ИСП РАН

Учебный курс «Архитектура ЭВМ и язык ассемблера» с точки зрения системного программиста

Кирилл Батузов, Вартан Падарян, Михаил Соловьев
{batuzovk, vartan, icee}@ispras.ru

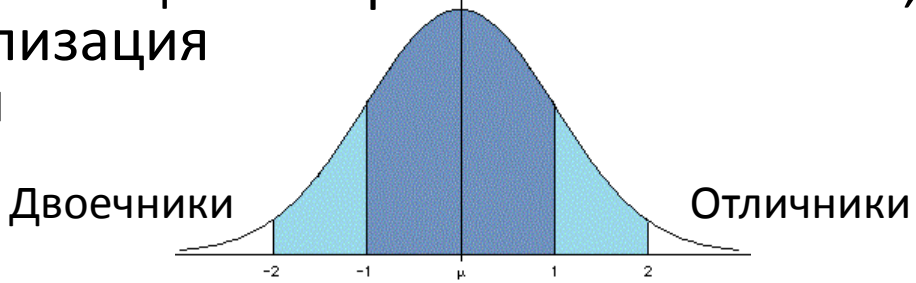
18 мая 2018

Кого пытаемся подготовить?

Цели курса

- Формирование у студентов связного представления об организации современных вычислительных систем.
- Понимание взаимосвязей между архитектурными решениями на уровнях аппаратуры ЭВМ и ПО.
- Понимание факторов, влияющих на качественные и количественные характеристики ЭВМ, производительность и безопасность всей вычислительной системы в целом.

- Научить основную часть студентов хотя бы одному языку программирования
- Первый семестр – реализация алгоритмов на языке Си, второй семестр – реализация конструкций языка Си



Проблемы (1/2)

- Лекции
  - Изучение только одной процессорной архитектуры провоцирует Синдром Утенка: студент через некоторое время не может даже представить, что компьютер может быть устроен как-то иначе.
  - Объем материала vs. Всего лишь 24 лекции
    из них 2 лекции частично заняты коллоквиумами
  - История развития ЭВМ vs. Современный уровень техники
  - Тесносвязанный материал: одновременное введение множества терминов, ссылки вперед в изложении
- Семинары
  - 1 поток, 6 групп, порядка 120-140 человек.
    Команда семинаристов, включая аспирантов – 12 человек.
  - Запуск курса потребовал подготовки методической литературы для семинаристов.

Проблемы (2/2)

- Неоднородный по подготовке состав групп
- Высокий «порог входа». Как правило, студенты не имеют предварительных знаний по теме курса. Даже «олимпиадники».
- Отношение к учебе со стороны студентов
  - Немотивированность
  - «Магическое мышление» применительно к непонятным устройствам.

Мучительный выбор архитектуры

- Архитектура IA-32 относительно сбалансирована
  - Одинаковый размер у РОН, исполнительного адреса и максимального размера непосредственно кодируемых операндов
  - Почти нет ограничений на использование в командах РОН
  - Плоская память
- На ВМК «традиционно» преподается архитектура x86
- Рассматривается подмножество набора команд IA-32
  - Подмножество команд x87, как пример стекового процессора
  - Архитектура x86-64 рассматривается в ключевых отличиях от IA-32
    - Соглашение вызова вбирает рассмотренные ранее в IA-32 компиляторные оптимизации
    - Простое построение позиционно независимого кода
    - Ограничения на адресацию памяти из-за совместимости машинной кодировки
- Архитектура RISC-V иллюстрирует идеи упрощенной ISA
  - Достоинства и недостатки лучше воспринимаются на конкретных примерах
  - Простой конвейер команд, который можно сравнивать с «модельным» конвейером x86.

Рассматриваемые темы

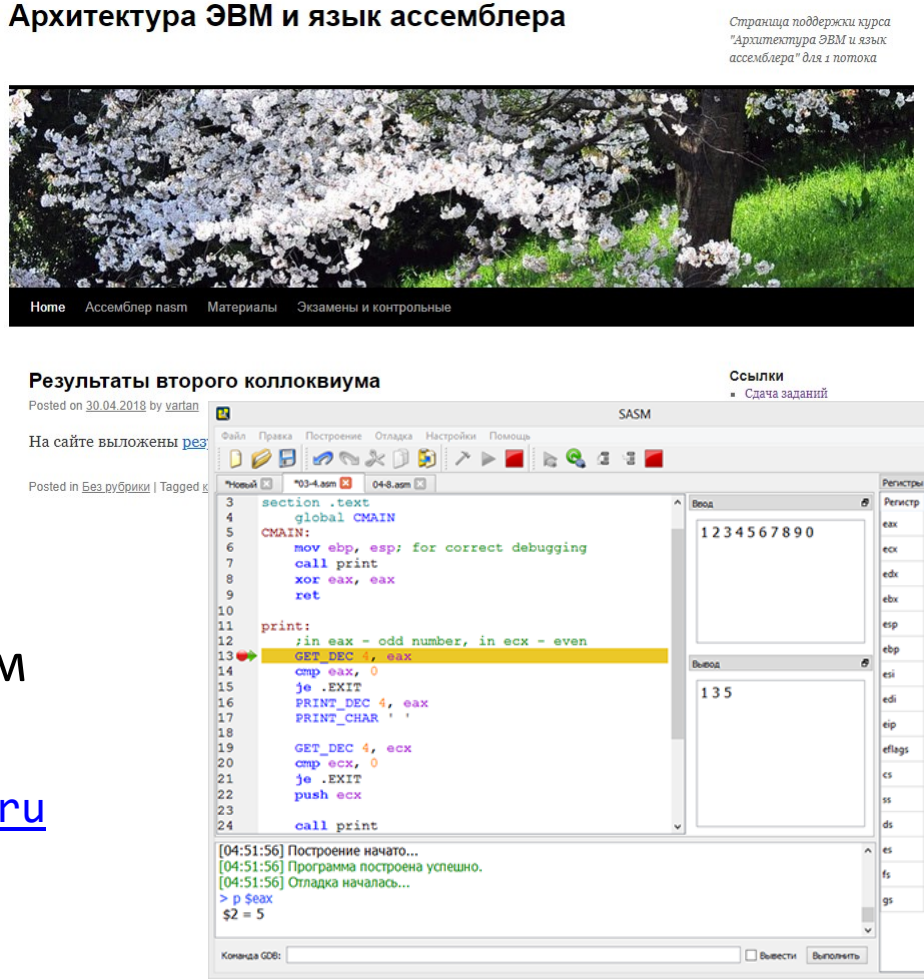
- Введение в архитектуру ЭВМ. Основные компоненты компьютера. Архитектура IA-32.
  - Сопоставление Си-кода и ассемблера помогает лучше понять, как работает машина.
- Архитектура набора команд
  - Сопоставление Си-кода и ассемблера помогает лучше понять, как работает Си-код, почему у Си-программ высокая производительность и каковы причины уязвимостей в программах.
- Двоичный интерфейс приложения
  - Почему возможно составить программу из кусков кода, написанных на разных языках
- Компоновка и загрузка
  - Как получается работающий код. Какие механизмы позволяют запускаться и работать программам в реальной Linux-системе.
- Устройство аппаратного обеспечения компьютера
  - Какими законами обусловлено развитие аппаратуры, с какими непреодолимыми преградами это развитие сталкивается
- Многозадачная работа компьютера

«Сквозные» темы

- Важные или трудные для понимания вопросы поднимаются в течении курса несколько раз
  - Конвейер команд – с первой и до последней темы
  - Выравнивание адресов
  - Предсказание переходов
  - Прерывания
    - Первое упоминание – x87, в нем пользовательский код может маскировать часть исключений
    - Работа с блочным устройством – уведомление о завершении прямого доступа в память
      (на самом деле – завершение PCI-транзакций)
    - Организация многозадачной работы, системные вызовы
- Рассматриваем одно и тоже, но с разных точек зрения

Поддержка курса

- Сайт курса http://asmcourse.cs.msu.ru
- «Автоматическая» сдача домашних заданий в ejudge http://earth.ispras.ru/
  На самом деле – предварительная фильтрация перед проверкой преподавателем
- Среда разработки SASM
- Telegram-канал asm@cs.msu.ru
  (на момент 2018.05.18 решение о блокировке Telegram еще не вступило в законную силу)
- Выборочная проверка на плагиат MOSS https://theory.stanford.edu/~aiken/moss/
- Методические пособия



Отчетность

- Лекции
  - Два коллоквиума по 5 задач на 50 минут
    экзамен 10 задач на 150 минут
    Подсчитывается процент набранных баллов
  - ***Оценка ставится за работу в течение всего семестра***
    0.2 * c1 + 0.2 * c2 + 0.6 * ex,
    где c1 и c2 – коллоквиумы, ex – экзамен
  - Меньше 40% – ***неуд***, 40-59% – ***удовл***, 60-79% – ***хор***, 80% и выше – ***отл***
- Семинары
  - Шесть домашних заданий. Первое ДЗ – сортировка, пишется на Си.
  - Четыре ДЗ, сдающихся через ejudge
  - Шестое ДЗ – многомодульная программа на Си и ассемблере, подсчет площади криволинейной трапеции.
    Есть вариант задания для «олимпиадников», требующий автоматического построения кода для сопроцессора x87.

Комплект задач в контестах – 28
24 задачи – отл
19 задач – хор
13 задач – удовл

	Простые	Средние	Сложные	Обратные
Выражения	**2**	**4**	**1**	**н/д**
Передача управления	**2**	**2**	**1**	**2**
Вызов функций	**2**	**2**	**1**	**2**
Ввод/вывод	**2**	**3**	**2**	**н/д**

Домашние задания

- Научиться программироваться можно только программируя (с) В.П. Иванников
- Студент долгосрочно запоминает только тот материал, по которому он писал работающий код
- Подготовкой контестов для ejudge занималась команда из 8 человек
  - Разработка задач, формулировка – 79
  - Авторские решения
  - Тесты – 1743
  - Настройка контестов
- Обязательная контрольная работа в машзале в течение семестра
  - Комиссия по практикуму на ЭВМ – машзал

Пример задачи средней сложности

Задача 03-9: Недостаточные числа

Ограничение времени: 1 с
Ограничение памяти: 64 M
Максимальное количество посылок: 24

Недостаточным числом называется такое натуральное число, сумма собственных делителей (то есть делителей, отличных от самого этого числа) которого меньше этого числа.

На стандартном потоке ввода задаётся единственное натуральное число $K \le 7000$. Требуется на стандартный поток вывода вывести K-е недостаточное число.

Указание: проверку числа на недостаточность вынести в отдельную функцию в соответствии с конвенцией `cdecl`.

Примеры

Входные данные	Результат работы
1	1
14	16

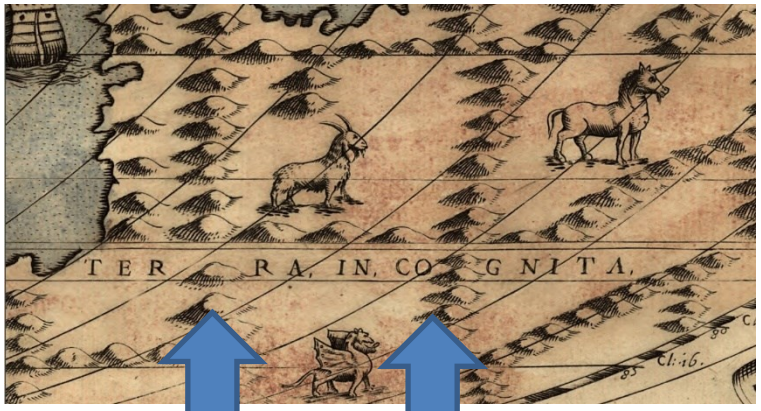
Пример обратной задачи

Дан код программы на языке Ассемблера. Вам необходимо восстановить семантику данной программы и выразить её в виде программы на языке Си.

```
%include "io.inc"

        SECTION .text

        GLOBAL CMAIN
        CMAIN:
            GET_UDEC            4, EAX
            CALL                F
            PRINT_UDEC          4, EAX
            NEWLINE
            XOR                 EAX, EAX
            RET

        F:
            CMP                 EAX, 0
            JNZ                 REC
            MOV                 EAX, 1
            RET
        REC:
            DEC                 EAX
            CALL                F
            LEA                 EAX, [EAX + 2 * EAX]
            RET
```


Известно, что входное число находится в диапазоне от 0 до 20 включительно.

Связи с другими курсами



1 поток «непрерывная» математика

2 поток дискретная математика

Конструирование компиляторов

Языки программирования

Введение в сети ЭВМ

ФОП ЭВМ

Конструирование ядра ОС

Специализация, распределение по кафедрам

2 год

1 год

Системы программирования

Операционные системы

Архитектура ЭВМ и язык ассемблера

Алгоритмы и алгоритмические языки

Что выпадает за пределы и возможности курса?

- Не рассматривается, но студенты проявляют интерес
  - Векторные команды
  - GPU, FPGA, ASIC
- Нет возможности апробировать в рамках семинарских занятий и домашних заданий
  - Системные команды
  - Процесс загрузки
- Каждый год лекционный материал обновляется ~10%