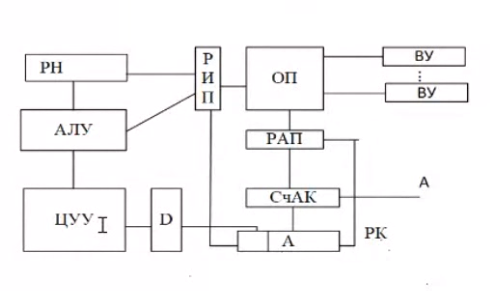
Компьютерные системы

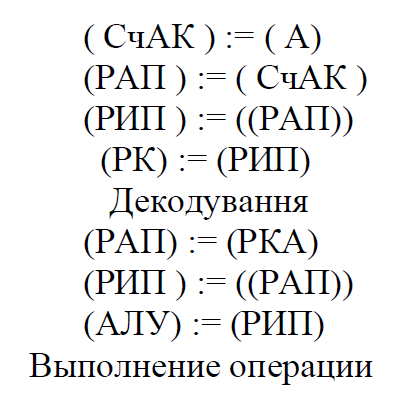
# 01.09.2020

## Особенности машин первого и второго поколения



регистр команд, регистр информации памяти, регистр адреса памяти, регистр накопления

1. Все функции управления сосредоточены в единственном ЦУУ (центральном устройстве управления). То есть все устройства (АЛУ, оперативная память и тд) лишены автономности, устройство может работать только если к нему подключено ЦУУ. ЦУУ подключается к устройствам последовательно, из-за чего появляется большое количество простоев. Увеличение количества элементов проблему не решит. Решение - дать автономность устройствам, чтобы они могли выполнять свои функции без подключения к ним ЦУУ.
2. Огромное количество пересылок и невозможность их минимизировать



1. Узкая специализация регистров, из-за чего повышать их количество также не имеет смысла
2. В машинах первого поколения был непосредственный доступ (во всём нужно участие человека), во втором - пакетная обработка.

## Машины третьего поколения (мультипрограммная организация вычислений)

1. Децентрализация функций управления и реализация автономности всех устройств в системе. ЦУУ задал работу элементу - он сам её дальше выполняет.
2. Для определения завершения работы устройства была введена система прерываний
3. Коллективное использование оперативной памяти
4. Есть первый вариант параллелизма благодаря параллельной работе специализированных процессоров ввода-вывода и центрального процессора, который выполнял прикладную задачу.
5. Универсализация регистров

01.09 2 часть 75 минут - 4 поколение

# 08.09.2020

Транспьютеры

* впервые введено понятие поток
* введено аппаратное прерывание, которое выполнялось за 1 такт

4-ое поколение машин было проблемно-ориентированным

5-ое поколение (современное) - большая производительность, “система массового распараллеливания”

## Цели параллелизма

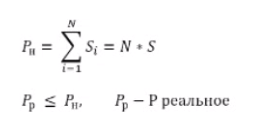
1. Повышение надёжности
2. Повышение *отказоустойчивости* (живучести - возможность корректной работы системы в условиях отказа).

*Надёжность* - время безотказной работы. Отказоустойчивость - логический элемент надёжности, её основным свойством является избыточность. Надёжность повышается физическим способом за счёт каждого отдельного элемента.

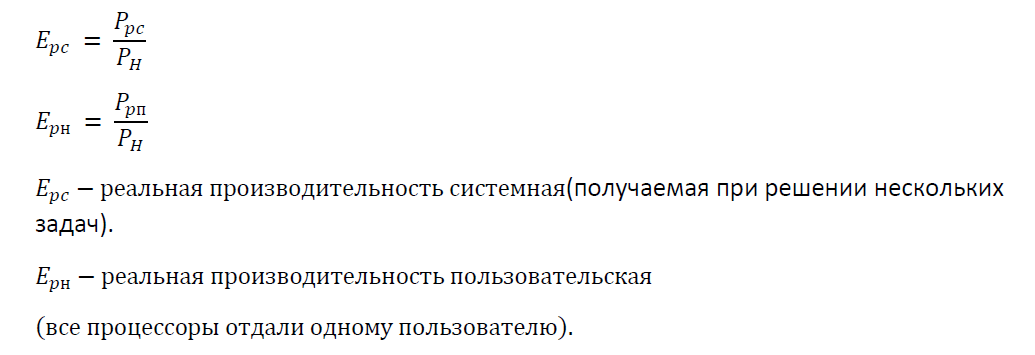
1. Уменьшение плотности потоков информации по каналам связи (уменьшение стоимости каналов связи)
2. Повышение производительности



Номинальная производительность - это сумма производительностей всех элементов



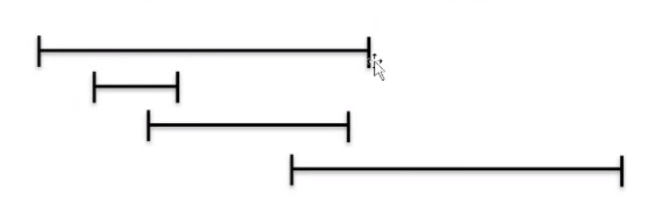
Реальная производительность всегда меньше номинальной из-за накладных расходов параллельной обработки (взаимодействие, планирование и разработка расписания)



Самой важной производительностью является пользовательская.

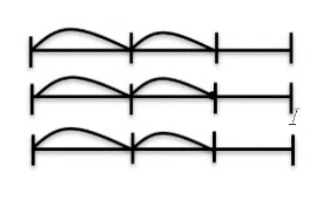
## Способы организации параллелизма

### Асинхронный



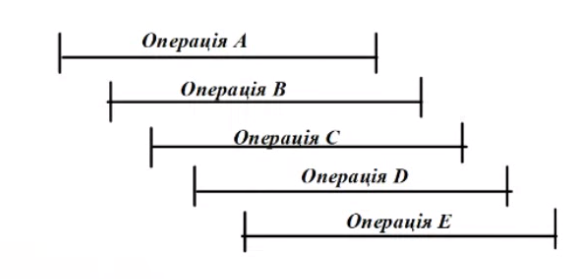
* Все процессы работают асинхронно и независимо друг от друга, то есть моменты начала, периоды развития и моменты завершения процессов или потоков никак не связаны между собой и могут реализовываться в любые моменты времени.
* Примером асинхронной системы являются мультипроцессорные системы
* Взаимодействие происходит на уровне процессов или потоков

### Синхронный



* Взаимодействие происходит на уровне операций или простых процедур
* Возникновение и завершение элементов происходит в общих точках

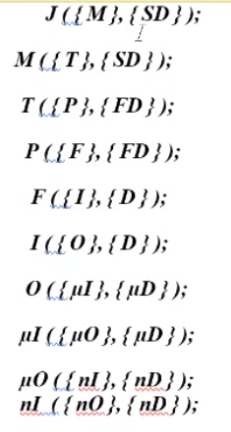
### Перекрывающий (конфликтный)



* обычно на уровне операций, но иногда на более высоком
* примером является конвейерная организация
* В конвейерной организации происходит объединение пространственной и временной организации параллелизма

## Уровни распараллеливания

1. Уровень задания
2. Уровень задач
3. Уровень команд
4. Микроком
5. Наноком



J - job (задание), SD - set of data (файл файлов, например библиотека), FD - file of data. M - шаги задания. T - task (задача, реально выполняемая программа). P - процесс. F - flow (поток). I - instruction (команда). O - operation. mI - microinstruction, mD - micro data. mO - miro operation

*возможно нужно подробнее про уровни написать*

# 15.09.2020, 22.09.2020, 29.09.2020

Накладные расходы - это прежде всего расходы на взаимодействие элементов между собой.

Процесс - экземпляр программы во время выполнения, независимый объект, которому выделены системные ресурсы (например, процессорное время и память). Каждый процесс выполняется в отдельном адресном пространстве: один процесс не может получить доступ к переменным и структурам данных другого.

Поток — определенный способ выполнения процесса. Когда один поток изменяет ресурс процесса, это изменение сразу же становится видно другим потокам этого процесса.

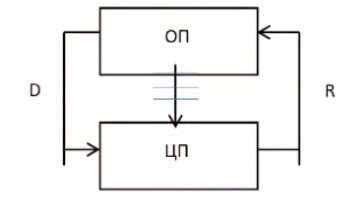
Аналогия со спектаклем: текст спектакля - это задача. Одна постановка спектакля - это процесс, актёры - это потоки. В каждой постановке могут участвовать разные актёры, быть разный настрой, поэтому как не может быть одинаково сыгранных спектаклей так не может быть одинаково выполненных процессов. Все ресурсы (здание, декорации, костюмы) выделяются спектаклю, а не каждому актёру в отдельности, они пользуются ресурсами спектакля так же как потоки, на которых исполняется процесс пользуются ресурсами, выделенными системой этому процессу.

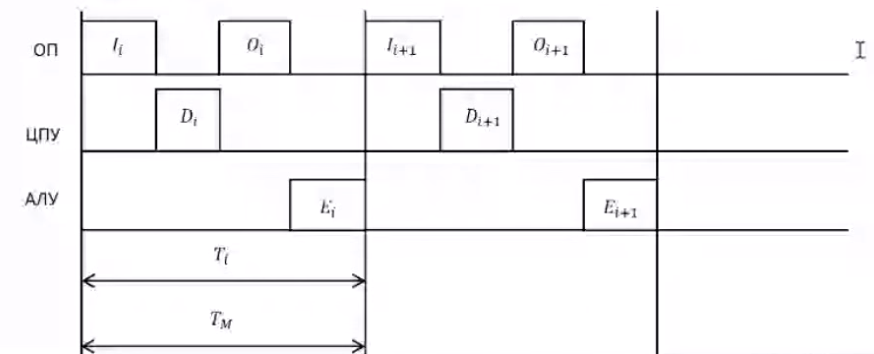
## Классификация ВС с точки зрения параллелизма

Самая простая и самая эффективная - классификация Флинна

Наряду с понятием параллелизма Флинн ввёл понятие множественности. На основании этого получилось 4 класса систем:

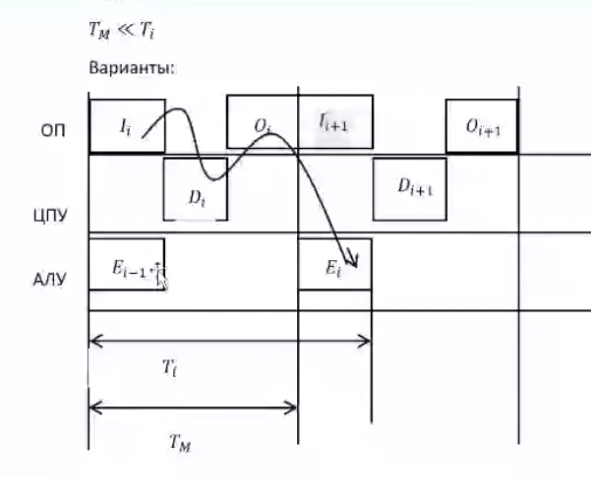
### **SISD** (Single instruction single data)

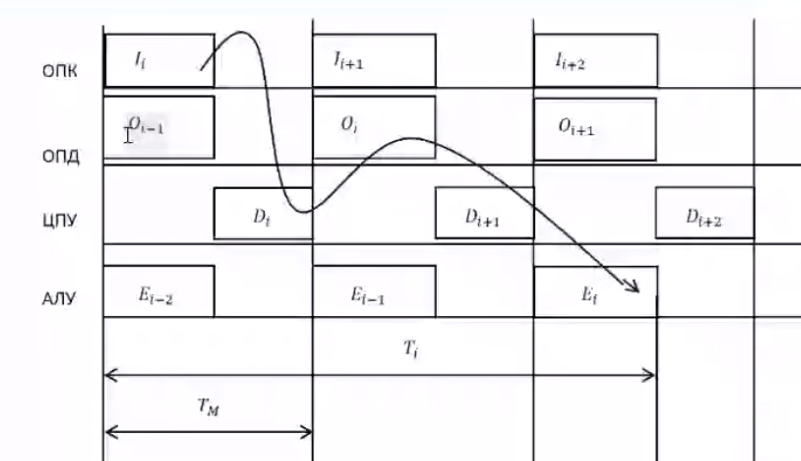


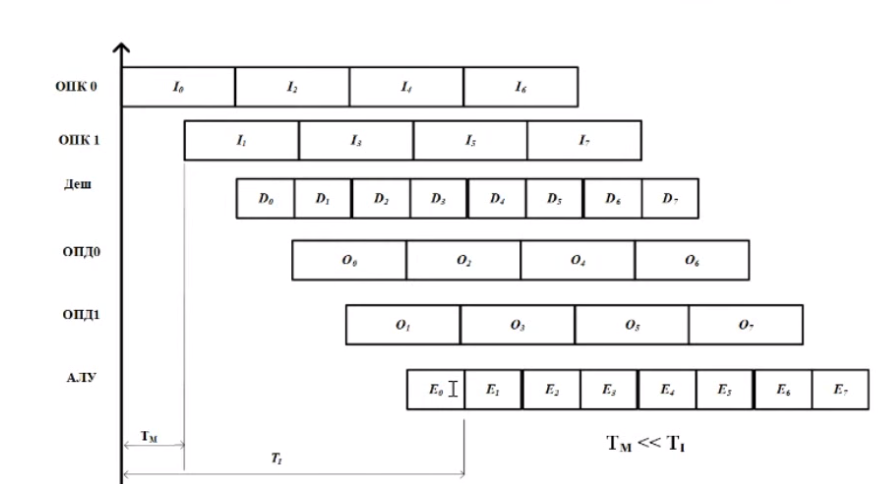


Получаем команду из ОП, декодируем её в ЦПУ, получаем операнд из ОП, выполняем её в АЛУ

Варианты модификации для уменьшения машинного цикла (ТМ):





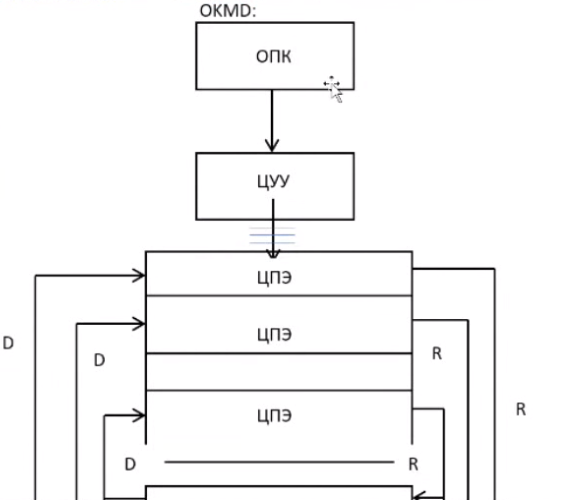


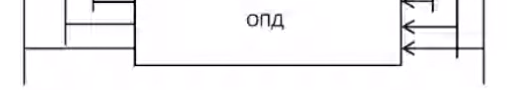
В модификациях не меняется ТИ

Дальнейшие улучшения без перехода к другому классу по Флинну невозможны.  
Данная схема работает только для полностью независимых друг от друга команд

Предельное время повышения производительности таких систем - время работы дешифратора

### **SIMD** (Single instruction multiple data)

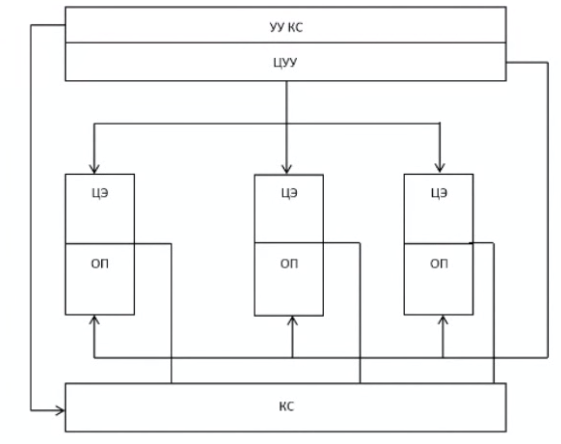




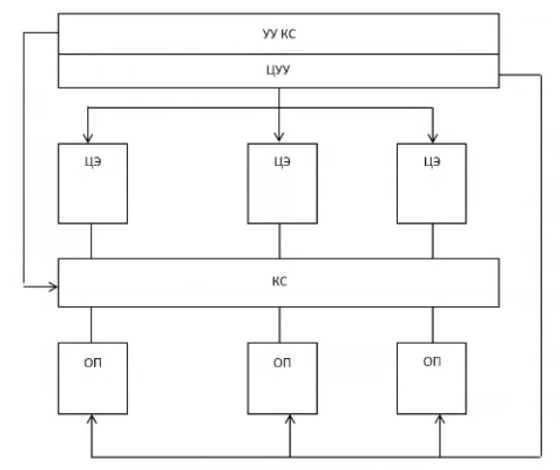
ОПК (оперативная память команд) выделена в отдельный элемент. Одна и та же команда поступает в центральные процессорные элементы одновременно и одновременно там выполняется. Выполняется всё в единственном последовательном потоке.

В данном классе различают следующие типы организации:

#### Матричная



*С индивидуальной памятью*

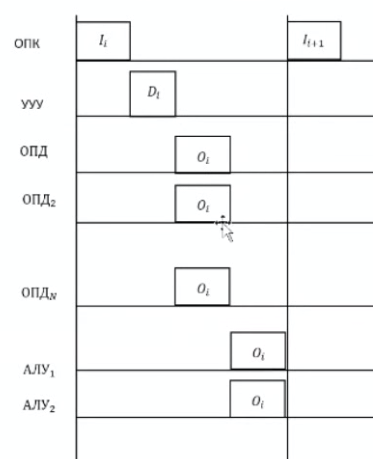
**

*С общей (разделяемой памятью)*

УУ КС - управляющее устройство коммутационной сети, ЦУУ - центральное устройство управление. Вся последовательность управляющих сигналов (в том числе и операций) распространяется по всем процессорным элементам.

Вариант с общей памятью практически не нашёл применения из-за большого числа конфликтов доступа к памяти и сложности их разрешения.

Машины с матричными системами - это по сути обычные машины 1 и 2 поколения с тем отличием, что вместо обычного ЦП в них используется матричный ЦП, все элементы которого выполняют одну и ту же команду ЦУУ. Однако по скорости, быстродействию и технологии это машина 4 поколения.



Выбирается команда, декодируется команда, выбирается несколько операндов, декодированная команда выполняется над всеми операндами в разных АЛУ.

Преимущества:

* дешёвые элементы
* простая организация
* можно добиться существенного улучшения быстродействия для конкретного типа задач

Недостатки:

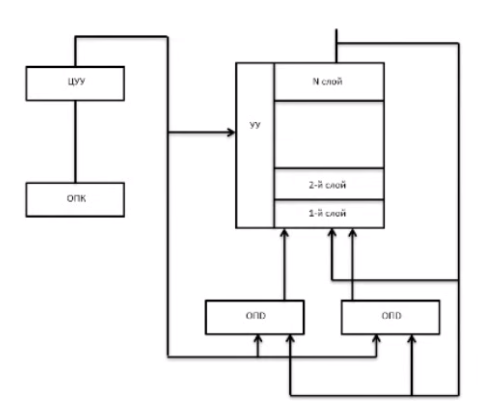
* узкая специализация и проблемноориентированность приводит к простою процессоров при выполнении скалярных операций
* несоответствие логического и физического векторов
* значительные затраты на операции взаимодействия (много пересылок - 1 команда = 1 пересылка)
* Деградация производительности при больших количествах ветвей

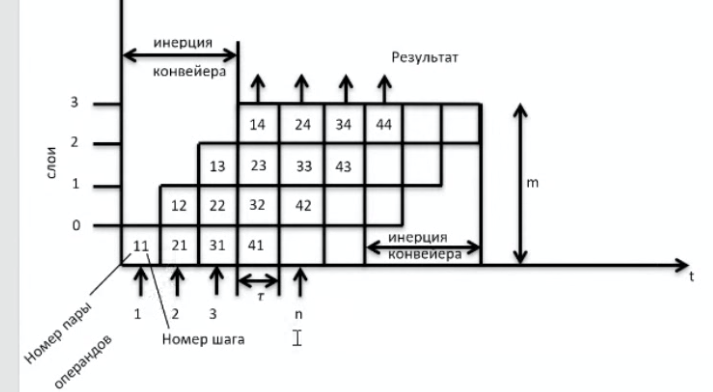
Первая реализация - машина Унгера, 1024 ПЭ, 1967 год.

В машине Соломона появилось маскирование - передача вектора размерности равной количеству процессоров. Если в этом векторе на месте какого-то процессора был 0 - то процессор не принимал участие в обработке. Таким образом программисту больше не нужно было управлять всеми процессорами вручную. Так же благодаря тому что все ПЭ управлялись одной мощной вычислительной машиной, можно было перенаправить на неё скалярные вычисления, а на матричной системе проводить только векторные (способ преодоления проблемноориентированности)

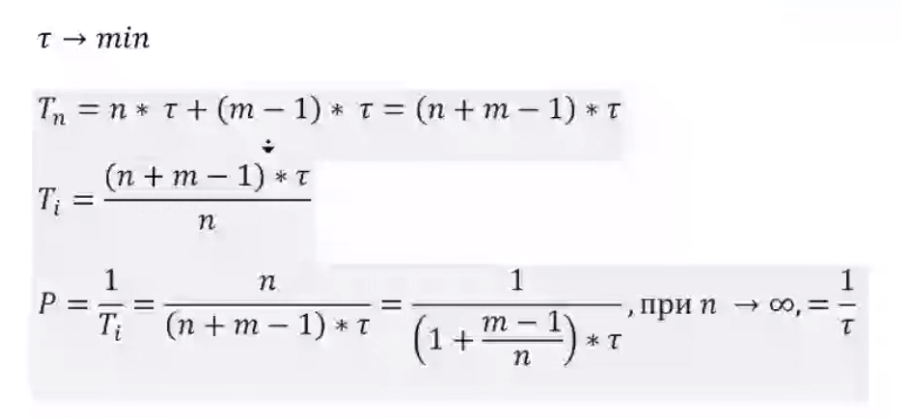
*15.09 ~100-130 минута - про илиак-4*

#### Конвейерная (векторная)



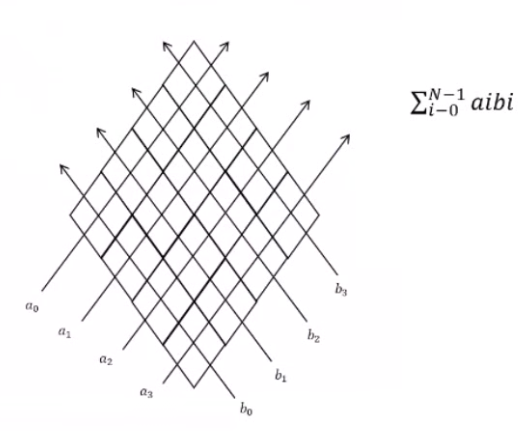


Пространственно-временная реализация параллелизма. чистый конвейер выполняет на каждом месте разные операции, поэтому такую организацию правильнее называть векторной.



Систолическая организация - это конвейерная организация, у которой реализованы только близкие связи (принцип близкодействия). Связи как таковые отсутствуют - выход элемента подключён напрямую к входу его соседа.

Структурные свойства системы базируются на использовании в качестве элементов построения такие топологические организации, которые предполагают возможность формирования вычислительной среды без налеганий и зазоров. К таким элементам в двумерном представлении можно отнести равносторонний треугольник, квадрат и правильный шестиугольник.

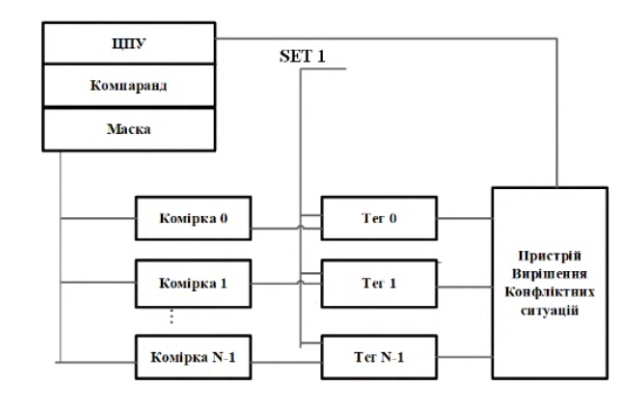


22.09 74 мин - пример умножения на конвейере

#### Ассоциативная

Фактически это матричная организация с использованием ассоциативной памяти. Особенности ассоциативной памяти:

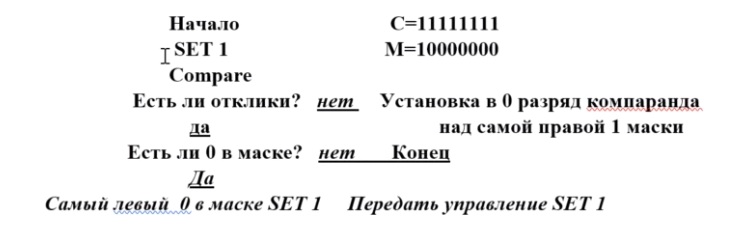
* естественным образом решается вопрос параллелизма, так как при её использовании все данные подвергаются одновременному анализу/преобразованию (все элементы массива). Поэтому ассоциативная память прежде всего оперирует массивами данных
* решается задача уменьшения количества операций пересылок, так как ассоциативная память позволяет некоторые операции проводить внутри самой памяти, не извлекая данные из неё

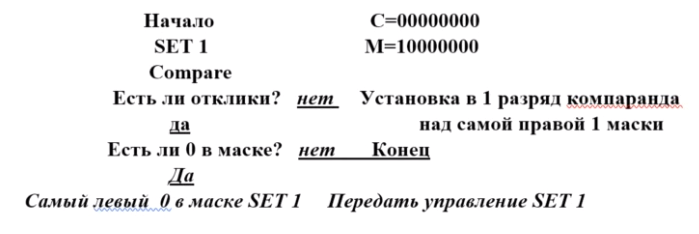


*Ядро ассоциативной памяти*

Компаранд - слово, с которым сравниваются все слова, находящиеся в памяти. Теги - это триггеры, которые сначала установлены в 1. Потом происходит сравнение слова с маской и все теги, кроме тех, где и в маске и в слове стоят 1, устанавливаются в 0.

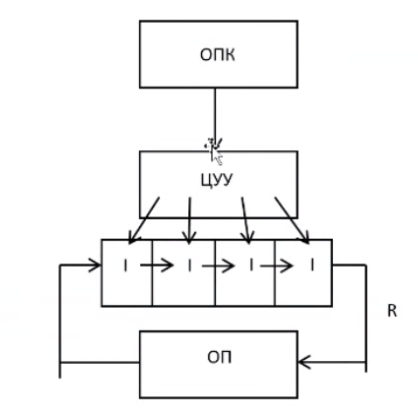
29.09 51-57 минута - поиск максимального и минимального числа в памяти



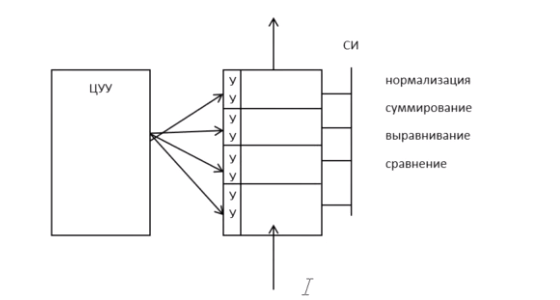


29.09 58-95 минута - схемотехника ассоциативной памяти

### **MISD** (Multiple instruction single data)



ЦУУ разделяет команду поступающую из ОПК на отдельные инструкции

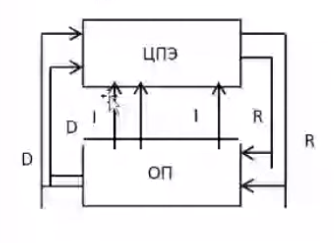


*Пример - операция над числами с плавающей точкой*

Фактически это тот же векторный конвейер, только вместо единого устройства управления, здесь своё УУ на каждом слое и на каждом слое выполняются разные операции.

поток данных должен быть достаточно скоростным, чтобы соотноситься с t(тау), которое мы пытаемся уменьшать.

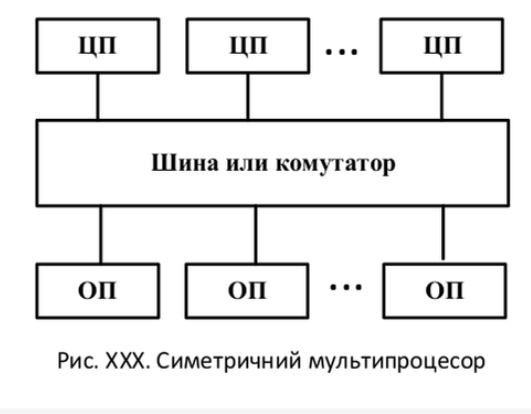
### **MIMD** (Multiple instruction multiple data)



# 24.11.2020

Что такое мультипроцессорные системы (SMP - Symmetric Multiprocessor) и какие у них особенности?

* 2 и больше процессоров
* У них всё разделяется: общая разделяемая оперативная память, общая разделяемая внешняя память, общее разделяемое устройство ввода-вывода, общая операционная система, общее виртуально-адресное пространство
* Все процессоры одинаковые
* есть 3 варианта взаимодействия (какие)
* Использование общего ресурса (в особенности - оперативной памяти) сильно ограничивает возможности (все обращаются одновременно к одному ресурсу, из-за этого создаётся очередь). с увеличение количества процессоров увеличивается количество конфликтов (*насыщение*)

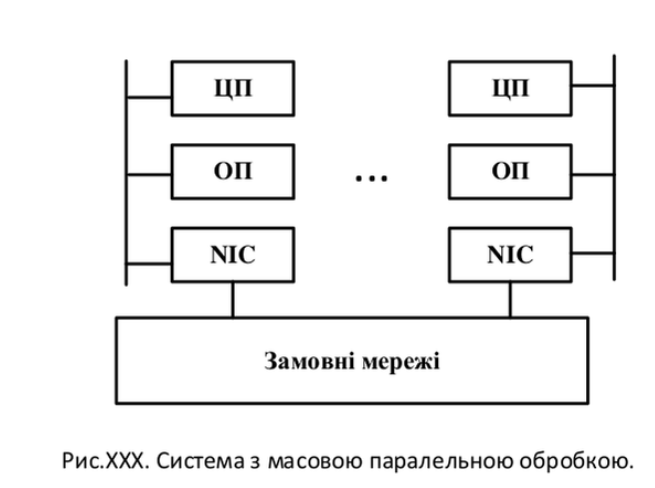


Мультикомпьютерные системы

* Всё разделено: каждый компьютер имеет свою собственную память, устройство ввода-вывода и тд. Общим является коммутатор
  + + : эта система масштабируемая (потому что нет общих ресурсов, а значит с увеличением количества элементов не увеличивается количество конфликтов)
* Фактически это сеть
* Ограничены только возможностями коммутатора

## Массивно-параллельный процессор (мультиконвейерный ?)

* Считается что к МПП относятся системы, которые имеют 128 и больше процессоров
* Элементы (сеть, программное обеспечение) заказные, поэтому такие системы дорогие
  + Передача данных идёт через пакеты, пользователей очень много, поэтому на разработчика ПО ложится сложная задача формирования пакетов сообщений и распределения ресурсов
* По сравнению с SMP решается проблема широкого распараллеливания, но происходит значительное усложнение системы
* 26 min



# 01.12.2020

SMP-системы

Плюсы

* не нужно думать про способы и алгоритмы передачи данных

минусы

* конфликты доступа к памяти

*были ещё вопросы, надо дополнить*

*начиная с 26 минуты:*

## Универсальные однородные структуры с индивидуальным поведением элементов

Основные идеи такой среды:

1. Исключение физических связей как таковых. Переход на логические связи.
2. Реализация, принципы быстродействия в которой должен быть реализован в максимальном варианте, что означает непосредственную связь “выход одного элемента - вход другого”
3. Возможность динамической перестройки связи