# RISC-архитектуры

## (Reduced Instruction Set Computer)

# Основные черты CISC-архитектур

- **В основе архитектуры CISC (Complex Instruction Set Computer) лежит эффективное использование памяти и относительная простота ассемблерного программирования (но достаточно сложная аппаратная реализация).**
- **Каждая команда CISC-процессора включает в себя несколько операций. Это уменьшает количество операторов, требующихся для реализации конкретной программы, и предоставляет для программиста удобный и быстрый набор команд. Для CISC-процессоров характерны следующие черты.**

- **Большое число машинных команд, некоторые из которых семантически нагружены аналогично операторам ЯВУ и выполняются за много тактов.**
- **Большое количество методов адресации памяти, включая специализированные способы индексации с помощью массивов.**
- **Формат команд: операция и два операнда: источник и адрес результата.**
- **Команды переменной длины, длина часто меняется в зависимости от способа адресации.**
- **Команды обмена типа Р-П, П-Р, Р-Р.**
- **Небольшое количество (8-12) РОН. Это результат наличия команд, оперирующих непосредственно с памятью и ограниченности места на микросхеме, предназначенных для целей, отличных от декодирования и исполнения команд и хранения микрокоманд.**
- **Микропрограммная реализация набора команд.**

# Проблемы и недостатки CISC-архитектур

- Усложнение набора команд и аппаратной реализации с каждым новым поколением компьютеров.
- Различным командам для выполнения требуется разное количество машинных тактов.
- Многоформатность команд.
- Многие специализированные команды используются недостаточно часто. В обычной программе используется около 20% наиболее употребительных команд.

## Возникновение RISC-архитектур

- Прорыв в схематехнике.
- Семантический разрыв. Эта проблема возникла  перехода к ЯВУ: сложные операторы, характерные для ЯВУ, существенно отличаются от простых машинных операций, реализуемых в большинстве ВМ.

# Идея RISC-архитектуры

**Основная идея RISC-архитектуры состояла в том, что последовательность простых команд, давая тот же результат, что и последовательность составных команд, может выполняться на более простой (и быстрой) физической архитектуре процессора при условии, что память так же не будет отставать.**

**CISC-процессоры проектировались под нужды asm-программистов, RISC-процессоры проектировались в расчете на типовой код, генерируемый компилятором.**

# Реализация RISC-архитектур

Результаты глубокого анализа показали, что основные усилия должны быть направлены на

- упрощение набора команд,
- эффективную реализацию конвейерной обработки путем тщательного планирования компилятором его загрузки

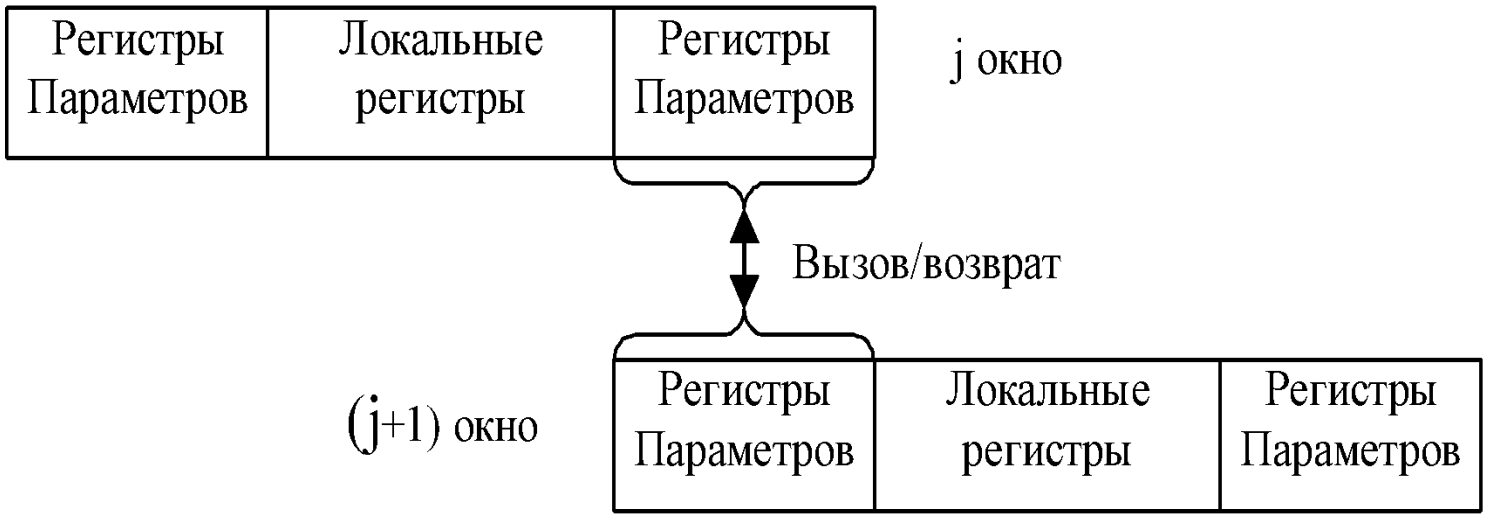
-- отделение медленной памяти от быстрой памяти (высокоскоростных регистров) (две инструкции Load/Store),

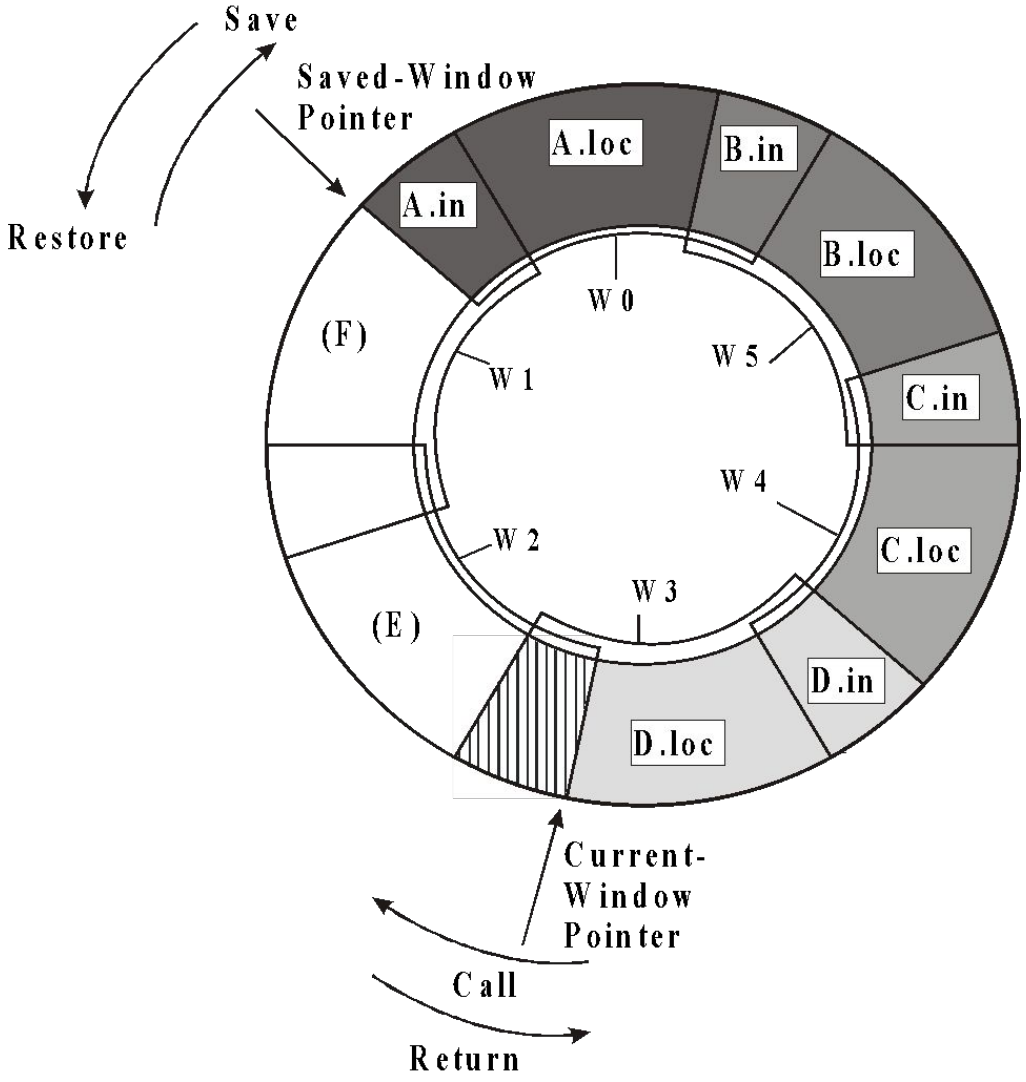
-- использование регистровых окон.

# Два способа оптимального использования регистров

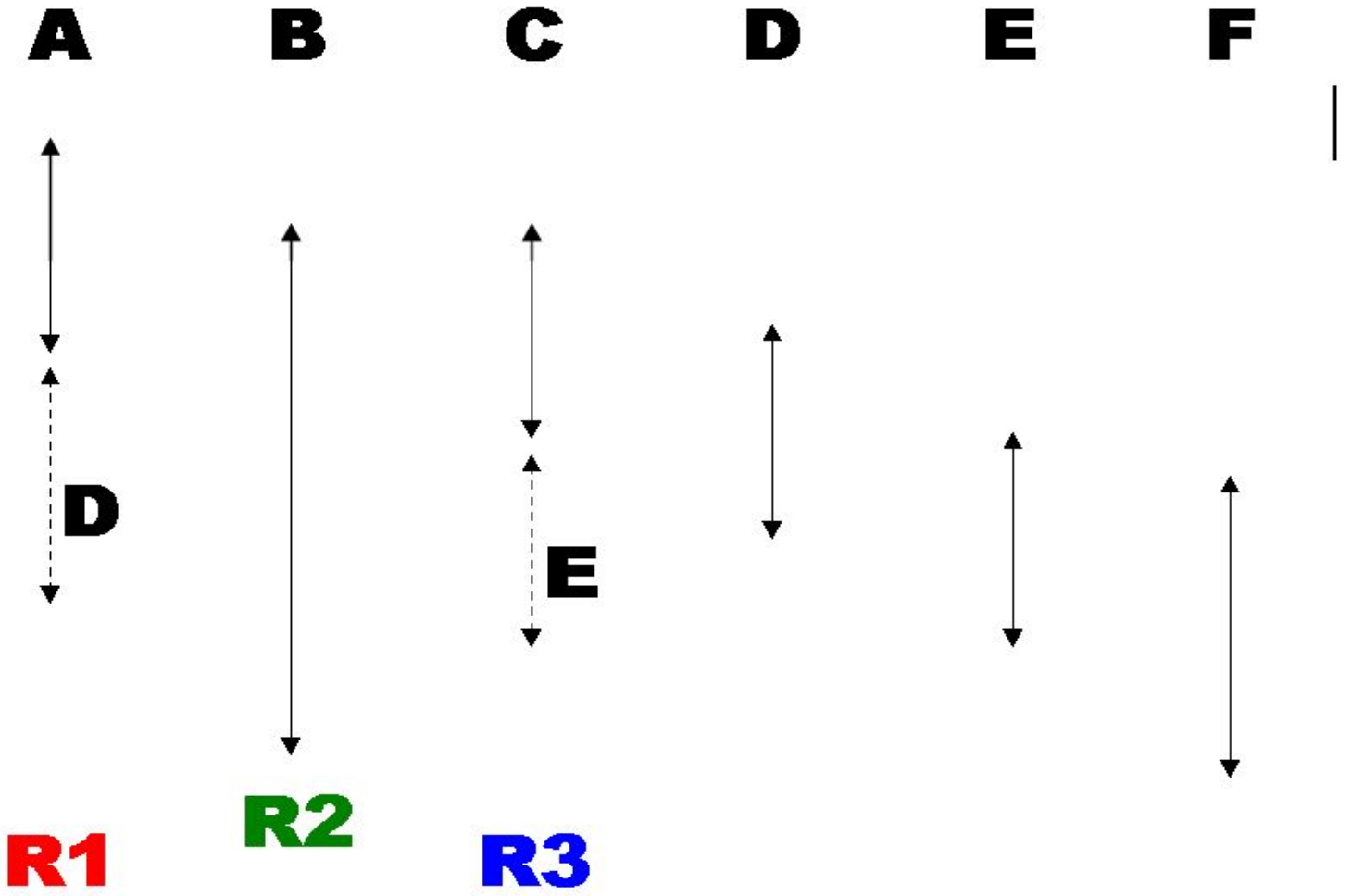
- **Программный способ.**
- **Аппаратный способ.**

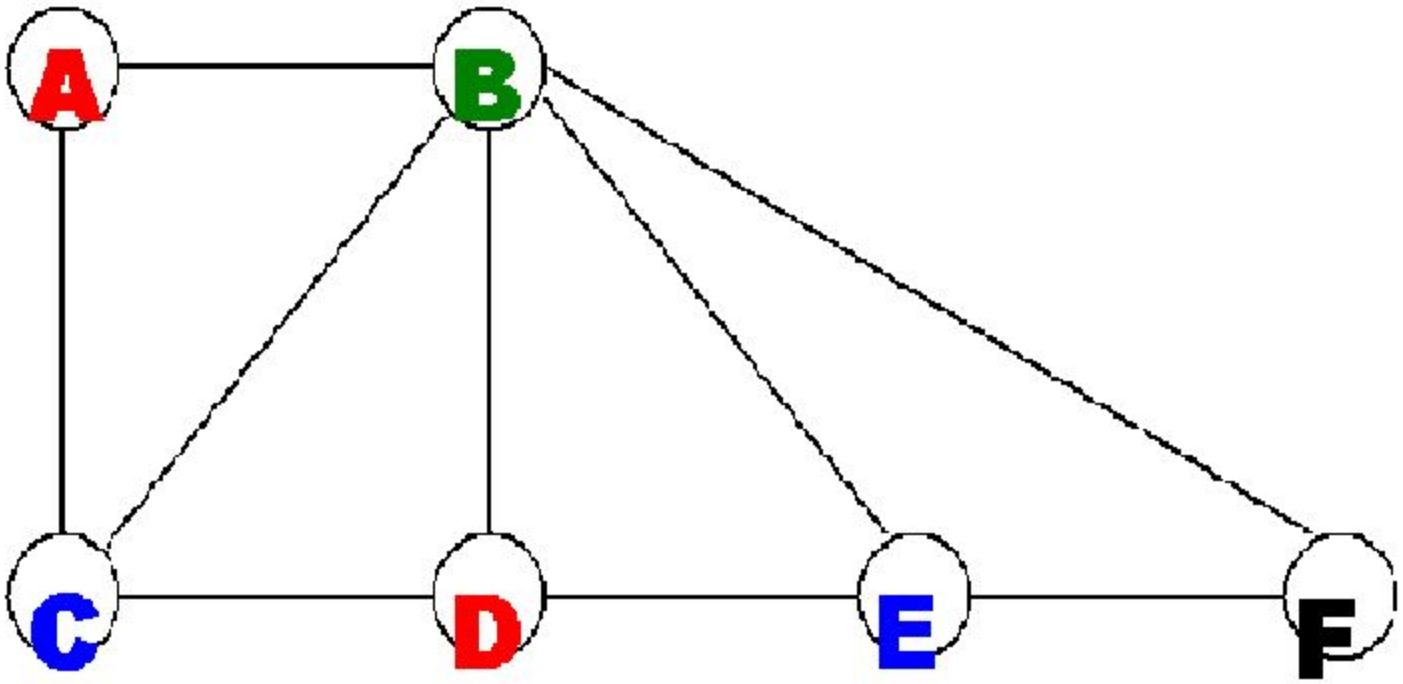
# Аппаратный способ



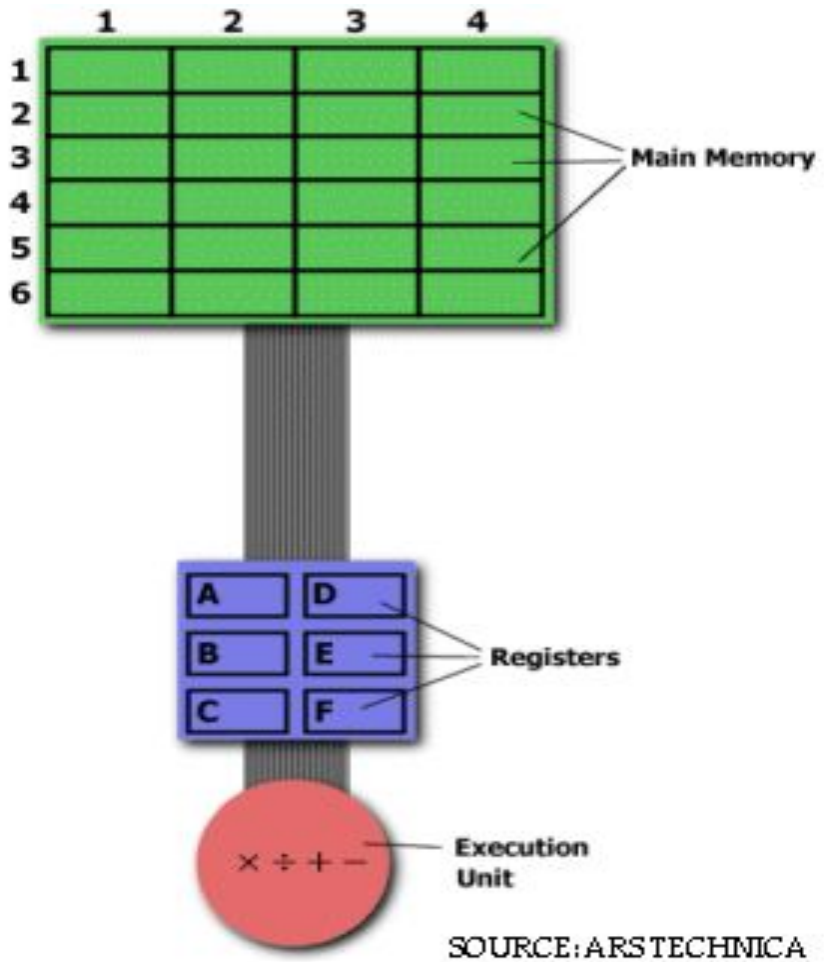

Save
Saved-Window Pointer
Restore
A.loc
B.in
A.in
B.loc
W 0
W 5
(F)
W 1
C.in
W 4
C.loc
W 2
W 3
(E)
D.in
D.loc
Current-Window Pointer
Call
Return

# Программный способ




A
B
C
D
E
F

# Пример. Умножение двух чисел.



# Две реализации умножения двух чисел

- **The CISC Approach**

**MULT 2:3, 5:2**

- **The RISC Approach**

**LOAD A, 2:3**
**LOAD B, 5:2**
**PROD A, B**
**STORE 2:3, A**

# Сравнительная характеристика **RISC**-компьютеров

| | Alpha | MIPS I | PA-RISC 1.1 | PowerPC | SPARC v.8 |
|---|---|---|---|---|---|
| Date announced | 1992 | 1986 | 1986 | 1993 | 1987 |
| Instruction size (bits) | 32 | 32 | 32 | 32 | 32 |
| Address space (size, model) | 64 bits, flat | 32 bits, flat | 48 bits, segmented | 32 bits, flat | 32 bits, flat |
| Data alignment | Aligned | Aligned | Aligned | Unaligned | Aligned |
| Data addressing modes | 1 | 1 | 5 | 4 | 2 |
| Protection | Page | Page | Page | Page | Page |
| Minimum page size | 8 KB | 4 KB | 4 KB | 4 KB | 8 KB |
| I/O | Memory mapped | Memory mapped | Memory mapped | Memory mapped | Memory mapped |
| Integer registers (number, model, size) | 31 GPR × 64 bits | 31 GPR × 32 bits | 31 GPR × 32 bits | 32 GPR × 32 bits | 31 GPR × 32 bits |
| Separate floating-point registers | 31 × 32 or 31 × 64 bits | 16 × 32 or 16 × 64 bits | 56 × 32 or 28 × 64 bits | 32 × 32 or 32 × 64 bits | 32 × 32 or 32 × 64 bits |
| Floating-point format | IEEE 754 single, double | IEEE 754 single, double | IEEE 754 single, double | IEEE 754 single, double | IEEE 754 single, double |

# Характеристики RISC-архитектур

- **Время выполнения большинства команд составляет один машинный такт.**
- **Простые команды (аппаратная реализация).**
- **Все команды имеют одинаковую длину так, что каждая команда может быть исполнена за один такт применения микропрограмм.**
- **Набор команд обычно включает в среднем 128 наиболее часто используемых команд.**
- **Небольшое число способов адресации (4). Операнды располагаются в регистрах. Стек памяти редко используется для передачи параметров процедур, вместо этого используются регистры.**
- **Формат чаще трехадресный: команда берет операнды из двух регистров и помещает результат в третий регистр.**
- **Доступ к памяти только посредством команд Чтение и Запись.**
- **Все операции, кроме Чтения и Записи, имеют тип Р-Р.**
- **Устройство управления с жесткой логикой.**
- **Большой файл регистров общего назначения.**

# Сравнение архитектур

- **CISC**
  - **Emphasis on hardware**
  - **Includes multi-clock complex instructions**
  - **Memory-to-memory: "LOAD" and "STORE" incorporated in instructions**
  - **Small code sizes, high cycles per second**
  - **Transistors used for storing complex instructions**

- **RISC**
  - **Emphasis on software**
  - **Single-clock reduced instruction only**
  - **Register to register: "LOAD" and "STORE" are independent instructions**
  - **Low cycles per second, large code sizes**
  - **Spends more transistors on memory registers**