ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ

(АРХИТЕКТУРНО- АЛГОРИТМИЧЕСКИЕ ОСНОВЫ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ)

Лектор: **профессор** ***Райхлин Вадим Абрамович***

ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ

Основные цели изучения дисциплины

- показать приоритетную роль параллельных вычислений в современных информационных технологиях, действительность и перспективу параллельных систем;
- дать базовые представления о принципах организации параллельных систем и оценок их эффективности;
- выявить целесообразность специальных разработок параллельных алгоритмов с учетом особенностей параллельной архитектуры;
- освоить методы аппаратно-программной организации информационных кластеров.

Основные задачи изучения дисциплины

- знакомство с различными принципами классификации параллельных систем и способами оценок их производительности;
- показ на конкретных примерах адекватности параллельной обработки современным задачам информатики и особенностей разработки параллельных алгоритмов;
- изучение принципов организации основных классов современных параллельных компьютеров и суперпроцессоров, подсистем коммутации и памяти, тенденций развития кластерных технологий;
- практическое знакомство с архитектурой современных параллельных СУБД.

МЕСТО ДИСЦИПЛИНЫ В УЧЕБНОМ ПРОЦЕССЕ

Дисциплина «Параллельные вычисления» входит в профессиональный цикл образовательной программы бакалавра. Материал курса основан на знаниях, навыках и умениях, почерпнутых студентами из курсов «Электронные вычислительные машины», «Схемотехника ЭВМ», «Сети и телекоммуникации», «Микропроцессорные системы», «Операционные системы», «Базы данных», «Защита информации».

Студенты должны быть знакомы с архитектурой ЭВМ («Электронные вычислительные машины»), с микросхемами операционных узлов ЭВМ («Схемотехника ЭВМ»), с основными разновидностями локальных сетей («Сети и телекоммуникации»), с основами микропроцессорной техники («Микропроцессорные системы»), с сетевыми операционными системами («Операционные системы»), с понятием реляционной базы данных и языком SQL («Базы данных»), с основными понятиями защиты информации («Защита информации»).

Полученные при изучении дисциплины знания, умения и навыки будут использованы студентами при изучении дисциплин «Параллельное программирование», «Персональные суперкомпьютеры», «Кластерное дело» и при подготовке выпускной работы бакалавра.

РАЗДЕЛЫ ДИСЦИПЛИНЫ И ВИДЫ ЗАНЯТИЙ

№ п/п	Раздел дисциплины	Лекции (час.)	ПЗ (час.)	ЛР (час.)	СР (час.)
1	Начальные понятия и предпосылки	12	–	12	24
2	Представители параллельных систем	12		12	24
3	Специализированные системы, память, суперпроцессоры	12		12	24
	ИТОГО:	**36**		**36**	**72**

ЛАБОРАТОРНЫЙ ПРАКТИКУМ

№ п/п	№ раздела дисциплины	Темы занятий	Объем (час)
1-3	1	Параллелизм на микроуровне	12
4-6	2	Параллельные СУБД на кластерной платформе	12
7-9	3	Системы защиты пространственных данных	12
		ИТОГО:	**36**

Распределение фонда времени по неделям и видам занятий

№ п/п	Разделы дисциплины и темы	Семестры	Неделя сем ЛК, ЛР	Всего час.	Виды учебной деятельности, включая самостоятельную работу студентов, и трудоемкость (час.)			Формы текущего контроля успеваемости *(по неделям сем.)*. Формы итоговых аттестаций
					ЛК, час.	ЛР, час	СР, час.	
1	**Начальные понятия и предпосылки**	7	**1-6**	**48**	**12**	**12**	**24**	Тестирование 1 (6 нед.)
	1.1. Необходимость, ретроспектива, тенденции развития, классификация		1,	4	2	–	2	Контроль активности работы на лекциях и ЛР
	1.2. Ассоциативная обработка и параллелизм		2-3,1-6	32	4	12	16	
	1.3. Показатели производительности		4,	4	2	–	2	
	1.4. Предметные предпосылки параллелизма		5-6,	8	4	–	4	Защита ЛР 1-3
2	**Представители параллельных систем**	7	**7-12**	**48**	**12**	**12**	**24**	Тестирование 2 (12 нед)
	2.1. Кластерные системы		7-8,7-12	32	4	12	16	
	2.2. Процессорные матрицы (СВО)		9,	4	2	–	2	Контроль активности работы на лекциях и ЛР
	2.3. Матричный процессор ассоциативного типа		10,	4	2	–	2	
	2.4. Мейнфреймовые архитектуры		11,	4	2	–	2	
	2.5. Сети коммутации		12,	4	2	–	2	Защита ЛР 4-6
3	**Спец. системы, память, супер процессоры.**	7	**13-18**	**48**	**12**	**12**	**24**	Тестирование 3 (18 нед)
	3.1. Система защиты пространственных данных		13-14,13-18	30	3	12	15	
	3.2. Организация главной памяти		14-15,	6	3	–	3	Контроль активности работы на лекциях и ЛР
	3.3. RAID-массивы		16,	4	2	–	2	
	3.4. Графические процессоры		17,	4	2	–	2	
	3.5. Архитектура CELL		18.	4	2	–	2	Защита ЛР 7-9
	3.6. **Подготовка к экзамену**			**36**	–	–	**36**	**Экзамен**
	Всего за семестр (количество часов)			**180**	**36**	**36**	**108**	

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

- **Райхлин В.А.** Системы параллельной обработки данных – Казань: Изд-во ФЭН, 2010.
- **Райхлин В.А.** Начала параллельных вычислений. – Казань: Изд-во КГТУ, 2008.– http://modelling.kai.ru/LPC.zip
- **Райхлин В.А.** Суперпроцессоры и RAID-массивы – http://modelling.kai.ru/ SP_raid.zip
- **Абрамов Е.В.,Вершинин И.С.,Гибадуллин Р.Ф., Шагеев Д.О.** Практикум по параллельным вычислениям /Под ред. В.А. Райхлина – Казань: Изд-во КГТУ, 2008.
- **Воеводин В.В.,Воеводин Вл.В.** Параллельные вычисления. – С.-Пб.: “БХВ-Петербург”, 2004.
- **Корнеев В.В.** Вычислительные системы – М.: «Гелиос АРВ», 2004.
- ***http:*** //parallel.ru

Раздел I

НАЧАЛЬНЫЕ ПОНЯТИЯ И ПРЕДПОСЫЛКИ

Лекции 1-3

Лекция 1. НЕОБХОДИМОСТЬ, РЕТРОСПЕКТИВА, ТЕНДЕНЦИИ РАЗВИТИЯ, КЛАССИФИКАЦИЯ

СФЕРА ПРИМЕНЕНИЙ ВВС:

- В коммерции – работы со сверхбольшими базами данных и графическими приложениями: кино, телевидение.
- В военных целях – разработка ядерного оружия, конструирование самолетов, ракет, бесшумных подводных лодок.
- В научных исследованиях – физических, химических, метеорологических, геологических.
- В технике – решение проблем аэрокосмической, автомобильной, газовой и нефтедобывающей промышленности, связанных со своевременной обработкой больших объемов экспериментальных данных.

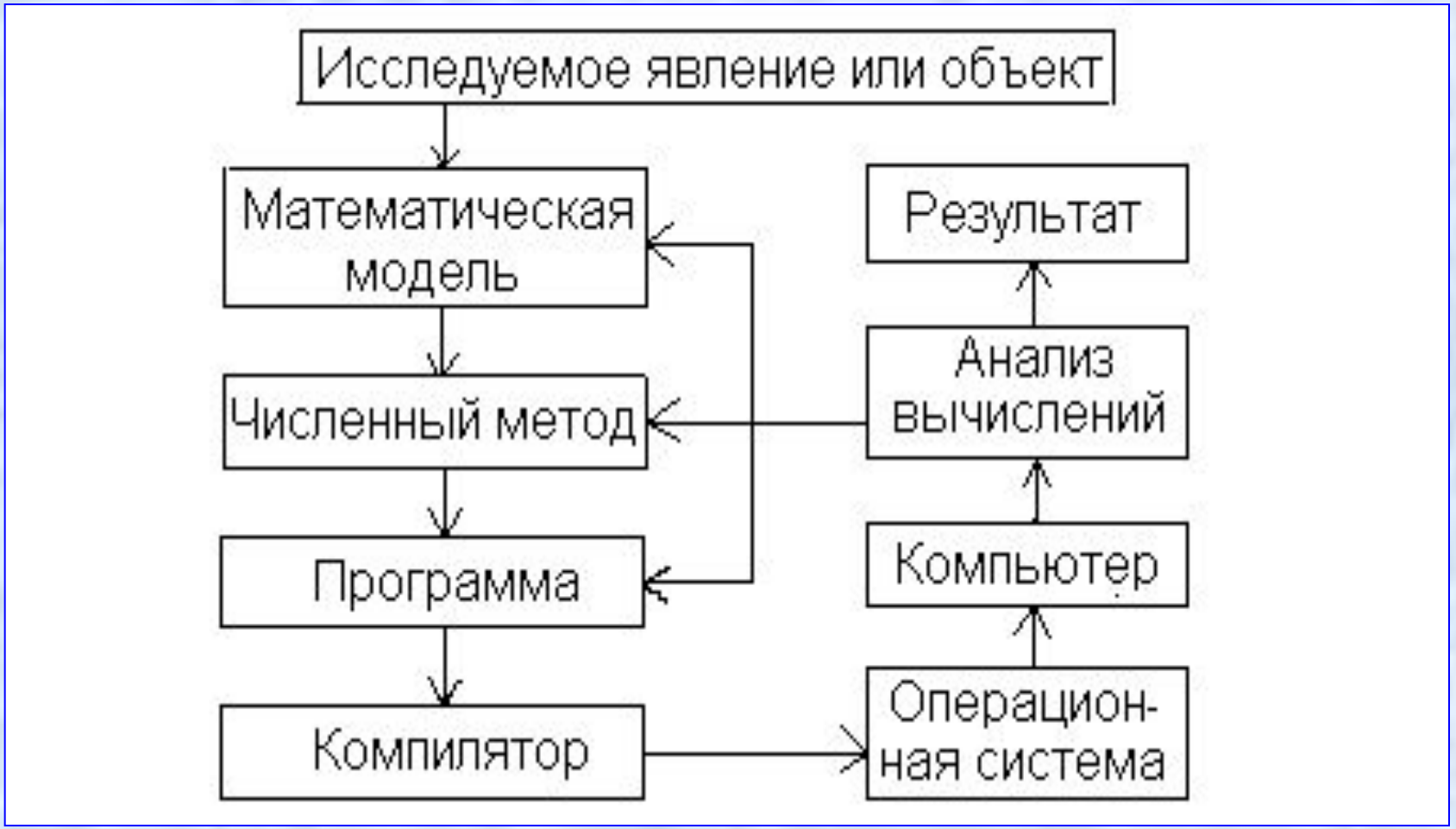
***ОСНОВА ПОВЫШЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ* ВС:**

введение параллелизма на всех уровнях: алгоритмическом, программном, структурном, архитектурном.

ПРИМЕНЕНИЕ ВВС СВЯЗЫВАЕТСЯ :

с проведением сложного компьютерного моделирования (вычислительного эксперимента).

ЭТАПЫ ЧИСЛЕННОГО ЭКСПЕРИМЕНТА



ВВЕДЕНИЕ ПАРАЛЛЕЛИЗМА В АРХИТЕКТУРУ ЭВМ

- ***КОНВЕЙЕРНАЯ ОБРАБОТКА*** – применение метода линии сборки с целью повышения производительности арифметического и управляющего устройств.
- ***ФУНКЦИОНАЛЬНАЯ ОБРАБОТКА*** – предоставление нескольким независимым устройствам возможности выполнения различных функций, таких как операции логики, сложения или умножения, обеспечивая взаимодействие с различными данными.
- ***МАТРИЧНАЯ ОБРАБОТКА*** – наличие матрицы идентичных процессорных элементов с общей системой управления, где все элементы в каждый момент времени выполняют одну и ту же операцию, но с разными данными, хранящимися в их собственной либо в общей памяти.
- ***МУЛЬТИПРОЦЕССОРНАЯ ОБРАБОТКА*** – осуществляется множеством процессоров, каждый из которых выполняет свои собственные команды, а все процессоры взаимодействуют друг с другом тем или иным способом.

РЕТРОСПЕКТИВА РАЗВИТИЯ ПАРАЛЛЕЛЬНЫХ СИСТЕМ

1. *СОВЕРШЕНСТВОВАНИЕ АРХИТЕКТУРЫ ФОН НЕЙМАНА*

- Быстродействующие скалярные компьютеры.
- Конвейерные векторные ЭВМ.
- Алгоритмические матричные процессоры.

2. *ПЕРЕХОД К НОВЫМ ПАРАЛЛЕЛЬНЫМ АРХИТЕКТУРАМ*

- Процессорные матрицы.
- Ортогональные и ассоциативные ЭВМ.

СОВРЕМЕННЫЕ ТЕНДЕНЦИИ

- Векторно-конвейерные суперкомпьютеры.
- SMP-системы.
- NUMA-технологии.
- MPP-системы.
- кластерные системы.

СИСТЕМАТИКА ПАРАЛЛЕЛЬНЫХ СИСТЕМ

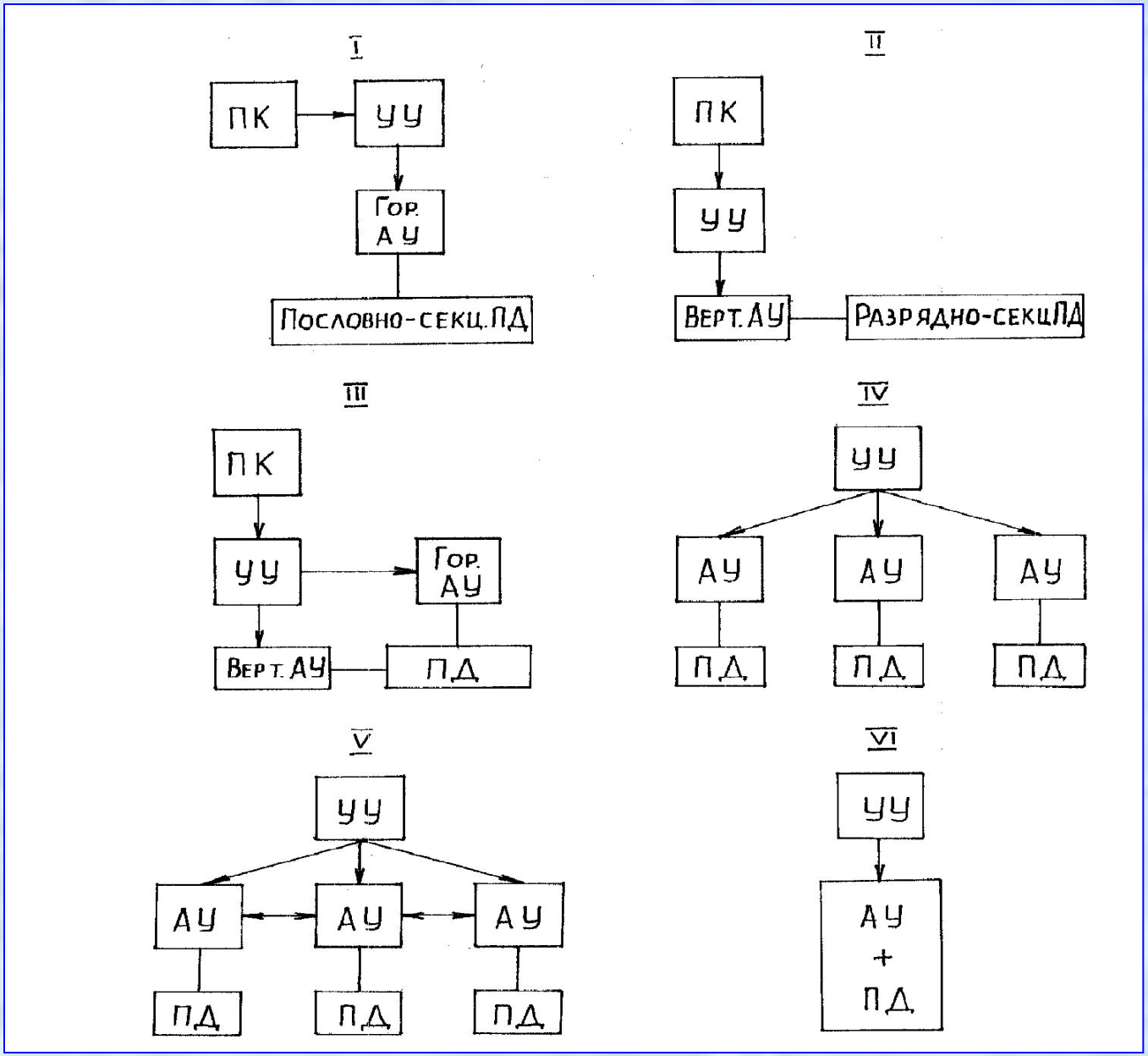
Задачи систематики

- Однозначное отнесение любой известной или предвидимой архитектуры к тому или иному классу (***«что есть что»***).
- Выделение существенных различий между элементами одного класса (***морфологический анализ***).
- Создание предпосылок к отбору ***наиболее перспективных направлений*** путем выявления взаимосвязи этих различий с областью специализации и достижимой эффективностью.

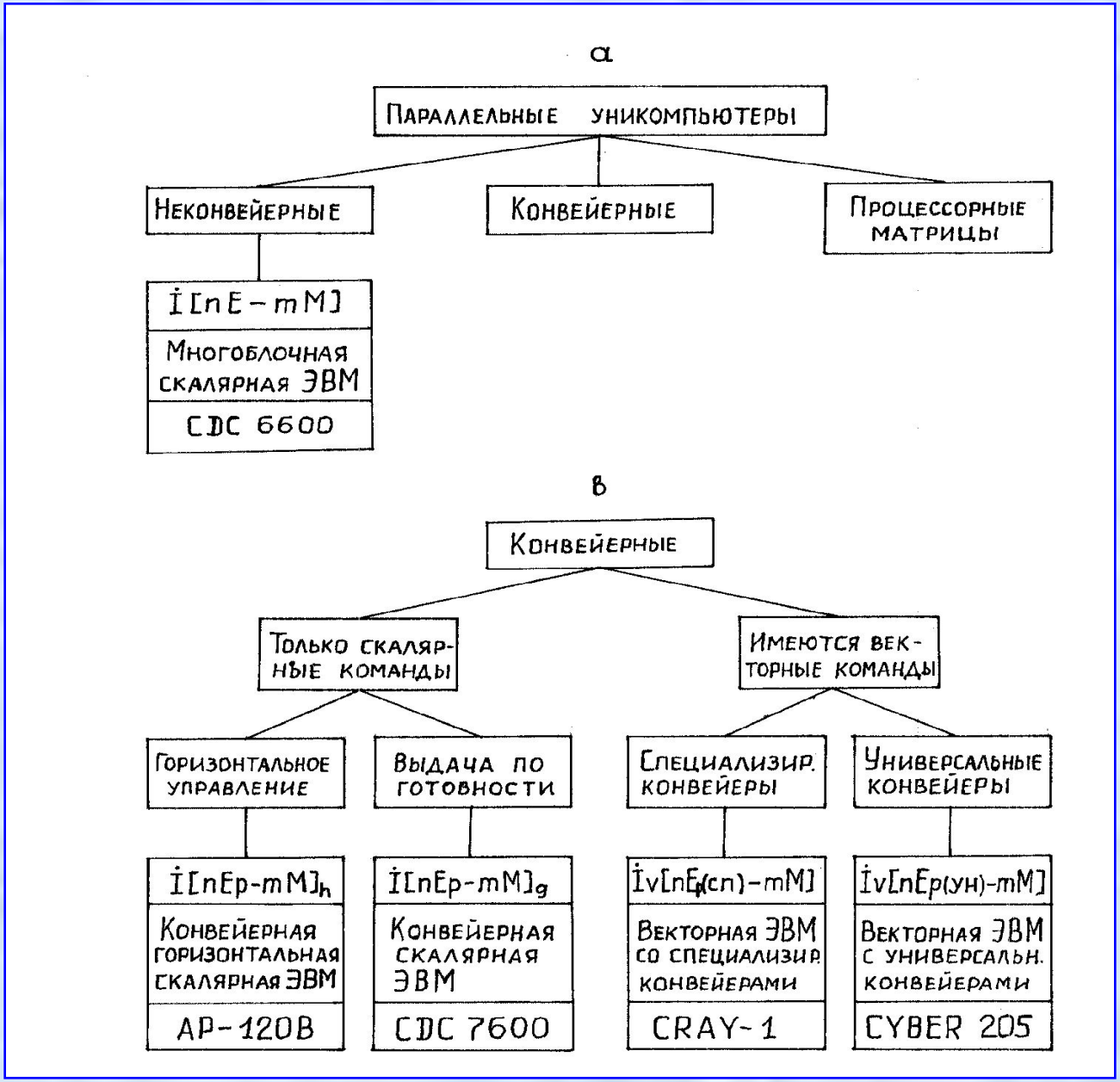
Систематика Флинна

- **ОКОД** ***(SISD)*** – **один поток команд/один поток данных** – как в ЭВМ ***фон Неймана***. Команда **>** одна скалярная операция над одним (парой) операндов.
- **ОКМД** ***(SIMD)*** – **один поток команд/много потоков данных**. Здесь сохраняется один поток команд, но уже векторных, которые инициируют множество операций. Каждый элемент вектора рассматривается как элемент отдельного потока данных. Этот класс включает все ***машины с векторными командами***.
- **МКОД** ***(MISD)*** – **много потоков команд/один поток данных**. Этот класс ***пока вакантен***, т. к. здесь несколько команд должны работать с одним элементом данных.
- **МКМД** ***(MIMD)*** – **много потоков команд/ много потоков данных.** Этот класс включает в себя все формы ***мультипроцессорных конфигураций***.

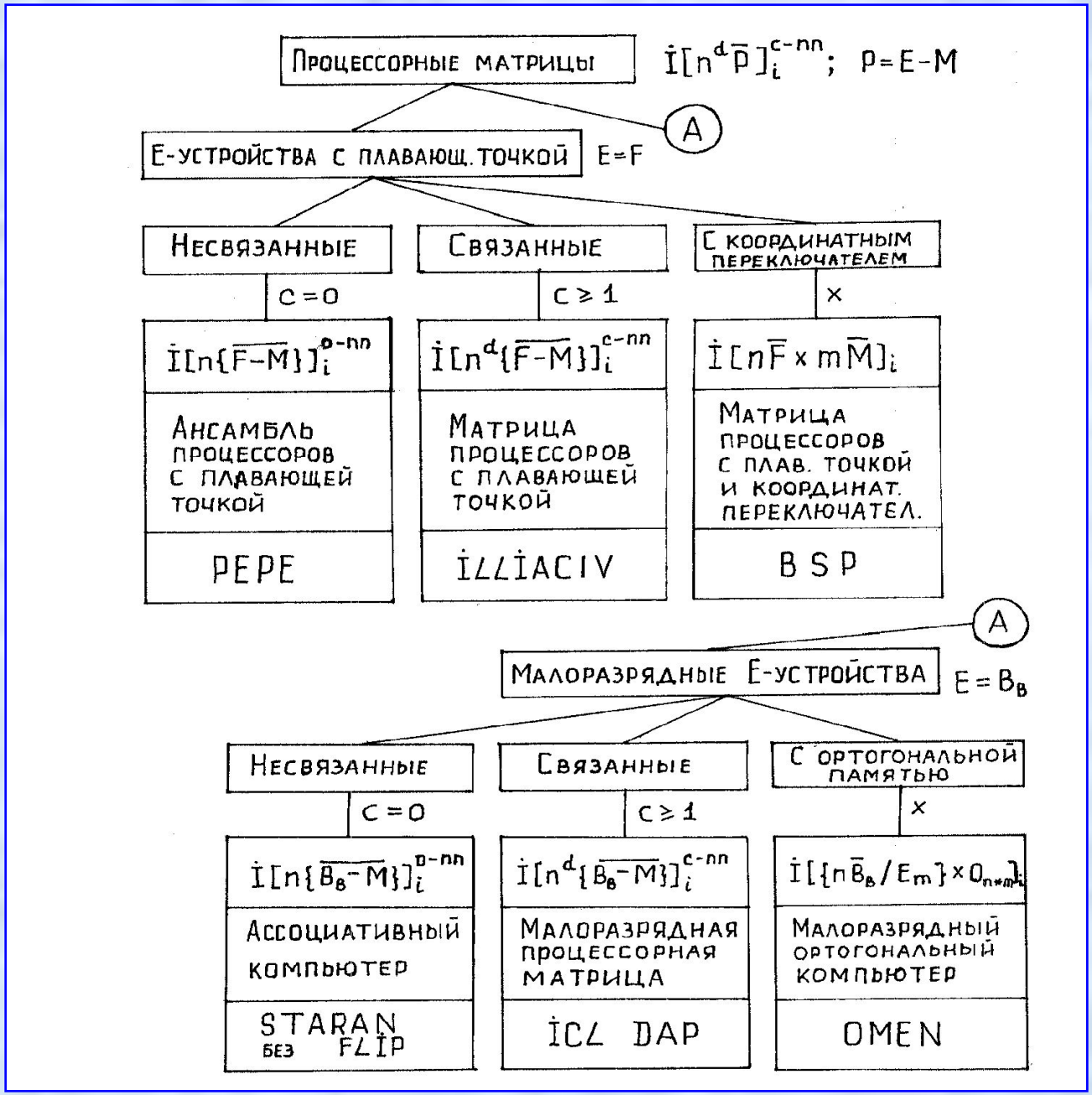
Классификация Дж. Шора



Структурная систематика – I



Структурная систематика – II



Лекции 2-3. АССОЦИАТИВНАЯ ОБРАБОТКА И ПАРАЛЛЕЛИЗМ

ВВОДНЫЕ ЗАМЕЧАНИЯ

Развитие ассоциативных принципов – одна из первых вех параллелизма. Использование этих принципов неизменно значимо при создании специализированных систем параллельной обработки **однородных массивов** данных.

Различают следующие виды такой обработки:

Структурная – охватывает множество процедур *числового поиска* (на « = », «≠ », « > », « < », min, max и др.); *распознавания элементов массива по некоторым признакам* (по ключам, вхождению определенной последовательности литер и др.); *упорядочения элементов массива по заданным критериям* (сортировка). Разряды слов, которые при обработке не несут информационной нагрузки (их значения безразличны) ***маскируются***.

Символьная – подразумевает *исключение некоторых символов со сжатием строк, замену одного символа другим, выравнивание текста по границе определенного символа* и др. Размер символа – 1 байт (8 бит).

Логическая – проводится над цепочками бит: *логическое умножение булевых матриц* (БМ), *транспонирование* БМ, *побитовая конъюнкция либо дизъюнкция* БМ и др.

Арифметическая обработка выполняется над массивами кодов либо чисел: *сложение и умножение матриц, вычисление определителей*, БПФ и др.

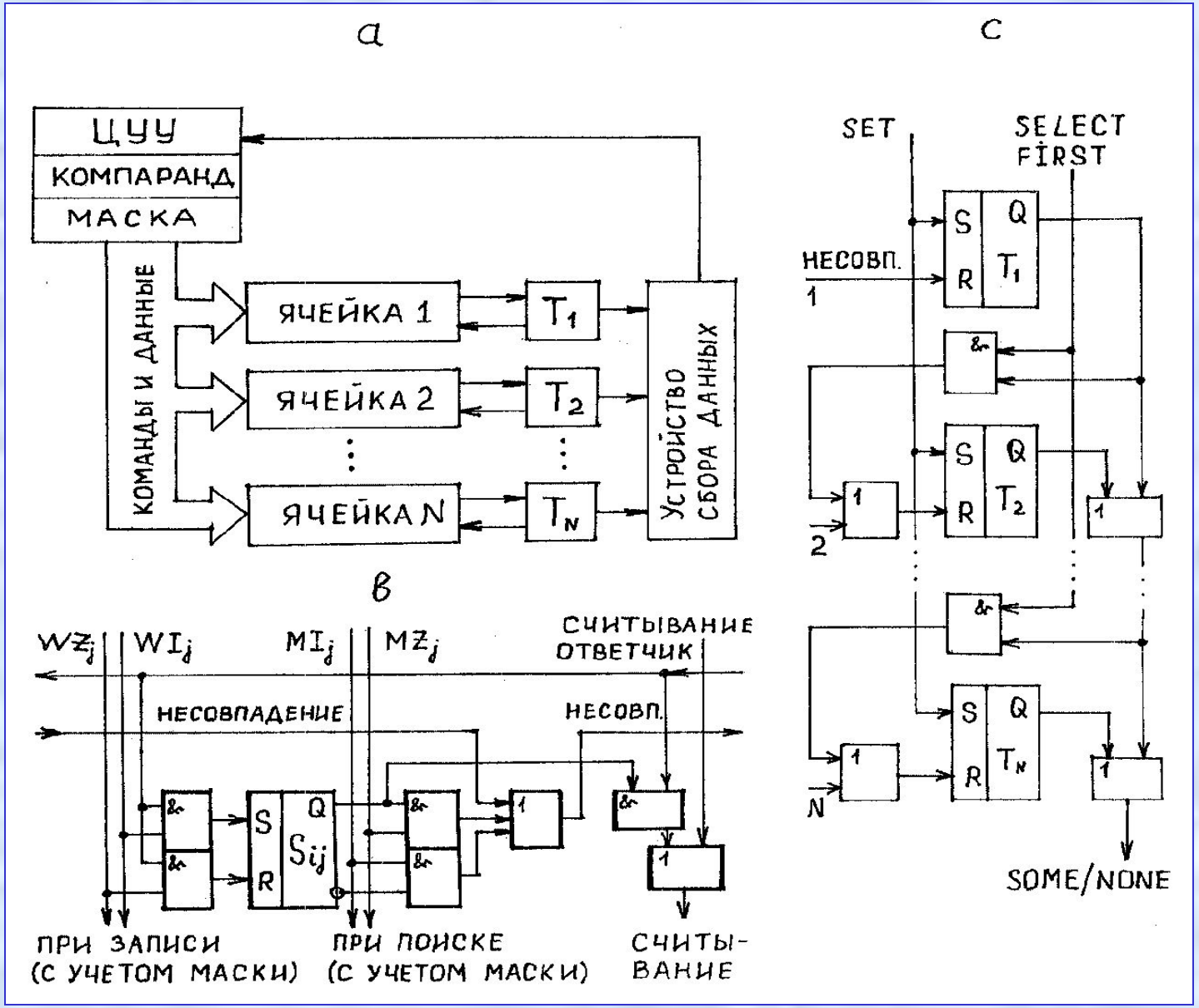
Ассоциирование объектов той или иной природы – это установление соответствия между ними по степени сходства, контраста, близости, единству причин, следствий и т.д. Механизм ассоциаций человеческого мозга чрезвычайно сложен и трудно познаваем. Однако

понятие ассоциативной обработки на ЭВМ существует, является вполне конкретным и достаточно узким. Оно выделяет определенный класс параллельных процессоров – ***ассоциативных параллельных процессоров*** (АПП).

Принципы построения АПП начали обсуждаться с середины 50-х годов прошлого века. Практически одновременно шло развитие теории ОВС – ***однородных вычислительных сред***. Классика АПП и ОВС положила начало исследованиям по ***операционным логико-запоминающим средам*** (ЛЗС) – ***процессорным матрицам ассоциативного типа***.

Проблематика операционных ЛЗС связана с развитием принципов параллельной обработки ***однородных массивов данных*** путем микропрограммной реализации массовых операторов на матрицах спец. БИС. Такие среды относятся к классу ***параллельных систолических структур*** в том смысле, что обработка инициируется в них локально и постепенно охватывает всю среду. Перспективы их применений связаны с развитием технологии программируемых логических интегральных схем (ПЛИС).

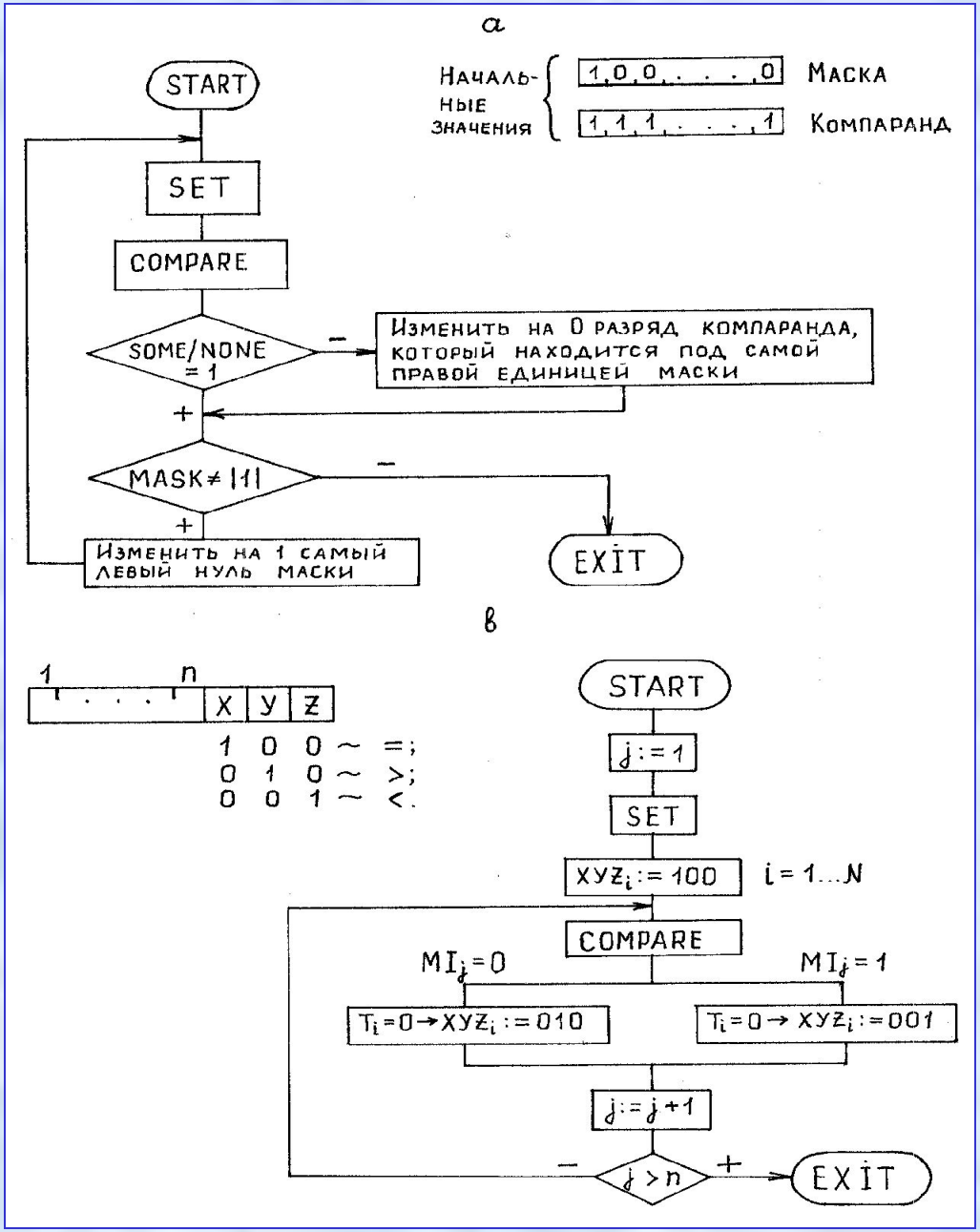
1. АССОЦИАТИВНЫЕ ПАРАЛЛЕЛЬНЫЕ ПРОЦЕССОРЫ



КОМАНДЫ АПП

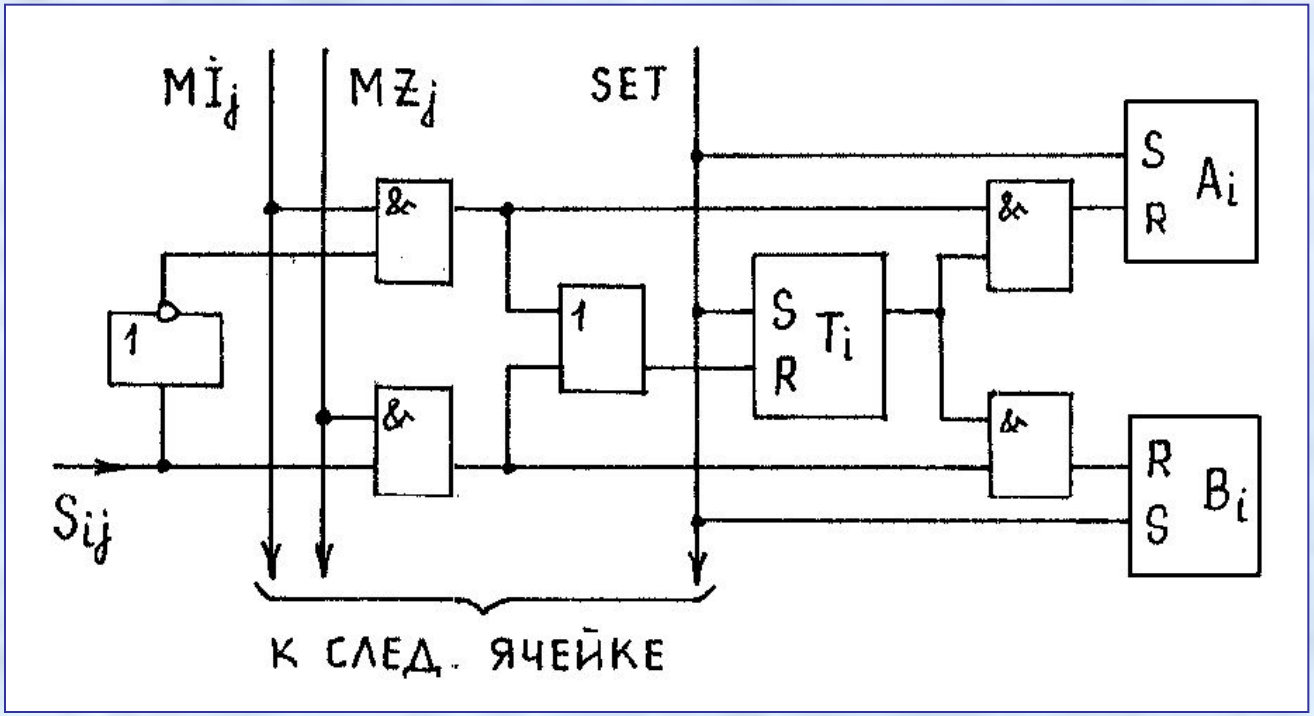
- **SET – установка теговых разрядов в «1»**
- **COMPARE – сравнить**
- **WRITE – мультизапись**
- **FIRST – выбор первого ответчика**
- **COUNT – каково число ответчиков?**
- **REPORT – есть ли ответчик ?**
- **MOVE - сдвиг теговых разрядов**

ЛОГИЧЕСКИЕ АЛГОРИТМЫ для АПП



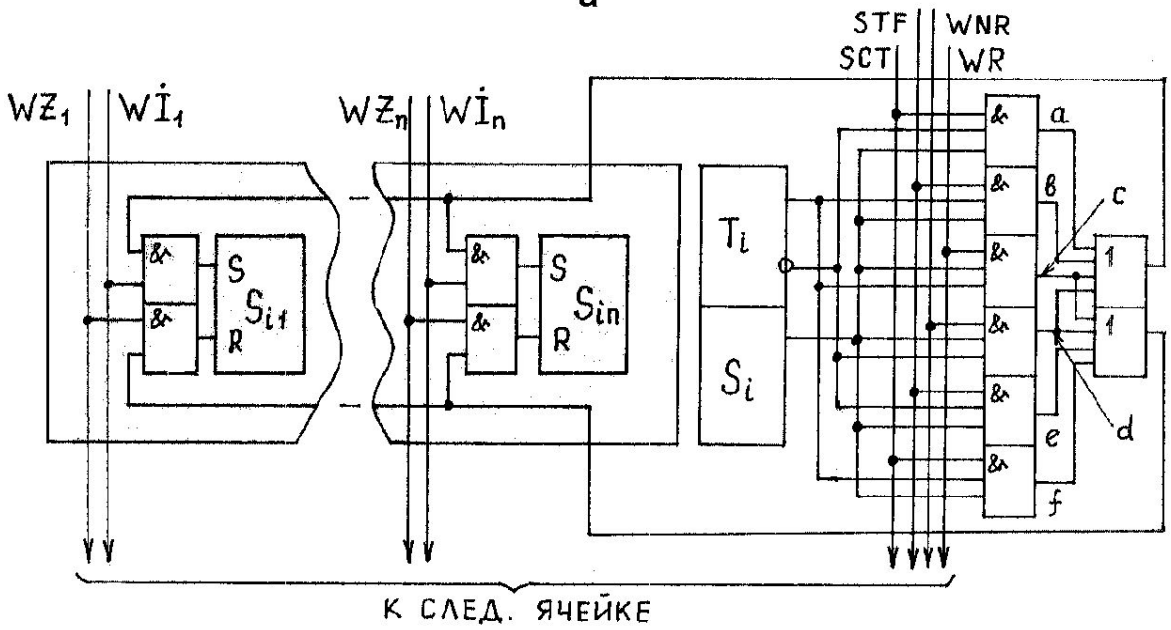
Аппаратная поддержка алгоритма

деления массива на три части

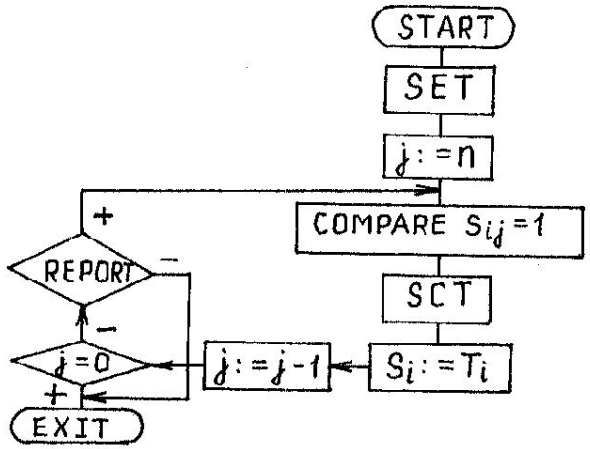


АРИФМЕТИЧЕСКИЕ АЛГОРИТМЫ для АПП

a



b



c

N вар. (дейст.)	Нач. СВА	Рез. С В	Комментарий
0	0 0 0	0 0	Нет действий
1(2)	0 0 1	0 1	Делать после 3
2	0 1 0	0 1	Нет действий
3 (1)	0 1 1	1 0	Делать перед 1
4 (3)	1 0 0	0 1	Делать перед 6
5	1 0 1	1 0	Нет действий
6 (4)	1 1 0	1 0	Делать после 4
7	1 1 1	1 1	Нет действий

Сигналы от ЦУУ:

WR – записать в *ответчики*. Для всех ячеек, которые являются ответчиками (T_i=1) и выбраны (S_i=1), содержимое компаранда поразрядно пишется в эти ячейки с учетом разрядной маски.

WNR – записать в *неответчики*.Запись – аналогично предыдущему, но уже в ячейки, для которых $T_i = 0$ и $S_i = 1$.

STF – записать T_i. Для выбранных ячеек ($S_i = 1$) содержимое тега переписывается в j-разряд, если он размаскирован. Операция проводится в два приема. Сначала – запись 1 в ответчики ($WI_j = 1$) . Затем – запись 0 в неответчики ($WZ_j = 1$).

SCT – хранить дополнение до Ti, т.е. $\overline{T_i}$ (инверсная запись). Сначала – запись1 в неответчики ($WI_j = 1$). Затем – запись 0 в ответчики ($WZ_j = 1$).

К алгоритму сложения полей:

Пример. Сложить число A в разрядах 1-10 с числом B в разрядах 21-30, сохраняя результат в разрядах 21-30.

Два разряда каждой строки отведем под флажки – выбора X_i и переноса C_i.

Действие 1 :

SET $\rightarrow$ **COMPARE** $X_i = 1$ $\rightarrow$ **COMPARE** $S_{ij} = 1$ $\rightarrow$ **COMPARE** $S_{i,j+20} = 1$

$\rightarrow$ **COMPARE** $C_i = 0$ $\rightarrow$ **WRITE** $S_{i,j+20} = 0$ $\rightarrow$ **WRITE** $C_i = 1$.

Это дает 7 обращений к памяти. Действие 2 требует 6 обращений. Действие 3 – семь. Действие 4 – шесть. Всего имеем 26 обращений на разряд.

ОРТОГОНАЛЬНАЯ ПАМЯТЬ

Специально для ассоциативного процессора:

n слов данных разрядностью **n** – по обратным диагоналям матрицы **n x n.**

Каждому слову – свой вес **G**.

Для рис. *a,c*:

n = 4, слова – **a (G=0), b (G=1), c (G=2)** и **d (G=3).**

Разрядный срез – вектор $\langle d_i, c_i, b_i, a_i \rangle$.

Его компоненты – в разряде **i** модулей памяти M_0 – M_3 (**рис. *a***).

Номер модуля **μ,** где хранится **i**-компонента слова,

$$\mu = |G + i|_n \qquad (a_2 \text{ – в } M_2, \text{ а } d_2 \text{ – в } M_1).$$

Доступ к компонентам вектора и разрядам слова – без конфликтов.

Глобальный адрес **битового среза** – **i**.

Суммирование (**рис. *c***) не проводится. При чт.-зп. разрядных срезов (a_i справа) – ***циклический сдвиг вправо*** (***при чтении***) или ***влево*** (***при записи***) на **i** разрядов.

Глобальный адрес **слова** – **(n – G)**.

Номер разряда i *этого слова в модуле* ***μ***: $i = |(n - G) + \mu|_n$.

Для приведения слова к **стандартной форме** – ***аналогичный сдвиг*** на **G** разрядов.

Фирма **Goodyear** разработала иную схему хранения данных (**рис. *b*; случай n = 8**). Разряд **i** слова **W** хранится в разряде **i** модуля $\mu = i \oplus W$. В итоге: ***бесконфликтный доступ по словам, битовым срезам и фрагментам различных подмножеств слов .***

a

N разряда →

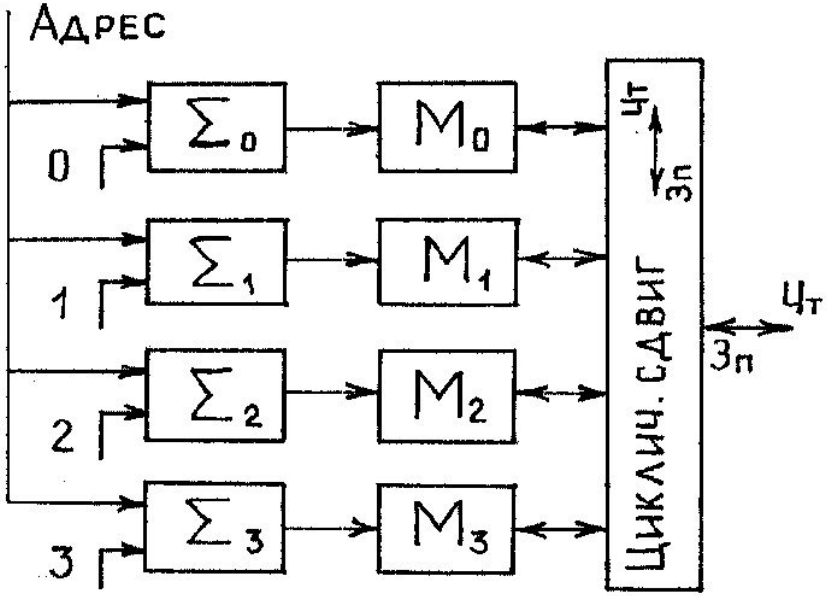
N модуля ↓	0	1	2	3
0	a_0	d_1	c_2	b_3
1	b_0	a_1	d_2	c_3
2	c_0	b_1	a_2	d_3
3	d_0	c_1	b_2	a_3

b

W(μ)

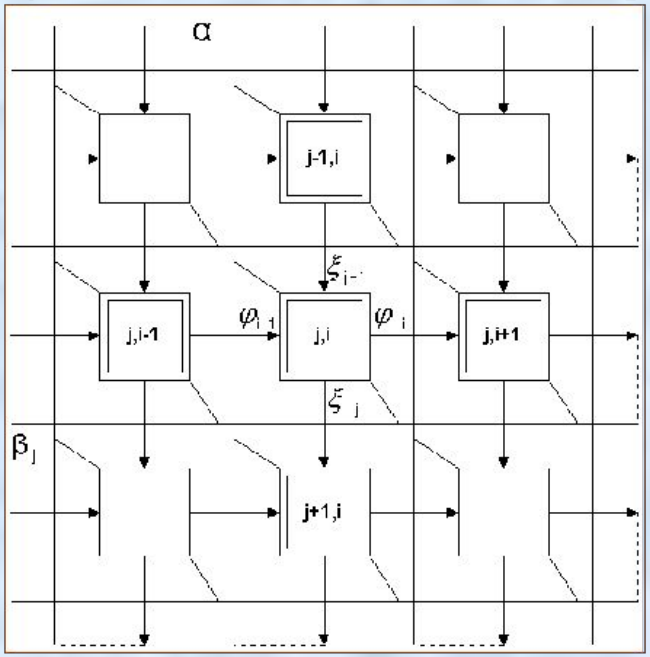
μ (W)	Разряд 0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	0	3	2	5	4	7	6
2	2	3	0	1	6	7	4	5
3	3	2	1	0	7	6	5	4
4	4	5	6	7	0	1	2	3
5	5	4	7	6	1	0	3	2
6	6	7	4	5	2	3	0	1
7	7	6	5	4	3	2	1	0

c



2. ОПЕРАЦИОННЫЕ ЛЗС КАК АССОЦИАТИВНЫЕ МАТРИЦЫ

Операционная ЛЗС – это итеративная двумерная структура, которая реализует заданное множество процедур независимо от размеров среды. Элемент (j, i) среды содержит ячейку памяти для хранения 1 бита исходной информации и автоматную часть.



Признаки α_i, β_j являются общими для i-столбца и j-строки. Они могут выполнять функции *настройки, маскирования, разрешения считывания или записи, числовых разрядов, результатов анализа содержимого строк или столбцов* и т.д.

Выходы элемента (j, i) являются ***автоматными функциями*** входов и содержимого $a_{j,i}$ ячейки памяти. Элементы могут иметь и отдельные выходы. Допускается ***коммутация шин*** по краям среды (*пунктир* на рис.).

«Обрамление» среды составляют группы регистров, в которые помещаются исходные данные (признаки), маски и конечные результаты. Запись (чтение) информации во всех ячейках памяти среды происходит ***параллельно*** из (в) одного слоя трехмерной памяти данных (буфера). Размеры слоев и операционной матрицы ***одинаковы***.

Алгоритмам, реализуемым ЛЗС, свойственны характерные признаки ***ассоциативной*** обработки: *зависимость производительности от размеров матрицы*, высокий удельный вес ***поисковых*** операций, широкое использование операций ***над масками***. Поэтому **операционные среды – это ассоциативные матрицы**, в которых обработка массивов данных выполняется ***параллельно по словам и последовательно по слоям*** трехмерной «быстрой» памяти.

Чем шире набор микроопераций ЛЗС, тем короче микропрограмма, т.е. быстрее выполняется процедура. Но это приводит к усложнению базового модуля, который в ***универсальном*** варианте приобретает черты **секционного микропроцессора**. Обычно ***размеры матрицы*** ЛЗС ***сравнительно невелики***. Обработка больших массивов данных ведется **по фрагментам**. Быстродействие системы в целом во многом зависит от успешного решения проблемы ***совмещения операций обработки и обменов***.

Процедуры обработки **однородных** (по структуре данных) массивов информации имеют *высокий удельный вес* в современных ИВС. Поэтому требуются спец. подсистемы, в которых эффективно реализуются операторы над массивами, – ***процессоры массивов***. Они работают в комплексе с основной (Host) ЭВМ, являясь для нее ***акселераторами*** (ускорителями процессов) на заданном множестве процедур массовой обработки. С выполнением этих процедур связано решение множества задач проблемной ориентации: обработки изображений, цифровой обработки сигналов, распознавания образов, линейной алгебры, математической физики, управления базами данных и др.

Определенному разбиению этого множества будет отвечать некоторая библиотека базовых модулей ЛЗС. Двумерная матрица взаимосвязанных однотипных модулей из этой библиотеки с общим микропрограммным управлением быть взята за основу построения процессоров массивов.

При этом возможны два пути:

1. ***Широкая ориентация на основные виды параллельной обработки*** с целью построения достаточно универсального средства. Эта ориентация наиболее приемлема для современного уровня технологии, отвечает тенденциям развития микропроцессорной техники и минимизирует состав пресловутой библиотеки. ***Она будет рассмотрена далее на примере матричного процессора ассоциативного типа*** (ПМА).

2. ***Ориентация на определенный класс решаемых задач*** (**узкая специализация**). Практические возможности узкой специализации ограничены необходимостью производства спец. БИС. Но выпуск крупных серий одного -двух типов заказных БИС для особо эффективных применений ЛЗС может иметь достаточное *технико-экономическое обоснование*.

На примере **узкой специализации** можно сравнительно просто и достаточно «выпукло» показать некоторые особенности организации и функционирования достаточно сложных систолических матриц. В качестве примера выбрана ***задача реализации двумерного ассоциативного оператора*** (построения спецпроцессора-идентификатора). Она имеет самостоятельное познавательное значение и определенный практический интерес.

СПЕЦПРОЦЕССОР-ИДЕНТИФИКАТОР НА ОСНОВЕ ЛЗС

РАССМАТРИВАЕМАЯ ЗАДАЧА – ***лексическое распознавание*** объектов стилизованных бинарных изображений в терминах ***«объекты – координаты»*** на основе операционных **ЛЗС**. Каждый тип объекта представлен своей двоичной матрицей-***эталоном***. Размеры всех эталонов одинаковы. Допустимые ***искажения объектов*** (в сравнении с эталоном) ***ограничены*** возможной инверсией их отдельных элементов, определенных ***маской***.

Задана троичная матрица

$$X = |x_{pq}| = \begin{vmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \dots & \dots & \dots & \dots \\ x_{m1} & x_{m2} & \dots & x_{mn} \end{vmatrix}, \quad x_{pq} \in \{0, 1, -\}.$$

Здесь « – » означает безразличное значение x_{pq} (0 либо 1).

Требуется определить все ее вхождения в двоичный массив

$$A = |a_{ji}|, \; j = 1 \dots K, \quad i = 1 \dots L; \quad K > m, \; L > n,$$

который превышает по размерам матрицу X .

ИЗВЕСТНОЕ (*курса «Схемотехника ЭВМ»*) ПОЛОЖЕНИЕ: ***Если размеры операционной матрицы равны размерам анализируемой сцены K x L бит и эти размеры значительны, то в случае однотактного распознавания сравнительно крупных объектов для построения ЛЗС потребуется чрезмерно большое число корпусов БИС.***

ПРЕДВАРИТЕЛЬНЫЕ ЗАМЕЧАНИЯ

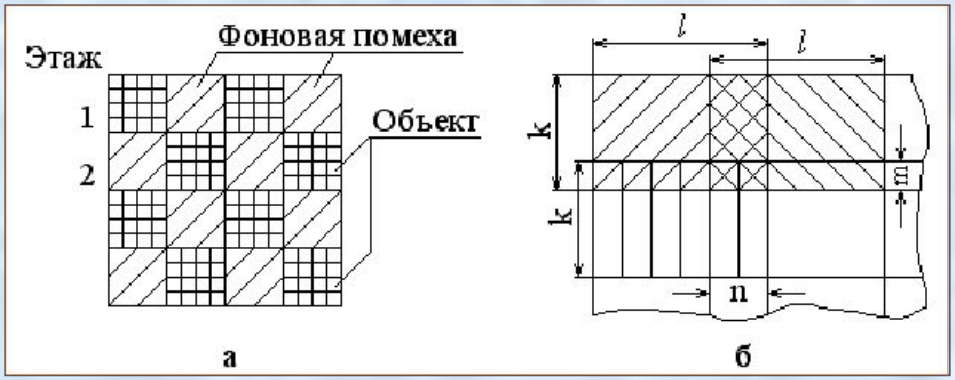
Реальные возможности ***анализа сцен*** (двоичных матриц A) *приемлемо больших размеров* K x L бит *все же существуют*. Процесс распознавания может быть построен на операционных матрицах меньших размеров k×***l*** бит (k < K, ***l*** < L) со значением k и ***l*** до 128. При этом анализ изображения выполняется *последовательно по кадрам* k×***l*** бит и ***многотактно*** – фрагментами $(1\times1)n_1$ бит, $n_1 \geq 2$, – для каждого эталона. Тогда в кадре оказывается возможным распознавание объектов различных размеров $(d\times c)n_1$, где d =1 … $[k/n_1]$ и c = 1 … $[l/n_1]$, при разумной сложности операционной матрицы. В случае k = ***l*** = 128, n_1 = 4 количество корпусов W = 2816. Если же n_1=2, то W=1024. Это приемлемо.

Многотактная организация процедуры распознавания приводит к необходимости разработки ***спецпроцессора-идентификатора***, работающего в комплексе с универсальной (Host) ЭВМ. Эта ЭВМ выполняет управляющие функции: диспетчирование, прием – передача и формирование массивов данных для обработки и хранения. Спецпроцессор реализует параллельную часть обработки.

Основной вклад в объем его оборудования вносит обрабатывающая часть. *Помимо операционной матрицы, она содержит многослойную буферную память с параллельным обменом данными между каждым слоем буфера и матрицей.* В спецпроцессоре имеется свое устройство управления, оперативная память и так называемый координатный блок. Все это определяется ***алгоритмом многотактного распознавания***.

АЛГОРИТМ МНОГОТАКТНОГО РАСПОЗНАВАНИЯ

Размеры всех объектов будем полагать *одинаковыми*, а сами *объекты – непересекающимися*. Оценку дадим для случая "шашечного" расположения объектов (рис. а), когда *количество объектов на каждом "этаже" одинаково и равно* ε. В качестве варьируемых параметров возьмем *размеры объектов* m×n, *изображения* в целом K×L, *число типов объектов* γ. Анализ проведем в предположении наличия между объектами *непрерывной фоновой помехи*, полагая величины L, ***l*** и n *кратными байту*.



Идентификацию на комплексе будем проводить в два этапа:

Этап 1. Формирование кадров – реализуется *на базовой* ЭВМ.

Этап 2. Идентификация объектов каждого кадра, *последовательно по кадрам* – реализуется *на спецпроцессоре*.

При работе с отдельным кадром k×***l*** бит распознаются лишь те объекты, которые *полностью входят в кадр*. Поэтому кадры формируются *с перекрытием* (рис. б): *по горизонтали* – на n бит, *по вертикали* – на m бит. Число кадров по горизонтали $Q =](L - n)/(l - n)[$ и по вертикали $P =](K - m)/(k - m)[$. Общее количество кадров в изображении равно PQ. *Каждый эталон разделяется на* cd *непересекающихся фрагментов* $||\lambda_{pq}||_{su}$ ($p, q = 1....n_1$; $n_1 \geq 2$; $s = 1...c$; $u = 1...d$).

АЛГОРИТМ :

1. $w := 1$, $u := 1$.
2. $s := 1$.
3. Опросить кадр на Λ_{su}.
4. По результатам опроса сформировать двоичную матрицу
$$X' = \|x'_{ji}\| \quad (j = n_1 \ldots k; \quad i = n_1 \ldots \ell).$$
5. $s := s + 1$.
6. Если $s \le c$, идти к шагу 7. Иначе переход к шагу 10 (для $w = 1$) или к шагу 11 (при $w = 2$).
7. Опросить кадр на Λ_{su}.
8. По результатам опроса сформировать матрицу $X'' = \|x''_{ji}\|$ и образовать ее покомпонентную дизъюнкцию с X'.
9. Опросить матрицу $(X' \vee X'')$ на признак-строку $\Gamma = \|1' - - - 1''\|$, где число безразличных состояний равно $(n_1 - 1)$. Переход к шагу 4.
10. По результатам опроса сформировать матрицу $Y' = \|y'_{ji}\|$. Переход к шагу 13.
11. По результатам опроса сформировать матрицу $Y'' = \|Y''_{ji}\|$ и образовать ее покомпонентную дизъюнкцию с Y'.
12. Опросить матрицу $(Y' \vee Y'')$ на признак-столбец Γ^T (транспонированная матрица Γ). Переход к шагу 10.
13. $w := 2$; $u := u + 1$.
14. Если $u \le d$, идти к шагу 2. Иначе – к шагу 15.
15. Результат опроса принять за конечный результат.

Рисунок сжато иллюстрирует последовательность процесса многотактного распознавания по этому алгоритму в случае $k = 6$; $l = 14$; $m = 4$; $n = 6$; $n_1 = 2$ ($c = 3$, $d = 2$) на примере матриц:

$$X = \left\| \begin{array}{ccc|ccc|ccc} & \Lambda_{11} & & & \Lambda_{21} & & & \Lambda_{31} & \\ 1 & - & & - & 0 & & 1 & 1 & \\ 0 & 1 & & 1 & 1 & & 0 & 1 & \\ \hline 1 & 0 & & 0 & 1 & & 0 & - & \\ 1 & 1 & & - & 1 & & - & 1 & \\ & \Lambda_{12} & & & \Lambda_{22} & & & \Lambda_{32} & \end{array} \right\|,$$

$$A = \left\| \begin{array}{cccccccccccccc} 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ \hline 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \end{array} \right\|.$$

На рисунке представлены:

а – результат опроса A на $\Lambda_{11} \to X'$;

б – результат опроса A на $(\Lambda_{11} \vee \Lambda_{21}) \to (X' \vee X'')$;

в – результат опроса $(X' \vee X'')$ на $\Gamma = \|1' - 1''\|$ (опроса A на $\Lambda_{11}\Lambda_{21}$) → новая матрица X';

г – результат опроса A на $(\Lambda_{11}\Lambda_{21} \vee \Lambda_{31}) \to$ новая матрица $(X' \vee X'')$;

д – результат опроса $(X' \vee X'')$ на Γ (опроса A на $\Lambda_{11}\Lambda_{21}\Lambda_{31}) \to X' \to Y'$;

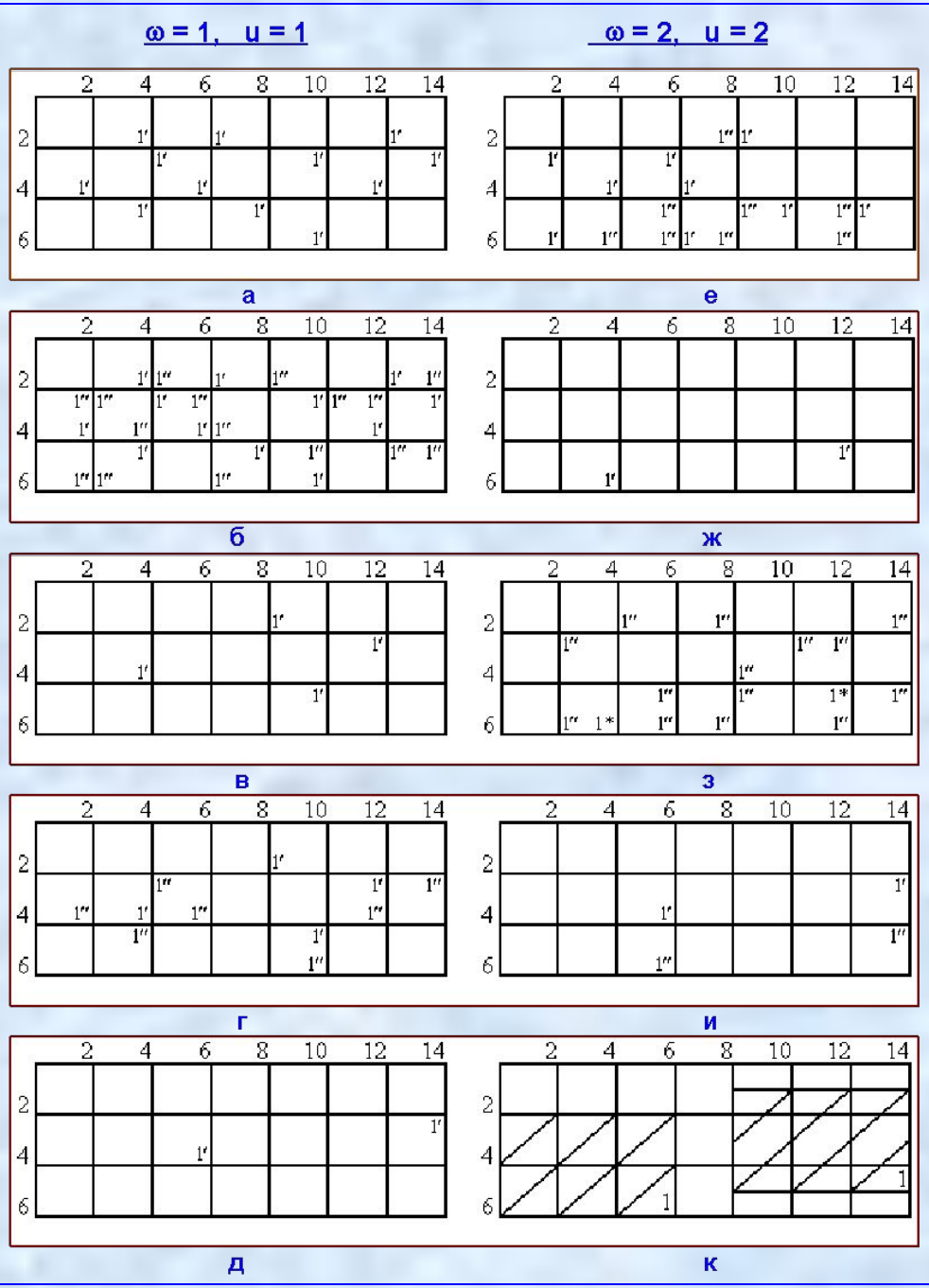
е – результат опроса A на $(\Lambda_{12} \vee \Lambda_{22}) \to (X' \vee X'')$;

ж – результат опроса A на $\Lambda_{12}\Lambda_{22} \to X'$;

з – результат опроса A на $(\Lambda_{12}\Lambda_{22} \vee \Lambda_{32}) \to (X' \vee X'')$, $1^* = 1' \vee 1''$;

и – результат опроса A на $(\Lambda_{11}\Lambda_{21}\Lambda_{31} \vee \Lambda_{12}\Lambda_{22}\Lambda_{32}) \to (Y' \vee Y'')$;

к – результат опроса $(Y' \vee Y'')$ на Γ^T (опроса A на X) → $Y' \to$ конечный результат распознавания.



СПЕЦПРОЦЕССОР

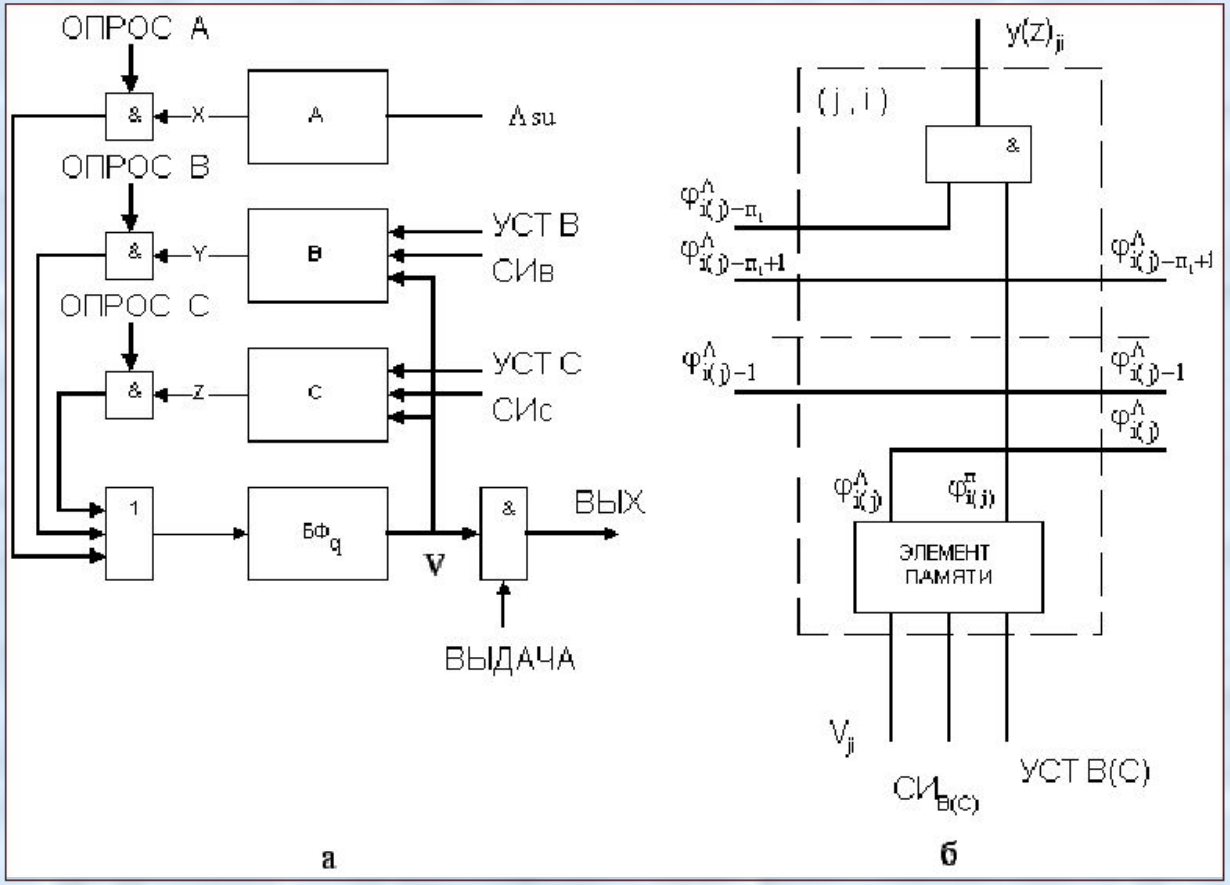
СТРУКТУРА

Для реализации АЛГОРИТМА необходимо образовать обрабатывающий массив (рис.а *на след. слайде*) из буферной памяти ($БФ_q$ – q-й слой буфера) и трех операционных матриц:

A (*базовая матрица*) – структура однотактного распознавания фрагментов Λ_{su};

B – среда формирования матриц X' и (X' v X") и распознавания строчных фрагментов Г;

C – среда формирования матриц Y' и (Y' v Y") и распознавания столбцовых фрагментов $Г^T$



ФУНКЦИОНИРОВАНИЕ

Анализируемый кадр размещается в элементах памяти матрицы A. По сигналам ОПРОС A, B, C результаты опроса соответствующих сред $X = \|x_{ji}\|$, $Y = \|y_{ji}\|$, $Z = \|z_{ji}\|$ записываются в $БФ_q$.

При подаче сигнала $СИ_{B,C}$ происходит переключение матрицы B или C *в соответствии с* информацией $V = \|v_{ji}\|$, записанной ранее в $БФ_q$. *Одновременно формируются результаты очередного опроса на* Г *или* $Г^T$.

Сигналы УСТ В,С подаются *перед началом распознавания*. Они устанавливают элементы памяти сред B и C *в исходное состояние*. ***В шаге 2*** алгоритма необходимо предусмотреть дополнительную установку матрицы B. По сигналу ВЫДАЧА из $БФ_q$ считывается результат.

Матрицы X′ (Y′) и X″ (Y″) различаются *"окраской"* содержащихся в них единиц (1′ и 1″), что достигается использованием в средах **B** и **C** специальных элементов памяти. Значения сигналов на выходах этих элементов: 00 – если данный элемент не является границей ни одного из двух фрагментов, последовательно распознаваемых предыдущей средой; 10, 01 и 11 – если данный элемент является границей только первого из этих фрагментов, только второго и обоих фрагментов соответственно.

Согласно **АЛГОРИТМУ** матрица X′ (Y′) в среде **B (C)** формируется по каждому *нечетному* синхроимпульсу, а матрица X″ (Y″) – по каждому *четному*. Это отражено в таблице переходов рассматриваемых элементов, где v – соответствующая компонента матрицы V; s^k – состояние; $\varphi_1\varphi_2{}^k$ – выходы.

S^{k-1} \ V	0	1
1	2, 00	3, 10
2	2, 00	1, 01
3	1, 10	1, 11

$S^k;\ \varphi_1\varphi_2{}^k$

При составлении таблицы учтено, что для принятой кодировки выходов в матрицах **B** или **C** распознаются фрагменты $\Phi = \|(1-)(--)\ ...(--)(-1)\|$ либо Φ^T с двухбитными компонентами. Поэтому, если после четного синхроимпульса на выходах элемента памяти имеем 00 либо 10, то очередное значение v = 0.

Организация элемента **(j,i)** среды **B(C)** показана на рис.б. Входные сигналы на границе среды – нулевые. Через $\varphi_{i(j)}{}^{л}$, $\varphi_{i(j)}{}^{п}$ обозначены левый и правый выходы элемента памяти.

ДОПОЛНИТЕЛЬНЫЕ БЛОКИ И ЭФФЕКТИВНОСТЬ

Анализируемые кадры r = 0 … (PQ – 1) размещаются в буферной памяти спецпроцессора, по одному слою $\mathbf{БФ_r}$ на каждый кадр r. Информация о троичных эталонах хранится в *оперативной памяти* спецпроцессора. Имеется *байтовый регистр*, в который последовательно заносятся фрагменты Λ_{su}.

В буфере выделяется рабочий слой $\mathbf{БФ_q}$. Он постоянно используется для получения промежуточных результатов распознавания в виде единичных отметок в соответствующих битах рассматриваемого в данный момент кадра по каждому типу объекта. Эти результаты передаются в *координатный блок*, который функционирует параллельно с обрабатывающей частью спецпроцессора.

Координатный блок служит для определения координат идентифицированных объектов. Он содержит простейшую *операционную матрицу* k × ***l*** бит с *регистровым обрамлением*, выполняющую поиск в строках на "≠ 0", и микропрограммное управление. Итоговая информация {t, r, j, i} о каждом факте идентификации заносится в оперативную память спецпроцессора.

Результаты расчетов, проведенных для случая k = ***l*** =128, даны в таблице. Здесь $T_п$ и T_c – времена анализа изображения в целом только на Host и на комплексе соответственно при одинаковой длительн. тактов.

K=L	m=n	γ	$T_п/T_c$
1008	24	64	91,68
1008	24	32	87,80
504	24	64	92,72
1008	**16**	64	**242,0**

При этом реализация трех операционных матриц (**A**, **B**, **C**) потребует 2294 *корпусов* **БИС** на 60 *выводов*. Использование кристаллов **ПЛИС** на 264 *сигнальных выводов* позволит *снизить* необходимое *число корпусов* до 406.