Глава 2. Система памяти ЭВМ.

Курс лекций «ЭВМ и периферийные устройства»

Лектор - доцент, к.т.н.
Кузьмин Александр Александрович

Система памяти и запоминающие устройства ЭВМ
Базовые определения

Памятью ЭВМ условимся называть всю совокупность устройств, функционально предназначенных для запоминания, хранения и выдачи информации (в форме числовых и нечисловых кодов).

Отдельные устройства, входящие в эту совокупность, условимся называть устройствами памяти или запоминающими устройствами того или иного назначения, или типа.

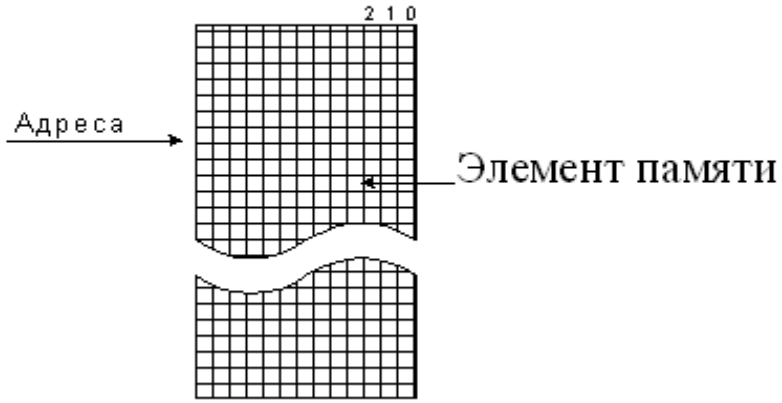
Содержимое памяти:

- коды команд (программ);
- данные (константы, исходные, промежуточные и результаты, представленные в определенных форматах);
- адреса (коды адресов, формализованные ассоциативные признаки, критерии поиска, дескрипторы и т.д.);
- коды состояний устройств ЭВМ, подсистем и всей вычислительной системы.

Понятие линейной адресации

Элементарное описание линейной адресации:

элементы памяти образуют многоразрядные (многобитные) ячейки памяти, нумеруемые по порядку расположения (последовательным, «линейным» перечислением).



Предполагается, что номер соответствует «линейному адресу». Линейно в памяти можно адресовать слова информации или их части, например байты, образующие слова информации.

Важнейшие понятия: элемент памяти, бит, байт, ячейка памяти, адрес ячейки памяти, машинное слово, поле, запись, файл, массив.

Типы ЗУ и их основные характеристики

Основным требованием к ЗУ является надежное хранение информации без изменения ее начального содержания в течение продолжительного времени и своевременное предоставление по запросу.

Основные характеристики любого ЗУ – емкость и быстродействие.

Быстродействие ЗУ оценивается временем обращения (за информацией) или временем цикла:

$$t_{обр}^{чт} = t_{дост}^{чт} + t_{счит} + t_{рег}$$

$$t_{обр}^{зп} = t_{дост}^{зп} + t_{подг} + t_{зап}$$

Время обращения - минимальный временной интервал между двумя последовательными обращениями к ЗУ:

$$t_{обр} = \max(t_{обр}^{чт}, t_{обр}^{зп})$$

Время выборки: $t_{выб} = t_{поиска} + t_{сч}$

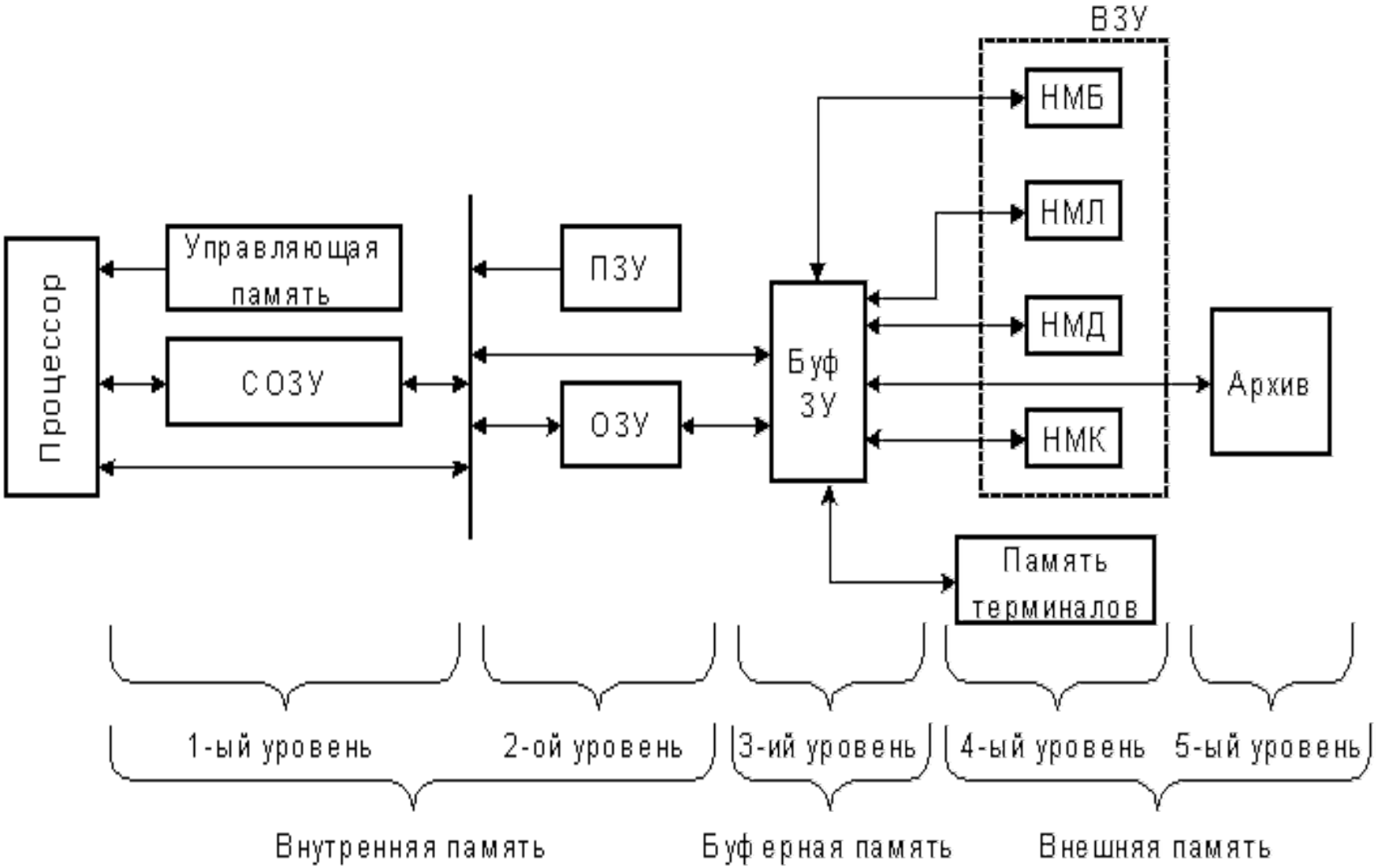
Регенерация (восстановление) может быть необходима в случае считывания с разрушением, а также в особых случаях технологии ЗУ.

Иерархия системы памяти

Каждое конкретное ЗУ при любом его физическом исполнении обладает определенными значениями быстродействия и емкости. С увеличением емкости быстродействие уменьшается, что объясняется, в частности, усложнением поиска требуемой информации.

Иерархическая структура памяти предполагает объединение нескольких ЗУ различных типов, отличающихся по емкости и быстродействию, в общую память вычислительной системы (машины).

Иерархическая структура ЗУ ЭВМ позволяет *сочетать хранение большого объема* информации с *быстрым доступом* к информации в процессе ее обработки, что способствует в совокупности *повышению производительности* (глобально – эффективности) вычислительных систем.


ВЗУ
НМБ
НМЛ
НМД
НМК
Архив
Процессор
Управляющая память
СОЗУ
ПЗУ
ОЗУ
Буф ЗУ
Память терминалов
1-ый уровень
2-ой уровень
3-ий уровень
4-ый уровень
5-ый уровень
Внутренняя память
Буферная память
Внешняя память

Классификации запоминающих устройств

По специфике использования в ЭВМ (функциональному назначению):

1) ЗУ со сменой информации в процессе работы (СОЗУ, ОЗУ, БЗУ, ВЗУ);

2) ЗУ без смены информации (ПЗУ или односторонние ЗУ);

3) ЗУ с медленной сменой информации (программируемые ПЗУ или ЗУ БРИ, то есть ЗУ без разрушения информации при считывании) – запись возможна, но специальными средствами и медленно.

По способу размещения и способу поиска **информации**:

- адресные;
- безадресные:
  - а) ассоциативные;
  - б) стековые и магазинные;
  - в) ортогональные.

По способу обращения к ячейкам **запоминающей среды**:

1) ЗУ с произвольным доступом;

2) ЗУ с последовательным доступом:

- периодические (циклические), ещё их именуют ЗУ с прямым доступом;
- апериодические или ЗУ с последовательным доступом.

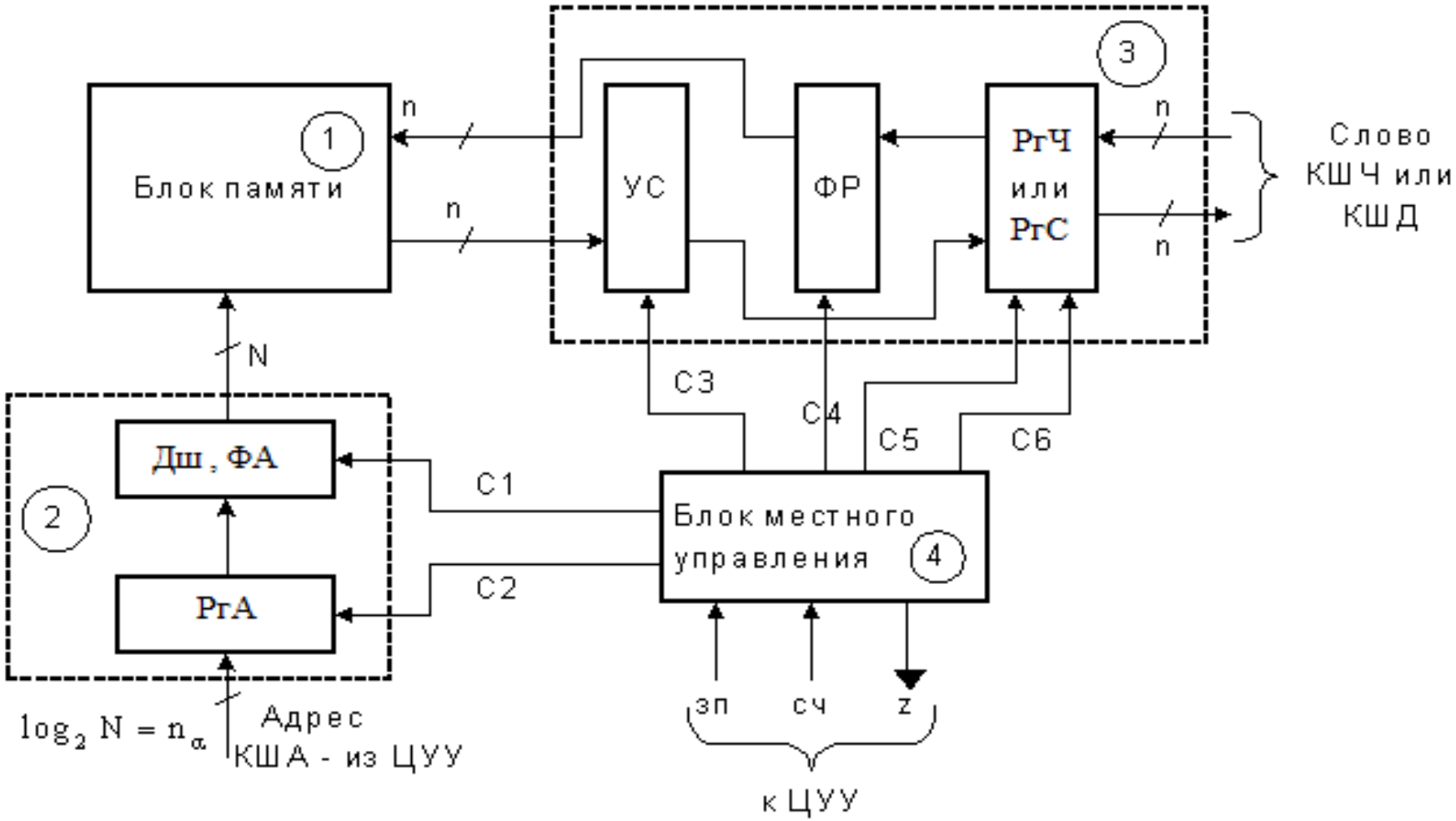
Оперативные запоминающие устройства

Общие принципы организации

Элемент физической среды, использующийся для хранения единицы информации, именуется элементом памяти.

Совокупность элементов памяти называется запоминающей средой или запоминающим массивом. В этом понятии не определена структура совокупности, правила распределения элементов в массиве.

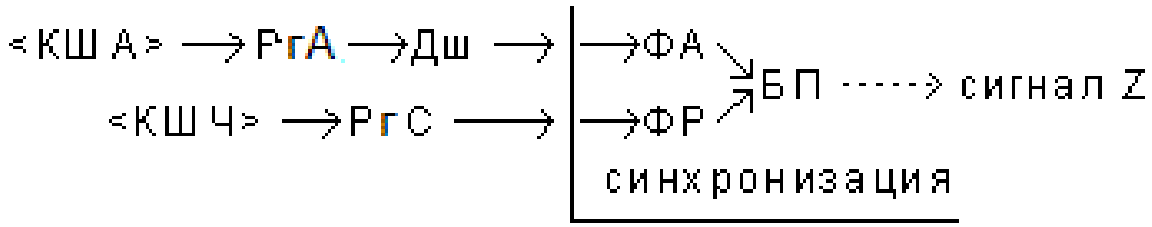
Организация ОЗУ с произвольным доступом показана далее на рисунке.


1
Блок памяти
n
n
3
УС
ФР
РгЧ
или
РгС
n
n
Слово
КШ Ч или
КШ Д
N
2
Дш , ФА
РгА
С1
С2
С3
С4
С5
С6
Блок местного
управления
4
зп
сч
z
к ЦУУ
log2 N = nα
Адрес
КШ А - из ЦУУ

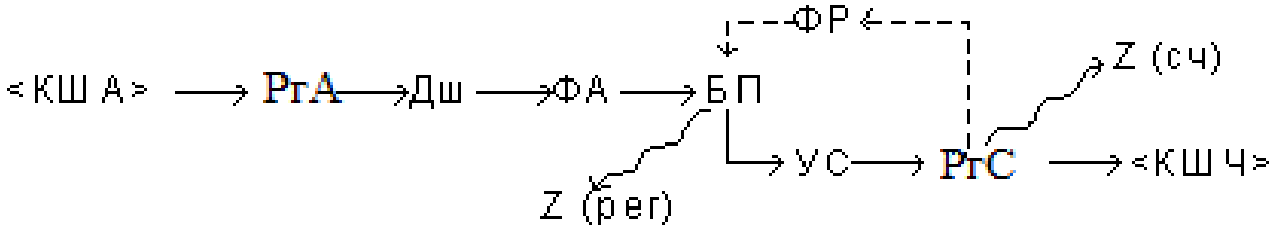
Общие принципы организации ЗУ

Условные диаграммы работы ЗУ с произвольным доступом в режимах записи и считывания информации:

Запись



Считывание



БИС ОЗУ с произвольным обращением

I.

Достоинства:

- высокое быстродействие;
- компактность;
- достижима низкая стоимость (при больших партиях);
- легко обеспечиваемая совместимость по сигналам с другими схемами ЭВМ;
- общие с другими элементами ЭВМ конструкционные и технологические принципы.

Недостаток – энергозависимость.

По технологии изготовления и типу запоминающих элементов интегральные схемы весьма различны.

II.

Таблица. Сравнение технологий полупроводниковых ЗУ в относительных единицах.

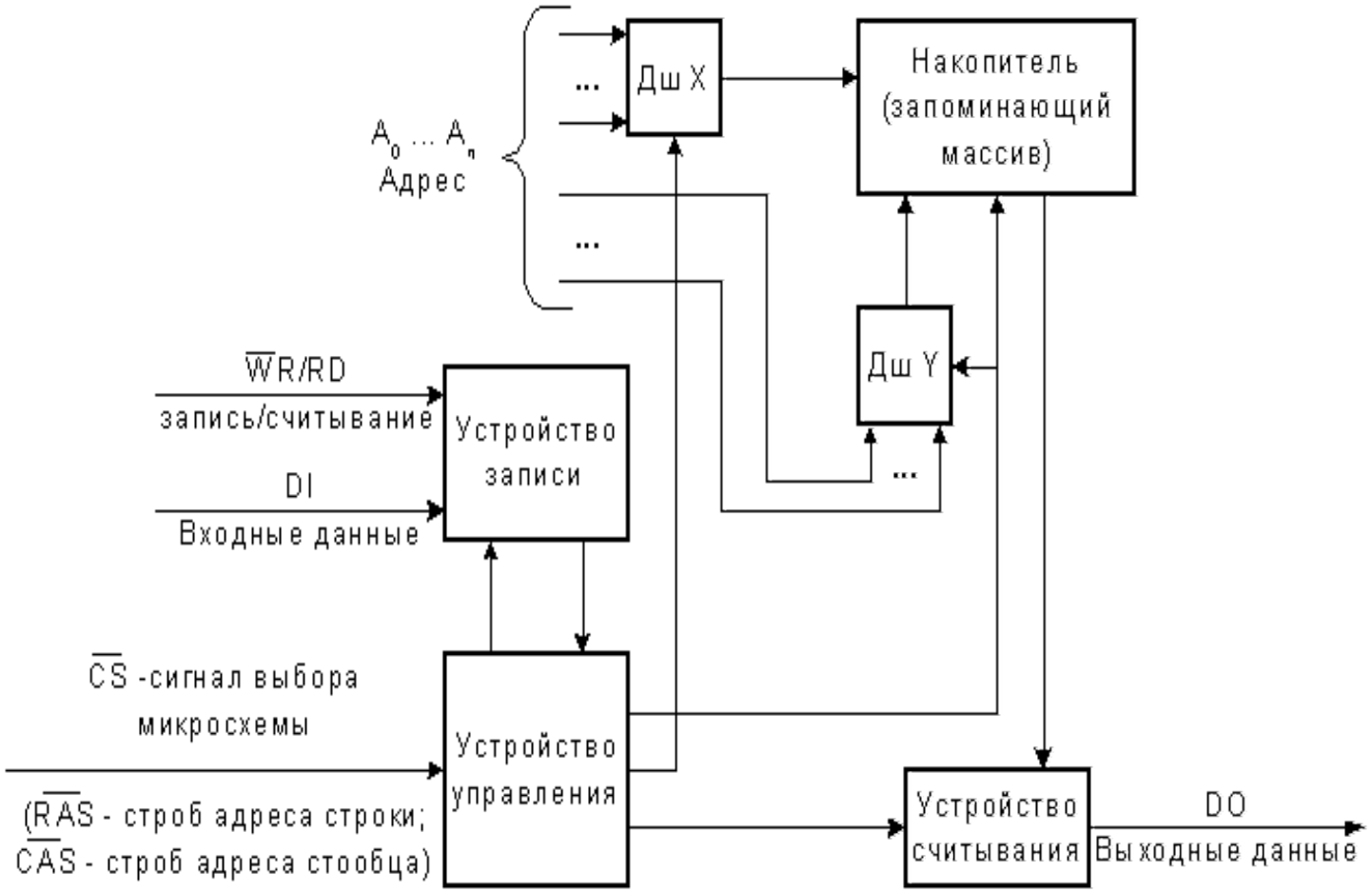
Типы технологий элементов БИС ЗУ	**Емкость**	**Время выборки**	**Потребляемая мощность**
Статические ЭСЛ	1	1	120
Статические ТТЛ	4	5	120
Статические ИИЛ	4	15	60
Статические n-, p-МОП	4	6	10
Статические КМОП	8	20	4
Динамические МОП	512	10	1

III.

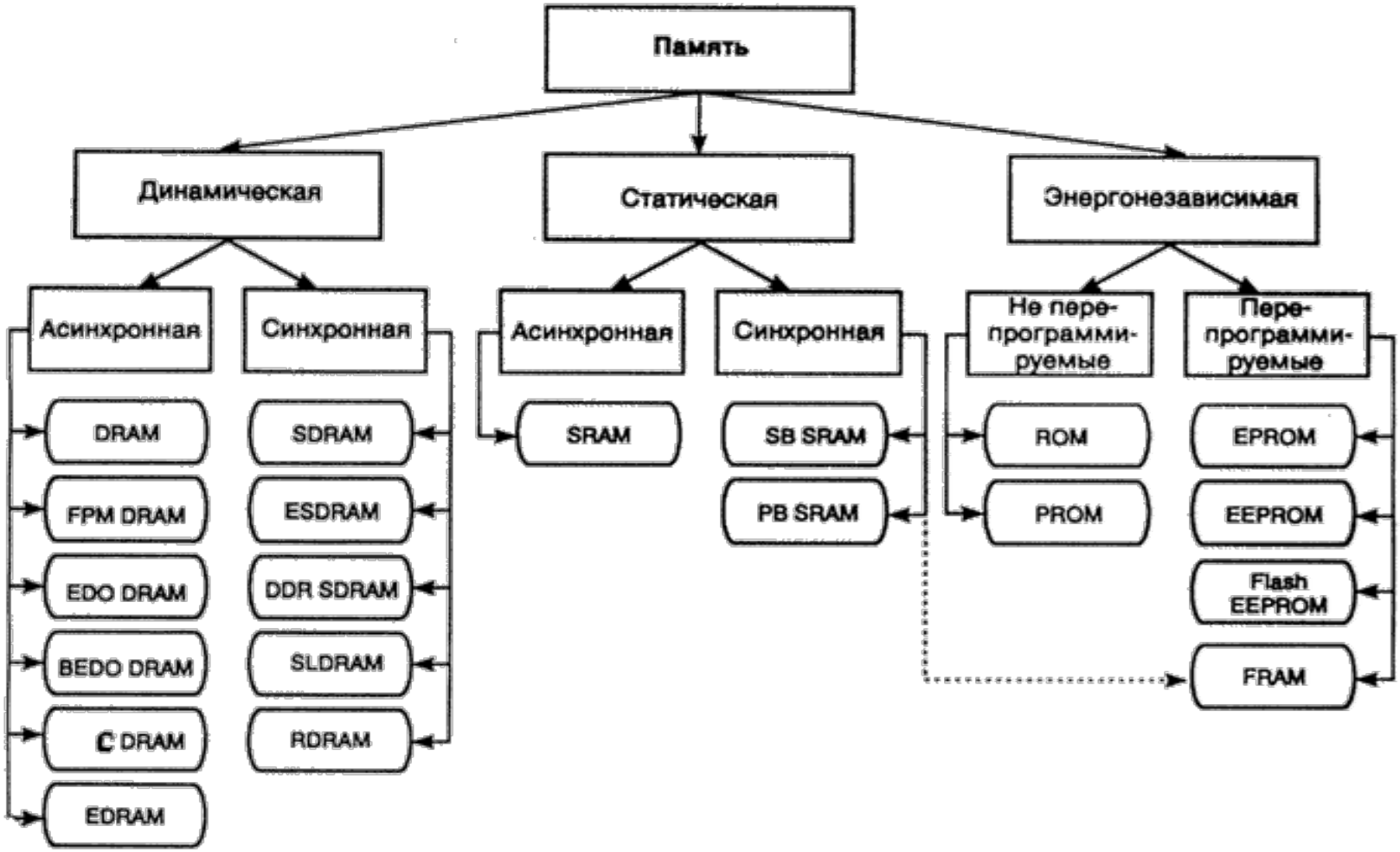
Таблица. Области применения БИС ЗУ.

Тип технологий элементов полупроводниковых ЗУ	ОЗУ супер-ЭВМ	ОЗУ боль-ших ЭВМ	ОЗУ персональ-ных ЭВМ	СОЗУ, БЗУ	Каналь-ные ЗУ	ОЗУ ПУ
Статические ЭСЛ	+			+		
Статические ТТЛ	+	+		+	+	
Статические ИИЛ			+		+	
Статические n-, p-МОП		+				+
Статические КМОП		+	+			+
Динамические МОП		+	+		+	+

IV. Обобщенная структура ИС ЗУ.



V. Модули памяти и элементы памяти (БИС).



(Ознакомьтесь с приложениями П2.1 и П2.2.)

Контроль функционирования БИС ЗУ.

Под функционированием какого-либо объекта понимается выполнение предписанного ему алгоритма функционирования при применении объекта по назначению.

Функциональный контроль (ФК) решает две основные задачи: определение факта наличия неисправности в объекте (задача контроля работоспособности) и определение места неисправности (задача диагностики).

Методы ФК основаны на сравнении с эталонными сигналами выходных реакций (сигналов) тестируемой схемы на заданные входные воздействия.

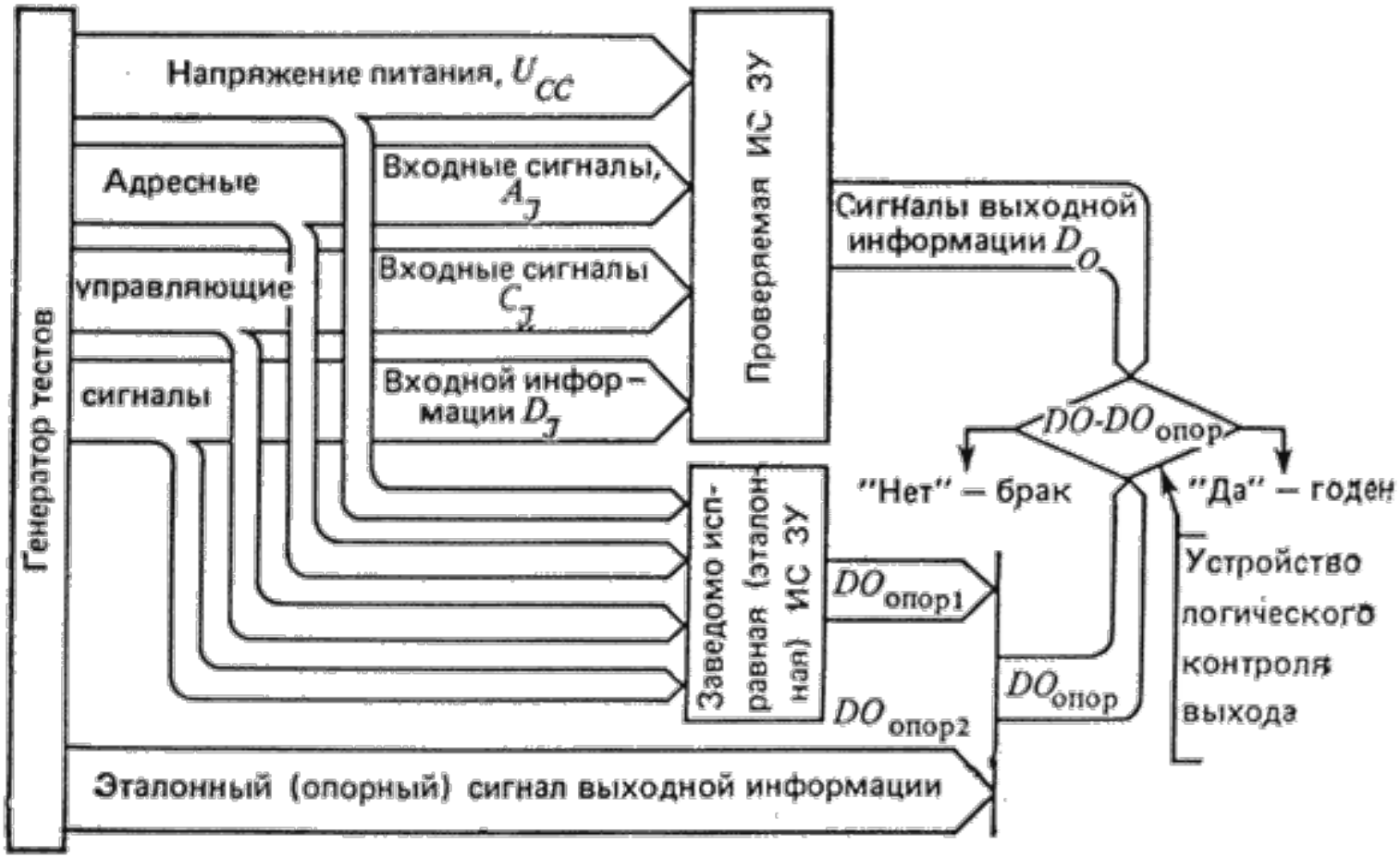
Одним из основных узлов системы ФК является генератор тестов, предназначенный для формирования последовательности тестирующих и эталонных сигналов по заданному «закону» (точнее, алгоритму). В понятие теста включают состав, параметры и порядок следования электрических сигналов, подаваемых на испытуемую схему с целью измерения какого-либо параметра или контроля работоспособности.

Наборы входных сигналов, задаваемые в виде машинных слов (кодов), определяют порядок обращения к элементам памяти и последовательность выполняемых операций.

Коды эталонных сигналов должны соответствовать выходным кодам исправной БИС ЗУ при заданных входных воздействиях, т. е. эквивалентность выходных и эталонных сигналов, определяемая путем логического сравнения, указывает на правильность функционирования БИС.

Заключение о правильности функционирования ЗУ делают либо по результатам выполнения очередного элементарного теста («останов по ошибке»), либо по конечному результату выполнения полного теста.

Общая структурная схема ФК приведена далее на рисунке


Генератор тестов
Напряжение питания, U_{CC}
Адресные
управляющие
сигналы
Входные сигналы, A_J
Входные сигналы C_J
Входной информации D_J
Проверяемая ИС ЗУ
Заведомо исправная (эталонная) ИС ЗУ
Сигналы выходной информации D_O
DO-$DO_{опор}$
"Нет" – брак
"Да" – годен
$DO_{опор1}$
$DO_{опор2}$
$DO_{опор}$
Устройство логического контроля выхода
Эталонный (опорный) сигнал выходной информации

ПОЛИТЕХ

Эффективность ФК решающим образом определяется построением теста.

Существуют различные способы генерации тестовых последовательностей для контроля БИС ЗУ. Наиболее широко используются алгоритмические функциональные тесты (АФТ), содержащие последовательность элементарных тестов, изменяемых по выбираемому алгоритму. Это связано с простотой генерации, малым объемом занимаемой памяти управляющей ЭВМ и высокой воспроизводимостью результатов ФК.

Эталонный сигнал выхода ЗУ вырабатывается, как правило, также алгоритмически генератором тестов, но можно использовать и эталонную схему ЗУ. АФТ должны обладать двумя противоречивыми свойствами: с одной стороны, обеспечивать достаточную полноту контроля БИС ЗУ, а с другой — быть достаточно короткими по времени, чтобы обеспечить производительность проверки БИС ЗУ при их большой информационной емкости.

Для обеспечения полного функционального контроля БИС ЗУ необходимо проверить запись/считывание всех возможных комбинаций состояний запоминающих элементов, то есть 2 в степени (N+K), где *N* – число запоминающих элементов, бит; *K*— число функциональных входов.

Непосредственный перебор всех состояний ОЗУ становится нереальным при N+K>64.

По количеству циклов обращения тестируемой схеме, выраженному через ее информационную емкость, алгоритмы ФК условно делятся на три типа: N, N^2, $N^{3/2}$, где N — емкость ЗУ, бит.

(Ознакомьтесь с приложением П2.3.)

При разработке алгоритмов ФК идёт поиск минимальной тестовой последовательности входных сигналов, для которой имеет место изменение выходной последовательности сигналов тестируемой схемы при отказе любого из ее элементов. Решение этой задачи осложняется наличием у БИС ОЗУ ряда неисправностей, не описываемых булевыми функциями (например, множественная выборка), а также неисправностей, связанных с динамическими состояниями элементов.

Для современных СБИС ЗУ эффективное тестирование достигается специальными весьма изобретательными инженерными решениями:

- использование априорных данных распределения элементов и ячеек памяти на кристалле;
- девиация режимных параметров;
- комбинации тестов (в том числе АФТ);
- использование обратной связи о реакции на тестирование и др.

Контроль функционирования может быть совмещен с измерением (контролем) статических и динамических параметров ЗУ, если позволяет точность и быстродействие аппаратуры контроля. Принципиальных методических отличий от уже рассмотренных методов измерения параметров ЗУ в этом случае нет.

Организация многоблочной оперативной памяти

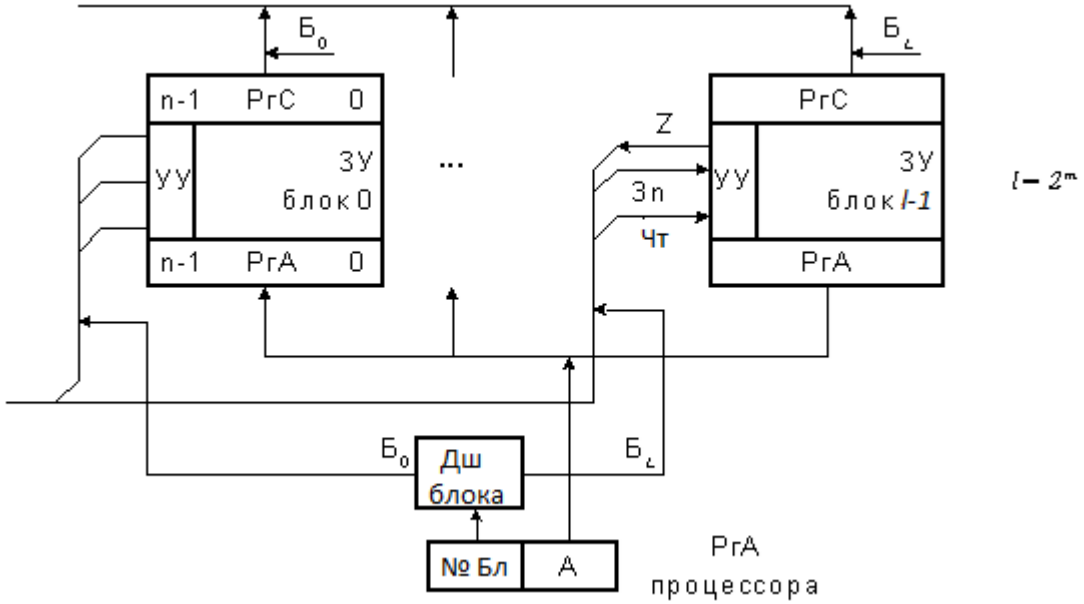
Такая организация, как и иерархия памяти, позволяет повысить быстродействие.

ОЗУ большой емкости обычно выполняют в виде отдельных блоков (модулей) ЗУ, в которые входят РгА и РгС (числа). Если в ЗУ 2^m блоков, в каждом из которых 2^k слов, то суммарная емкость:

2^{k+m} слов.

В функциональном отношении такие ЗУ можно рассматривать как одно с быстродействием одного блока.

Организация многоблочной оперативной памяти



Структура адресного кода:

Совмещение работы во времени разных блоков ЗУ возможно при условии, что каждое последующее обращение происходит к другому (следующему!) по номеру блоку ЗУ.

Основные проблемы организации многоблочной памяти:

- использование блоков при условных и безусловных переходах
- размещение информации в памяти для получения наибольшего эффекта расслоения.

Есть несколько **способов организации**. Наиболее распространены два:

1) Функциональное распределение.
2) Расслоение адресов.

Максимальное число одновременно выполняемых к памяти обращений называется **коэффициентом расслоения**.

Реальные эмпирически полученные значения коэффициентов расслоения в зависимости от классов решаемых задач:

при 2-х блочной ОП – коэффициент –1,2 – 1,3 ; при 4-х блочной - 1,7 – 2,2; при 8-ми - 3,4 – 4,2.

Организация ОЗУ с многоканальным доступом

Ресурсы оперативной памяти используются многими устройствами: процессором и каналами ввода/вывода (под управлением контроллеров). Эти устройства могут (и должны!) функционировать одновременно и независимо друг от друга.

Память, ресурсы которой распределяются между несколькими потребителями, называется памятью с многоканальным доступом.

Выбор канала осуществляется согласно принятому в системе приоритету. Обычно используются относительные приоритеты.

Пусть $\lambda_1, \lambda_2, \ldots, \lambda_m$ – сигналы запросов от 1, 2, …, m каналов. Будем полагать приоритет 1-го канала высшим. Необходимо сформировать $p_1, p_2, \ldots, p_m$ – сигналы *разрешения* доступа к ОП соответственно 1, 2, …, m каналу.

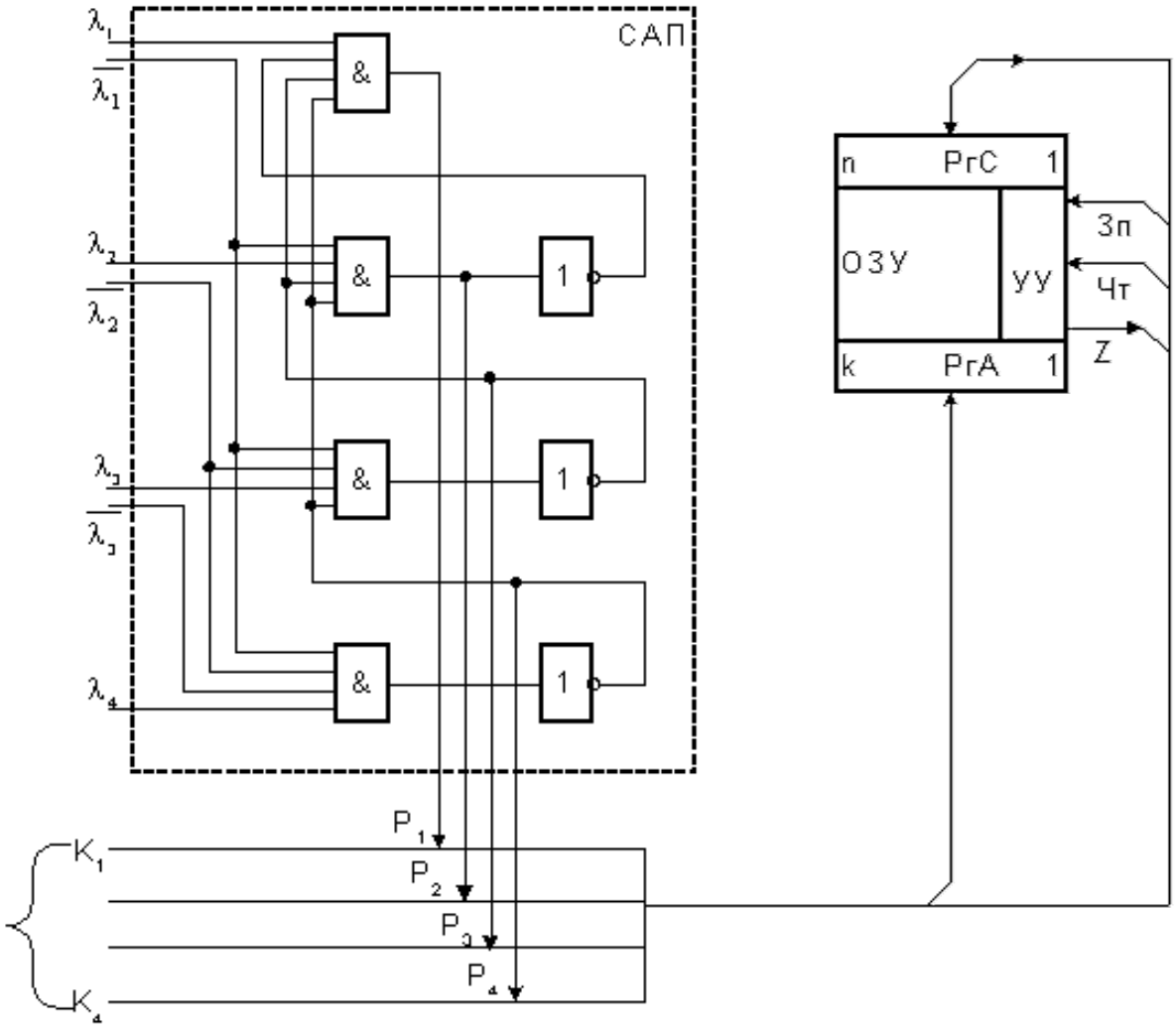
Будем полагать $p_k=1$, если:

- имеется запрос λ_k, т.е. если $\lambda_k=1$;
- отсутствуют запросы более высоких приоритетов $\lambda_1 \ldots \lambda_{k-1}$;
- не начато обслуживание каналов с (k+1)-го по m - ый, т.е. $p_{k+1}=p_{k+2}=\ldots=p_m=0$.

Нетрудно записать логические выражения для формирования сигналов $p_1, p_2, \ldots, p_m$.

Сделайте это самостоятельно.

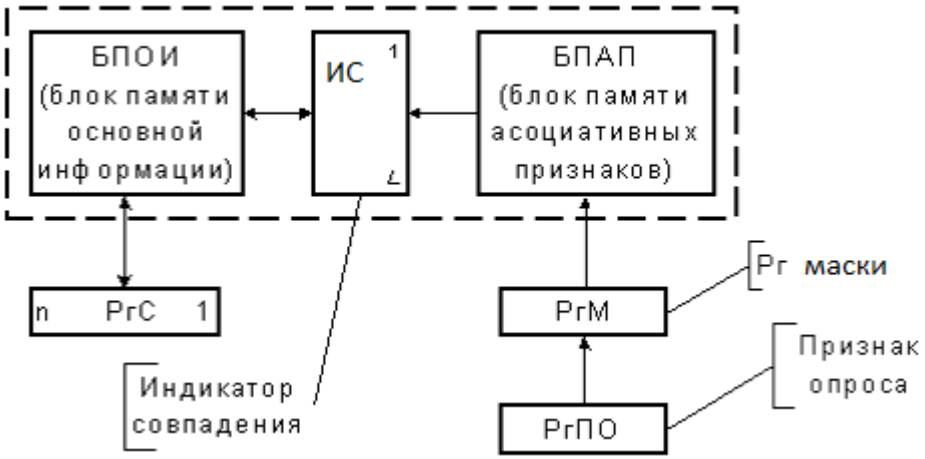
На следующем рисунке показана простая схема анализа приоритетов (САП) доступа для четырёх абонентов ОП.



Ассоциативные ЗУ

В ассоциативных ЗУ (АЗУ) выборка информации производится не по адресу, а по признакам самой информации, либо по некоторым критериям, связанным с искомой информацией. Все это называется ассоциативным поиском.

Следует заметить, что в больших информационно-поисковых системах (ИПС) при большом объеме памяти ныне ассоциативный поиск реализуется программно.



На рисунке: ПО – ассоциативный признак опроса или дескриптор.

Сверхоперативные ЗУ

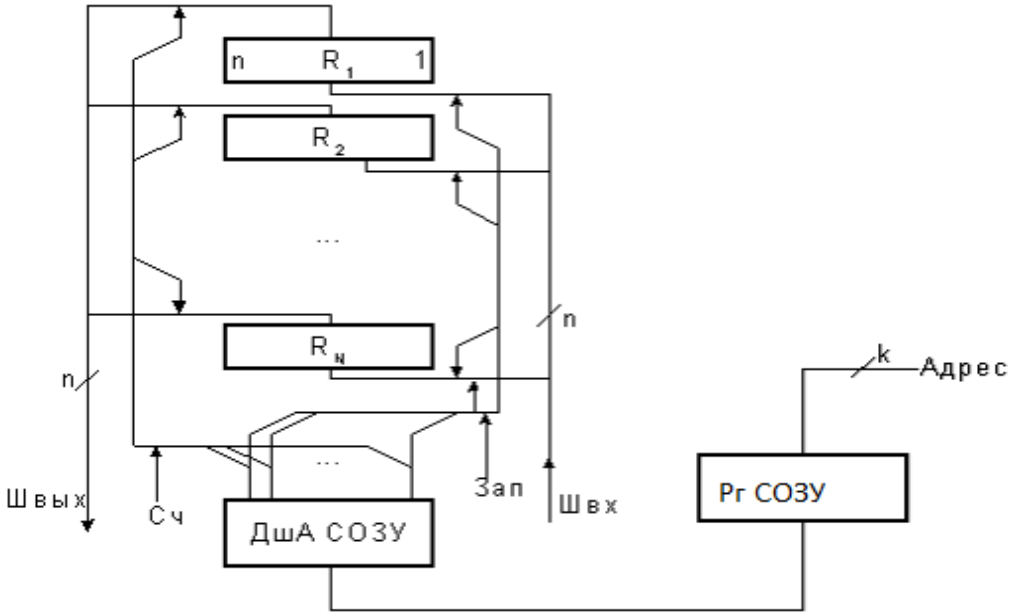
Назначение СОЗУ состоит во временной синхронизации быстродействующих логических устройств процессора с системой памяти компьютера, и прежде всего с ОЗУ, которое имеет меньшее быстродействие.

Как и вообще в оперативных ЗУ, в зависимость от метода поиска информации различают СОЗУ:

- с прямой адресацией;
- стековые;
- магазинные;
- ассоциативные

Организация СОЗУ с прямой адресацией

Другое название – регистры общего назначения (РОН). Номер регистра как раз и означает адрес ячейки СОЗУ.

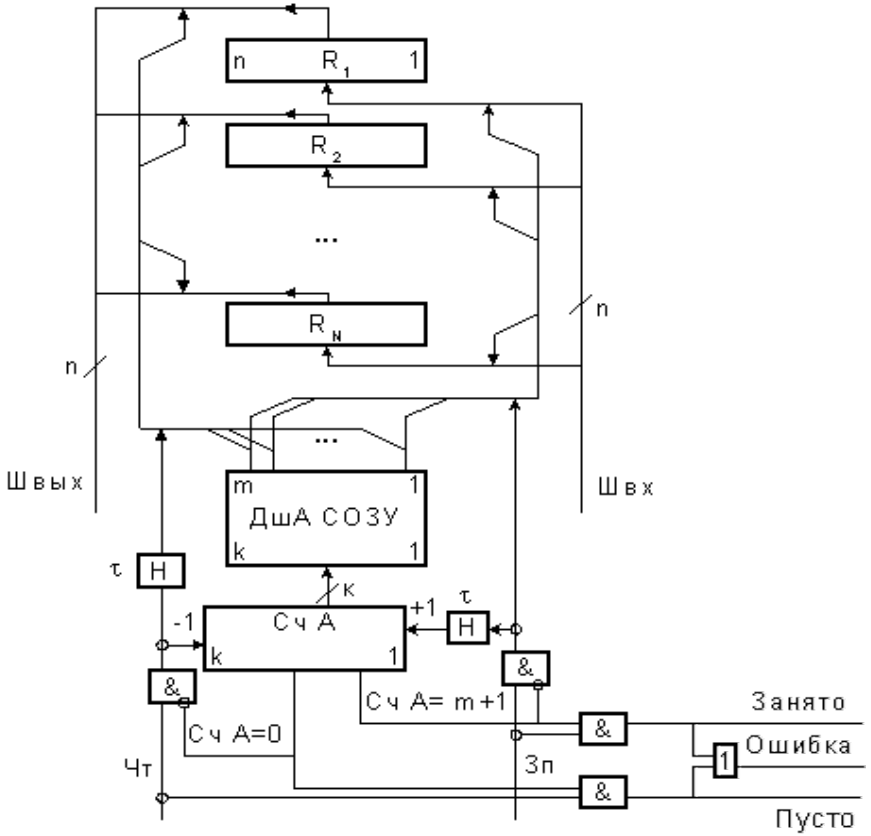


Регистровые СОЗУ широко распространены в современных процессорах наряду с кэш-памятью.

Организация стекового и магазинного СОЗУ

Необходимо различать *программные* и *аппаратные стеки* (магазины). При этом основное – функциональное различие в назначении указателя стека (УС). Однако программный стек – это уже не СОЗУ.

Структура стекового СОЗУ:



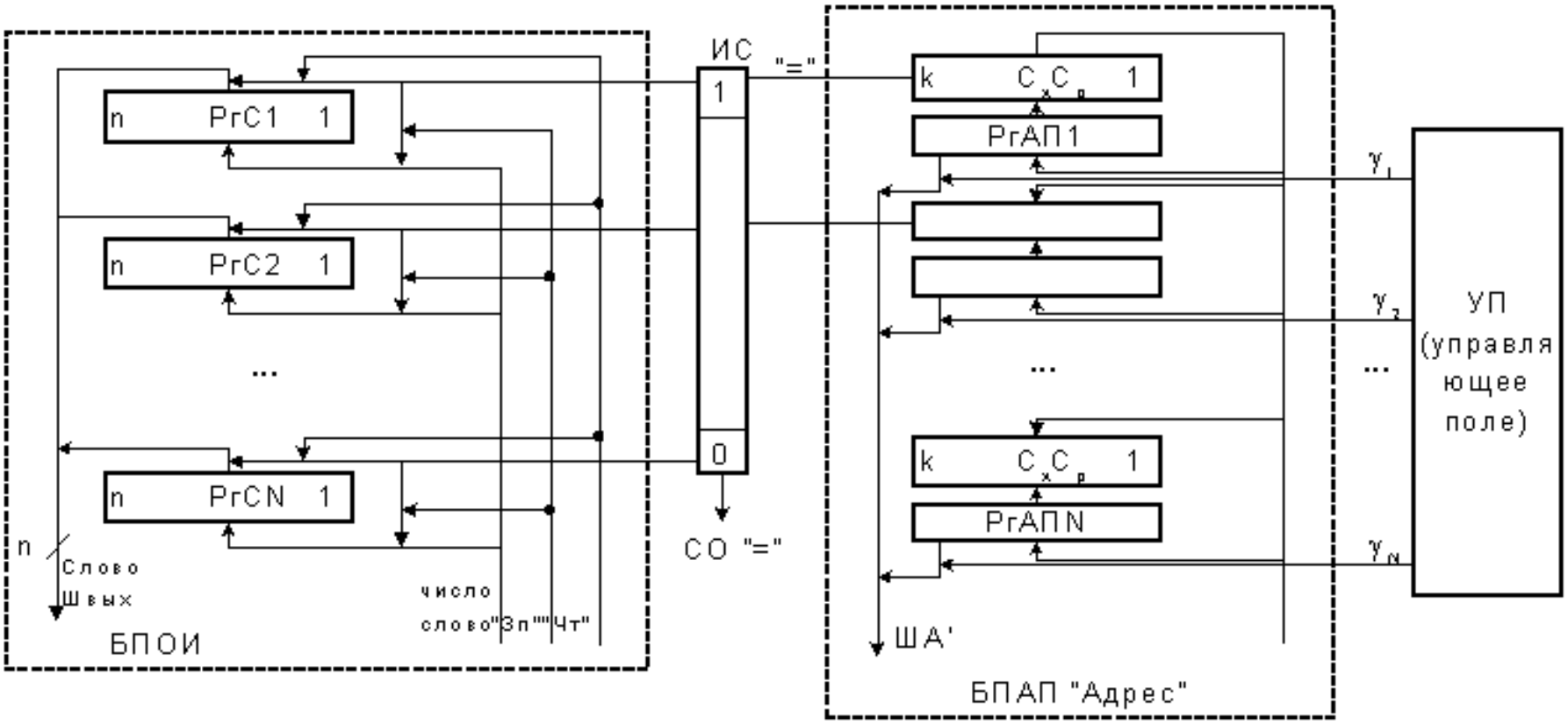
Организация ассоциативных СОЗУ

Ассоциативные СОЗУ имеют более распространённое наименование – кэш-память.

При использовании ассоциативных ЗУ в качестве *СОЗУ* они *выполняют роль буферных ЗУ.*

Ячейки СОЗУ используются *для подмены ячеек ОЗУ* с целью *повышения быстродействия* обмена. Такая подмена производится путем занесение в ячейку СОЗУ и содержимого ячейки ОЗУ и её адреса в ОЗУ.

В кэш-памяти *роль ассоциативного признака играет адрес ячейки ОЗУ (оперативной памяти)*, где до некоторых пор хранился операнд. При обращении к СОЗУ требуемая ячейка определяется не по своему адресу («физическому»), а по признаку хранимого в ячейке слова. Таким признаком в данном случае выступает адрес слова в ОЗУ.



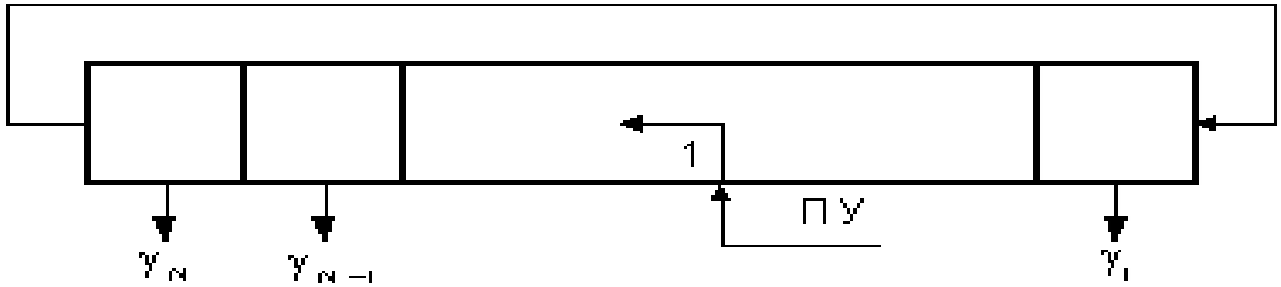
Кэш-память в роли буферного ЗУ допускает замену каждым регистром СОЗУ любой ячейки ОЗУ по факту обращения к ОЗУ, но подмена ячеек может производиться программно *по* специальным командам.

Поскольку с некоторого момента времени работы СОЗУ все его ячейки могут оказаться занятыми, то должна существовать процедура преднамеренного «освобождения» некоторых ячеек кэш-памяти с пересылкой их содержимого в ОЗУ для сохранения информации.

Критерием (идеальным!) для выбора ячейки на освобождение должно быть самое редкое обращение (самое давнее обращение). Но это потребует аппаратурных затрат: фиксация момента и отсчёт временных интервалов. Дисциплина именуется LRU (Least Recently Used – дольше всех неиспользуемый).

Существуют и другие дисциплины, среди которых для большинства программ эмпирически показана неплохая эффективность случайного освобождения ячейки.

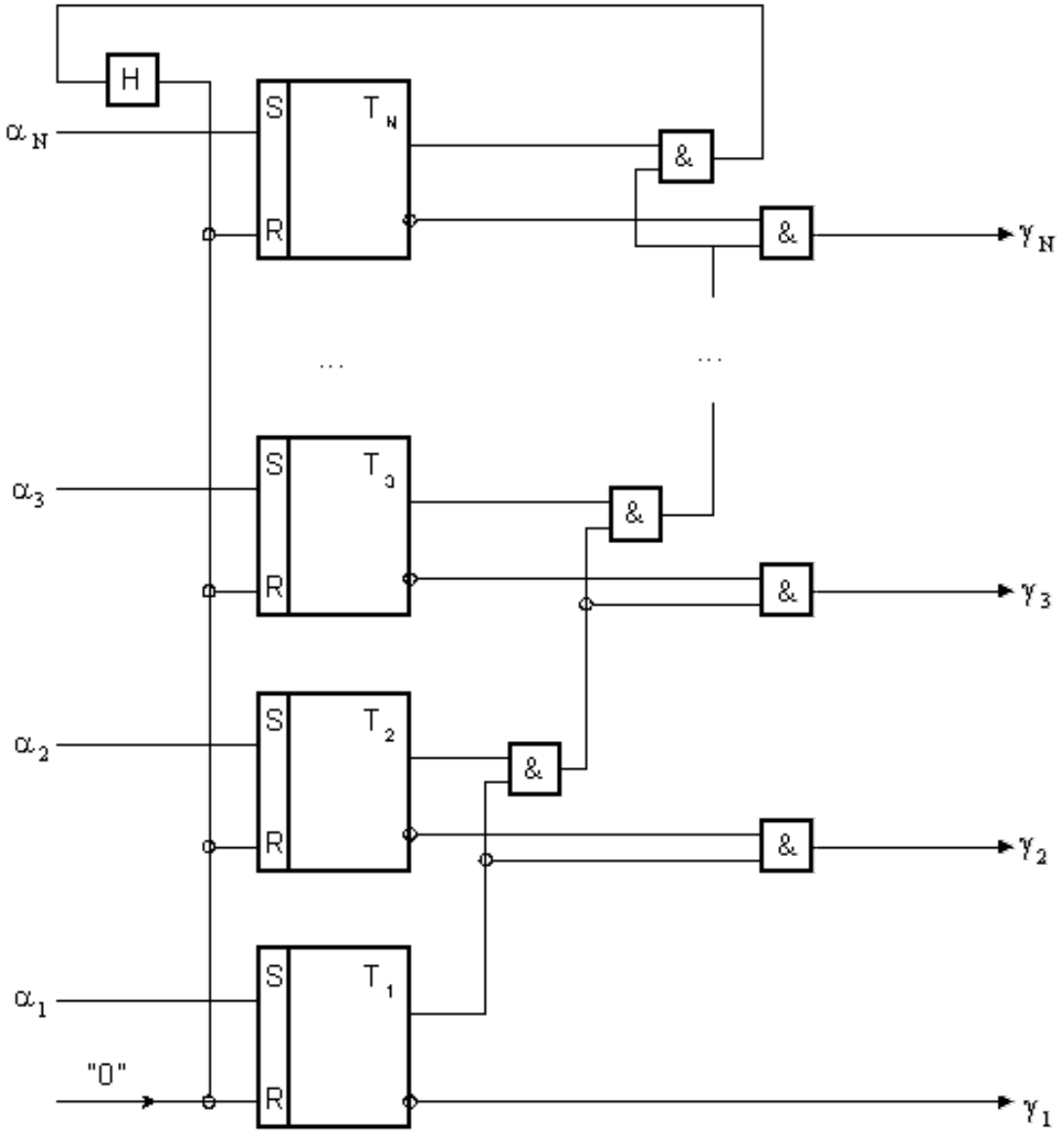
Безусловно, здесь тоже нужна специальная схема, но она существенно проще. *Случайное освобождение* может быть реализовано на сдвигающем регистре, в который предварительно записан код «000100…00». Длина регистра равна числу ячеек кэш-памяти.



ПУ – сигнал признака удаления, когда в адресном поле нет свободной ячейки. Там, где окажется «1» (она движется под действием сигналов ГТИ), содержимое ячейки и адрес удаляются.

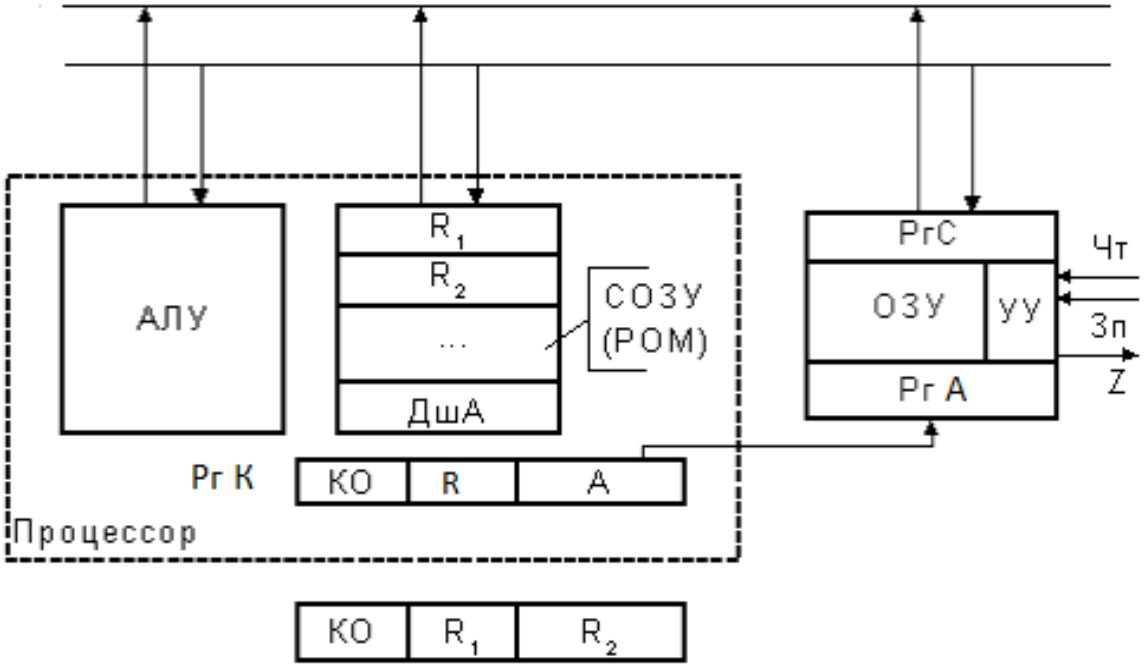
Стратегия *удаления слова*, к которому достаточно давно не было обращения, реализуется, например, с помощью такой схемы.

Первоначально во все триггеры записывается «0» (признак неактивности). Сигналы α_i подаются, когда к ячейке было обращение. Удаляется слово из той ячейки, которая имеет признак неактивности («0») и расположена ниже других. Когда везде будет «1» через интервал задержки произойдет обнуление.



Оценка эффективности использования СОЗУ в процессоре

Условное изображение функциональной схемы:



Пусть требуется вычислить R=a/b + b*c - d*e, причем a, b, c, d, e находятся в ОЗУ. Тогда получаем для двухадресной машины две следующие гипотетических программы:

1) если СОЗУ не используется

1	Деление	A	B	a/b → (A)
2	Умножение	B	C	b*c → (B)
3	Сложение	A	B	a/b + b*c → (A)
4	Умножение	D	E	d*e → (D)
5	Вычитание	A	D	(A) – (D) → (A)

2) Если регистровое СОЗУ используется

1	Пересылка	РОН1	A	a → (РОН1)
2	Деление	РОН1	B	a/b → (РОН1)
3	Пересылка	РОН2	B	b → (РОН2)
4	Умножение	РОН2	C	b*c → (РОН2)
5	Сложение	РОН1	РОН2	a/b+b*c → (РОН1)
6	Пересылка	РОН2	D	d → (РОН2)
7	Умножение	РОН2	E	d*e → (РОН2)
8	Вычитание	РОН1	РОН2	a/b + b*c - d*e → (РОН1)

Сравнительный расчет времени доступа к памяти

В среднем время обращения к ОЗУ в 10 раз больше времени обращения к СОЗУ (обозначим его t).

В первой программе потребуется 20 обращений ОЗУ: 5 за командами; по 2 в каждой команде для чтения операндов и по 1 в каждой команде для записи результата. Следовательно, общее время доступа к памяти составит

$$T_1 = 10t*20 = 200t.$$

Во второй программе суммарно потребуется 8 обращений к ОЗУ за командами, 6 обращений к ОЗУ для чтения операндов, а также 15 обращений к СОЗУ для записи/чтения операндов. Общее время доступа к памяти составит

$$T_2 = 10t*(8+6)+15t = 155t.$$

Эффективность применения СОЗУ определяется отношением:

$$\eta = T_2 / T_1 = 155/200 = 0,775.$$

Продолжим оценку эффективности применения СОЗУ.

Для эффективного использования стекового СОЗУ в систему команд ЭВМ целесообразно включить команды, не имеющие аналогов в ЭВМ с обычной структурой. Это команды: *«дублирование»*, осуществляющая передачу слова из Рг1 в Рг2 (содержимое Рг2 и последующих сдвигается вниз по стеку, содержимое Рг1 остается неизменным, а в Рг2 также помещается (Рг1)); *«реверсирование»* – команда перестановки содержимого пары соседних регистров (обычно Рг1 и Рг2) без какого-либо продвижения информации по стеку.

Таблица, поясняющая получаемую программу с применением аппаратного стека, показана на следующем слайде. В правой части таблицы условно показано содержимое регистров стека Рг1, Рг2 и Рг3.

1	Вызов в стек (B)	b	-	-
2	Дублирование	b	b	-
3	Вызов в стек (A)	a	b	b
4	Деление	a/b	b	-
5	Реверсирование	b	a/b	-
6	Вызов в стек (C)	c	b	a/b
7	Умножение	b*c	a/b	-
8	Сложение	a/b + b*c	-	-
9	Вызов в стек (D)	d	a/b + b*c	-
10	Вызов в стек (E)	e	d	a/b + b*c
11	Умножение	d*e	a/b + b*c	-
12	Реверсирование	a/b + b*c	d*e	-
13	Вычитание	a/b + b*c –d*e	-	-
		(Рг1)	(Рг2)	(Рг3)

Сравнительный расчет времени доступа к памяти

В этой третьей программе потребуется 13 обращений к ОЗУ за командами, всего 5 обращений к ОЗУ для чтения операндов и суммарно 23 обращения к регистрам стека для чтения/записи операндов. Следовательно,

$$T_3 = 10t*(8 + 5) + 23t = 203t.$$

С эффективностью произошла катастрофа:

$$\eta = T_3 / T_1 = 203/200 > 1.$$

Вывод: применение аппаратных стеков оказалось неэффективным.

Дополнение к расчётам

Приведённый расчёт не основан на специально подобранной формуле. При аналогичных расчётах по формуле

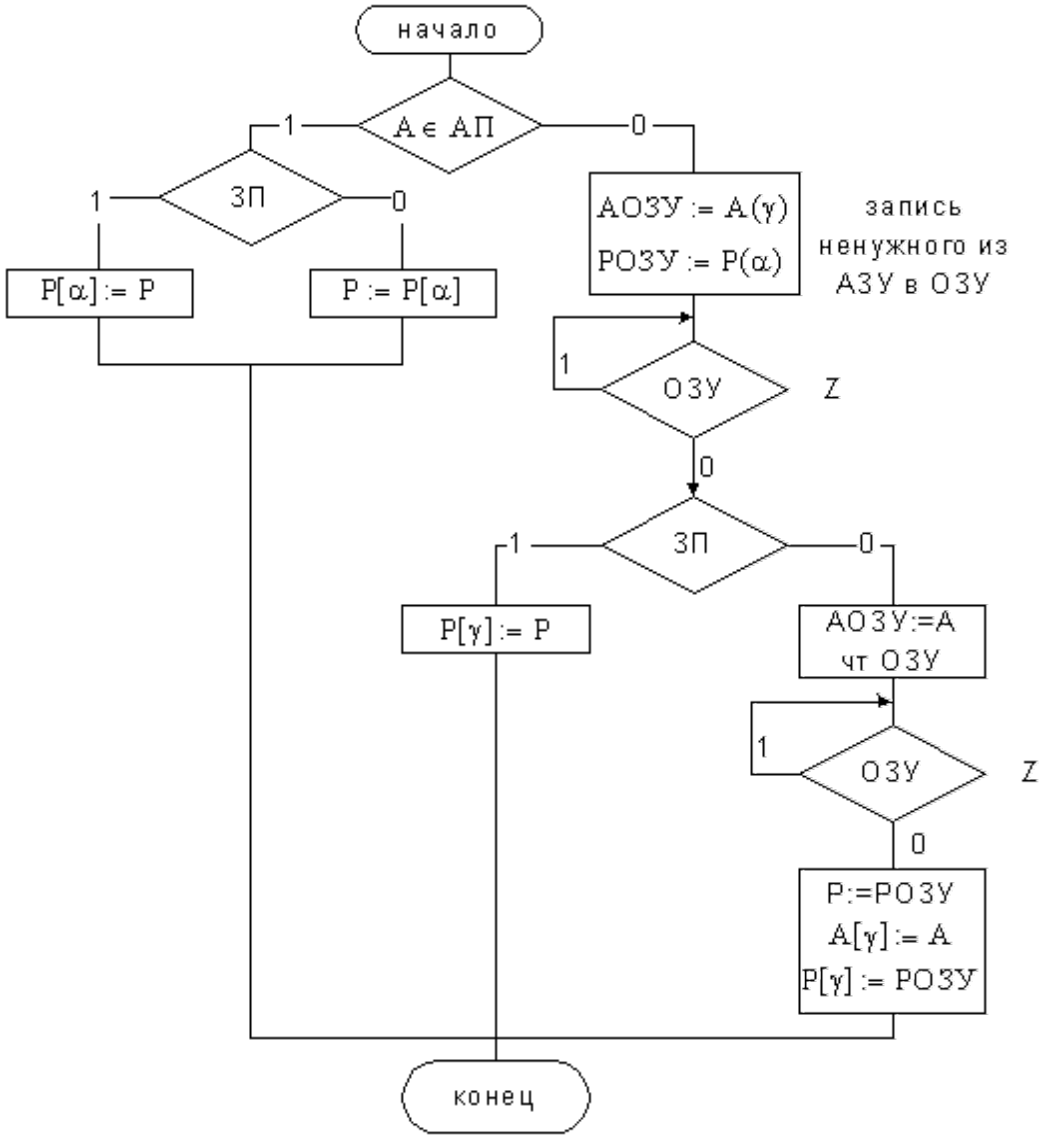
$$R = a + b^2/c – a*d + c/3 – 1$$

получаются следующие значения суммарного времени обращения к памяти:

- используется только ОЗУ $T_1 = 30*10t = 300t$;
- ОЗУ и регистровое СОЗУ $T_2 = 18*10t + 21t = 201t$;
- ОЗУ и стековое СОЗУ $T_3 = 29*10t + 39t = 329t$.

Оценка эффективности кэш-памяти

При наличии хотя бы небольшого числа свободных ячеек результат будет тождественен применению регистрового СОЗУ, т.е. время доступа составит T_2. А при полном кэш-СОЗУ процесс получения результата удобно представить структурной схемой алгоритма:



По этому алгоритму можно иначе подойти к оценке эффективности.

Пусть за некоторое время T выполняется N обращений к памяти. Из структуры схемы алгоритма следует, что обращение к памяти с точки зрения затрат времени обслуживается одним из трех способов:

- простым обращением к АСОЗУ;
- сложным обращением с целью записи слова, когда выполняется обращение к ОЗУ и затем обращение к АСОЗУ;
- сложным обращением при чтении слова, когда выполняется 2 обращения к ОЗУ и затем к АСОЗУ.

Пусть за время T выполняется $N_П$ простых обращений, $N_{С/Зп}$ – сложных обращений с целью записи; $N_{С/Чт}$ – сложных обращений с целью чтения. Естественно, что

$$N = N_П + N_{С/Зп} + N_{С/Чт} .$$

Пусть – T_0 и T_1 обозначения длительности обращения к ОЗУ и АСОЗУ, соответственно. Если нет буфера в виде СОЗУ, то

$$\nu_0 = N*T_0 ,$$

а если буфер есть, то

$$\nu_1 = N_П* T_1 + (T_0 + T_1)*N_{С/Зп} + (2T_0 + T_1)*N_{С/Чт} *T_0 .$$

Аналогично предыдущему сравнению введём относительную оценку:

$$\eta = \nu_1/\nu_0 =$$

$$= T_1/T_0 * N_П /N + (1 + T_1/T_0) * N_{С/Зп} /N + (2 + T_1/T_0) * N_{С/Чт} /N .$$

Обозначим $A = N_П /N$; $B = N_{С/Зп} /N$; $C = N_{С/Чт} /N$.

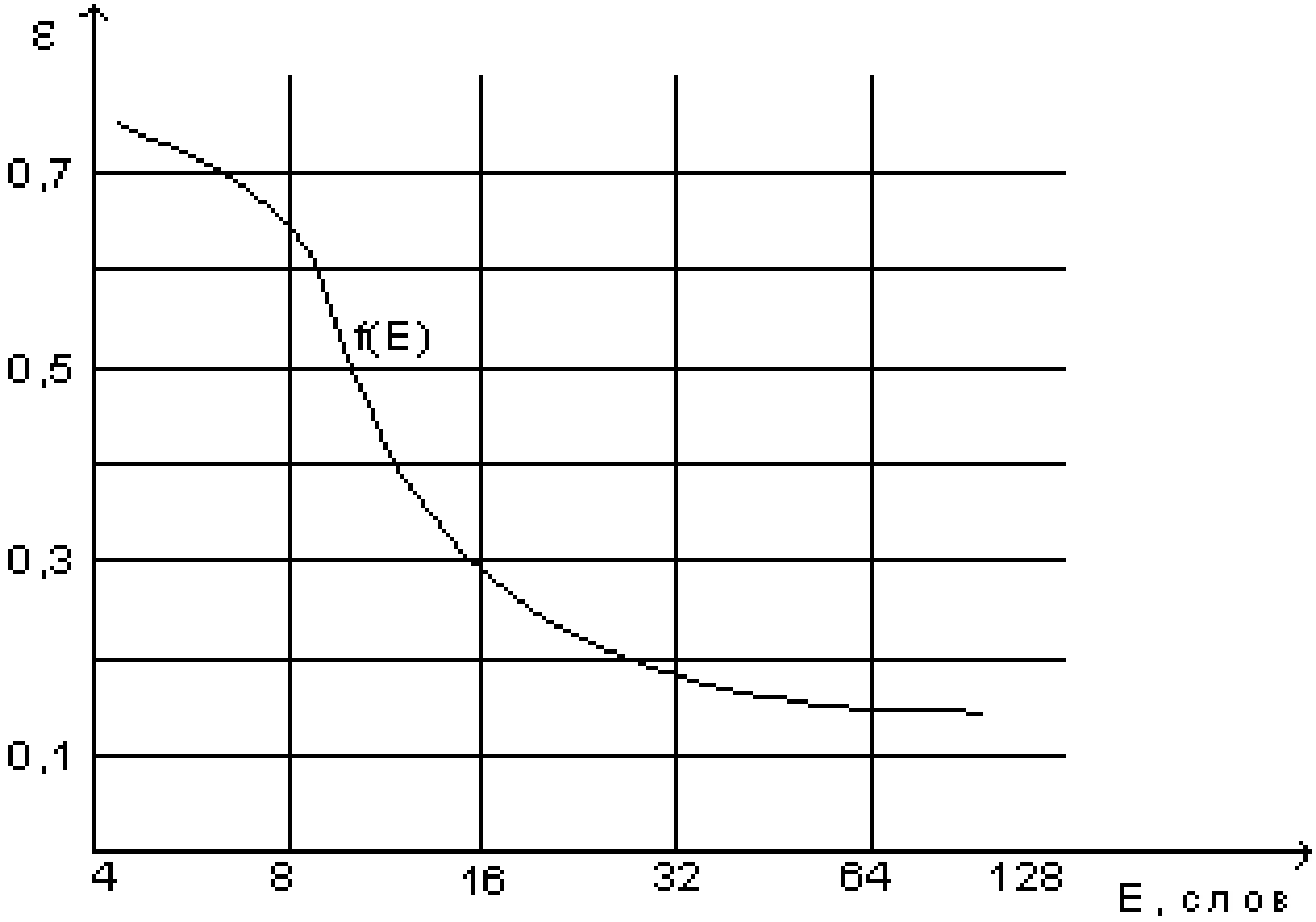
При $t \rightarrow \infty$ A, B и C – суть вероятности каждого из трёх способов обращения к памяти, причём $A + B + C = 1$.

Если обозначить $\eta_1 = T_1/T_0$, то

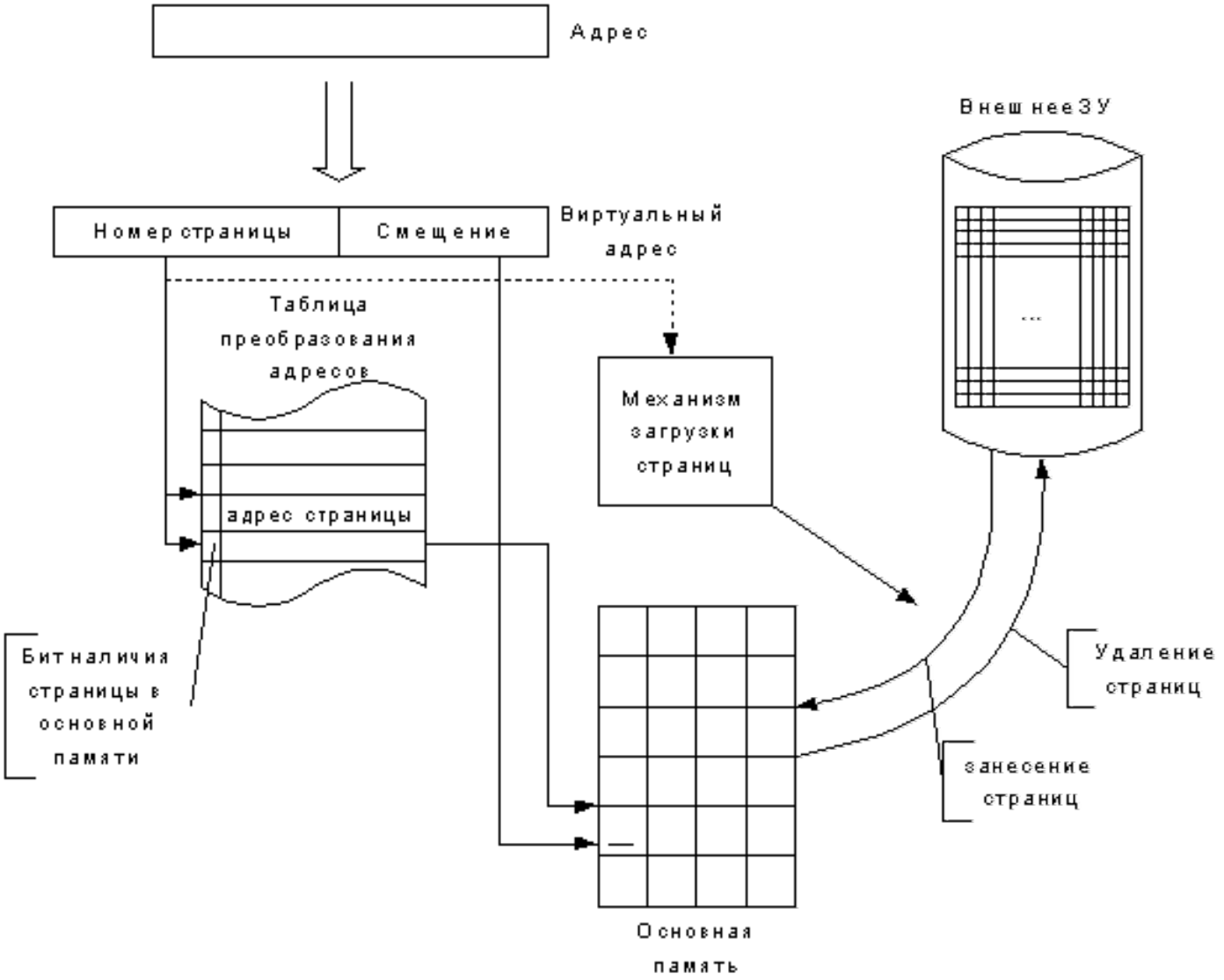
$$\eta = \eta_1 * A + (1 + \eta_1) * B + (2 + \eta_1) * C = \ldots = \eta_1 + (B + 2*C) = \eta_1 + \varepsilon.$$

Коэффициент ε зависит от способа назначения слов на удаление и от емкости кэш-памяти (длительности пребывания слов в СОЗУ) и др.

Тогда коэффициент ε можно рассматривать как некоторый организационный параметр. Интересна зависимость $\varepsilon = f(E)$, где E – емкость кэш-СОЗУ. Она получена статистическим (имитационным) моделированием.



Концепция виртуальной памяти



Реально существующая память в ОЗУ – физическая (у нее физические адреса). Остальная память рассматривается как логическая или виртуальная (у нее виртуальные или логические адреса). Соответствие между физическими адресами устанавливается совместно аппаратными и программными средствами ЭВМ. Устройства (системы устройств) реализующие концепцию виртуальной памяти совмещают много функций (управление адресами, сегментация адресного пространства, защита и т.д.)

Адресное пространство может быть разбито на страницы и сегменты. Страницы: деление на части фиксированной длины вне зависимости от содержания информации. Сегменты: деление по логическим признакам, задаваемым программистом, обычно сегмент соответствует массиву данных, программе или подпрограмме и т.д. и имеет переменную длину.

В заключение главы предлагаю ознакомиться с приложением П2.4, в котором изложена предыстория и специфика распределения ОЗУ в Intel-совместимых персональных компьютеров, а также с приложением П2.5, освещающим технологические аспекты организации флэш-памяти.