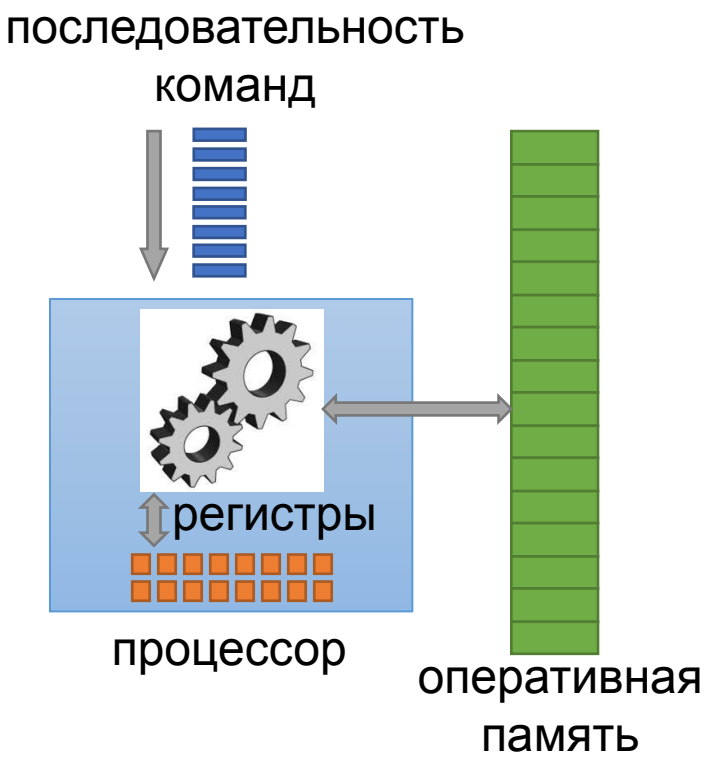
Краткий обзор языка ассемблера архитектуры x86/x86-64

Инструменты

- Получение ассемблерного листинга
  - gcc -S prog.c → prog.s
  - icc -S prog.c → prog.s
- Компиляция ассемблерного листинга
  - gcc prog.s → a.out
- Дизассемблирование объектного кода
  - objdump -d a.out >prog.txt → prog.txt
- Полезный ресурс:
  - Compiler Explorer, https://godbolt.org/

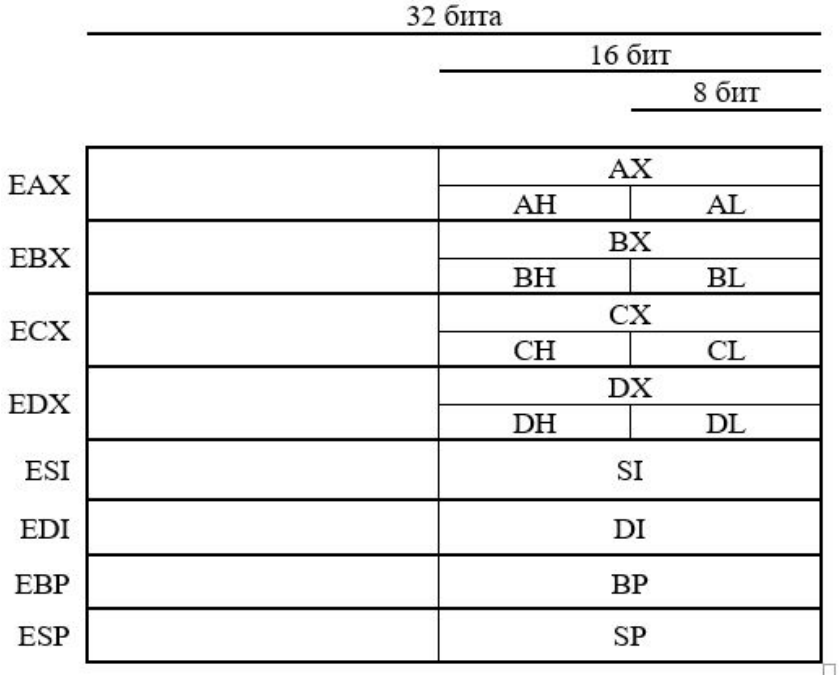
Упрощённая схема процессора



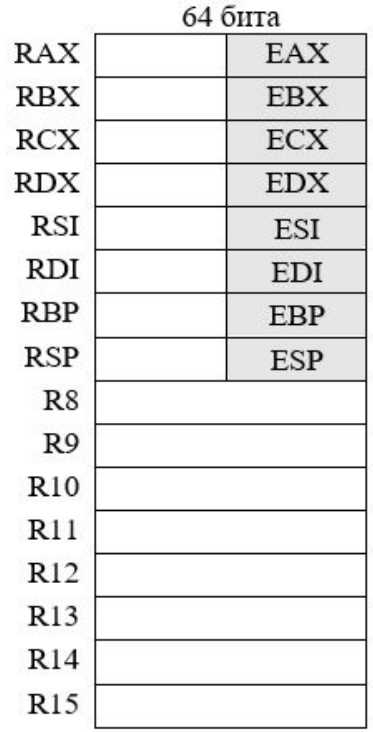
Регистры

- регистры общего назначения
  - имеют размер машинного слова (размер адреса в данной арх.)
  - некоторые части регистров тоже являются регистрами
  - количество регистров зависит от версии архитектуры: 8 / 16
  - используются для целочисленной арифметики и адресации памяти
- регистры сопроцессора
  - имеют фиксированный размер: 10 байт (80 бит) и количество: 8
  - используются для хранения вещественных 10-байтных значений
  - организованы в круговой стек, имена регистров задают смещение относительно вершины стека: st(0), st(1), st(2), ...
  - в 32-битном режиме компилятор использует их для хранения вещественных значений,
  - в 64-битном режиме компилятор их использует, только если явно задана расширенная точность (long double).
- векторные регистры
  - имеют размер и количество в зависимости от версии архитектуры
  - могут хранить скалярные (одиночные) значения различных типов и вектора значений - столько, сколько входит в регистр
- флаговые регистры
  - 1-битовые регистры
  - хранят характеристики результата выполнения “последней” команды (определённые команды модифицируют определённые флаги)

Регистры общего назначения



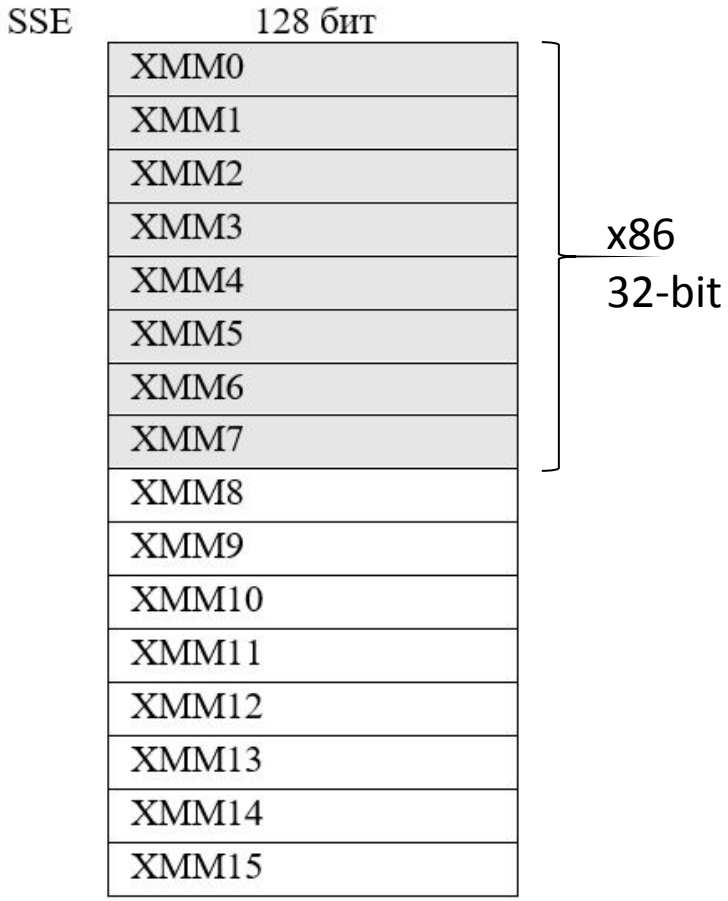
x86 32-bit

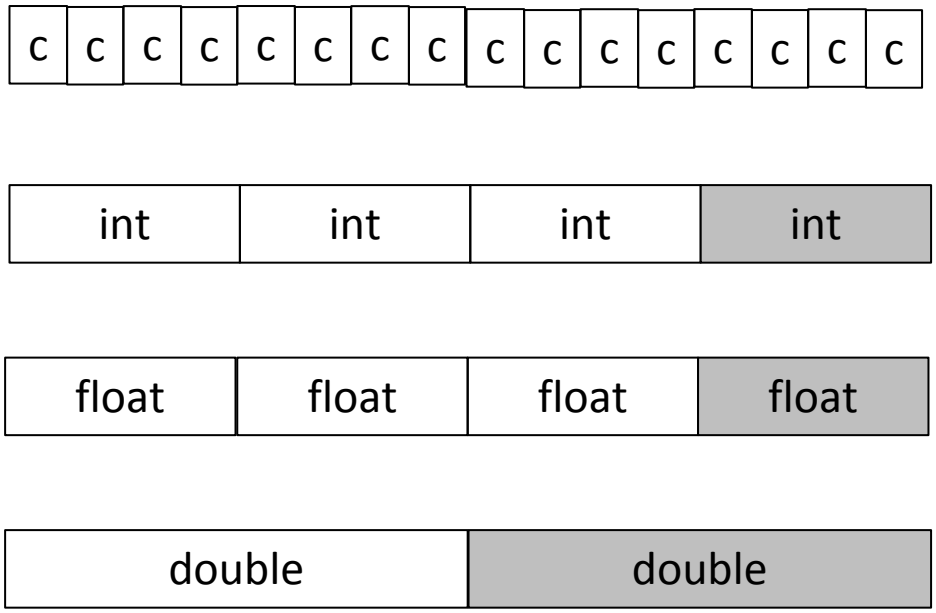


x86-64

Регистры сопроцессора

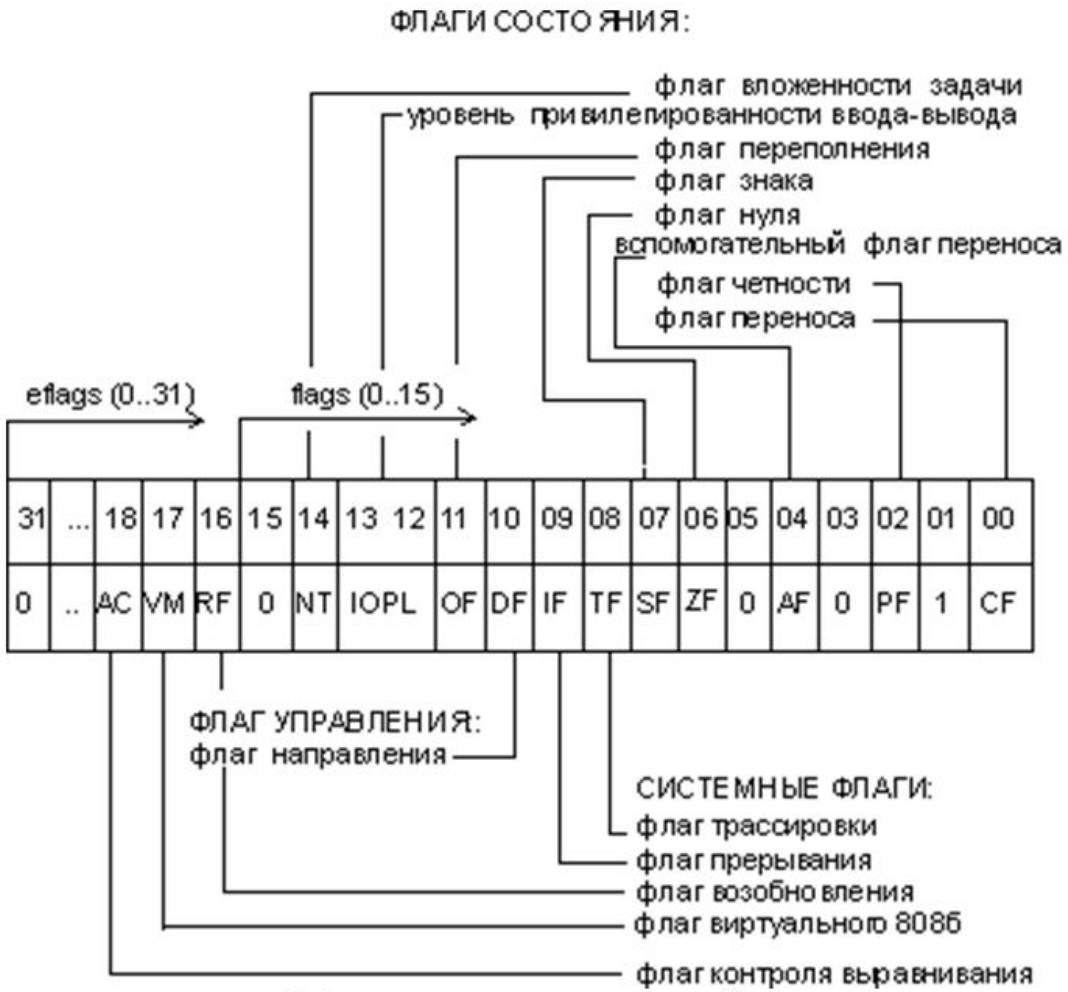
Векторные регистры





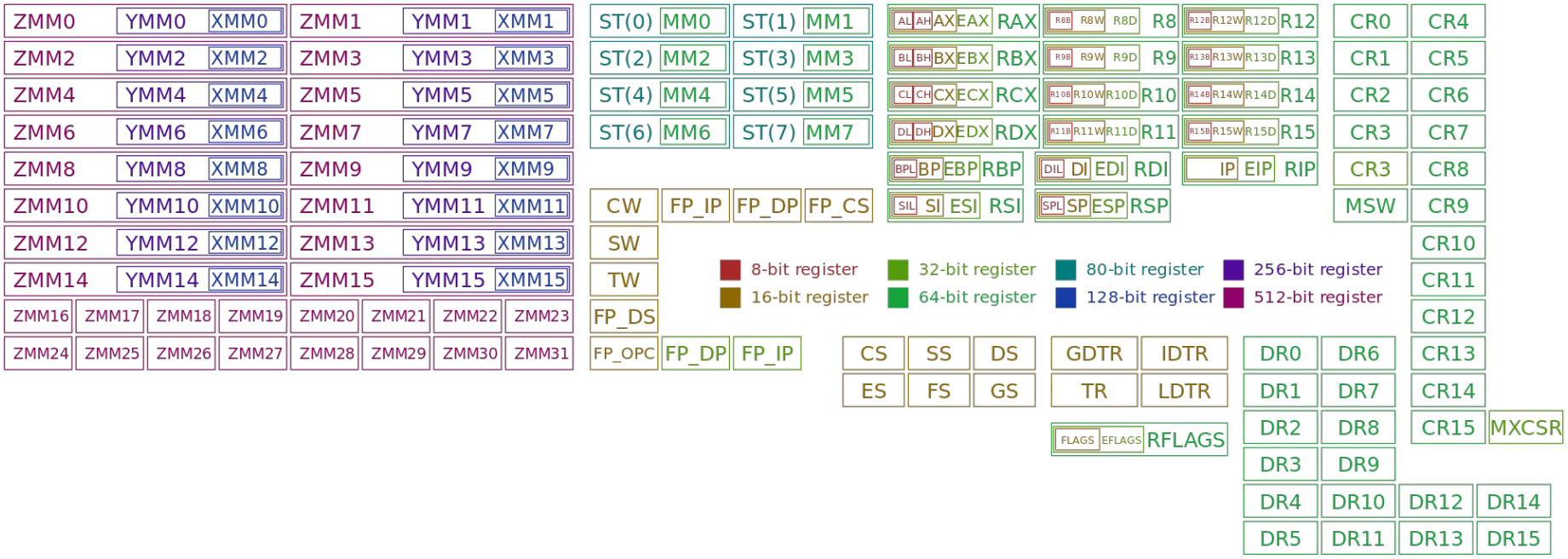
XMM: 16 байт
YMM: 32 байта
ZMM: 64 байта

Флаговые регистры (флаги)



Содержимое регистра eflags

Регистры x86



Wikipedia

Адресация памяти

- задание адресного выражения:
  - Общий вид: d(a,b,c) = d + a + b * c
  - Сокращённый вид: (a), d(a), (a,b), (,b,c), ...
- задание метки:
  - Например: .LC1
- неявное задание адреса
  - Например: movsb → [rdi++] := [rsi++];

Команды

Формат команды:

- Префикс
  - rep команда → while (rcx--) команда;
  - lock команда
  - ...
- имя команды – какую операцию выполнять
  - add, mov, cmp, ...
- суффикс - тип данных
  - b, w, l, q, sd, ps, ...
- параметры команды:
  - константы, регистры, адресные выражения

Основные виды команд

- копирование: mov, xchg, ...
- арифметико-логические:
  - арифметические: add, sub, mul, div, sqrt, …, lea, ...
  - логические побитовые: and, or, not, xor, …
  - сравнения: test, cmp, …
  - сопроцессор: fadd, fmul, ...
- передача управления:
  - jmp, j__ (e / ne / g / ge / l / le /...)
  - call, ret, ...
- работа со стеком: push, pop
- работа с флагами: cl_, st_, lahf, sahf, pushf, popf, …
- ...

Строение стека