© 2015 МГУ/ВМК/СП

Лекция 1

7 февраля

© 2015 МГУ/ВМК/СП

Цель курса

- Изучение языка Си с другой точки зрения
  - Было
    - Абстрактные типы данных
    - Асимптотический анализ сложности
  - Будет
    - Изучение особенностей реализации языка Си на уровне аппаратуры
    - Область применимости различных способов реализации
- Результат
  - Понимание причин хорошей/плохой производительности программы
  - Понимание причин ошибок
  - Основа для других курсов: ОС, сети, компиляторы, параллельная обработка данных …

© 2015 МГУ/ВМК/СП

Проблема №1 Свойства чисел

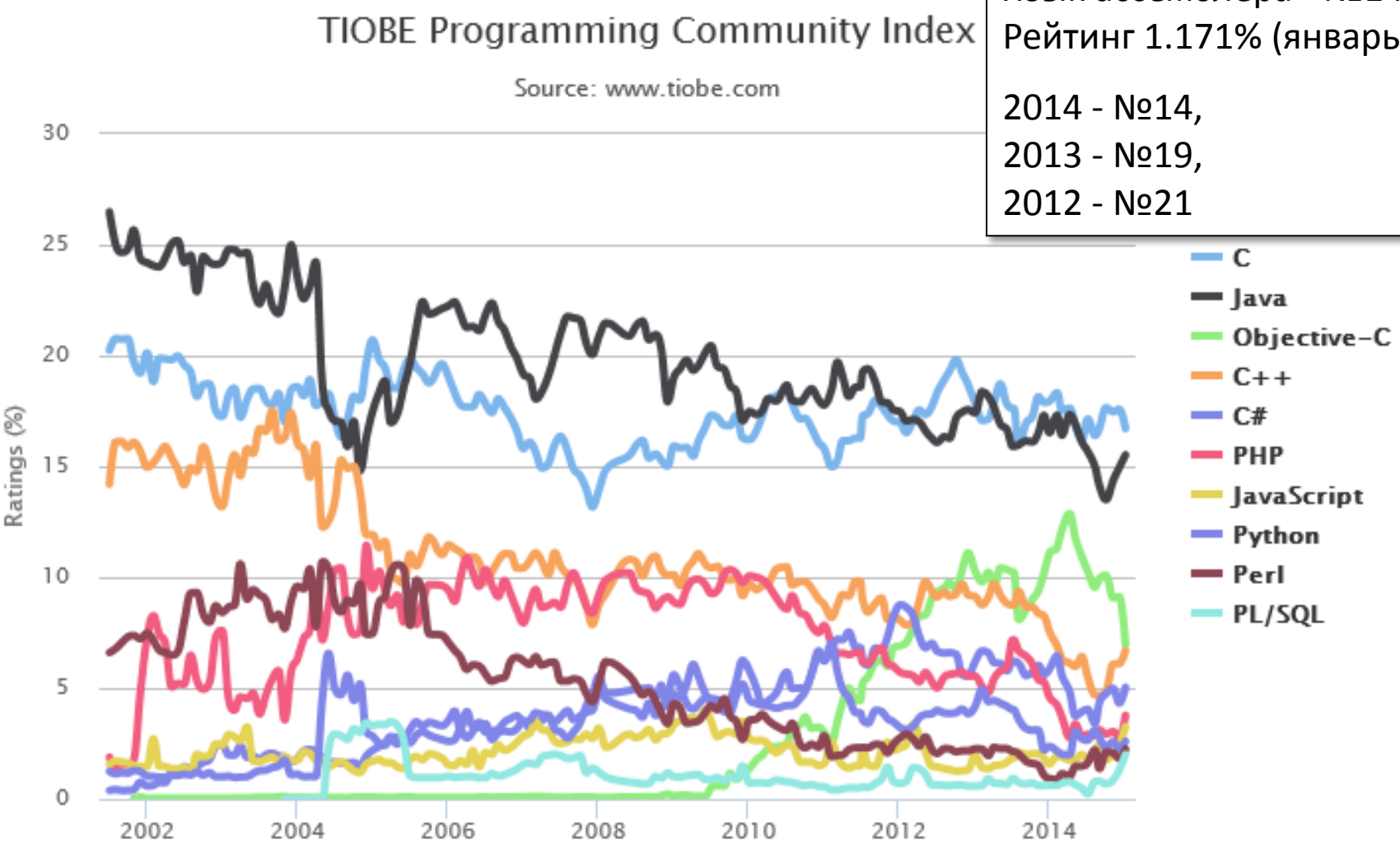
- $X^2 \geq 0$?
  - float – всегда да
  - int ?
    - 40000 * 40000 = 1600000000
    - 50000 * 50000 = ?
- (x + y) + z = x + (y + z) ?
  - signed/unsigned int – всегда да
  - float
    - (1e20 + -1e20) + 3.14 → 3.14
    - 1e20 + (-1e20 + 3.14) → ??
- Свойства типов данных могут быть использованы для причинения вреда

© 2015 МГУ/ВМК/СП

Проблема №2 Язык ассемблера

Язык ассемблера - №14
Рейтинг 1.171% (январь 2015)

2014 - №14,
2013 - №19,
2012 - №21



Компилятор Си составит программу ничуть не хуже

© 2015 МГУ/ВМК/СП

Проблема №2 Язык ассемблера

- Язык ассемблера позволяет понимать поведение машины
  - Отладка ошибок разработчиком программы
  - Настройка производительности программы
    - Почему оптимизация программы компилятором не дает ожидаемого результата
    - В чем причина плохой производительности
  - Системное программное обеспечение создается людьми, в том числе выпускниками ВМК
    - Компиляторы
    - Операционные системы
  - Безопасность программного обеспечения (software security, cyber security)
    - Сертификация программ
    - Средства защиты
    - Вредоносный код: ботнеты, вирусы/черви, руткиты, …

© 2015 МГУ/ВМК/СП

Проблема №2 Язык ассемблера Отладка ошибок

- Оптимизации компилятора способны нарушать безопасность кода

```
memset(password, '\0', len);
free(password);
```

-O3 ⇒

```
free(password);
```

- Язык высокого уровня не специфицирует существенные для безопасности аспекты

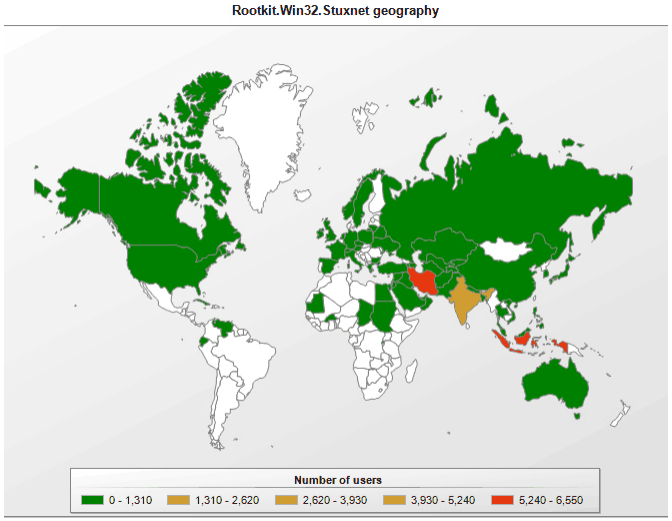
```
int a[2], b;
memset(a, 0, 12);
```

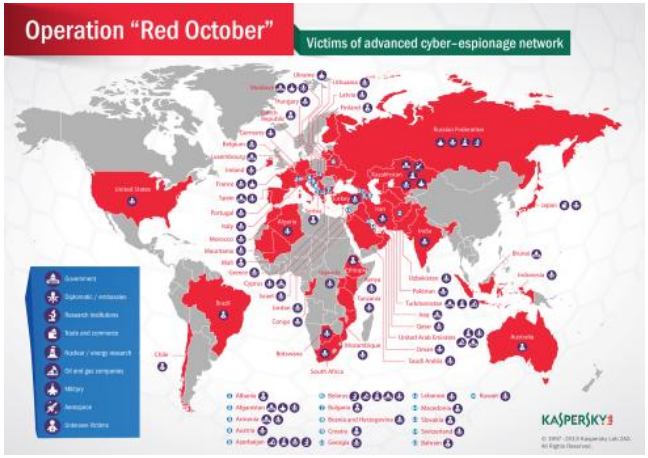
- Соответствие исходного и исполняемого кода
  - доступ к системе сборки и конфигурационным файлам
- Исходный код может быть просто недоступен

© 2015 МГУ/ВМК/СП

Проблема №2 Язык ассемблера
Что происходит на моем компьютере?

- Повсеместное объединение устройств в сети
  - Сервера/рабочие станции, Мобильные устройства, SCADA, маршрутизаторы , …
- «Эпидемии» вредоносного ПО
  - MyDoom, Conficker, Stuxnet, Duqu, Flame, …
  - Red October действовал более 5 лет, взламывая **защищенные** компьютеры
- Причина - ненадежность программной составляющей
  - Дефекты работы с динамической памятью
  - Организация многопоточной работы





© 2015 МГУ/ВМК/СП

Проблема №3 Работа с памятью

- Память ограничена
  - Памятью необходимо управлять
  - Работа многих программ зависит от объема доступной памяти
- Ошибки при работе с памятью тяжело исправлять
  - Возникновение ошибки может быть «далеко» от ее проявления
- Обращение к разным переменным длится разное время
  - Настройка программы под особенности устройства памяти компьютера может дать ощутимое ускорение работы

© 2015 МГУ/ВМК/СП

Проблема №3 Работа с памятью

```
double fun(int i) {
  volatile double d[1] = {3.14};
  volatile long int a[2];
  a[i] = 1073741824;
  return d[0];
}
```

```
fun(0)  →   3.14
fun(1)  →   3.14
fun(2)  →   3.1399998664856
fun(3)  →   2.00000061035156
fun(4)  →   3.14, а потом segmentation fault
```

© 2015 МГУ/ВМК/СП

Проблема №3 Работа с памятью

- Язык Си не предполагает защиты памяти
  - Выход за пределы массива
  - Некорректные указатели
  - malloc/free
- Тяжело отлаживаемые ошибки
  - Проявление ошибки может зависеть от окружения
- Как бороться
  - Java, Ruby, …
  - Понимать как работает компьютер
  - Использовать программные инструменты для отладки
    - Valgrind, Pin, …

© 2015 МГУ/ВМК/СП

Проблема №4 Асимптотическая сложность

```
void copyij(int src[2048][2048], int dst[2048][2048]) {
  int i,j;
  for (i = 0; i < 2048; i++)
    for (j = 0; j < 2048; j++)
      dst[i][j] = src[i][j];
}
```

Память имеет иерархическую структуру

Скорость работы программы зависит от порядка выполняемых действий

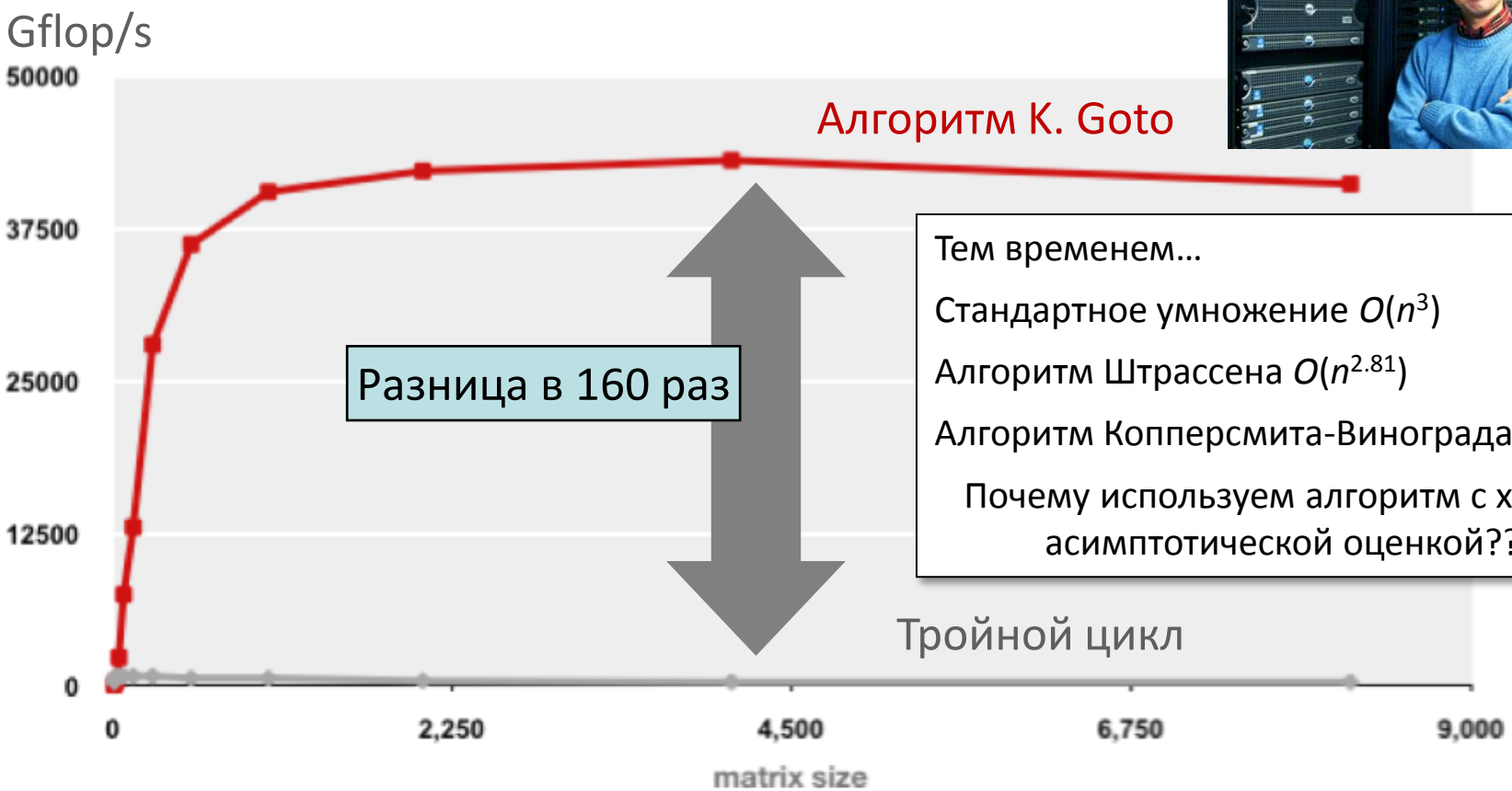
```
void copyji(int src[2048][2048], int dst[2048][2048]) {
  int i,j;
  for (j = 0; j < 2048; j++)
    for (i = 0; i < 2048; i++)
      dst[i][j] = src[i][j];
}
```

Медленней в 21 раз (Pentium 4)

© 2015 МГУ/ВМК/СП

Проблема №4 Асимптотическая сложность

Умножение матриц 2 x Core 2 Duo 3 GHz (double)
Библиотека GotoBLAS



Тем временем…

Стандартное умножение $O(n^3)$

Алгоритм Штрассена $O(n^{2.81})$

Алгоритм Копперсмита-Винограда $O(n^{2.38})$

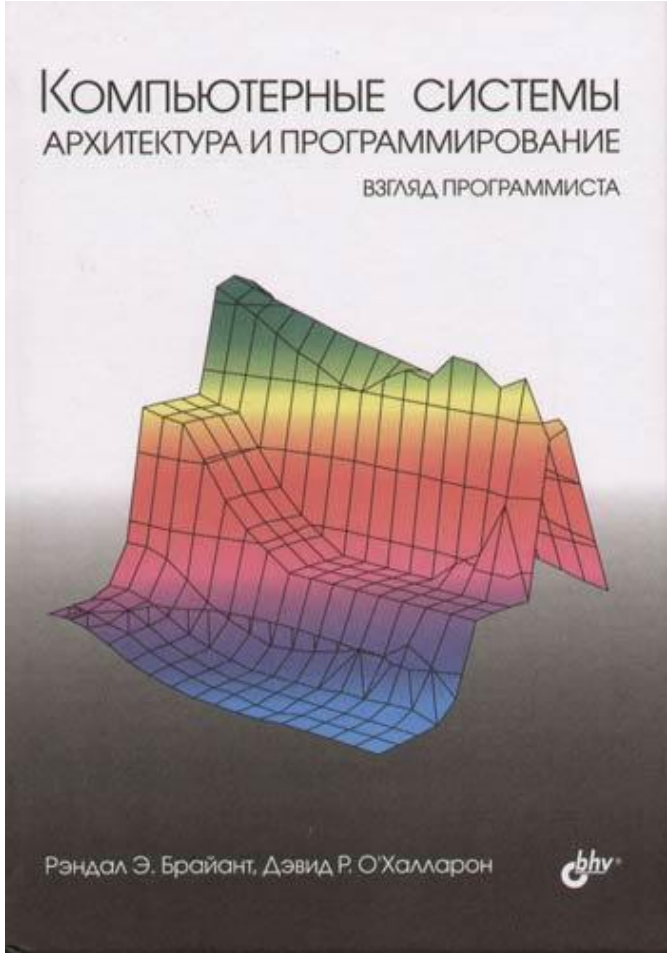
Почему используем алгоритм с худшей асимптотической оценкой???

В точности одинаковая сложность ($2n^3$)

Kazushige Goto and Robert A. van de Geijn. 2008. Anatomy of high-performance matrix multiplication. *ACM Trans. Math. Softw.* 34, 3, Article 12 (May 2008)

© 2015 МГУ/ВМК/СП

Литература



- Рэндал Э. Брайант, Дэвид О'Халларон Компьютерные системы: архитектура и программирование Computer Systems: A Programmer's Perspective

© 2015 МГУ/ВМК/СП

Литература

- Кип Р. Ирвин. Язык ассемблера для процессоров Intel. 4-е издание. Вильямс, 2005.
- Э. Таненбаум. Архитектура компьютера. 5-е изд. - СПб.: Питер, 2007

© 2015 МГУ/ВМК/СП

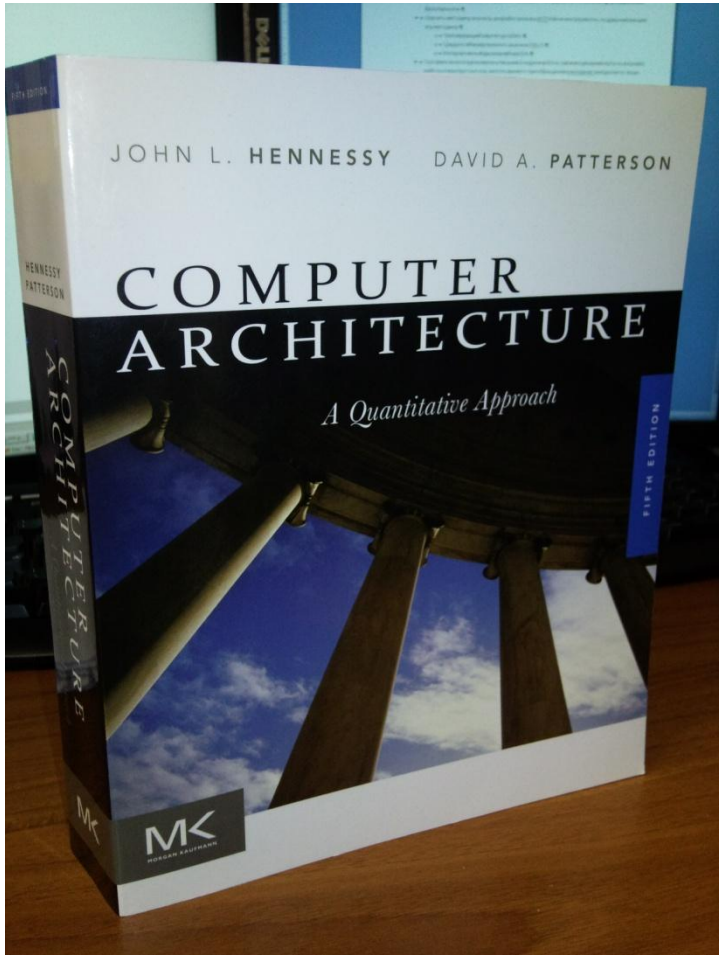
Дополнительная литература

- Д. Паттерсон, Дж. Хеннесси. Архитектура компьютера и проектирование компьютерных систем. Четвертое издание. Питер, 2012
- Рудольф Марек. Ассемблер на примерах. Базовый курс. — СПб: Наука и Техника, 2005.
- Юров Виктор Иванович. Assembler. 2-е изд. Питер, 2010 год.
- Юров Виктор Иванович. Assembler: Практикум. 2-е изд. Питер, 2007 год.

© 2015 МГУ/ВМК/СП

Дополнительная литература

- John L. Hennessy and David A. Patterson. Computer Architecture, 5th Edition: A Quantitative Approach. Morgan Kaufmann Publishers 2012.
- Часть тем курса рассматривается в первых двух главах (~130 страниц)



© 2015 МГУ/ВМК/СП

Техническая литература

- Стандарт языка Си 99. ISO/IEC 9899:1999
- The Intel® 64 and IA-32 Architectures Software Developer's Manual. February 2014. http://www.intel.com/products/processor/manuals/
- Using the GNU Compiler Collection http://gcc.gnu.org/onlinedocs/gcc-4.8.2/gcc.pdf

© 2015 МГУ/ВМК/СП

Из чего состоит курс

- Лекции
  - 48 часов / 24 лекции
  - Два письменных коллоквиума **c1** и **c2**
  - Письменный экзамен **ex**
- Семинары
  - 72 часа / 36 занятий
    - Аудитория
    - Машинный зал
  - Домашние задания
    - всего – 7 заданий
    - 4 задания сдаются в системе Ejudge
  - Зачет с оценкой
  - Комиссия: машинный зал
- Сайт поддержки http://asmcourse.cs.msu.ru/

Итоговая оценка

0.2 * c1 + 0.2 * c2 + 0.6 * ex

- удовлетворительно >= 40%
- хорошо >=60%
- отлично >=80%

Пропуск коллоквиума – штраф -5%.

Ограничение на количество посылок задачи:

- простые задачи – 16,
- обычные задачи – 24,
- повышенной сложности – 32.

На сайте поддержки курса в разделе «Экзамены и контрольные» детально описаны правила выставления оценок за лекционный курс и практикум http://asmcourse.cs.msu.ru/?page_id=1020

© 2015 МГУ/ВМК/СП

План лекций

- Введение
  на примерах ассемблерных программ
  - 4 лекции
- Взаимосвязь языка Си, языка ассемблера и особенностей архитектуры IA32
  - 10 лекций
- Архитектура ЭВМ
  - 5 лекций
- Система программирования языка Си и ее связь с архитектурой ЭВМ
  - 4 лекции

Ориентировочное время проведения коллоквиумов – середина марта и конец апреля

© 2015 МГУ/ВМК/СП

История первых компьютеров

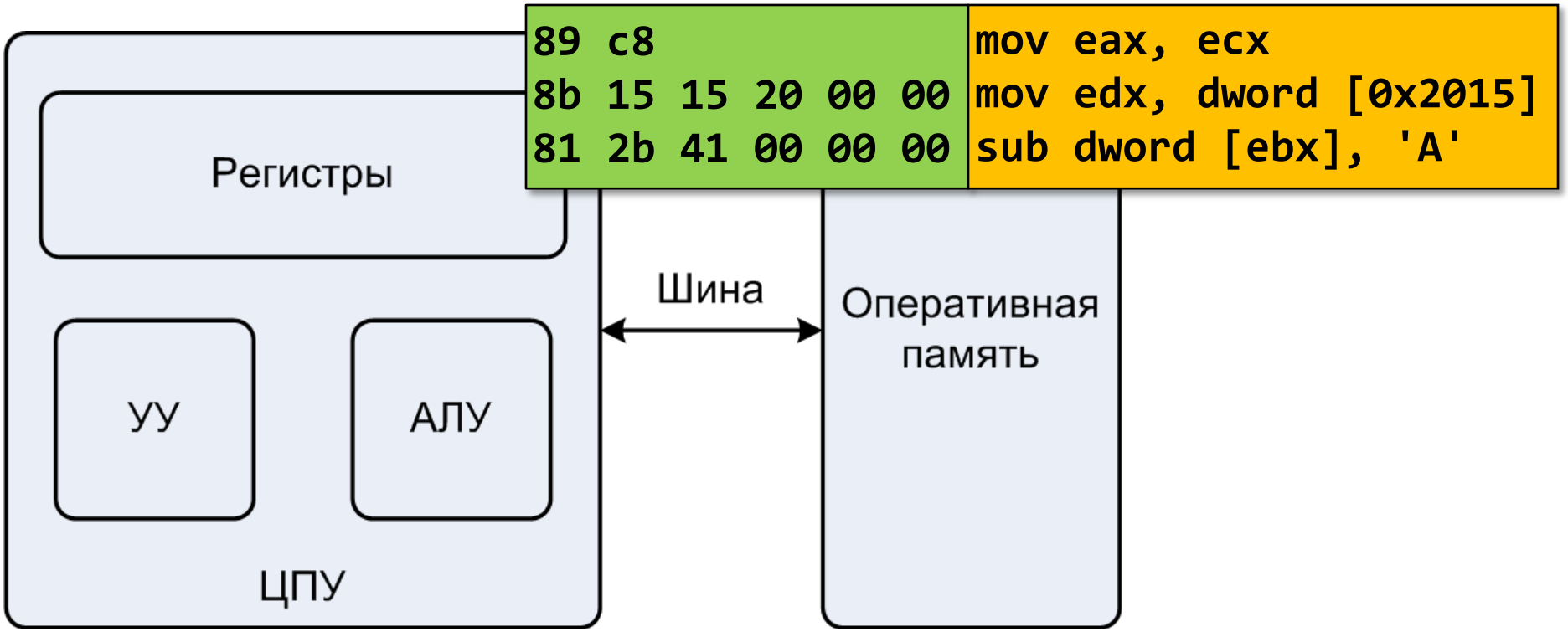
- Чарлз Бэббидж, Ада Лавлайс 1820 – 1833
- Конрад фон Цузе
  - Z1 1938г., Z2 1939 г., Z3 1941г., Z4 1945г.
- Экерт и Моучли
  - ENIAC 1946 г.
  - фон Нейман
    - First Draft of a Report on the EDVAC, 30 июня 1945
- Атанасов и Берри
  - ABC 1942 г.
- Mark I
  - 1949 г.
- Исаак Брук
  - М-1 1951 г.
- Сергей Лебедев
  - БЭСМ-1 1952 г.

Задача первых компьютеров: автоматизация вычислений
- задачи математической физики
- шифрование/дешифрование сообщений

Современное применение компьютеров
- хранение информации и предоставление удобного доступа для работы с ней
- передача информации
- управление технологическими процессами
- и многое другое …

© 2015 МГУ/ВМК/СП

Архитектура фон Неймана

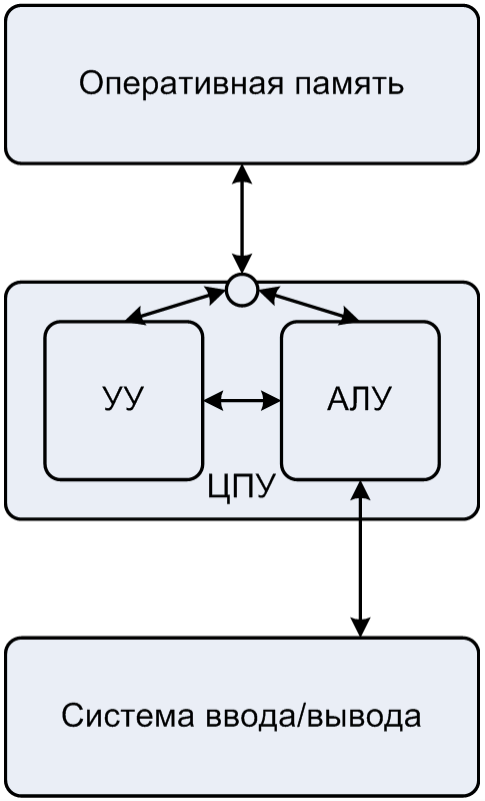


1. Двоичное кодирование информации
2. Неразличимость команд и данных
3. Адресуемость памяти
4. Последовательное выполнение команд

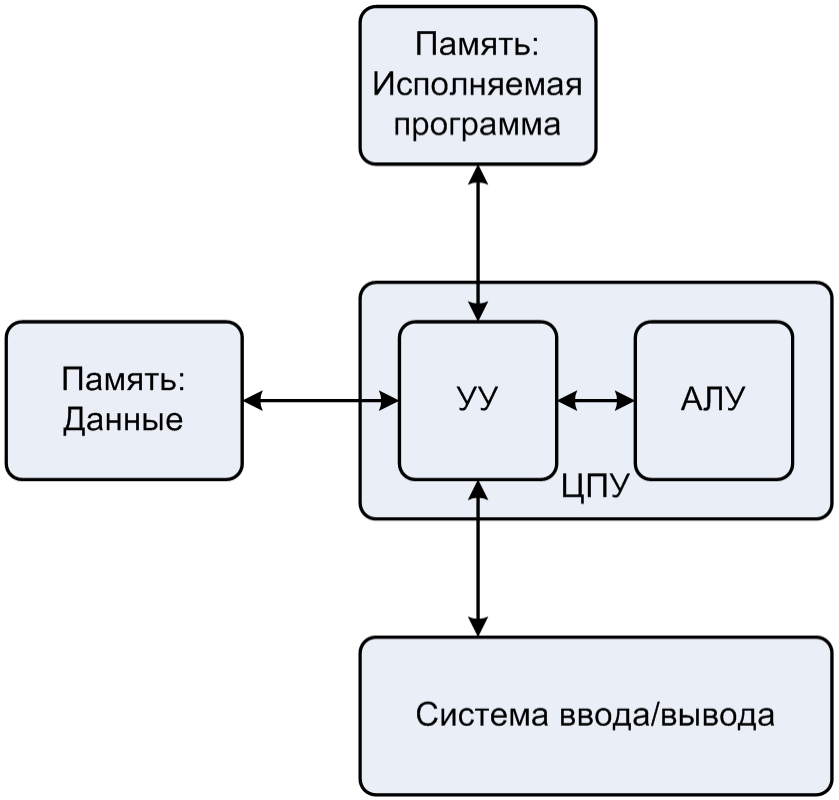
Вспоминаем Си-машину…

© 2015 МГУ/ВМК/СП

Единственный вариант архитектуры?



Еще один вариант изображения архитектуры фон Неймана…



Гарвардская архитектура

Вспоминаем Си-машину…

© 2015 МГУ/ВМК/СП

Цикл работы

1. **Извлечение инструкции из памяти**
   Используя текущее значение *счетчика команд,* процессор извлекает некоторое количество байт из памяти и помещает их в *буфер команд*
2. **Декодирование команды**
   Процессор просматривает содержимое *буфера команд* и определяет *код операции* и ее *операнды*
   Длина декодированной команды прибавляется к текущему значению *счетчика команд*
3. **Загрузка операндов**
   Извлекаются значения операндов
   Если операнд размещен в ячейках памяти – вычисляется *исполнительный адрес*
4. **Выполнение операции над данными**
5. **Запись результата**
   Результат может быть записан в том числе и в *счетчик команд* для изменения естественного порядка выполнения

© 2015 МГУ/ВМК/СП

Промежуточные итоги: ключевые термины и понятия

- Регистр, имя регистра, значение регистра, разрядность регистра, машинное слово
- Память, ячейка памяти, адрес ячейки памяти, время доступа к памяти
- Цикл, тактовая частота, счетчик команд
- Машинная команда, код операции, операнд, адресность команды
  Исполнительный (действительный) адрес
  англ. *effective (executive) address*
- Способ адресации
  прямая, непосредственная, косвенная