Соглашения о вызовах

- Соглашения о вызовах определяет порядок вызова процедур (функций, подпрограмм), передачи им параметров и возврата значений
- Соглашение о системных вызовах определяет порядок переключения в привилегированный режим, передачи параметров и получения результата

Модель процессора ix86

- Регистры %eax, %ebx, %ecx, %edx, %esi, %edi, %ebp, %esp
- Регистры FPU: %fp(0), ..., %fp(7)
- Непосредственные операнды: $5
- Вид инструкций:
  - movl $5, %eax ;; загрузка 5 в %eax
  - movl 8(%ebp), %ecx ;; ecx = [ebp + 8]

Соглашение о вызовах Си

- Параметры передаются через стек
- Заносятся в обратном порядке
- Результат возвращается в %eax или %eax, %edx или %st(0)
- Регистры %esi, %edi, %esp, %ebp не должны модифицироваться вызываемой функцией
- Регистры %eax, %ebx, %ecx, %edx могут модифицироваться
- Стек очищает вызывающая функция
- Стек выравнивается по границе 16 байт

Пример

```
        .section        .rodata
txt:    .asciz  "Hello, world, %d\n"
        .text
        .globl  main
main:   pushl   %ebp
        movl    %esp, %ebp
        call    getpid
        pushl   %eax
        pushl   $txt
        call    printf
        addl    $8, %esp
        xorl    %eax, %eax
        popl    %ebp
        ret
```

Стандартное соглашение Win

- Параметры передаются через стек
- Параметры заносятся в обратном порядке
- Очистку стека выполняет вызванная функция
- Для защиты от передачи неправильного количества параметров имя функции декорируется, например, memcpy@8

Передача параметров через регистры

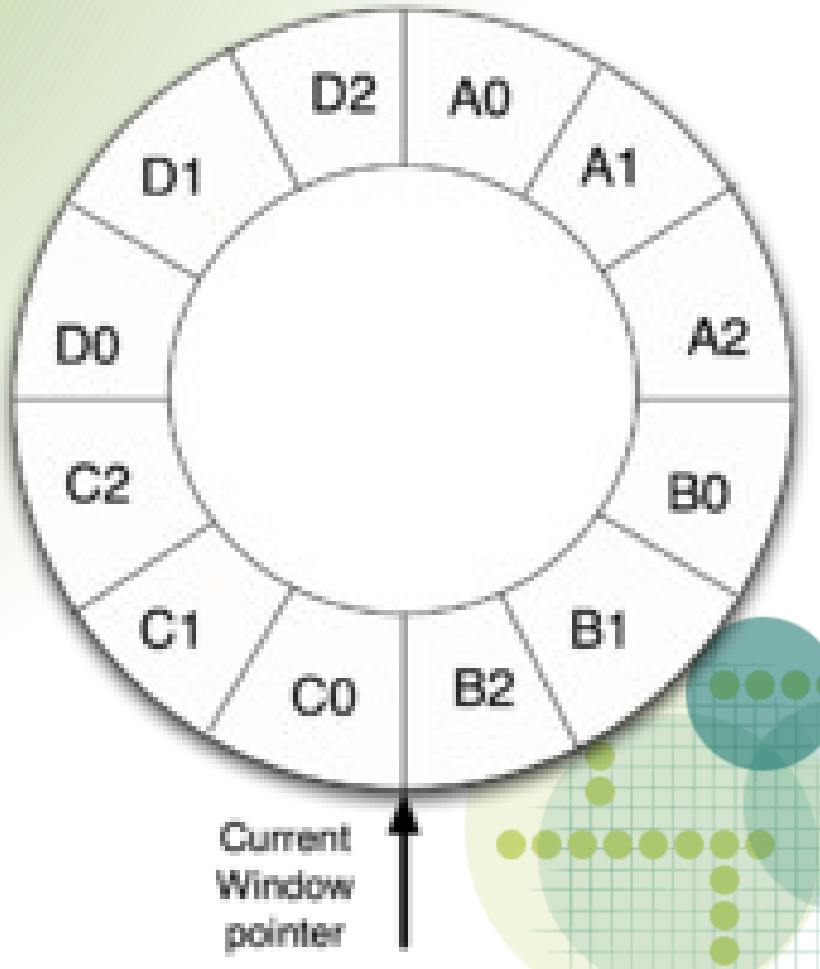
- Первые три параметра передаются через %eax, %edx, %ecx
- Остальные параметры передаются через стек

Регистровые окна

- Каждая функция имеет: входные регистры, рабочие регистры, выходные регистры
- Набор регистров одной функции: окно
- Набор всех регистров: файл
- При вызове функции окно «поворачивается», выходные становятся входными
- Когда цепочка вызовов становится слишком глубокой, регистровый файл выгружается на стек

SPARC

- Глобальные: g0(0), g7
- Входные: i0...i7
- Рабочие: L0...L7
- Выходные: o0...o7



Системные вызовы

- Соглашение о передаче параметров определяется ОС
- Linux (традиционно):
  - Номер системного вызова - %eax
  - Параметры - %ebx, %ecx, %edx, %esi, %edi, %ebp
  - int 0x80
  - Значение возвращается в %eax

Пример: open

- int open(char const *path, int flags, int mode);

```
open:
   pushl   %ebp
   movl %esp, %ebp
   movl $__NR_open, %eax
   movl 8(%ebp), %ebx
   movl 12(%ebp), %ecx
   movl 16(%ebp), %edx
   int     0x80
   leave
   ret
```

UNIX

- Переносимая многозадачная многопользовательская система разделения времени
- Философия Unix (Unix-way)
  - Хранение данных в текстовых файлах
  - Файловая система с единственным корнем
  - Все устройства и средства межпроцессного взаимодействия — файлы
  - Каждая утилита делает одно дело, но делает его хорошо

История UNIX

- 1971 — первая версия (на ассемблере) для PDP-11
- 1973 — переписан на языке Си
- 1975 — портирование на Interdata 7/32
- До 1982 года распространялся бесплатно
- В начале 80-х появилось много разных и не совсем совместимых версий от разных компаний: System III — System V, SunOS, Xenix, BSD — начались войны за стандартизацию и конфликт между AT&T и Беркли

1990-е

- 1992 — 386BSD — версия без ограничений на распространение, дала начало:
  - FreeBSD — версия ориентированная на пользователей
  - OpenBSD — версия с упором на защищенность
  - NetBSD — версия с упором на переносимость
- 1987 — Minix — учебная версия Unix
- 1991 — Linux — написан с нуля, свободен от всех ограничений

Стандартизация

- Семейство стандартов POSIX (1988 и далее) описывает разные аспекты функционирования: API, набор утилит и т. д.
- Начало 90-х Single Unix Specification (The Open Group)
- SUS v3, POSIX:2001 (2002) слияние стандартов POSIX и SUS
- POSIX:2008 (декабрь 2008)

Стандартизация Linux

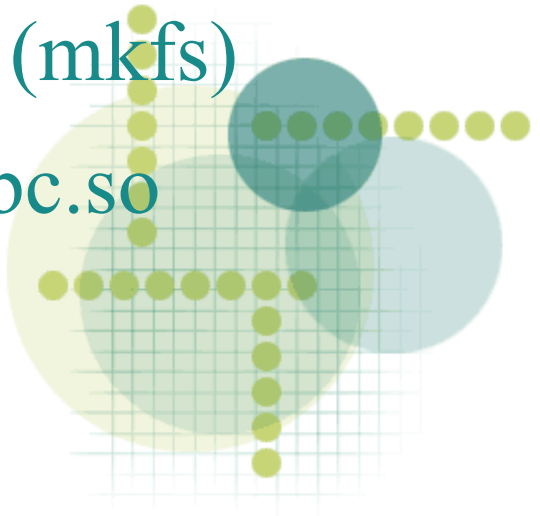
- За 90-е годы появилось много дистрибутивов Linux, некоторые подражали BSD (Slackware), некоторые подражали SystemV (RedHat)
- Linux Standard Base (LSB) (2001) де-факто стандарт для Linux

Дерево каталогов

- /usr/include — абсолютный путь
- task1/task1 — относительный путь (отсчитывается от текущего каталога процесса)
- . (точка) — текущий каталог
- .. (две точки) — родительский каталог

Структура дерева каталогов

- /etc — конфигурационные файлы: /etc/passwd
- /boot — загрузочные файлы (в том числе образ ядра): /boot/vmlinuz
- /bin — системные утилиты (необходимые для загрузки системы): sh, mount, …
- /sbin — административные утилиты (mkfs)
- /lib — системные библиотеки: /lib/libc.so
- /dev — файлы устройств



Структура дерева каталогов

- /root - «домашний» каталог пользователя root
- /mnt — точка монтирования для подключаемых устройств
- /var — рабочие каталоги серверных программ, журналы и прочая рабочая информация
- /home - «домашние» каталоги пользователей: /home/ivan
- /usr — все остальное

Структура дерева каталогов

- /usr/include — заголовочные файлы библиотек
- /usr/bin — основные исполняемые файлы
- /usr/lib — основные библиотеки
- /usr/sbin — сетевые системные программы (серверы)
- /usr/share — файлы, не зависящие от архитектуры
- /usr/src — исходный текст ядра и утилит
- /usr/local — каталог для локальной установки

Структура дерева каталогов

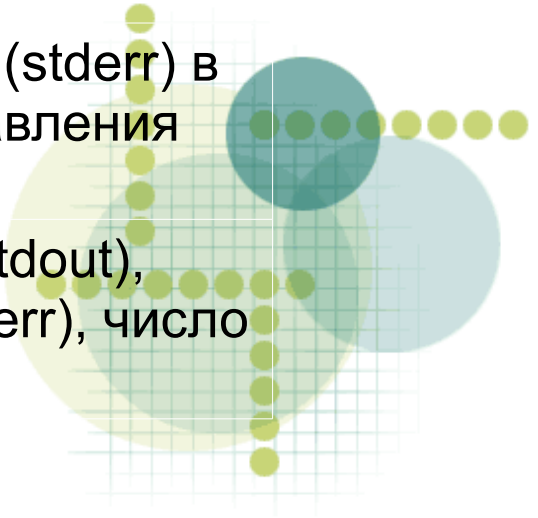
- /var/log — файлы журналов
- /var/mail — файлы с почтой пользователей
- /var/tmp — каталог временных файлов
- /var/www — каталог веб-сервера
- /var/lib — рабочие каталоги программ
- /var/lib/mysql
- /var/lib/texmf

Основные команды

- man — получение справки
- cp — копирование файлов
- mv — перенос файлов
- cat — конкатенация файлов
- ls — получение списка файлов в каталоге
- rm — удаление файлов и каталогов
- mkdir — создание каталогов
- cd — переход в другой каталог
- pwd — получение имени текущего каталога

Перенаправление потоков

prog > FILE	Перенаправление stdout программы prog в файл FILE в режиме перезаписи
prog 2> FILE	Перенаправление файлового дескриптора 2 (stderr) в файл FILE в режиме перезаписи
prog >> FILE	Перенаправление stdout в файл FILE в режиме добавления
prog 2>> FILE	Перенаправление ф. д. 2 (stderr) в файл FILE в режиме добавления
prog 2>&1	Создание копии ф. д. 1 (stdout), доступной как ф. д. 2 (stderr), число 1 может быть опущено



Перенаправление потоков

prog < FILE	Перенаправление стандартного потока ввода из файла FILE
prog << _END_ … data ... _END_	Текст до _END_ подается на стандартный поток ввода
prog1 | prog2	Стандартный поток вывода prog1 соединяется со стандартным потоком ввода prog2