один и тот же адрес, т. е. может возникнуть коллизия обращения к памяти (наложение адресов). Поэтому для одного из конфликтующих слов (обычно для второго) необходимо вычислить новый адрес.

Качество функции хеширования определяется ее способностью минимизировать число коллизий. Очевидно, что это может быть достигнуто путем равномерного распределения значений хеш-функции в заданном диапазоне чисел. Существующие методы устранения наложений можно разбить на две группы. Первая группа методов, получившая название открытой (или внутренней) адресации, предполагает, что при появлении наложения осуществляется поиск свободной ячейки в пределах той же области памяти, к которой происходит обращение в соответствии с используемой хеш-функцией. Адрес свободной ячейки формируется путем добавления к хеш-адресу хеш-смещения, величина которого определяется по некоторому правилу. Например, алгоритм двойного (равномерного) хеширования обеспечивает равномерное появление любых допустимых пар (хеш-адрес, хеш-смещение). Процедура, посредством которой в методе внутренней адресации находятся свободные ячейки, получила название пробинга. Трудоемкость методов поиска (успешного) первой группы (за исключением линейного метода) есть $T_1' = \frac{-\ln(1-z)}{z}$ [1—3]. Для линейного пробинга трудоемкость $T_{\text{ЛП}}' = \frac{1-z/2}{1-z}$ [1].

Вторая группа методов использует списковую организацию памяти для устранения наложений. С этой целью все разряды ячейки памяти делятся на две части: информационную и адресную. В первую часть заносится информация о самом первом элементе данных, который был распределен по вычисленному адресу. Во второй части при наличии повторного обращения по этому адресу указывается адрес ячейки, куда следует поместить наложившийся элемент данных. При наличии трехкратного наложения второй элемент данных снабжается указателем перехода на третий элемент и т. д. Поиск свободной ячейки для записи наложившегося элемента может осуществляться любым из методов первой группы либо путем использования дополнительной памяти. Извлечение же нужного элемента из памяти осуществляется всегда путем просмотра сформированного списка без вычисления значений функции расстановки. Трудоемкость методов поиска (успешного) второй группы $T_2' = 1 + z/2$ [1].

В методике ассоциативного кодирования алгоритм $A_{\text{зап}}$ снабжается хеш-функцией. Хеш-функция $f(w, j)$ для слова w и натурального j ($j = 1, 2, \ldots$) строит $\lceil \log_2 N + a \rceil$-разрядное слово ($N$ — число слов в массиве, $a > 0$ и определяется коэффициентом заполнения памяти). Подробные характеристики различных хеш-функций и алгоритмов пробинга приведены в [1—4, 7], откуда можно заключить, что при $z \lesssim 0{,}5$ и $m/\log_2 N \gtrsim 3$ даже простейший линейный метод обеспечивает показатели, близкие к равномерному хешированию, при минимальном времени вычисления хеш-адреса (методом свертки), равном $\sim \delta \log_2 N$, где δ — задержка вентиля.

Хеш-функция $f(w, j)$ обладает тем свойством, что для каждого j она отображает множество всех 2^m входных слов с равномерной плотностью распределения на множество, состоящее из $2^{\lceil \log_2 N + a \rceil}$ выходных слов (адресов), причем для различных j отображения независимы. Для этой цели можно использовать любой известный метод, обеспечивающий равномерное распределение вычисленных адресов, например метод случайного пробинга [1, 2].

Будем считать, что запоминающее устройство с произвольной выборкой образовано m-разрядными регистрами (ячейками) в количестве $M = 2^{\lceil \log_2 N + a \rceil}$. Пусть алгоритм $A_{\text{зап}}$ уже занес в память слова $w_1, w_2, \ldots, w_k$ и переходит к записи слова w_{k+1}:

$A_{зап}$: Вычислить $f(w_{k+1}, 1)$. Если ячейка с этим адресом пуста, занести в нее w_{k+1}. Если эта ячейка занята, повторить то же самое с $f(w_{k+1}, 2)$, $f(w_{k+1}, 3)$, ... и т. д. до тех пор, пока не будет найдена свободная (пустая) ячейка $f(w_{k+1}, j)$; в нее занести w_{k+1}, а в разряд признака занятости (например, первый) записать «1».

$A_{поиск}$: Вычислить $f(w, 1)$. Если w находится в этой ячейке, то w принадлежит множеству данных. Если $f(w, 1)$ содержит другое слово, отличное от w, то повторить то же самое с $f(w, 2)$, $f(w, 3)$, ... и т. д. до тех пор, пока не будет найдено или слово w, или пустая ячейка.

2. **Трудоемкость поиска на точное соответствие.** Оценку трудоемкости поиска будем искать при условии, когда на первом шаге считывается один разряд занятости, а на последующих шагах — r разрядов ($1 \le r \le m-1$). Поразрядное считывание ($r = 1$) дает возможность уменьшить число внешних выводов памяти до минимально возможной величины, а m-разрядное считывание позволяет увеличить быстродействие памяти до максимально возможной величины.

Пусть Z — доля занятых ячеек ($Z = N/M$), а $S = (1 - Z)$ — доля свободных. В процессе заполнения памяти доля свободных ячеек есть переменная s, меняющаяся от 1 до S. Среднее число обращений, приходящихся на одну запись в память, равно интегральному среднему от функции $f(s) = 1/s$ на интервале $[1, S]$:

$$T'_s = \frac{1}{S-1}\int_1^S \frac{ds}{s} = -\frac{\ln S}{1-S}.$$

При поиске слова, не принадлежащего множеству данных (предъявлено «чужое» слово), во всех $1/S$ просматриваемых ячейках, вплоть до попадания в свободную, просматривается первый разряд — признак занятости («1») или незанятости («0»). На это требуется в среднем $1/S$ одноразрядных обращений к памяти. В $(1/S - 1)$ ячейках производится поразрядная сверка, причем второй разряд проверяется обязательно. Вероятность того, что этот разряд совпадает, равна $1/2$. Вероятность того, что второй разряд совпадает, а третий не совпадает, равна $1/2 \cdot 1/2 = 1/2^2$. Вероятность совпадения трех первых и несовпадения четвертого разрядов равна $1/2^3$ и т. д. Отсюда среднее число обращений на поразрядной сверке есть

$$\sum_{i=1}^{m-1} i/2^i \le 2.$$

Пусть при сверке считываются группы по r разрядов ($1 < r \le m-1$). Вероятность того, что все r разрядов совпадут, равна $1/2^r$, а вероятность того, что хотя бы один из них не совпадет, равна $1 - 1/2^r$. Вероятность того, что $(i-1)$ последовательных групп совпадут, а i-я — не совпадет (т. е. будет i обращений на сверке), есть $(1 - 2^{-r})2^{-r(i-1)}$. Отсюда среднее число r-разрядных обращений на сверке «чужого» слова есть

$$C_r = \sum_{i=1}^{\lceil (m-1)/r \rceil} \frac{i(1-2^{-r})}{2^{r(i-1)}}.$$

Общее число обращений в среднем на одно «чужое» слово (т. е. среднее число обращений при неудачном поиске) есть

$$T'_н = \frac{1}{S} + \left(\frac{1}{S} - 1\right) C_r = \frac{1 + ZC_r}{S}.$$

При считывании за одно обращение всех m разрядов слова $C_r = 0$ и $T'_н = 1/S$.

При поиске слова, принадлежащего множеству данных (т. е. когда предъявлено «свое» слово), в точности повторяется процесс, имевший место при заполнении данной памяти. Среднее число проверок на одно слово равно

$\frac{-\ln S}{Z}$. Различие состоит лишь в том, что при заполнении искалась пустая ячейка, а при распознавании она содержит искомое слово. На каждой из $\frac{-\ln S}{Z}$ ячеек обязательно проверяется первый разряд (это составит $\frac{-\ln S}{Z}$ обращений) и затем выполняется r-разрядная сверка. Всего будет $\left(\frac{-\ln S}{Z} - 1\right)$ несовпадающих слов и одно (последнее) совпадающее. На несовпадающих словах среднее число r-разрядных обращений есть C_r. На совпадающем слове проверяются все $(m - 1)$ разрядов, что составляет $\lceil (m-1)/r \rceil$ обращений. Итого среднее число r-разрядных обращений при удачном поиске (при распознавании «своего» слова) равно

$$T'_{\text{у}} = \frac{-\ln S}{Z} + \left\lceil \frac{-\ln S}{Z} - 1 \right\rceil C_r + \left\lceil \frac{m-1}{r} \right\rceil.$$

$2^{\lceil \log_2 N \rceil - m}$, то при случайном поступлении распознаваемых слов среднее ожидаемое число обращений к ЗУПВ есть

$$T'_0 = T'_{\text{н}}(1 - 2^{\lceil \log_2 N \rceil - m}) + T'_{\text{у}} 2^{\lceil \log_2 N \rceil - m} \approx T'_{\text{н}},$$

так как обычно величина $2^{\lceil \log_2 N \rceil - m}$ пренебрежимо мала.

Отметим, что среднее ожидаемое число поразрядных сравнений определяется неудачным поиском и составляет $(1 + 2Z)/S$, так как при поразрядном сравнении (обращении) $r = 1$ и $C_r = 2$. При $Z = 0{,}5$ среднее ожидаемое число поразрядных сравнений (а следовательно, и поразрядных обращений к памяти) равно 4. Отсюда следует, что при считывании из памяти за одно обращение четырех разрядов слова минимальная трудоемкость поиска, равная 1, достигается уже при двухкратной избыточности памяти.

3. Временная сложность алгоритма поиска на точное соответствие. 1. Без учета ограничений, связанных с теплоотводом, оба подхода — хеширование и параллельный поиск — характеризуются примерно одинаковой временной сложностью, равной $O((mM)^{1/d})$, где d — размерность памяти. Однако из-за необходимости отвода тепла ситуация коренным образом меняется не в пользу АЗУ.

Действительно, поскольку отвод тепла осуществляется через поверхность, то при параллельной работе всех mM вентилей занимаемая ими площадь не может быть меньше величины mMp/ρ, где p — мощность, потребляемая вентилем при его переключении (предлагается, что мощность, рассеиваемая в статике, пренебрежимо мала), а ρ — удельная мощность рассеяния, ограниченная условием теплоотвода. Время поиска в АЗУ пропорционально линейному размеру памяти и составляет $O(\sqrt{mMp/\rho})$.

Если пренебречь мощностью, рассеиваемой в статике, то энергопотребление ЗУПВ в основном определяется мощностью, требуемой на переключение m вертикальных и m горизонтальных шин считывания/записи. Результирующая мощность, потребляемая ЗУПВ, равна $O(m\sqrt{M}p)$, что в $O(\sqrt{M})$ раз меньше, чем в случае АЗУ.

При достаточно высокой частоте обращений к памяти всегда можно разместить все ЗУПВ в пределах площади, обеспечивающей необходимый теплоотвод. При этом линейный размер ЗУПВ составит $O(\sqrt{m\sqrt{M}p/\rho})$. Таким образом, выигрыш в быстродействии ЗУПВ по сравнению с АЗУ равен $O(\sqrt{mMp/\rho})/O(\sqrt{m\sqrt{M}p/\rho}) = O(M^{1/4})$. Сравнение по комплексному критерию η, определяемому произведением площади П, занимаемой памятью, на время поиска T дает

$$\eta = (\Pi T)_{\text{АЗУ}}/(\Pi T)_{\text{ЗУПВ}} = O((mMp/\rho)^{3/2})/O((m\sqrt{M}p/\rho)^{3/2}) = O(M^{3/4}).$$

Поскольку поиск на точное соответствие в ассоциативной памяти (в том числе и оптоэлектронной) неэффективен, использование такой памяти вместо ЗУПВ в многопроцессорной вычислительной системе (ВС) (например, в ВС, процессоры которой взаимодействуют с памятью через коммутатор), вопреки устоявшемуся мнению, не может повысить производительность ВС, включая ВС нетрадиционной архитектуры (например, потоковые).

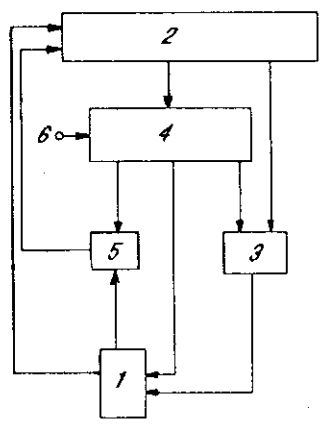
2. Резервы повышения быстродействия памяти. Собственные задержки современных вентилей на два-три порядка меньше, чем длительность такта цифровых устройств. Основным препятствием на пути уменьшения тактового периода являются задержки в межвентильных соединениях и межвентильные глобальные взаимодействия, реализуемые посредством общих шин (например, шин считывания информации из памяти).

ЗУПВ характеризуется двумя задержками, одна из которых обусловлена дешифрацией адреса (задержка на дешифрацию), а другая — доставкой элемента данных (задержка на передачу). При условии, когда можно пренебречь сопротивлением межсоединения, каждая из задержек пропорциональна линейному размеру запоминающей матрицы, содержащей один разряд всех M слов, т. е. $\sim \sqrt{M}$. Если собственная задержка вентиля больше, чем задержка на соединении, то задержка на дешифрацию увеличится на величину $0{,}5\delta\log_2 M$. Использование оптических шин при дешифрации адреса (для выполнения функций координатных шин) и считывания информации естественно не может устранить зависимость времени выборки информации от линейного размера памяти, однако позволит существенно уменьшить (вследствие нейтральности фотонов) коэффициент пропорциональности. При этом задержка на передачу снижается до минимально возможной величины, равной примерно $(\delta + l\sqrt{M}/v_c)$, где l — линейный размер вентиля, v_c — скорость света в оптической шине. Для полупроводниковой памяти задержка на передачу оценивается величиной $O(\delta\sqrt{M})$. Выигрыш в быстродействии при использовании оптических шин составит $O(\delta\sqrt{M})/O(l\sqrt{M}/v_c) = O(\delta/\tau)$, где $\tau = l/v_c$ — время распространения сигнала по линейному размеру элемента памяти. Учитывая, что собственная задержка транзистора определяется временем дрейфа носителей заряда, можно заключить, что потенциально возможный выигрыш в быстродействии при использовании оптических шин ограничен величиной, определяемой отношением скорости света v_c в оптической шине к дрейфовой скорости $v_д$ носителей заряда в полупроводнике. Например, при $v_c = 10^{10}$ см/с и $v_д = 10^7$ см/с выигрыш в быстродействии ограничен величиной $\sim 10^3$. Реальный выигрыш в быстродействии будет, возможно, на порядок меньше, так как межвентильные соединения реализуются посредством проводников.

Отметим, что эффективность использования оптических шин зависит от многих факторов, например сложности (емкости) элемента (модуля) памяти, его быстродействия и линейного размера, параметров оптико-электронных приборов, физико-технологических особенностей изготовления приборов и т. п. Для памяти большой емкости (10^{10} бит и более) с быстродействием $\sim 10^8$ Гц емкость каждого из модулей памяти, которые целесообразно объединять посредством оптических шин, составляет $\sim 10^3$ бит, так как собственное быстродействие модуля такой емкости в чисто полупроводниковом исполнении (без использования оптических шин) может составлять 10^9 Гц. При соединении таких модулей посредством оптических шин могут быть использованы модуляторы света с линейным размером $\sim 10^{-3}$ см.

4. **Структурная схема памяти с адресацией по содержанию.** Структурная схема памяти приведена на рисунке [8]. Запоминающее устройство содержит блок памяти *1*, блок управления *2*, блок формирования адреса *3*, входной регистр *4* и схему сравнения *5*, входы которой подключены к выходам регистра

4 и блока памяти 1. Выход регистра 4 подключен к информационным входам блока памяти и блока формирования адреса 3, подключенного своим выходом к адресным входам блока памяти. Выход схемы сравнения 5 и один из информационных выходов блока памяти 1 соединены с соответствующими входами блока управления.



Функционирование ЗУ рассмотрим в режиме полноразрядных обращений к блоку памяти 1.

Режим записи. Записываемое слово поступает с входа 6 в регистр 4. Часть этого слова (признак) или все слово с выхода регистра поступает на вход блока формирования адреса 3, вычисляющего функцию расстановки $f(w, j)$, где w — код признака, j — натуральный параметр.

Функция $f(w, j)$ каждому слову w ставит в соответствие некоторый адрес блока памяти 1, который с выхода блока формирования адреса 3 поступает на вход адреса блока памяти (вначале вычисляется адрес $f(w, 1)$). Далее производится чтение слова по данному адресу. Если соответствующая ячейка памяти свободна (признаком занятости служит «1» в первом разряде считываемого слова, устанавливаемая при записи слова в эту ячейку), то слово с регистра 4 записывается в данную ячейку. В случае занятости ячейки блок управления 2 по сигналу из блока памяти 1 переключает блок формирования адреса 3 на вычисление нового значения функции $f(w, 2)$ и снова производятся чтение и проверка занятости ячейки по адресу $f(w, 2)$ и т. д.

Режим выборки. Признак w, по которому производится выборка, поступает с входа 6 в регистр 4. Блок формирования адреса 3 вычисляет для данного признака значение функции $f(w, 1)$ и выдает его на вход адреса блока памяти 1. Далее производится чтение слова по данному адресу, которое сравнивается с признаком w с помощью схемы 5. Если сравнение успешно, то данное слово является результатом выборки. В противном случае производится проверка занятости данной ячейки памяти. Если она свободна, то это означает, что слово с признаком w в памяти отсутствует. Если же ячейка занята, то блок управления 2 по сигналу из блока памяти 1 переключает блок формирования адреса 3 на вычисление нового значения функции $f(w, 2)$ и снова производится чтение и сравнение с признаком. В случае успешного сравнения это слово является результатом выборки.

В заключение отметим, что для ЭВМ следующего поколения требуется память емкостью ~ 10^{10} слов с временем выборки ~ 10^{-8} с. Такие параметры могут быть достигнуты только при изготовлении памяти в виде больших полупроводниковых пластин диаметром ~ 150 мм. Для устранения электронных (зарядовых) межвентильных глобальных взаимодействий, снижающих быстродействие памяти, функции глобальных шин целесообразно реализовать посредством оптических шин, что на несколько порядков увеличит (вследствие нейтральности фотонов) быстродействие памяти. Большее, по сравнению с ассоциативной памятью, быстродействие ЗУПВ, достигаемое за счет более высокой (на несколько порядков выше, чем в случае АЗУ) плотности размещения элементов, позволит существенно сократить время поиска.

СПИСОК ЛИТЕРАТУРЫ

1. **Кнут Д.** Искусство программирования для ЭВМ. Т. 3. Сортировка и поиск.—М.: Мир, 1978.
2. **Кохонен Т.** Ассоциативные запоминающие устройства.—М.: Мир, 1982.
3. **Кричевский Р. Е.** Сжатие и поиск информации.—М.: Радио и связь, 1989.
4. **Величко В. М., Гусев В. Д.** Ассоциативное кодирование: реализация и применение // Вычислительные системы.—Новосибирск, 1975.—Вып. 62.
5. **Минский М., Пейперт С.** Персептроны.—М.: Мир, 1971.

6. **Мишин А. И., Седухин С. Г.** Вычислительные системы и параллельные вычисления с локальными взаимодействиями // Вычислительные системы.—Новосибирск, 1979.—Вып. 78.

7. **Гусев В. Д., Титкова Т. Н.** Экспериментальные исследования эффективности функций расстановки // Вычислительные системы.—Новосибирск, 1978.—Вып. 74.