Учебный курс

Введение в цифровую электронику

Лекция 6

Программирование микропроцессорной системы

кандидат технических наук, доцент
Новиков Юрий Витальевич

Языки программирования

- **Языки высокого уровня** — удобные для разработчика, не зависят от аппаратуры, имеют развитые готовые средства обработки и отображения, программы легко переносятся на другую аппаратуру; но формируют большие и медленные программы (Си, Паскаль, Фортран и т.д.);
- **Языки низкого уровня** — максимально приближены к аппаратуре, трудно писать сложные программы обработки, программы могут не работать на другой аппаратуре; но формируют максимально компактные и быстрые программы (язык машинных кодов, Ассемблер);
- Сочетание языков высокого и низкого уровней даёт оптимальные результаты.

Команды процессора

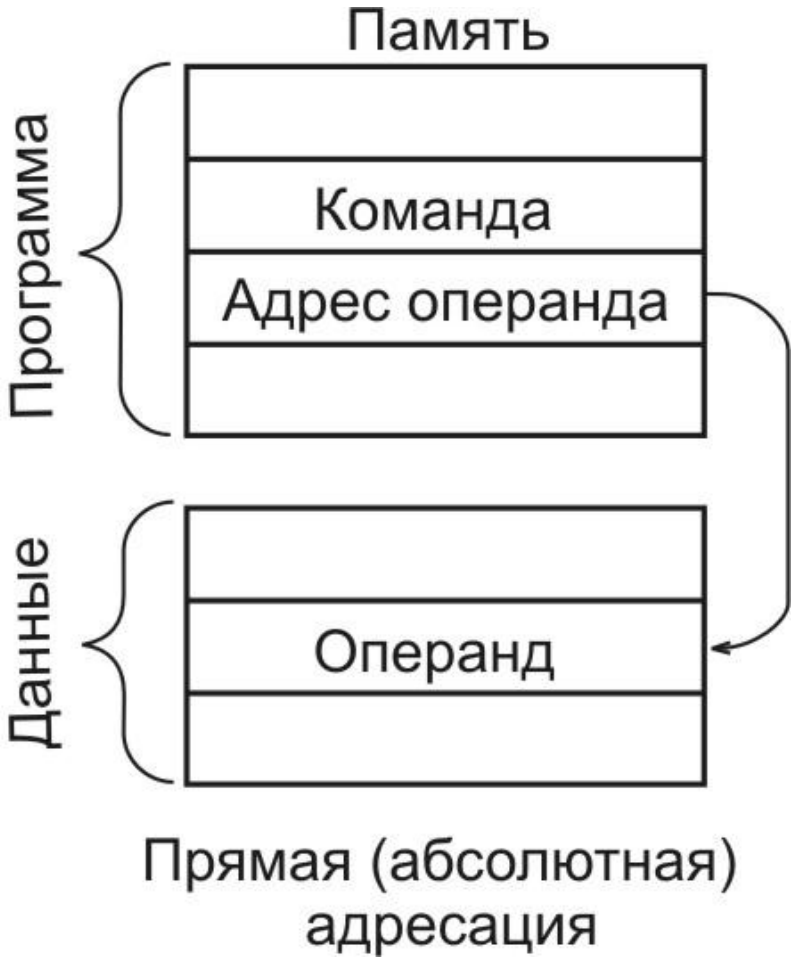
- Могут состоять из одного или нескольких байт; сколько именно байт команды читать процессору — указано в первом байте (слове). Короче команда — быстрее, длиннее команда — сложнее операция.
- Включают в себя код операции, которую должен выполнить процессор, а также указания на операнды, с которыми надо выполнять операцию.
- Образуют систему команд, сложность и полнота которой определяет быстродействие процессора, его универсальность и удобство использования.
- Преобразуются процессором в последовательность внутренних микрокоманд.

Операнды

- **Операнды** — это коды данных, с которыми производятся операции при выполнении программы (входные и выходные);
- **Расположение операндов**:
  - Во внутренних регистрах процессора — самое удобное и легкодоступное, но недостаточно места;
  - В ячейках памяти — самое часто встречающееся и достаточно удобное (доступно большое количество ячеек) — массивы;
  - В устройствах ввода/вывода — самый редкий случай, при обмене с внешними устройствами.
- **Адресация операндов** — это способ указания процессору на место расположения операндов (присутствует в каждой команде).

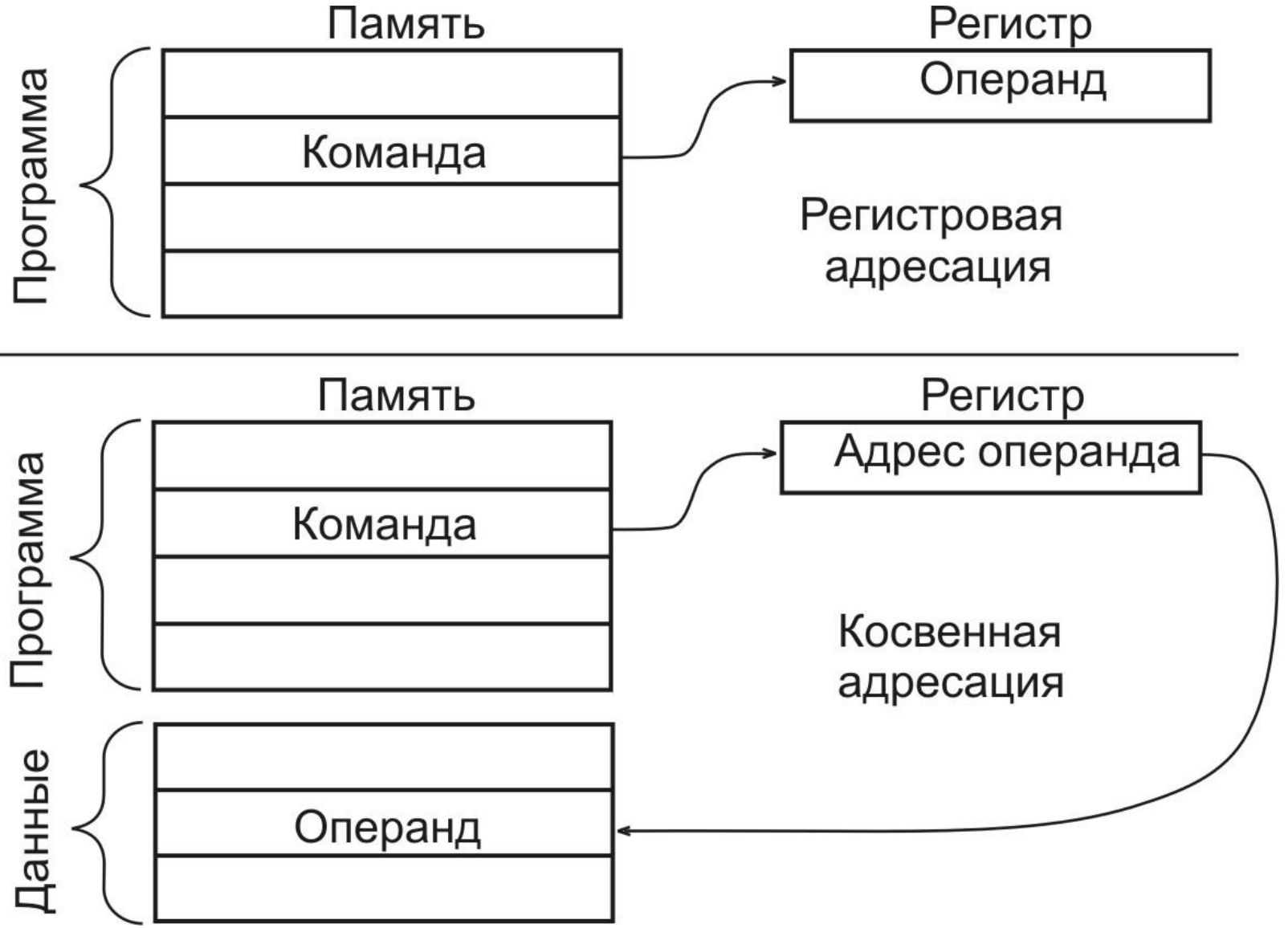
Непосредственная адресация и прямая адресация операндов

Непосредственная адресация



Прямая (абсолютная) адресация

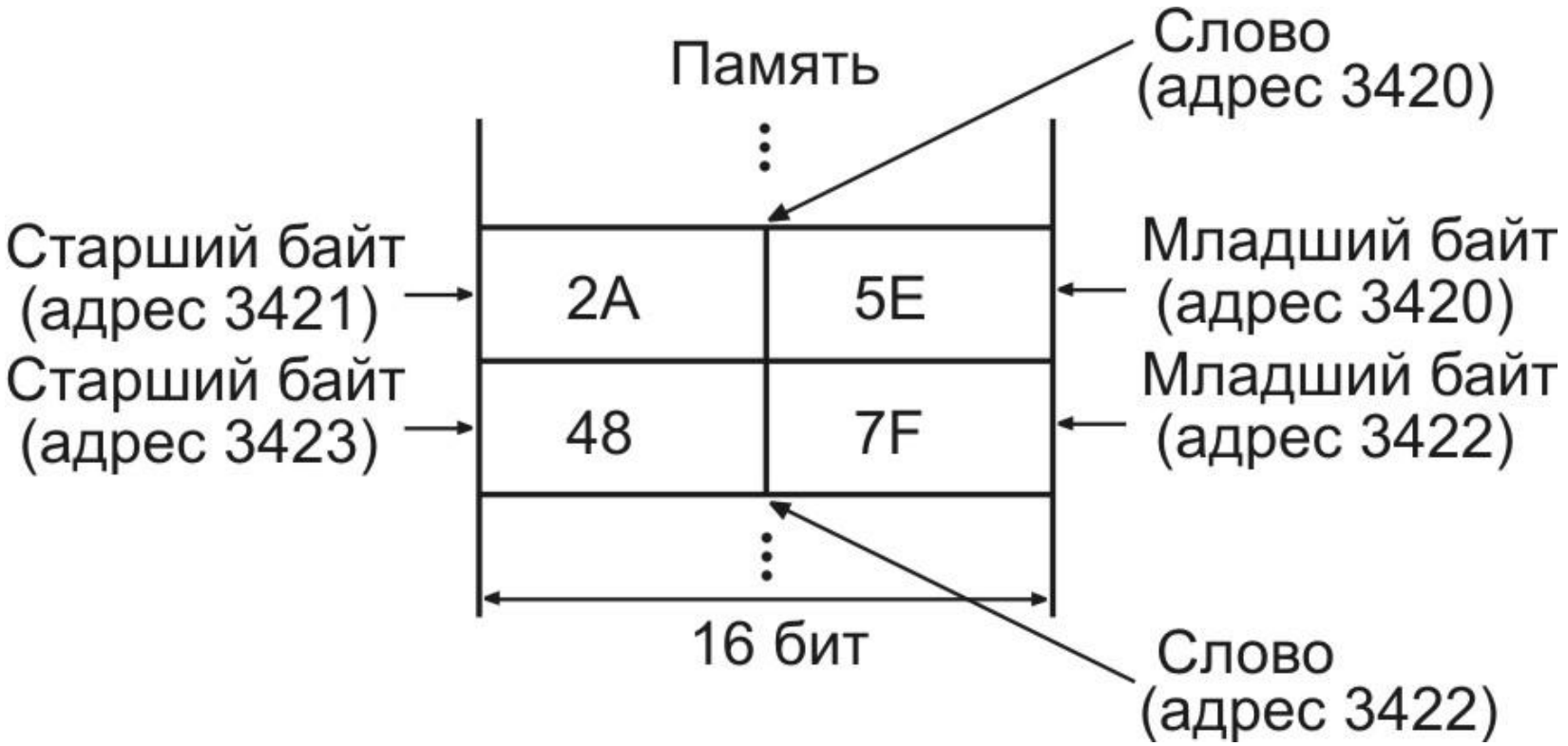
Регистровая адресация и косвенная адресация операндов



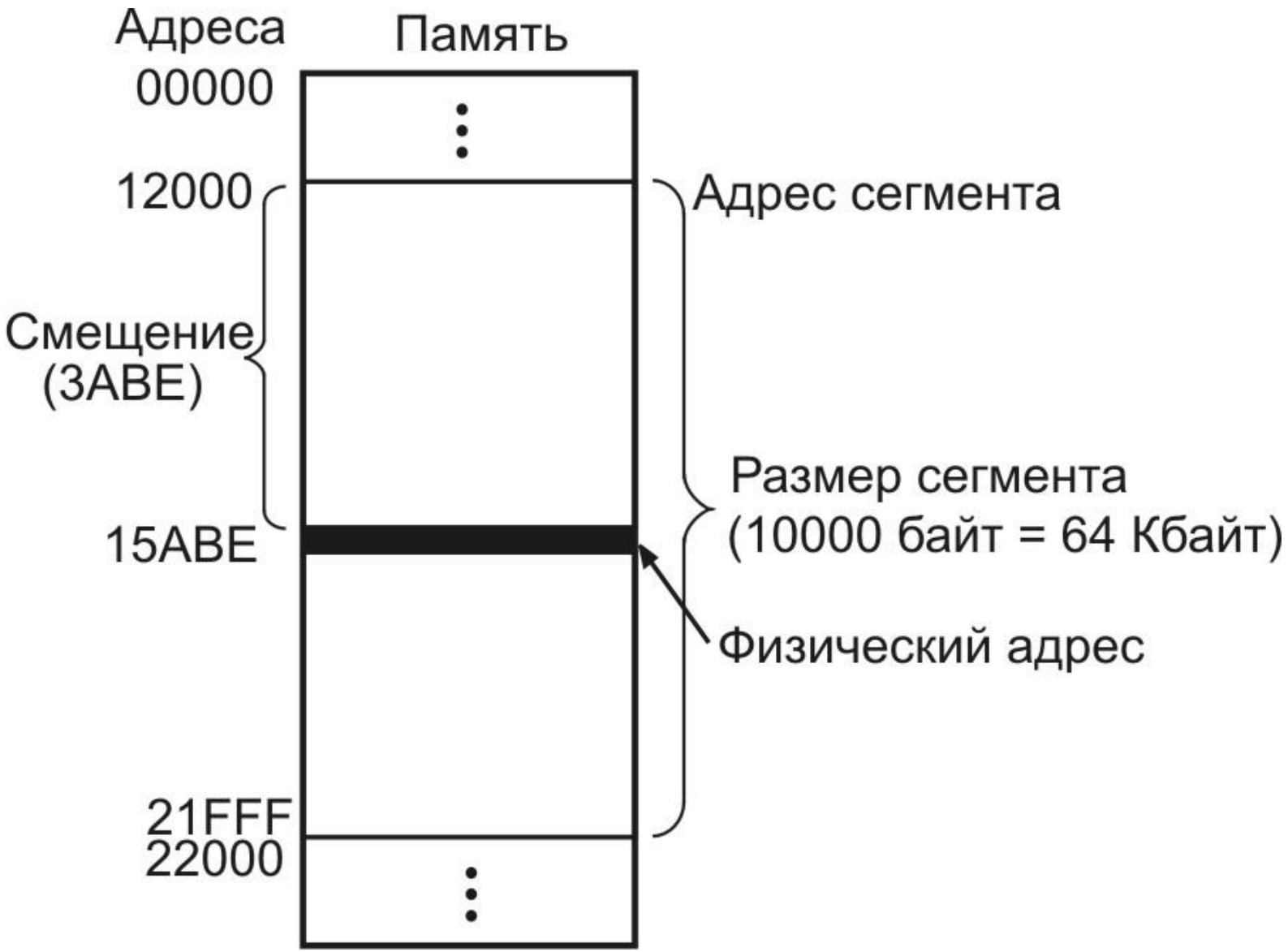
Автоинкрементная адресация и автодекрементная адресация

- **Автоинкрементная адресация**: похожа на косвенную, но *после* выполнения операции содержимое регистра увеличивается на 1 или на 2 (инкрементируется — постинкремент);
- **Автодекрементная адресация** работает, как косвенная, но *перед* выполнением операции содержимое регистра уменьшается на 1 или на 2 (декрементируется — предекремент);
- Оба типа адресации применяются для работы с массивами данных (последовательного их сканирования вверх или вниз);
- Если оба типа адресации используются одновременно, то мы получаем буфер типа LIFO (например, в стеке).

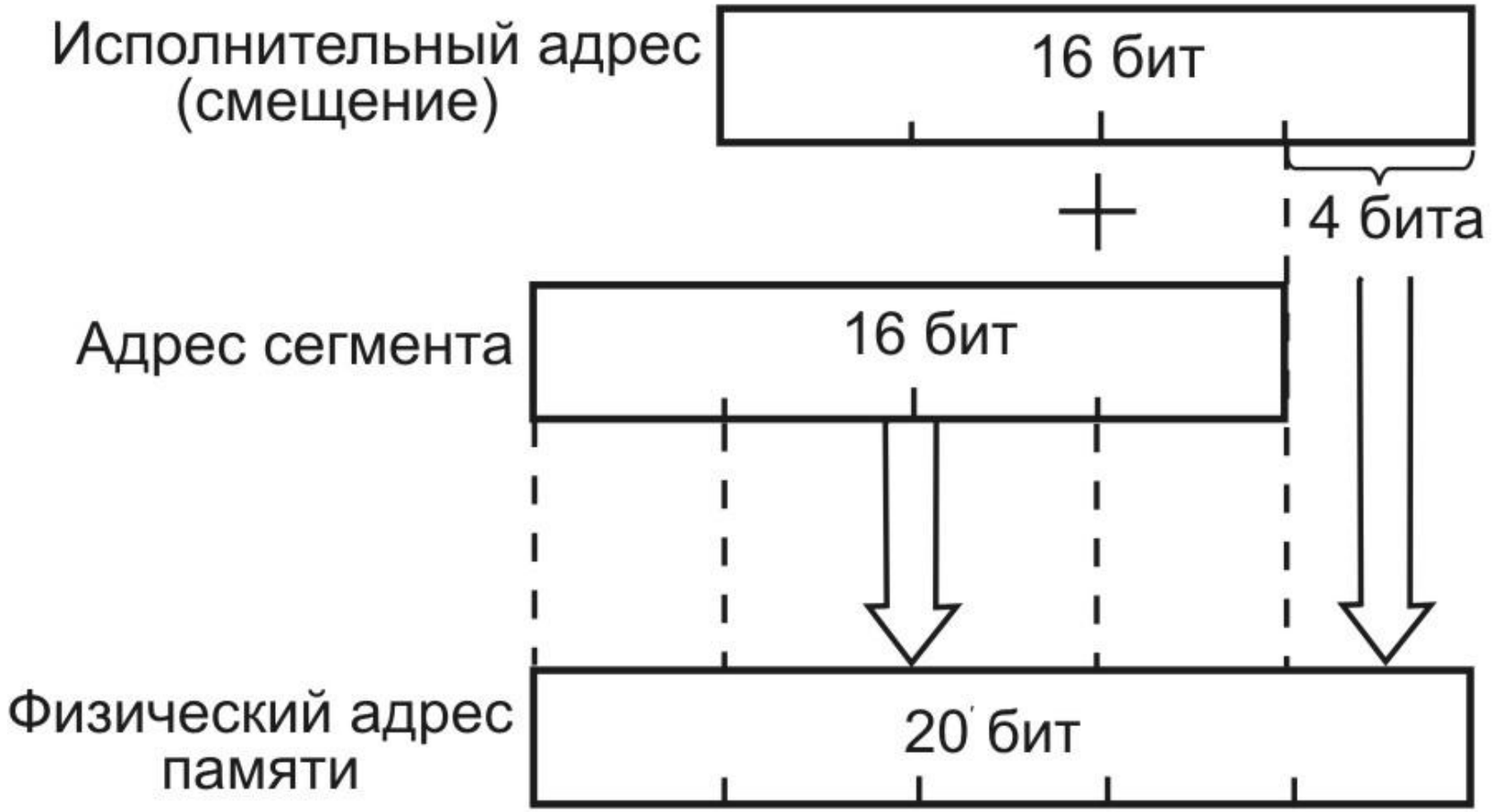
Адресация слов и байтов



Сегментирование памяти



Вычисление адреса в памяти при сегментировании



Основные группы команд процессора

- **Команды пересылки данных** — данные пересылаются (копируются) между памятью, регистрами процессора и УВВ. Не требуют выполнения каких-нибудь операций над данными;
- **Арифметические команды** — выполнение арифметических операций (сложение, вычитание, и т.д.). Один или два входных операнда и один выходной;
- **Логические команды** — выполнение логических операций (И, ИЛИ, инверсия, очистка, сдвиги). Один или два входных операнда и один выходной;
- **Команды переходов** — условные и безусловные. Операндов нет. Изменяется состояние регистра-счётчика команд. Вызов подпрограмм, ветвление алгоритмов.

Команды пересылки данных

- Загрузка (запись) содержимого во внутренние регистры процессора;
- Сохранение в памяти (в стеке) содержимого внутренних регистров процессора;
- Копирование содержимого из одной области памяти в другую область памяти (одиночные и строчные);
- Запись в устройства ввода/вывода и чтение из устройств ввода/вывода (одиночные и строчные);
- Обмен информацией между двумя регистрами или между регистром и памятью;
- Обмен информацией между байтами регистра или памяти.

Арифметические команды

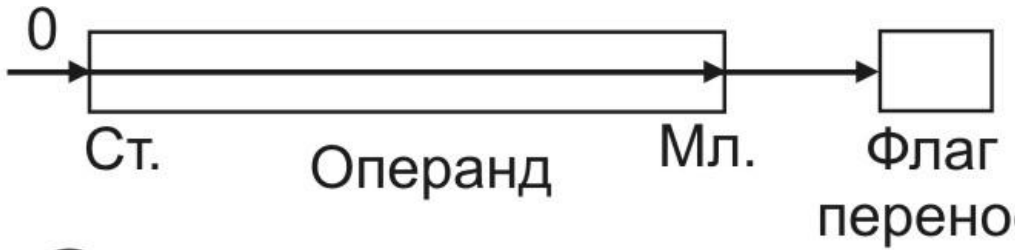
- Команды операций с фиксированной запятой (сложение, вычитание, умножение, деление) — числа как со знаком, так и без знака;
- Команды операций с плавающей запятой (сложение, вычитание, умножение, деление) — операнды в двух или более ячейках памяти, в сложных процессорах: тригонометрические, логарифмические, мультимедийные и т.д.;
- Команды очистки (выполняются быстрее команд пересылок, иногда считаются логическими командами);
- Команды инкремента и декремента — увеличение на 1 или уменьшение на 1;
- Команда сравнения — формирует флаги результата на основании сравнения (вычитания) операндов.

Логические (побитовые) команды

- Логическое И, логическое ИЛИ, сложение по модулю 2 (Исключающее ИЛИ) — маскирование битов в 0 или 1, побитная инверсия по маске;
- Логические, арифметические и циклические сдвиги — вправо или влево с разными значениями вдвигаемых битов;
- Проверка битов и операндов — устанавливает флаги состояния на основании проверки (на нуль, на знак);
- Установка и очистка битов (флагов) регистра состояния процессора (PSW) — для принудительного перевода процессора в тот или иной режим.

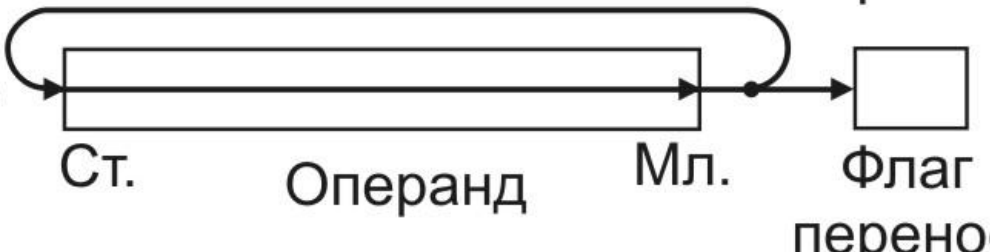
Выполнение сдвигов (вправо)

Логический сдвиг вправо



Арифметический сдвиг вправо

Циклический сдвиг вправо

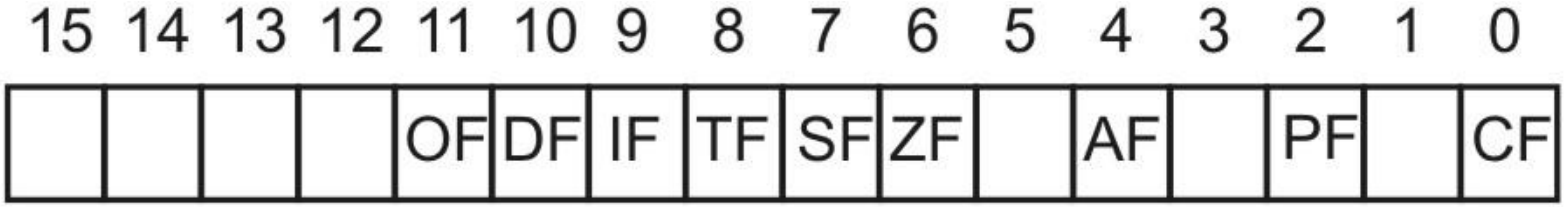


Циклический сдвиг вправо через перенос

Команды переходов

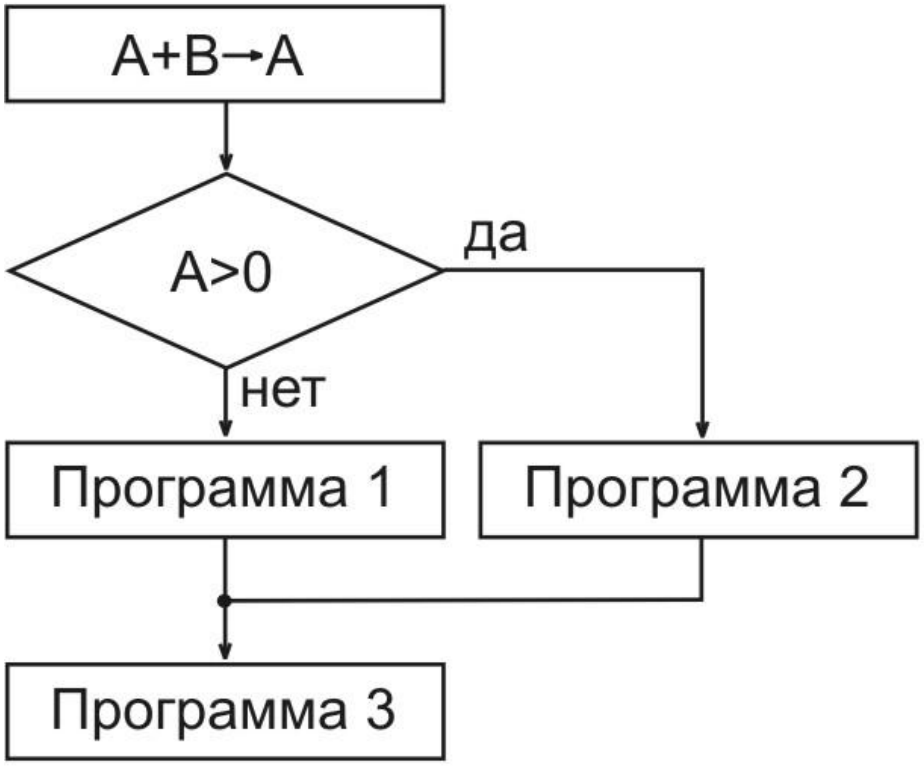
- Команды безусловных переходов (независимо ни от чего);
- Команды переходов с возвратом в исходную точку;
- Команды условных переходов (в зависимости от значений флагов регистра состояния процессора):
  - Переход, если равно нулю;
  - Переход, если не равно нулю;
  - Переход, если есть переполнение;
  - Переход, если нет переполнения;
  - Переход, если больше нуля;
  - Переход, если меньше или равно нулю и т.д.;
- Для проверки условий перехода можно применять команду сравнения, но флаги устанавливаются и любой другой командой, кроме команд переходов.

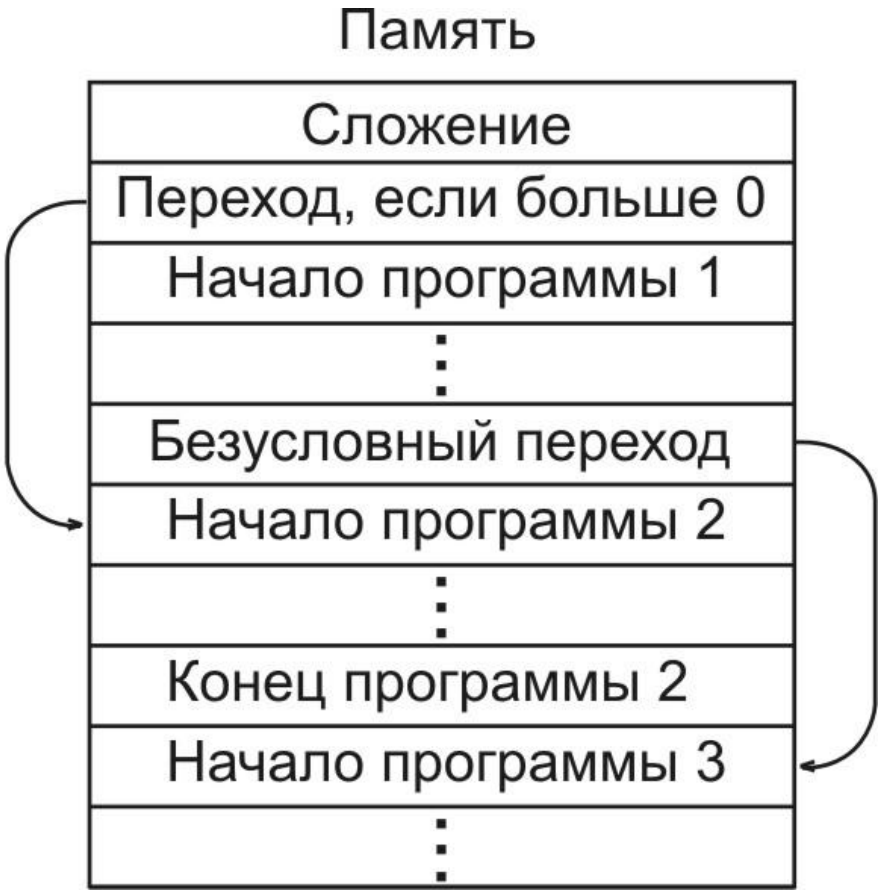
Регистр состояния (FLAGS) процессора Intel 8086



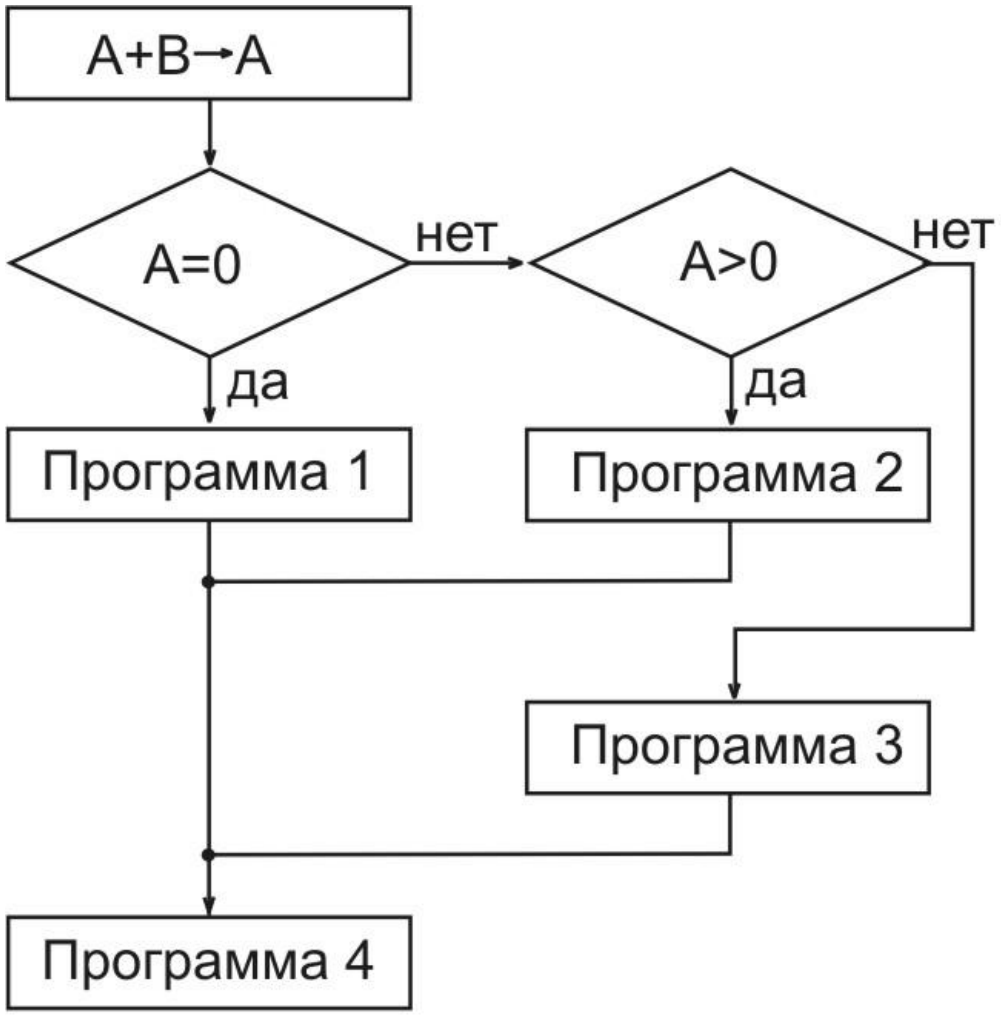
- **CF** — флаг переноса при арифметических операциях,
- **PF** — флаг четности результата,
- **AF** — флаг дополнительного переноса,
- **ZF** — флаг нулевого результата,
- **SF** — флаг знака (старший бит результата),
- **TF** — флаг пошагового режима (для отладки),
- **IF** — флаг разрешения аппаратных прерываний,
- **DF** — флаг направления при строковых операциях,
- **OF** — флаг переполнения.

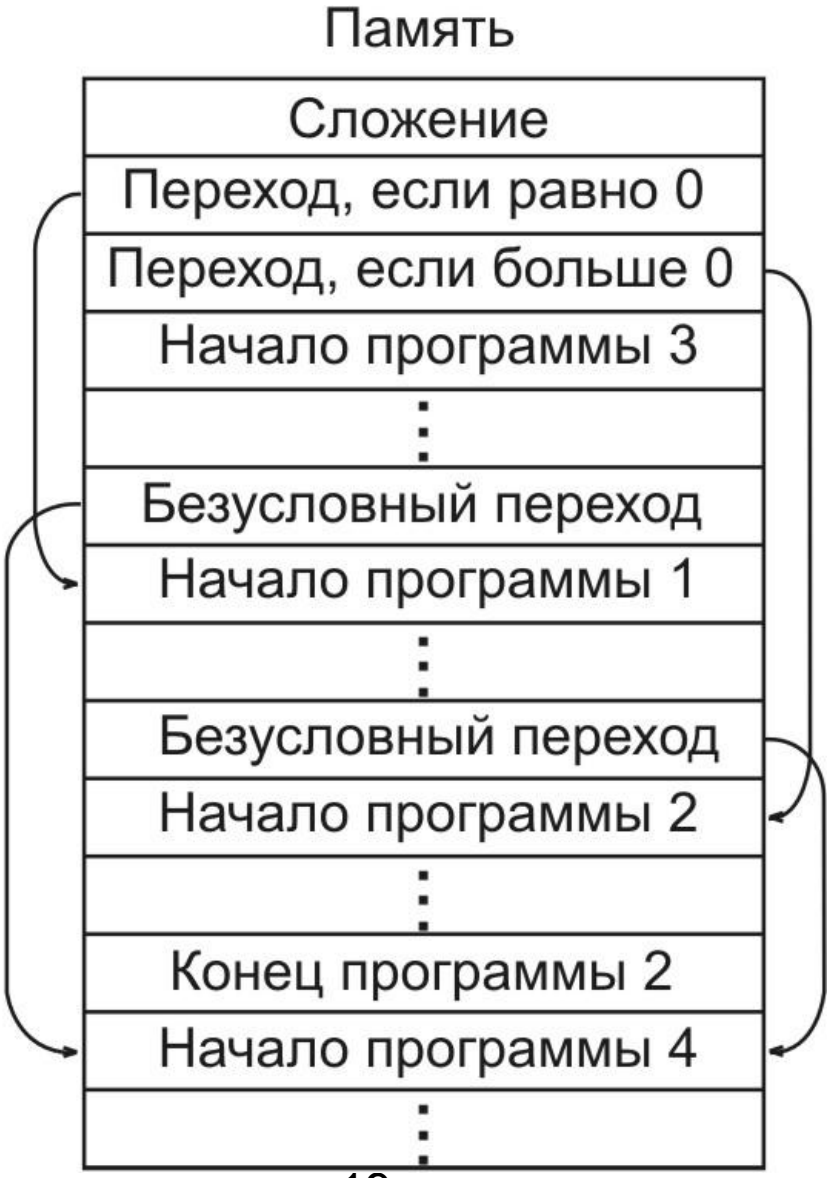
Реализация разветвления на две ветки





Реализация разветвления на три ветки





Команды перехода с возвратом в исходную точку (прерывания)

- Используются для вызова часто выполняемых подпрограмм;
- Обслуживаются по механизму прерываний (сохранение в стеке параметров возврата);
- Требуют задания входного числа — смещения в памяти для адреса начала подпрограммы или номера прерывания (номера элемента в таблице векторов прерываний) — программные прерывания;
- Для возврата в исходную точку используется специальная команда (безусловный переход) в конце подпрограммы, которая извлекает из стека параметры возврата.
- Упрощают написание программ, но замедляет их исполнение.

Методы увеличения быстродействия программ

- Использование вставок на языках низкого уровня в наиболее критичных местах;
- Минимизация количества команд в программе — рациональное построение алгоритма;
- Использование команд с минимальным временем исполнения (например, очистка, инкремент, декремент);
- Минимизация количества команд переходов;
- Минимизация количества подпрограмм, вызываемых по механизму прерываний — только самые необходимые и часто используемые;
- Оптимизация обращений к устройствам ввода/вывода;
- Рациональное расположение данных в памяти.