Нижегородский государственный университет
им. Н.И.Лобачевского

Факультет Вычислительной математики и кибернетики

Параллелизм как основа архитектуры ВС

Раздел 5

Динамическое планирование

Кудин А.В., к.т.н.

Содержание

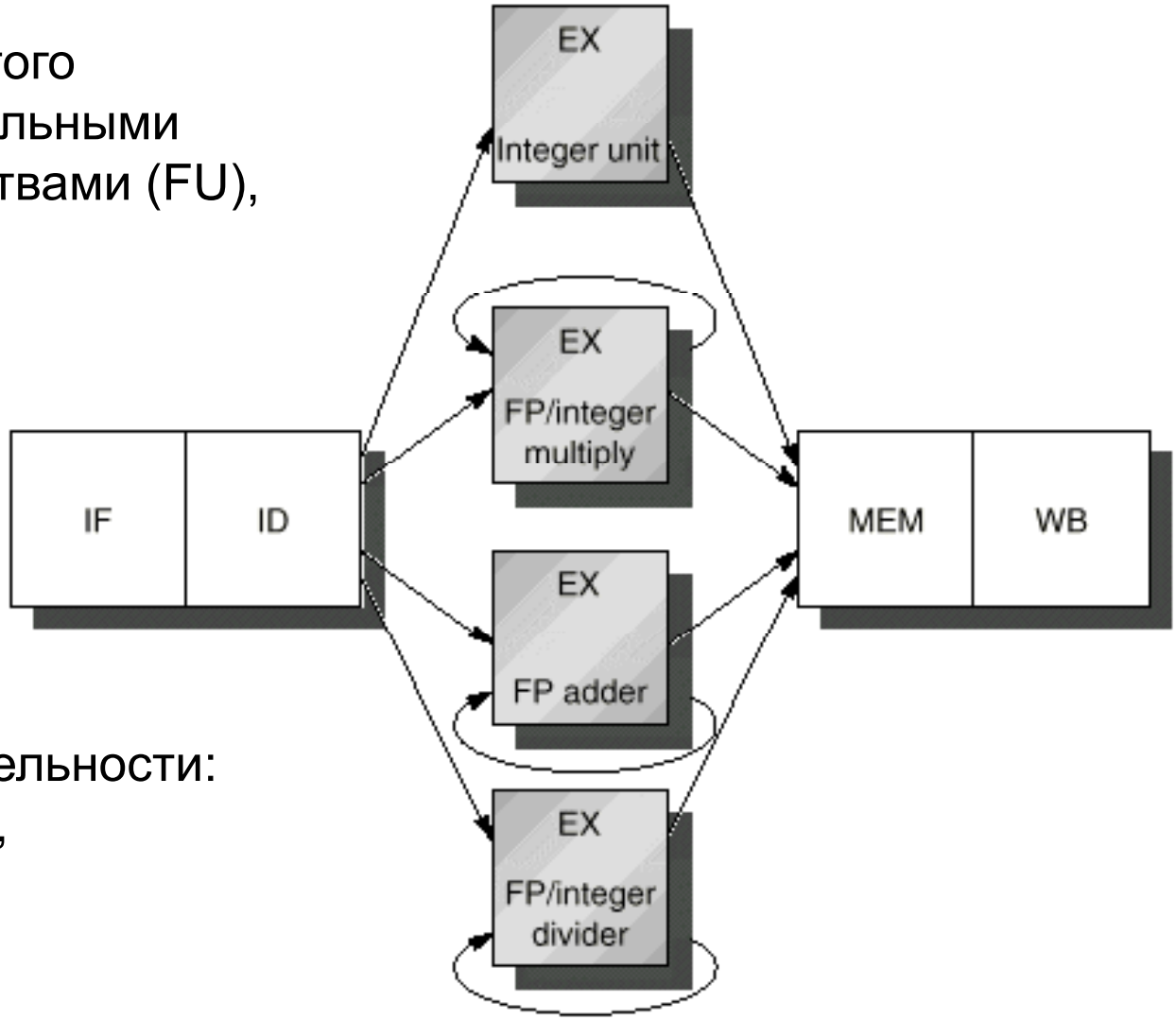
- Суперскалярные процессоры
- Неупорядоченная модель обработки
- Переименование регистров
- Переупорядочивание команд
- Алгоритм Томасуло

Суперскалярные процессоры

Расширение пятиступенчатого конвейера MIPS дополнительными функциональными устройствами (FU),

специализация FU

Цель: CPI < 1

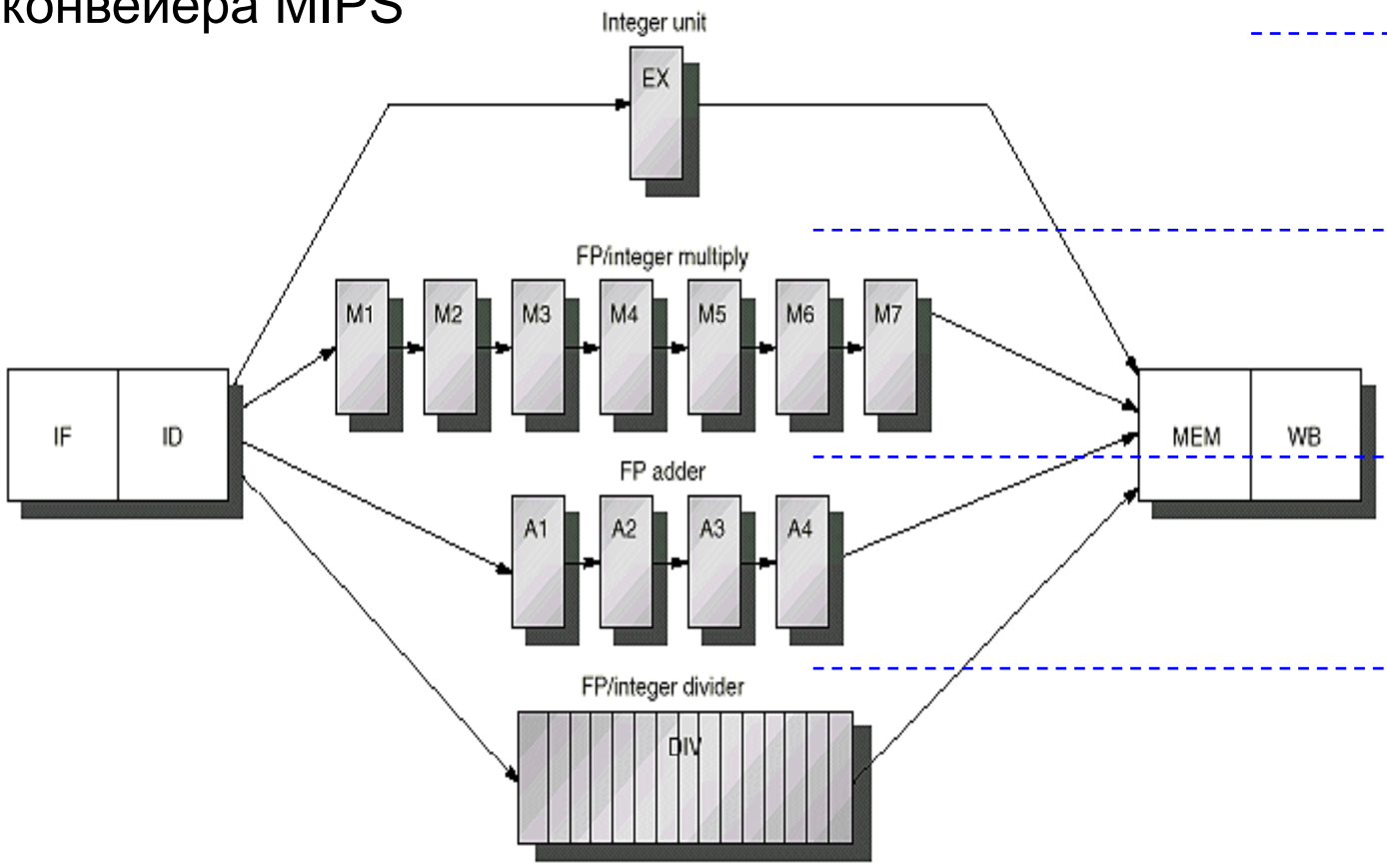


Новая метрика производительности:
Instruction Per Clock (IPC),

IPC = 1 / CPI

Суперскалярные процессоры

Расширение пятиступенчатого конвейера MIPS

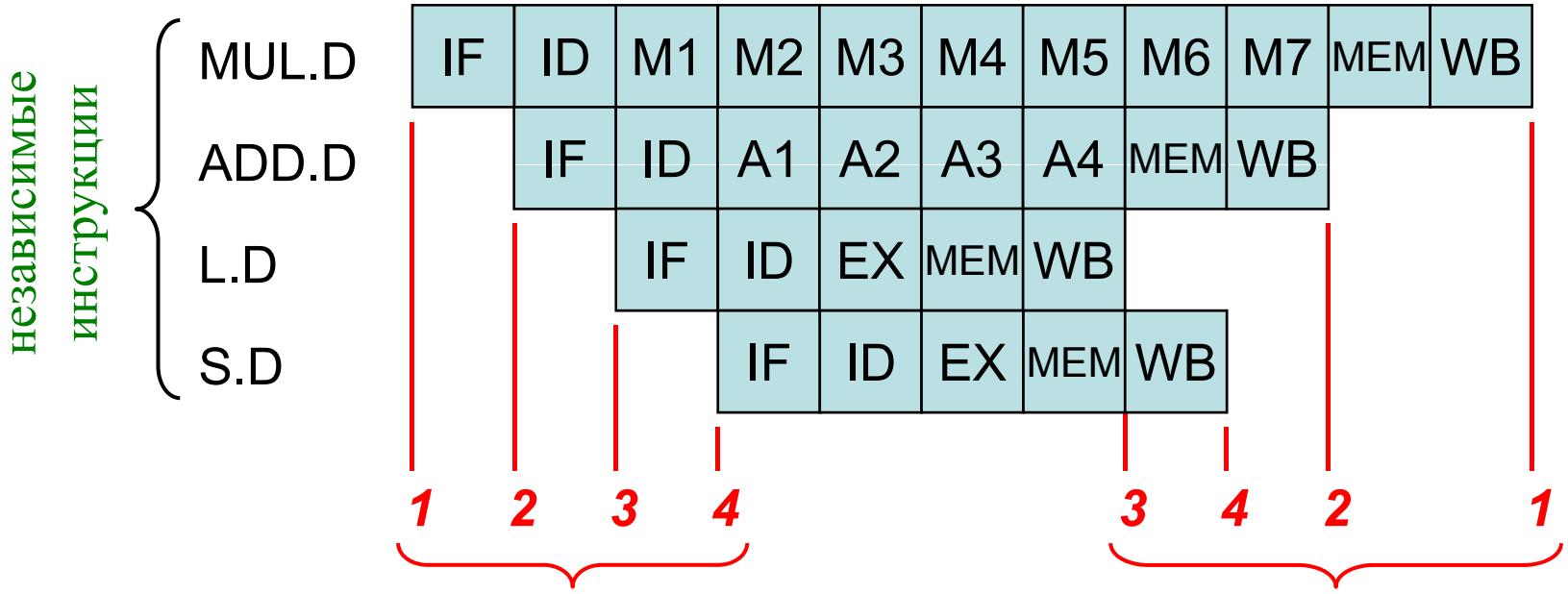


	Задержка	Время выдачи
Integer unit	1	1
FP/integer multiply	7	1
	5	2 в Pentium III
FP adder	4	1
FP/integer divider	25	25

равны для неконвейеризированных FU

Суперскалярные процессоры

- Упорядоченная выдача команд (in-order issue) неэффективна, так как требуемое FU может оказаться занятым.
- Упорядоченное завершение команд (in-order completion) неэффективно, так как остановка продвижения в одном FU приведёт к простою всех FU.



возникновение конфликтов WAW

Неупорядоченная модель обработки

Неупорядоченные выдача и завершение – неупорядоченная модель обработки (out-of-order execution) – дополнительный потенциал повышения производительности суперскалярного процессора.

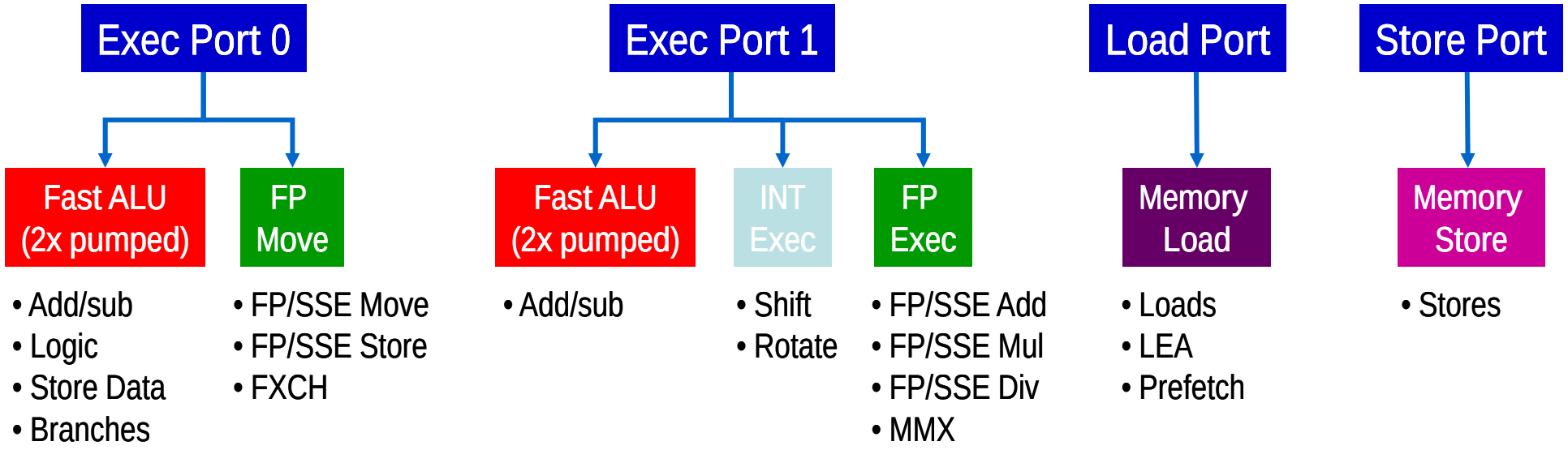
Современные суперскалярные CPU исполняют **от 2 до 10 инструкций** за такт и используют аппаратную логику анализа ILP перед выдачей команд. Такой аппаратный механизм переупорядочивания исполнения инструкций (out-of-order engine) называется **динамическим планированием**.

Компилятор и динамический планировщик не могут обойти все конфликты (структурные, по данным, по управлению) и задержки доступа к памяти (при кеш-промахах). Таким образом, фактическое число выданных в такте инструкций колеблется от нуля до максимально возможного (загрузка FU колеблется от 0% до 100%).

Пример суперскалярного процессора

Intel Pentium 4

- 4 порта выдачи
- Максимальная выдача: 6 микроопераций за такт



Загрузка суперскалярного процессора

Для обеспечения 100% загрузки всех FU суперскалярного процессора необходима **длинная последовательность независимых инструкций с удачным соотношением типов инструкций**.
Длина такой последовательности должна быть не менее глубины конвейера умноженной на количество FU. Однако типичный размер ББИ около семи инструкций.

Повышение степени ILP для суперскалярного процессора осуществляется

- ✓ оптимизирующим компилятором в статическом режиме (см. раздел “Статическая конвейеризация”),
- ✓ переупорядочиванием инструкций аппаратным планировщиком в динамическом режиме.

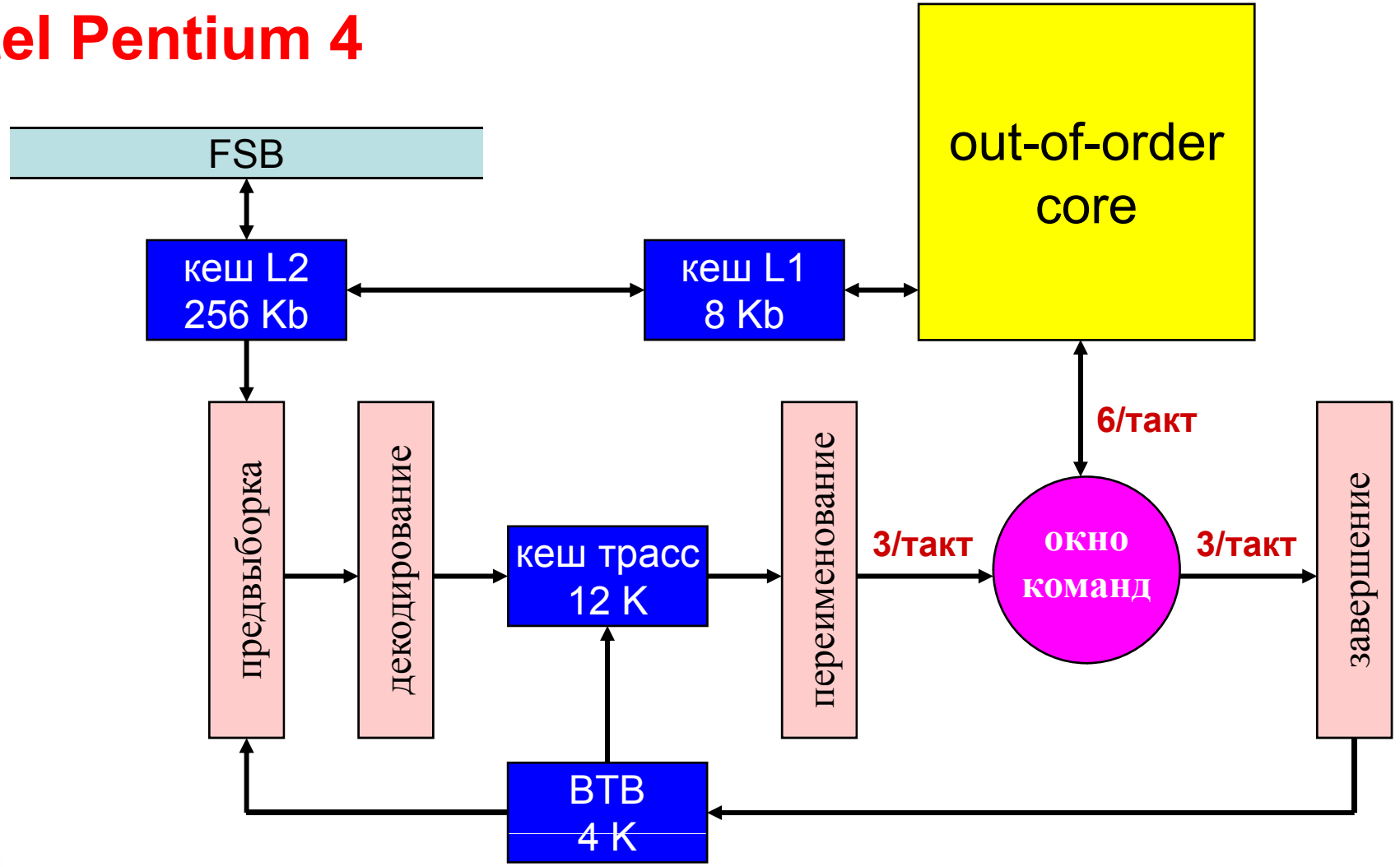
Динамическое переупорядочивание позволяет обрабатывать случаи, когда зависимости между инструкциями неизвестны во время компиляции (например, возможное перекрытие двух указателей).

Сравнение методов конвейеризации

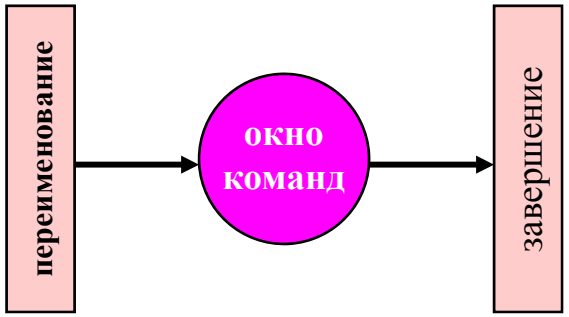
	статическая конвейеризация	динамическая конвейеризация
	оптимизирующий компилятор или программист	ядро CPU с неупорядоченной моделью обработки
имеющиеся ресурсы	большая память, много времени	минимальны
состояние вычислительного процесса	не определено	определено

Пример суперскалярного процессора

Intel Pentium 4

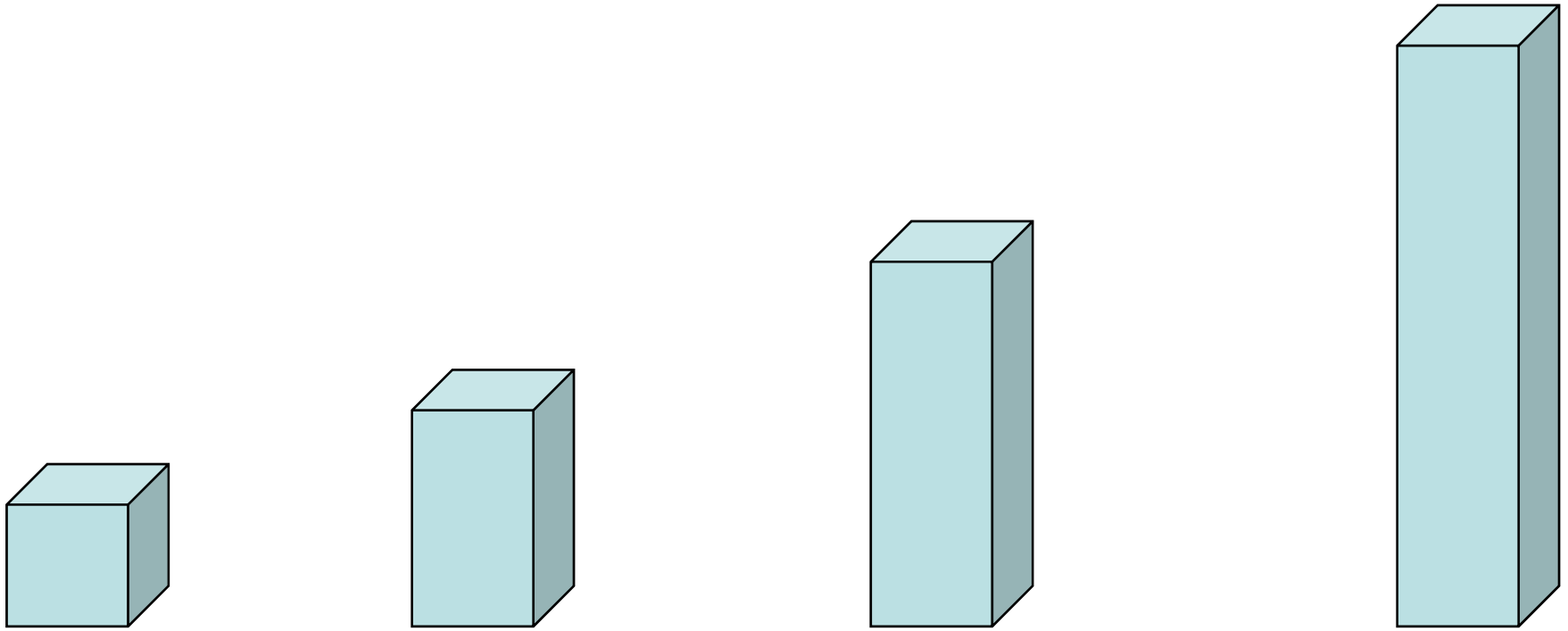


Поддержка неупорядоченного исполнения



- Блок предобработки микрокода (in-order front-end)
  - построение графа зависимостей инструкций
  - переход от логических регистров к физическим (теневым) регистрам
- Окно команд (shelving)
  - максимальное разъединение ступеней декодирования и исполнения при помощи кольцевого буфера памяти
  - выдача команд на исполнение определяется только готовностью операндов и свободностью соответствующего FU
- Блок упорядоченного завершения (in-order retirement unit)
  - восстановление исходного порядка среди выполненных команд
  - фиксирующая запись результатов исполнения

Переименование регистров



скалярный процессор

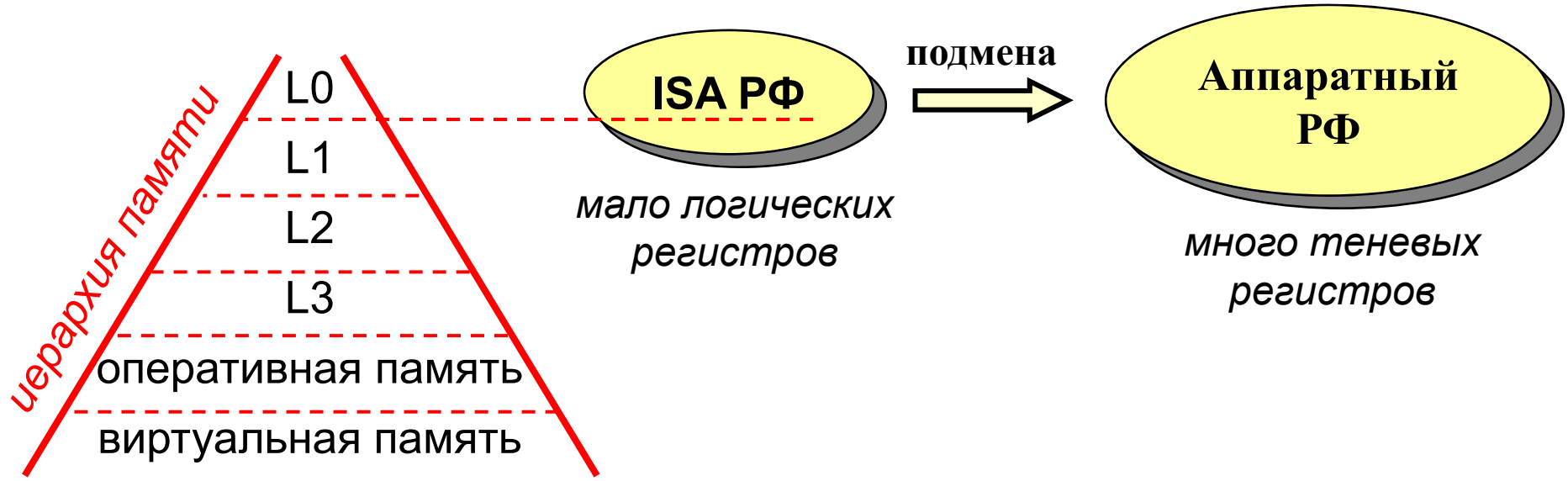
суперскалярный процессор

суперскалярный процессор с неупорядоченным исполнением

суперскалярный процессор с неупорядоченным исполнением и переименованием

Переименование регистров

Проблема: при неупорядоченном исполнении в регистрах могут быть “устаревшие” или “опережающие” значения (конфликты WAW/WAR).



Компилятор интенсивно использует логические регистры (имена), размещая в них наиболее часто используемые переменные. Небольшой объём логического регистрового файла обуславливает постоянное вытеснение одних данных другими (reusage).

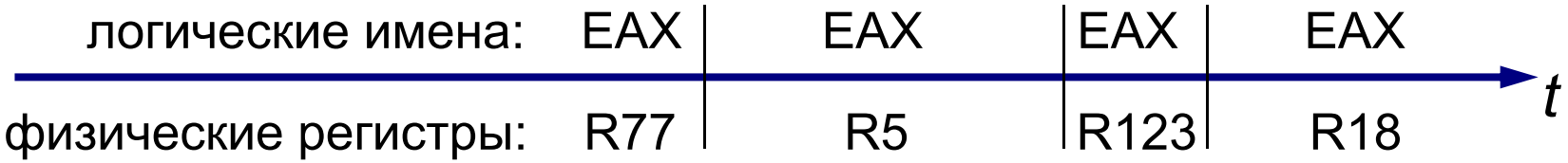
Переименование регистров

Переименование регистров – дополнительный этап декодирования:

- Каждый новый результат кладётся в свободный аппаратный регистр
- Ссылки на логический регистр в последующих командах заменяются на ссылки на выделенный аппаратный регистр

Пример

инструкции с операндом EAX перемешивать нельзя



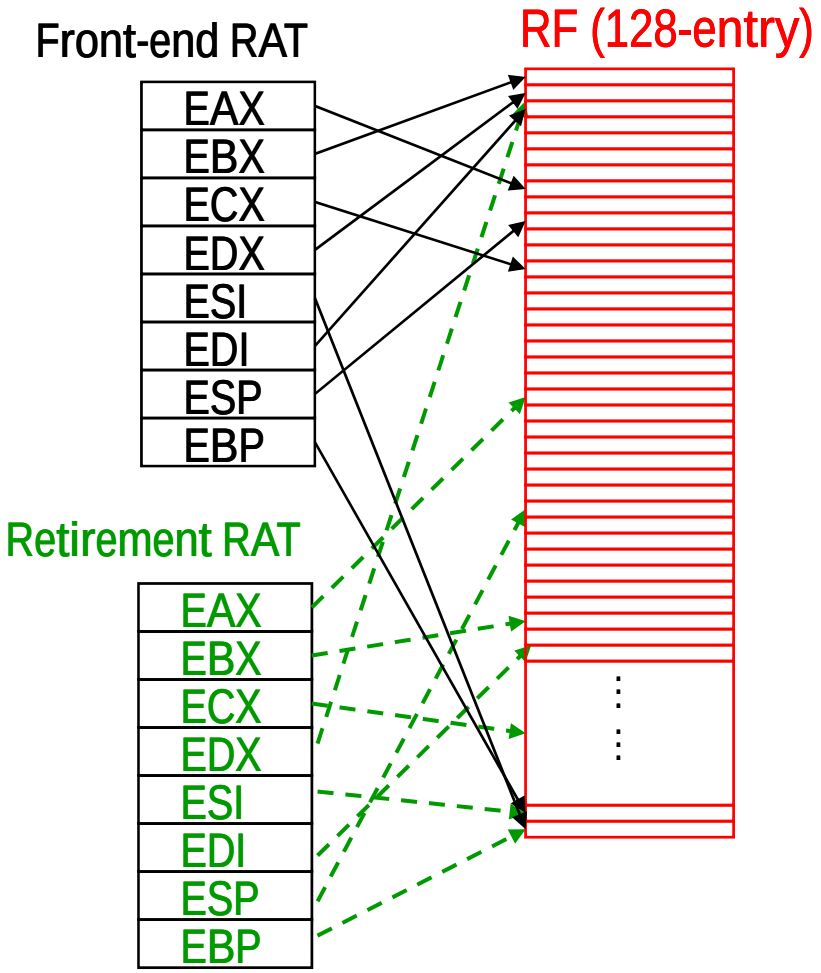
инструкции с операндами R77,R5,R123,R18 можно перемешивать без образования конфликтов WAW/WAR

Ключевая проблема реализации:
способность переименовывать регистр несколько раз за такт!

Пример суперскалярного процессора

Intel Pentium 4

Register Alias Table (RAT)

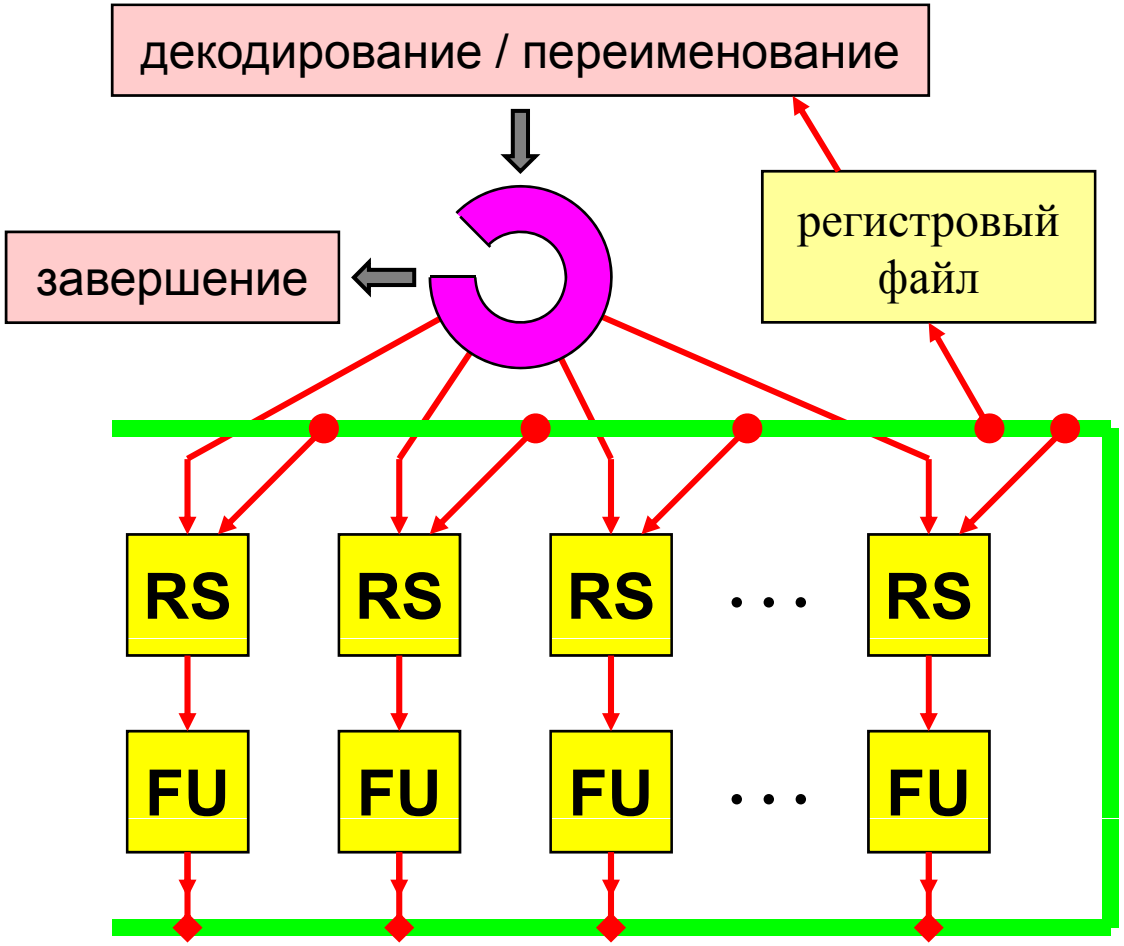


Переупорядочивание команд

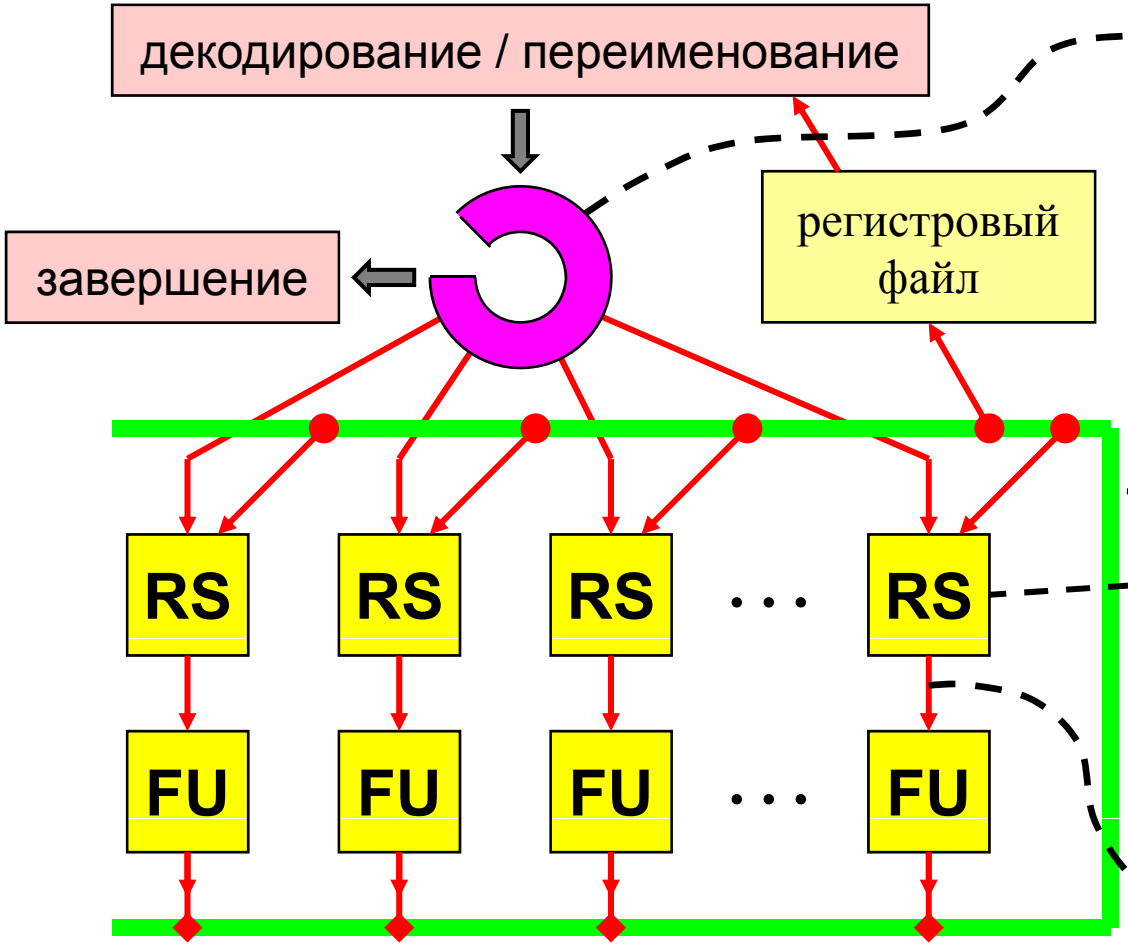
Два подхода к реализации динамического планирования:

- Централизованное окно команд в виде **табло** (scoreboard), 1963, Cray CDC 6600
  - **централизованное управление** (диспетчер – узкое место)
- Распределённое окно команд – **алгоритм Томасуло Р.Л.**, 1966, IBM 360/91
  - **распределённое управление** (масштабируемость по числу FU)

Алгоритм Томасуло



Алгоритм Томасуло



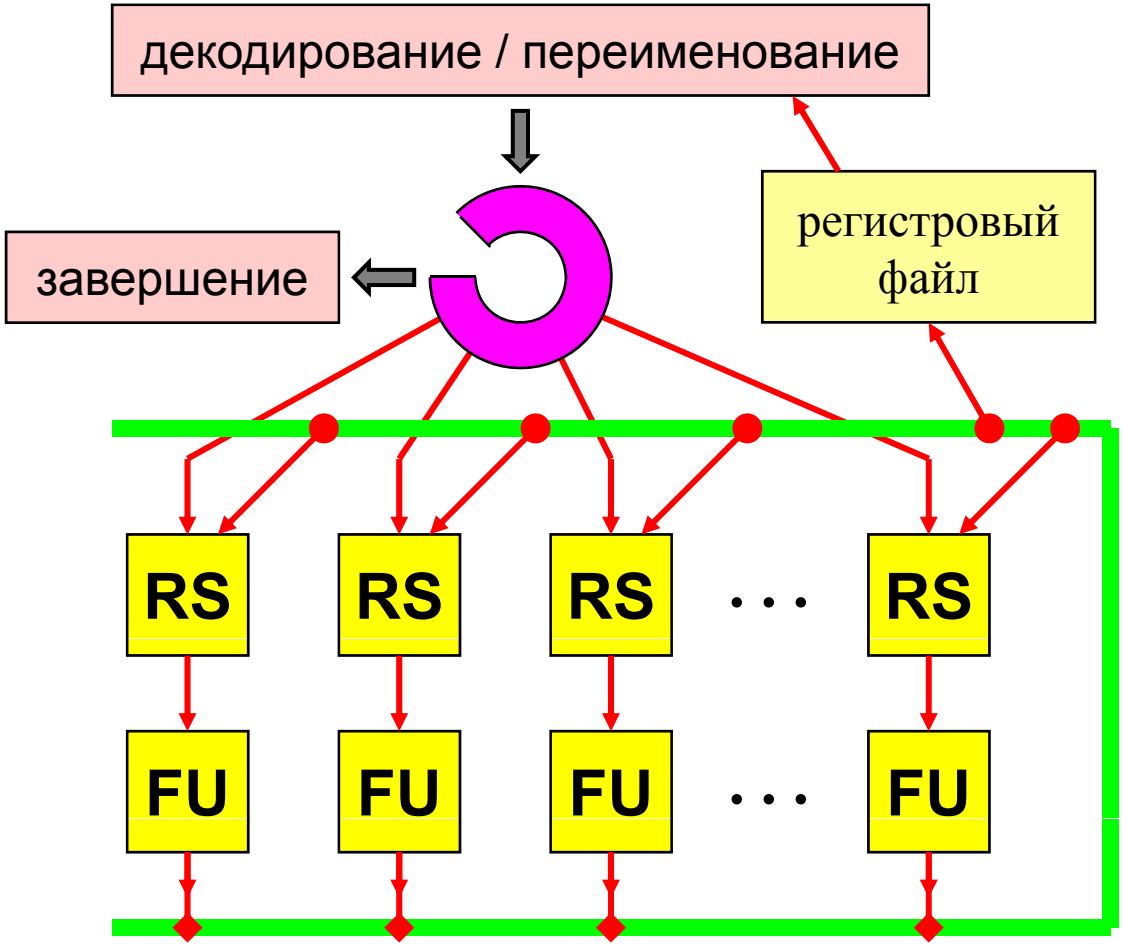
очередь команд
ROB (ReOrder Buffer)

шина широковещательной передачи данных
CDB (Common Data Bus)
масштабируемость ограничена производительностью шины

накопитель команд
RS (Reservation Station)
станция резервации нескольких команд

выдача определяется готовностью операндов и свободностью FU

Алгоритм Томасуло



управление распределено по RS

в командах регистры ISA заменены либо на значения, либо на указатели физических регистров

РФ содержит биты достоверности регистров

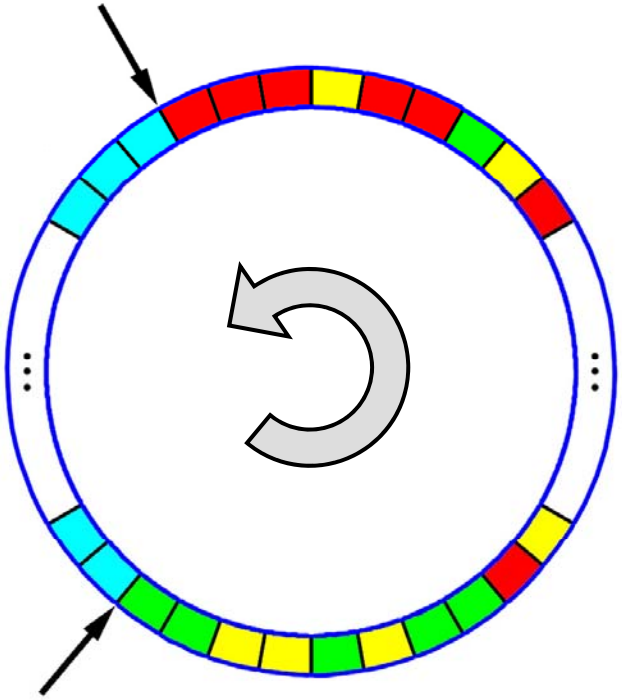
хотя RS имеют доступ к РФ, оперативный обмен между RS посредством CDB (а не через регистры)

RS выбирают допустимые команды из очереди команд в FIFO-порядке

если в RS одновременно готовы несколько команд, то используется FIFO-порядок выдачи

Восстановление последовательности

in-order указатель выдачи команд



in-order указатель завершения команд

- свободные слоты
- команда выдана
- команда исполняется
- команда завершена

удаление команды разрешено, только если она завершена и все предшествующие ей команды уже удалены из буфера восстановления последовательности (БВП)

фиксирующая запись результатов команды в память или логические регистры производится только при удалении данной команды из БВП (конфликты WAW/WAR невозможны)

при удалении из БВП команды, потребовавшей переименование логического регистра, предыдущий аппаратный регистр, ассоциированный с данным логическим регистром, высвобождается

при прерываниях сохраняются состояние логического РФ и указатель завершения команд

Smith and Pleszkun, 1988

Вопросы для обсуждения

- Какие уровни параллелизма используются при суперскалярном процессировании?
- Какой процессор (скалярный, суперскалярный, суперскалярный с неупорядоченным исполнением, суперскалярный с неупорядоченным исполнением и переименованием) наиболее (наименее) чувствителен к размеру ББИ?

Алгоритм Томасуло:

- Может ли ограниченность аппаратного РФ привести к структурному конфликту? Целесообразно ли многократное увеличение размера аппаратного РФ?
- Может ли ограниченность окна команд привести к структурному конфликту? Целесообразно ли многократное увеличение размера окна команд?
- Каково типичное состояние (полное/пустое) окна команд?
- По какой причине размер буфера RS не более 1÷5 команд?
- К чему приведёт многократное увеличение числа FU?
- Охарактеризуйте сопряжение с техникой предикатного исполнения
- Охарактеризуйте сопряжение с различными ISA (ROSC, CISC, RISC)

Следующая тема

- **Примеры дизайна многопроцессорных систем**