Е.А. Еремин (Пермь)
evg_eremin@mail.ru

Материалы для **круглого стола**

«Подготовка специалистов для создания современного аппаратного обеспечения и программных экосистем»

Взгляд преподавателя педагогического вуза

Преподавание ИТ в Российской Федерации
Тверь, 16 мая 2024

Тезис 1

Обучение по спирали. В довузовском обучении полезно дать начальное представление об устройстве вычислительной техники (ВТ). Это позволит:

- сориентировать способных молодых людей в области, связанные с разработкой и производством ВТ;
- облегчить студентам усвоение материала специализированных технических курсов;
- получить фундамент для деятельности как в области hardware, так и software.

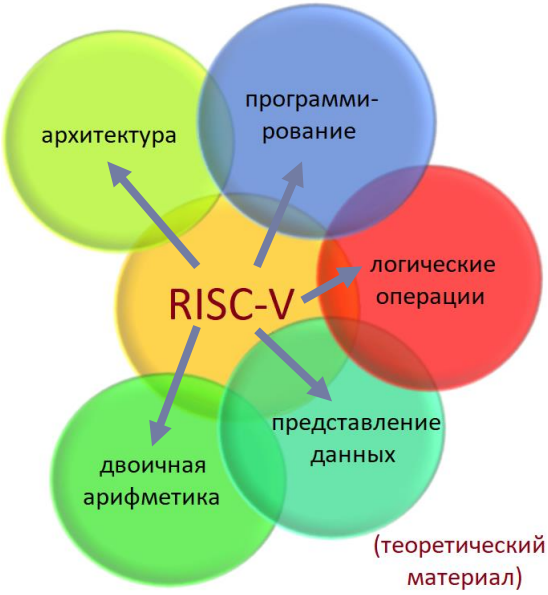
профориентация + база

Практическое предложение 1

Для предварительной подготовки и ориентации абитуриентов на обсуждаемые специальности нужны **популярные образовательные материалы по RISC-V на русском языке**.

Возможно, есть необходимость в **периодическом издании**, где будет соответствующий раздел.

Тезис 2



Систематичность и межпредметные связи. Компьютер – комплексная «многослойная» система (Д. Харрис, С. Харрис – с позиций hardware, Э. Таненбаум – с позиций software). Изучение объекта с разных точек зрения дает глубину и систематичность знаний. Еще обеспечивается повторение и закрепление материала.

«Областью, в которой "встречаются" программист и конструктор компьютеров, является набор машинных команд».

В. Столлингс

Практическое предложение 2

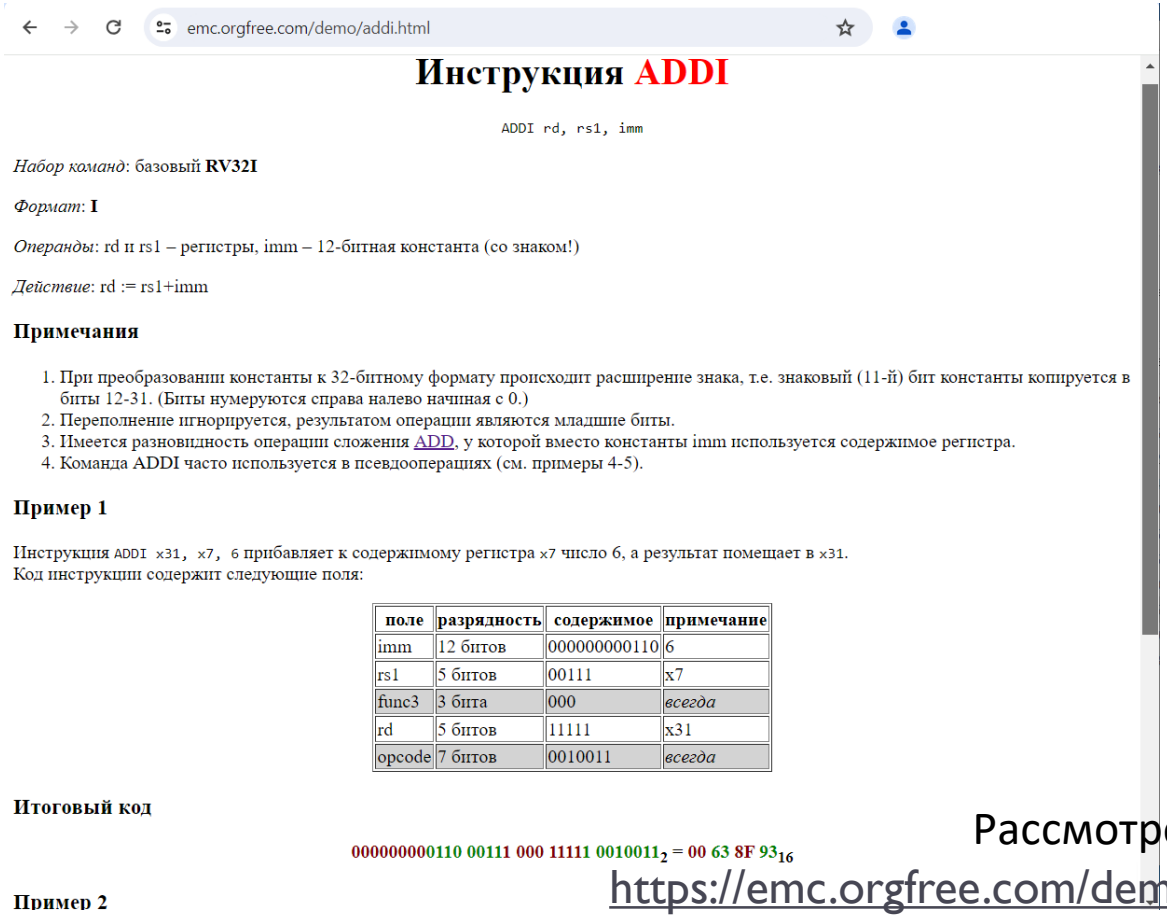
Система команд RISC-V может быть положена в основу **нескольких курсов** (архитектура, программирование и др.). На ее основе также можно глубже изучить **теоретические вопросы** (например, о двоичном представлении чисел, о ***расширении знака***, переполнении и т.д.).

Для этого полезно иметь соответствующие методические материалы.

А теперь несколько слов на основе опыта самостоятельного изучения RISC-V о том, где он, по-моему, может быть ***полезен*** сообществу.

Справка по инструкциям

Предлагаю разместить справочные материалы по RISC-V на сайте (возможно, принадлежащем Альянсу). Я вижу это так.



emc.orgfree.com/demo/addi.html

Инструкция ADDI

```
ADDI rd, rs1, imm
```

Набор команд: базовый **RV32I**

Формат: **I**

Операнды: rd и rs1 – регистры, imm – 12-битная константа (со знаком!)

Действие: rd := rs1+imm

Примечания

1. При преобразовании константы к 32-битному формату происходит расширение знака, т.е. знаковый (11-й) бит константы копируется в биты 12-31. (Биты нумеруются справа налево начиная с 0.)
2. Переполнение игнорируется, результатом операции являются младшие биты.
3. Имеется разновидность операции сложения ADD, у которой вместо константы imm используется содержимое регистра.
4. Команда ADDI часто используется в псевдооперациях (см. примеры 4-5).

Пример 1

Инструкция `ADDI x31, x7, 6` прибавляет к содержимому регистра `x7` число 6, а результат помещает в `x31`.
Код инструкции содержит следующие поля:

поле	разрядность	содержимое	примечание
imm	12 битов	000000000110	6
rs1	5 битов	00111	x7
func3	3 бита	000	*всегда*
rd	5 битов	11111	x31
opcode	7 битов	0010011	*всегда*

Итоговый код

$000000000110\ 00111\ 000\ 11111\ 0010011_2 = 00\ 63\ 8F\ 93_{16}$

Пример 2

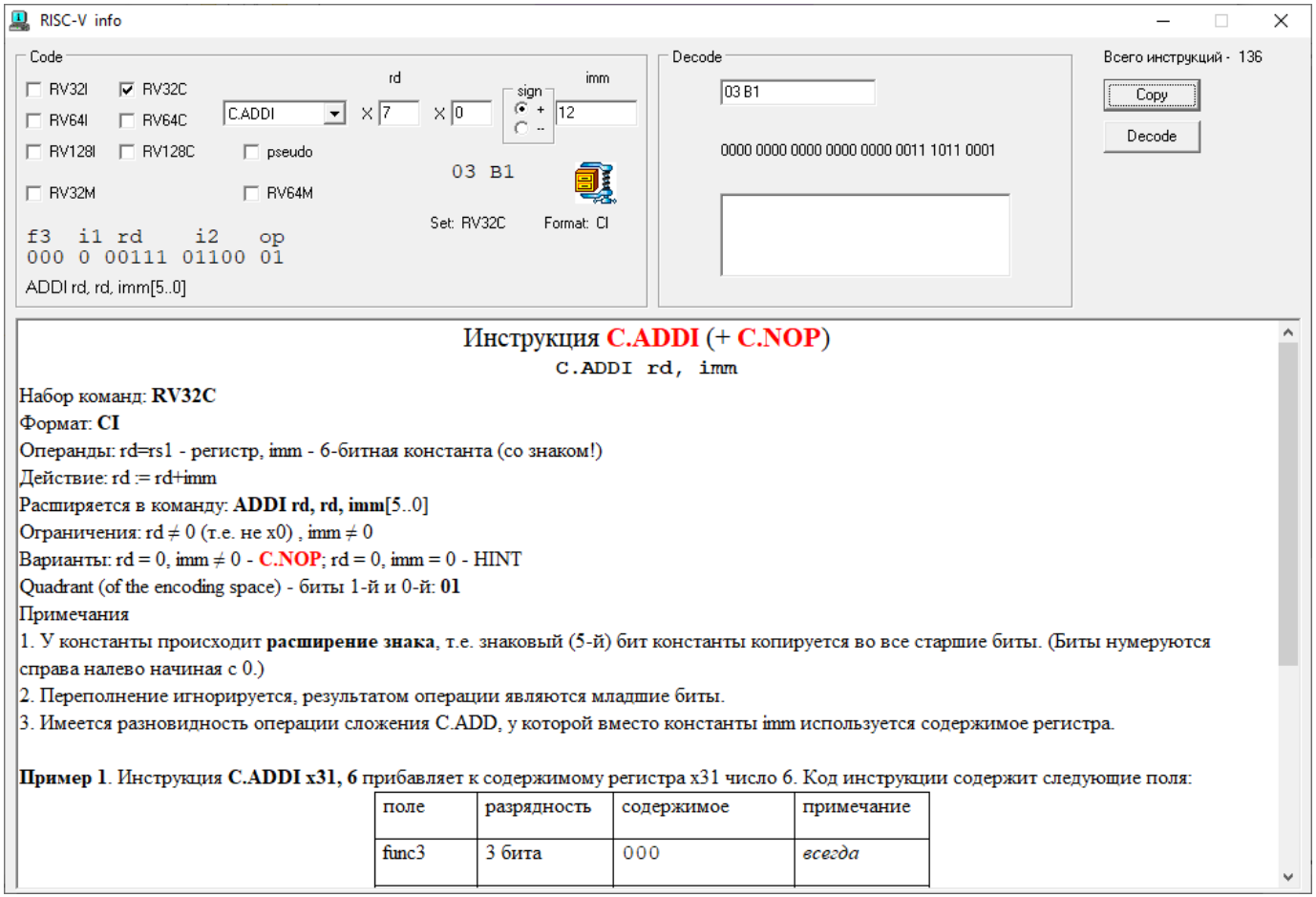
Рассмотреть подробнее – https://emc.orgfree.com/demo/apkit24.html

Как можно расширить возможности справочной системы?

Справочное приложение (в работе)

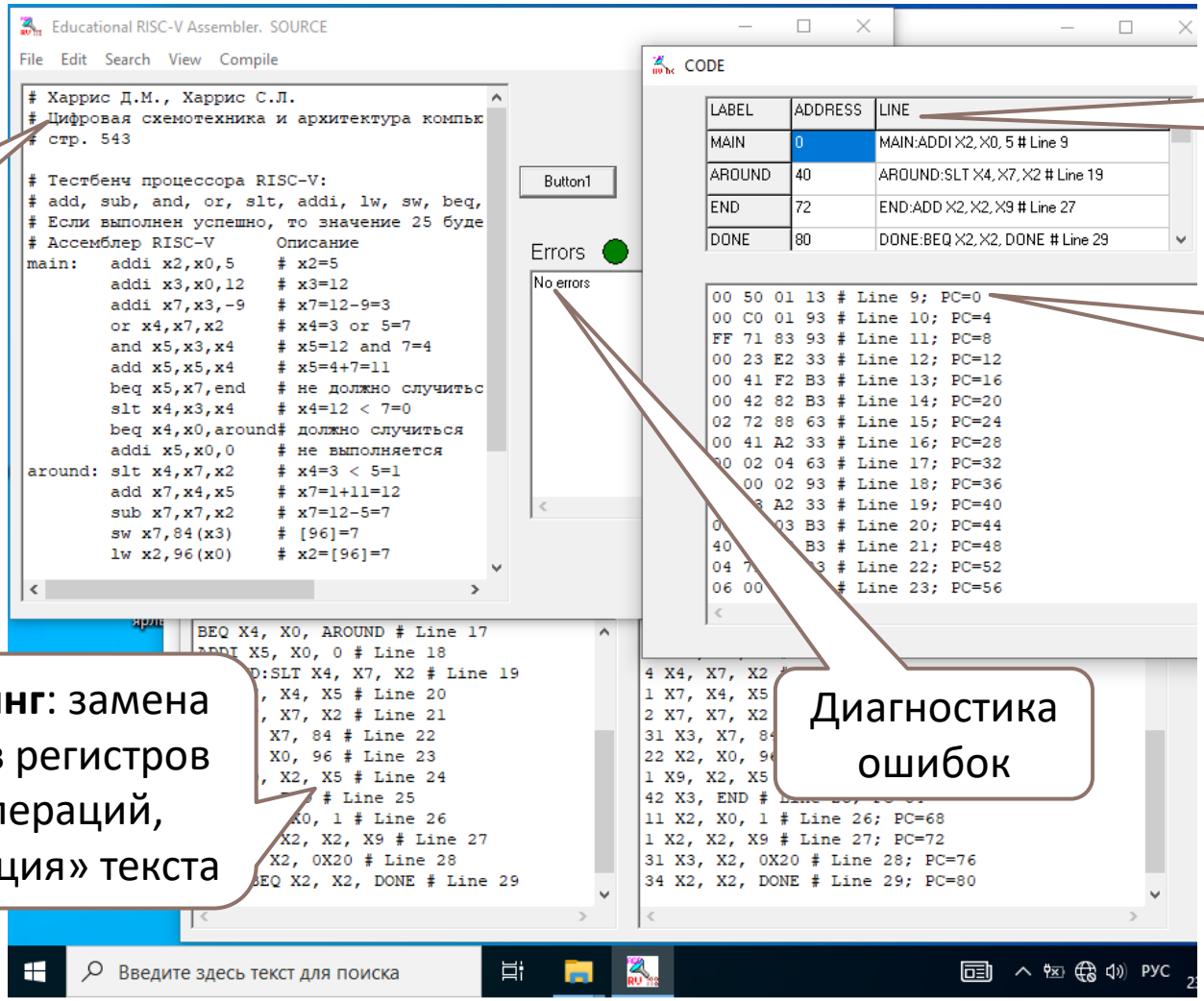
Помимо справки может закодировать набранную инструкцию.

(В перспективе можно будет и наоборот – по коду восстановить инструкцию).



Если научились кодировать ***одну*** команду, то следующий шаг – взять ***несколько*** команд.

Получаем бонус: учебный ассемблер (в работе)



Программа: тест из книги Харрисов

Таблица меток

Код RISC-V

Препроцессинг: замена псевдонимов регистров и псевдоопераций, «стандартизация» текста

Диагностика ошибок

Пользуясь случаем, обращаюсь к присутствующим здесь специалистам за подсказкой.

ОЖИ1: о соответствии кода и ISA

(ОЖИ = Очень Желаемая Информация)

RISC-V позволяет произвольно расширять ISA. Очевидно, что компилятор можно «попросить» сгенерировать код RISC-V для любой конкретной ISA (задать базовый набор + имеющиеся расширения).

А вот как потом при загрузке узнать исходную основу кода?

Подходит ли этот код для исполнения на данном процессоре с имеющейся у него ISA?

Характерный пример

Инструкция **C.LD** читает 64-битное целое в RV64C.

Точно такой же код имеет инструкция загрузки 32-битного вещественного числа **C.FLW**, входящая в набор RV32FC (там нет 64-битных целых!).

Надо как-то сравнивать конфигурацию процессора и кода.

Для определения первой достаточно ввести инструкцию вроде **CPUID** в Intel. А вот как узнать вторую?

ОЖИ2: о взаимодействии ядер

(ОЖИ = Очень Желаемая Информация)

Каким образом взаимодействуют ядра в многоядерном процессоре? Как происходит управление ими?

Характерный пример
Как стартует система, состоящая из абсолютно одинаковых ядер?

Детали постановки проблемы можно посмотреть в тезисах прошлогодней конференции.

Мне **очень** хочется поработать с многоядерным симулятором! А вам?

(дополнение:
какие они разные…)

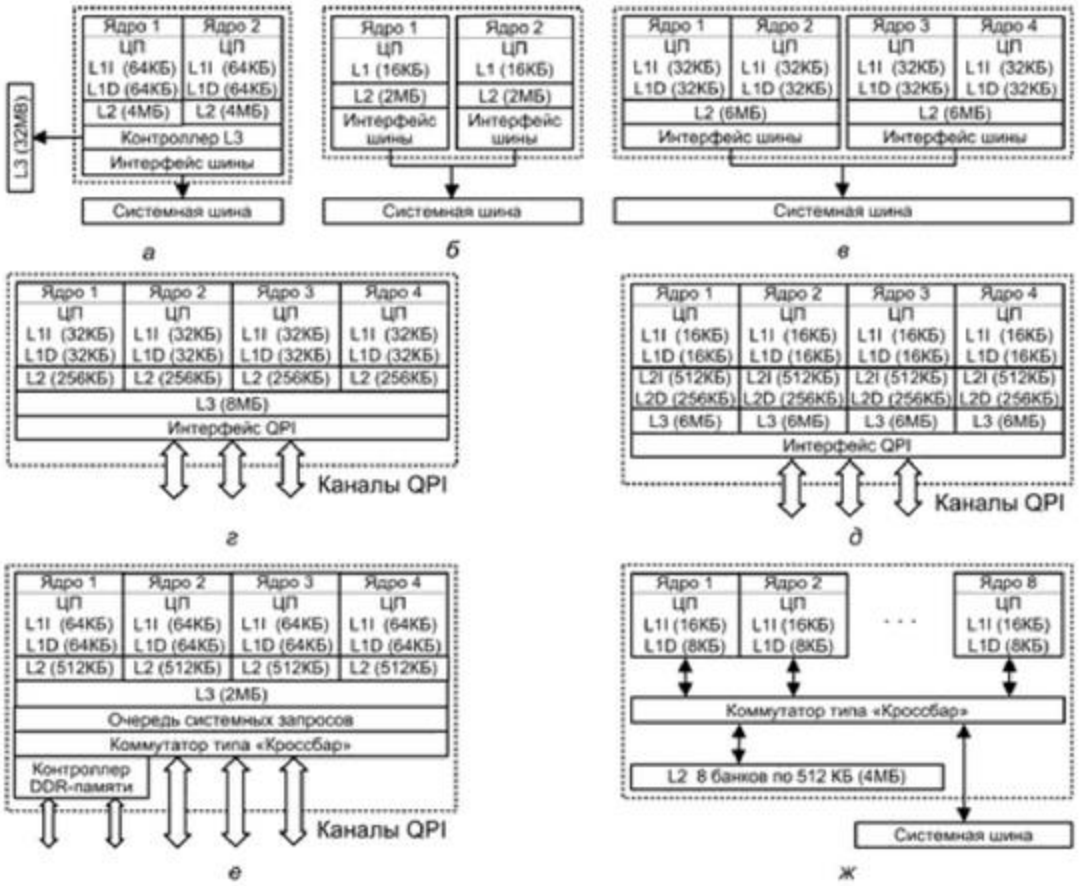


Рис. 9.42. Структура многоядерных процессоров различных производителей: *а* — IBM Power 6; *б* — Intel Pentium D; *в* — Intel Core 2 Quad; *г* — Intel Nehalem; *д* — Intel Itanium 3 Tukwila; *е* — AMD Phenom X4; *ж* — Sun UltraSPARC T2

Орлов С. А., Цилькер Б. Я.
О66 Организация ЭВМ и систем: Учебник для вузов. 2-е изд. — СПб.: Питер, 2011. — 688 с.: ил.

ISBN 978-5-49807-862-5

с. 447