ЛАБОРАТОРНАЯ РАБОТА № 4
"Считывание, декодирование и выполнение команд. Способ адресации операндов в командах"

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ
1.1. Классификация команд в ВМ

Алгоритм, написанный пользователем программы, в конечном счете, реализуется в виде машинных команд. Под командой понимают совокупность сведений, представленных в виде двоичных кодов, необходимых процессору для выполнения очередного шага. В коде команды для сведений о типе операции, адресной информации о нахождении обрабатываемых данных, а также для информации о месте хранения результатов выделяются определенные разряды (поля).

Множество реализуемых машиной действий образует её систему команд. Система команд часто определяет области и эффективность применения ВМ. Состав и число команд должны быть ориентированы на стандартный набор операций, используемых пользователем для решения своих задач.

Существует несколько различных классификаций команд в вычислительных машинах (ВМ). В частности команды можно выделить следующие типы команд:

- команды пересылки данных
- команды арифметической и логической обработки
- команды работы со строками
- команды SIMD
- команды преобразования
- команды ввода /вывода
- команды управления системой
- команды управления потоком команд.

Команды пересылки данных.

Наиболее распространённый тип машинных команд. Примеры: передача данных между ЦП и ОП, внутри процессора, внутри памяти.

Команда пересылки данных должна содержать следующую информацию:

- Адреса источника и получателя операндов - адреса ячеек памяти, номера регистров ЦП или информацию о том, что операнды - в стеке
- Длина подлежащих пересылке данных (в байтах или словах), заданная явно либо косвенно
- Способ адресации каждого из операндов (т.е. каким образом должен вычисляться физический адрес операнда).

Команды арифметической и логической обработки.

Обеспечивают арифметическую и логическую обработку информации в различных формах ее представления.

Помимо вычисления результата, формируются признаки (флаги) в АЛУ, характеризующие результат (например, Zero, Negative, Overflow, Carry).

Подробнее этот тип команд будет рассмотрен в лабораторной работе, связанной с АЛУ (арифметико-логическим устройством).

Команды SIMD — Single Instruction Multiple Data.

Эти команды обрабатывают сразу две группы чисел. Операнды обычно представлены в упакованном формате.

Команды работы со строками.

Обычно предусматриваются команды перемещения, сравнения и поиска строк. В большинстве ВМ данные операции имитируются с помощью других команд.

Команды преобразования.

Изменение формата представления данных. Например, перевод из двоичной системы счисления в двоично-десятичную.

Команды ввода/вывода.

Могут быть разделены на 3 типа:

- Команды управления периферийными устройствами (ПУ).
- Команды проверки состояния ПУ.
- Команды ввода /вывода.

Команды управления системой.

Эти команды являются привилегированными командами, выполняются когда ЦП находится в привилегированном состоянии либо выполняет привилегированную программу (обычно ОС).

Только с помощью данных команд могут быть изменено состояние ряда регистров устройства управления.

Команды управления потоком команд.

Вместо инкрементации счётчика команд команда загружает в счётчик адрес следующей команды.

Три типа таких команд:

- безусловные переходы;
- условные переходы (ветвления);
- вызовы процедур и возврат из процедур.

Для команд статического перехода один из вариантов - указание адреса перехода через счетчик команд (PC – program counter или IP – instruction pointer). В случае динамического управления должен быть иной способ указания адреса перехода.

Команды вызова процедуры обеспечивают переход из текущей точки программы к начальной точке процедуры. Команды возврата из процедуры обеспечивают возврат к команде следующей за командой вызова. Команда возврата должна обеспечиваться средствами сохранения текущего состояния счётчика команд и его восстановления.

1.2. Форматы команд

В команде для сведений о типе операции, адресной информации о нахождении обрабатываемых данных, а также для информации о месте хранения резуль-

татов выделяются определенные разряды (поля). В общем случае команду можно разбить на 2 части: операционную и адресную:

Операционная часть	Адресная часть

Типовая команда должна определять:

- подлежащую выполнению операцию;
- адреса исходных данных, над которыми выполняется операция;
- адрес, по которому должен быть помещён результат операции.

Выбор формата команд при создании ВМ влияет на многие характеристики будущей машины.

Необходимо принимать во внимание также следующие факторы :

- общее число различных команд;
- общую длину команды;
- тип полей команды (фиксированной или переменной длины) и их длина;
- простота декодирования;
- адресуемость и способы адресации;
- стоимость оборудования для декодирования и исполнения команды.

Длина команды

Влияет на организацию и ёмкость памяти, структуру шин, сложность и быстродействие ЦП.

Чем длиннее команда - тем медленнее её выборка. Длина должна быть кратна разрядности шины данных.

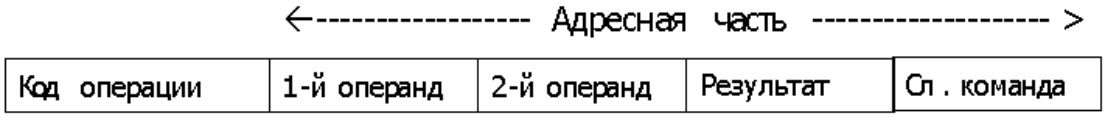
Различные способы адресации приводят к общему виду команды:

КОП – Код операции	СА – Способ адресации	Адресная часть

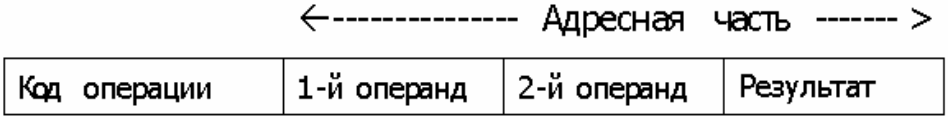
В большинстве ВМ одновременно используются несколько различных форматов команд.

Классификация по количеству адресов в команде:

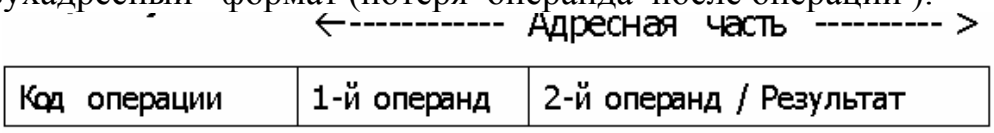
- Четырёхадресный формат:



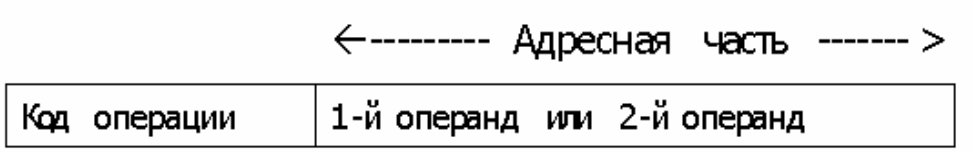
- Трёхадресный формат - добавляются команды переходов:



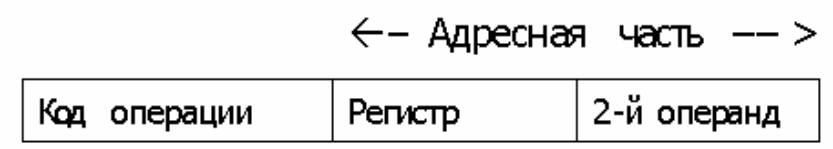
- Двухадресный формат (потеря операнда после операции):



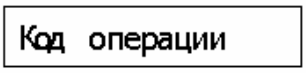
- Одноадресный формат (аккумуляторная архитектура):



- Полутораадресный или регистровый формат:



Нульадресный формат (отдельные команды):



Критерии выбора адресности команд:

- ёмкость запоминающего устройства;
- время выполнения программы;
- эффективность использования ячеек памяти при хранении программы.

С точки зрения увеличения ёмкости и эффективности использования памяти - наиболее оптимальным вариантом будут одноадресные команды.

Суммарные потери времени для трёхадресной команды складываются из:

- времени на выборку команды;
- времени на выборку первого операнда;
- времени на выборку второго операнда;
- времени на запись результата в память.

Для одноадресной команды составляющие потерь по времени следующие:

- время на выборку команды;
- время на выборку операнда.

Определяющим при выборе является тип алгоритмов, на которые ориентирована ВМ. В последовательных программах - результат предыдущей операции используется для следующей, и выгодна одноадресная команда. В параллельных программах - результат пересылается в память, и трёхадресные команды будут эффективнее. В комбинированных рограммах лучше подойдут одноадресные и полутороадресные команды.

Оттранслированные команды записываются в соседние ячейки памяти (памяти программ, если память программ и данных разделены) в порядке их следования в программе. При естественном порядке выполнения команд в программе, адрес каждой следующей команды определяется по содержимому специального счетчика команд (IP или PC), который входит в состав процессора. Содержимое этого счетчика автоматически наращивается на 1 при выполнении очередной команды. При организации ветвления цикла или для перехода на подпрограмму в счетчик в счетчик команд принудительно записывается адрес перехода, указанный в ходе команды.

1.3. Способ адресации операндов

Исполнительный адрес операнда (Аисп) - двоичный код номера ячейки памяти, служащей источником /приёмником операнда (адрес на ША или номер регистра).

Адресный код команды (Ак) - двоичный код в адресном поле команды, из которого необходимо сформировать исполнительный адрес операнда.

В современных ВМ как правило Аисп и Ак не совпадают, требуется соответствующее преобразование. Поэтому способ адресации можно определить, как способ формирования исполнительного адреса по адресному коду команды.

Существуют 2 различных принципа поиска операндов в памяти: ассоциативный и адресный.

Ассоциативный поиск (поиск по содержанию запоминающей ячейки) предполагает просмотр содержимого всех ячеек памяти для выявления кода, содержащего заданный командой ассоциативный признак.

Адресный поиск предполагает, что операнд находится по адресу, указанному в адресном поле команд.

Способы адресации операндов можно разбить на несколько групп.

1. по наличию адресной информации в команде (явная и неявная адресация).
2. по кратности обращения в оперативную память.
3. по способу формирования адресов ячеек памяти.

По способу формирования адресов ячеек памяти различают:

- Адресация с абсолютным способом формирования адресов ячеек памяти (двоичный код адреса ячейки памяти может быть целиком извлечен либо из адресного поля команды, либо из какой-нибудь другой ячейки в случае косвенной адресации);
- Адресация с относительным способом формирования адресов ячеек памяти (двоичный код адресной ячейки памяти образуется из нескольких составляющих).

По кратности обращения в оперативную память различают:

- непосредственную адресацию (direct addressing)
- прямую адресацию (immediate addressing)
- косвенную адресацию (indirect addressing)

Рассмотрим эти способы адресации и их различные варианты более подробно.

1.Непосредственная адресация (Immediate Addressing)

При непосредственной адресации операнд располагается в адресном поле команды. Используется в операциях сравнения, загрузки констант в регистры, арифметических операциях.

КОП	Операнд

Пример: mov AX,300

2.Прямая адресация (Direct Addressing)

При прямой адресации обращение за операндом производится по адресному коду в поле команды. При этом исполнительный адрес совпадает с адресным кодом команды.

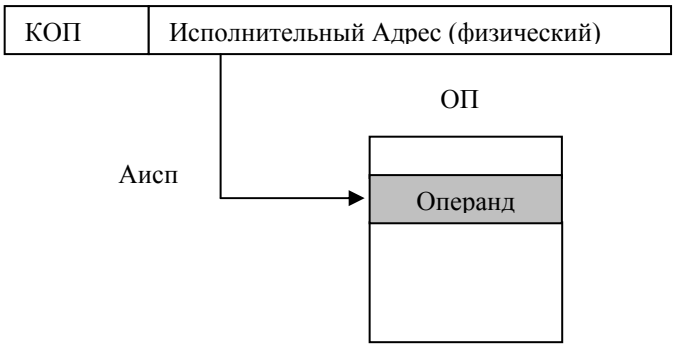


Рис. 1.1. Прямая адресация

3.Косвенная адресация (Indirect Addressing)

При косвенной адресации код команды указывает адрес ячейки памяти, в которой находится не сам операнд, а его исполнительный адрес.

Косвенная адресация памяти может быть многоуровневой с глубиной 2 и более, но с увеличением глубины после 10 увеличивается количество обращений за следующими адресами памяти. Косвенная адресация используется для организации цепных списков.

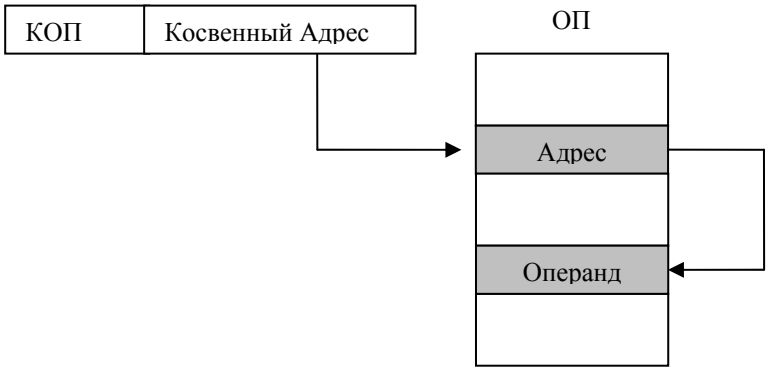


Рис. 1.2. Косвенная адресация

Плюсом этого способа является возможность изменения адреса операнда в процессе выполнения программы.

4.Прямая регистровая адресация (Register Direct Addressing)

Кроме оперативной памяти (ОП) операнды могут находиться в регистрах общего назначения (РОН). При прямой регистровой адресации в регистрах содержится сам операнд/операнды.

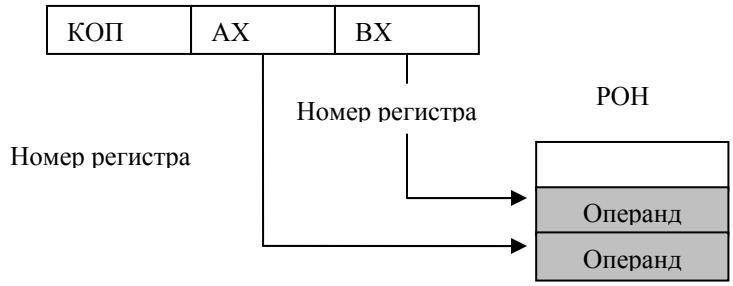


Рис. 1.3. Прямая регистровая адресация

Плюс прямой регистровой адресации – небольшая длина адресной части при условии небольшого числа РОНов.

5.Косвенная регистровая адресация (Register Indirect Addressing)

При косвенной регистровой адресации в регистрах содержится адрес операнда/операндов.

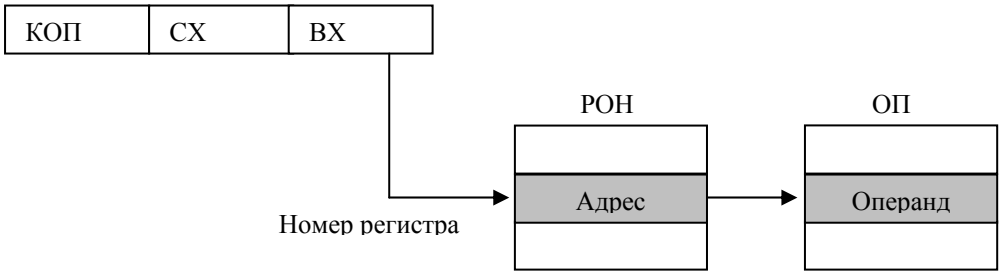


Рис. 1.4. Регистровая косвенная адресация

Адресация с относительным способом формирования адресов ячеек памяти предполагает, что двоичный код адресной ячейки памяти (исполнительный адрес) образуется из нескольких составляющих:

- Базы,
- Индекса,
- Смещения.

Аисп = [Базовый регистр] + [Индексный регистр] + Смещение.

Эти составляющие используются в различных сочетаниях.

- Аисп = [Базовый регистр] + Смещение – базовая адресация со смещением (рис. 6.4).
- Аисп = [Индексный регистр] + Смещение – индексная адресация со смещением (рис. 6.5).
- Аисп = [BX] + [SI] + Смещение – базово-индексная адресация со смещением (рис 6.6).
- Автоинкрементная индексная адресация: когда адрес, находящийся в индексном регистре, автоматически увеличивается (автоинкрементная адресация), или уменьшается (автодекрементная адресация) на постоянную величину до или после выполнения операции.
- Пост-автоинкрементная адресация: индексный регистр увеличивается после выполнения команды.

– Пред-автоинкрементная адресация: индексный регистр увеличивается до выполнения команды.

6. Адресация со смещением (Displacement)

Исполнительный адрес формируется суммированием адресного поля команды с содержимым одного или нескольких РОНов или специализированных регистров (базового или индексного).

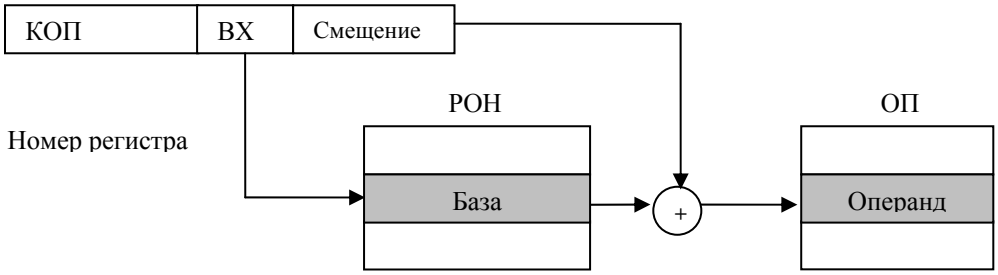


Рис. 1.5. Базовая адресация со смещением

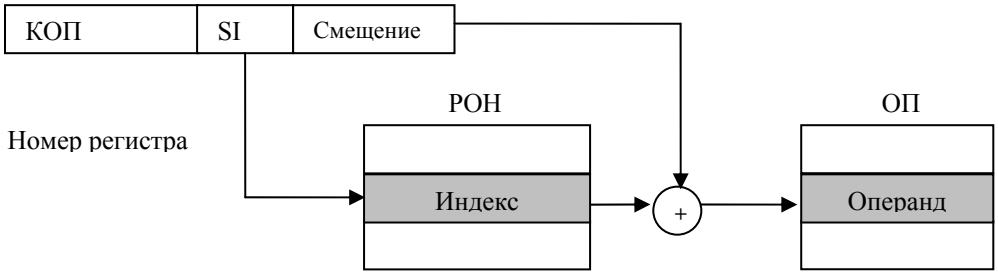


Рис. 1.6. Индексная адресация со смещением

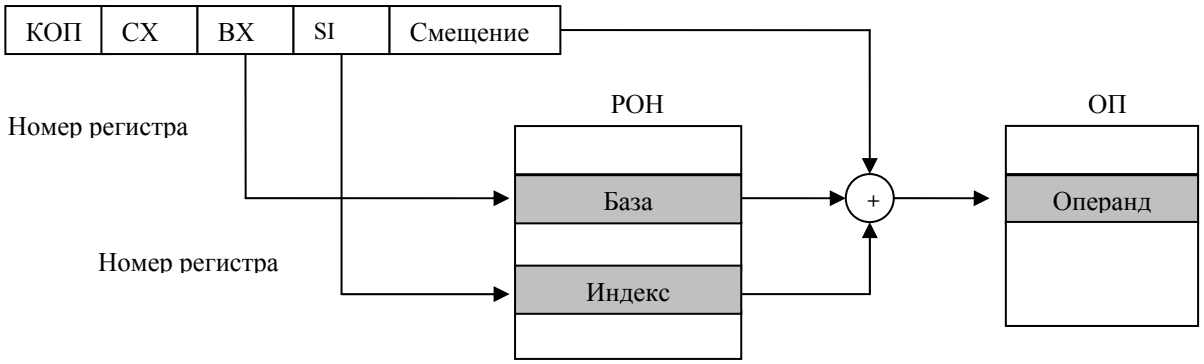


Рис. 1.7. Базово-индексная адресация со смещением

7.Относительная адресация со смещением

Для получения исполнительного адреса содержимое адресного поля складывается со счетчиком команд (IP или PC).

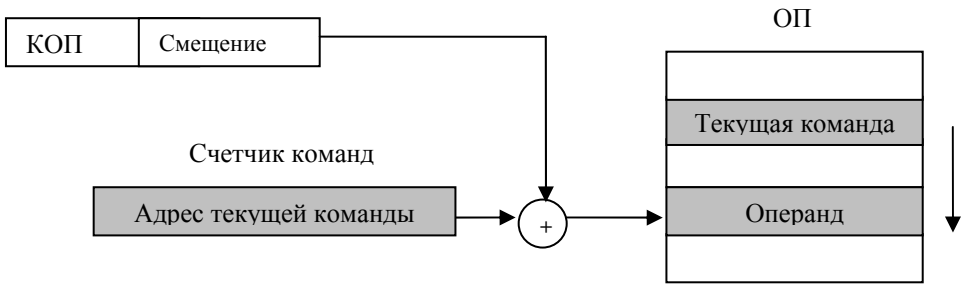


Рис. 1.8. Относительная адресация со смещением

8.Неявная адресация

При неявной адресации адресное поле в команде отсутствует, а адрес операнда подразумевается кодом операции. Например, из команды может быть исключен адрес приемника адресата, при этом подразумевается, что результат записывается на месте второго операнда.

Примеры: MUL BX
DIV BX

9.Блочная адресация

Используется в командах, обрабатывающих блок данных. Блок обычно описывается адресом первой или последней ячейки и количеством элементов в блоке. Иногда в качестве конца блока удобно использовать некоторый признак.

10.Стековая адресация

Для чтения записи доступен только один регистр - вершина стека. Этот способ адресации используется, в частности, системой прерывания программ при вложенных вызовах подпрограмм.

Стековая память реализуется на основе обычной памяти с использованием указателя стека и автоиндексной адресации.

Запись в стек производится с использованием автодекрементной адресации, а чтение - с использованием автоинкрементной адресации.

10.Страничная адресация

Проводится разбиение адресного пространства на страницы. Начальный адрес страницы - база. Старшая часть адреса страницы - в специальном регистре (РАС). Адресный код - смещение внутри страницы.

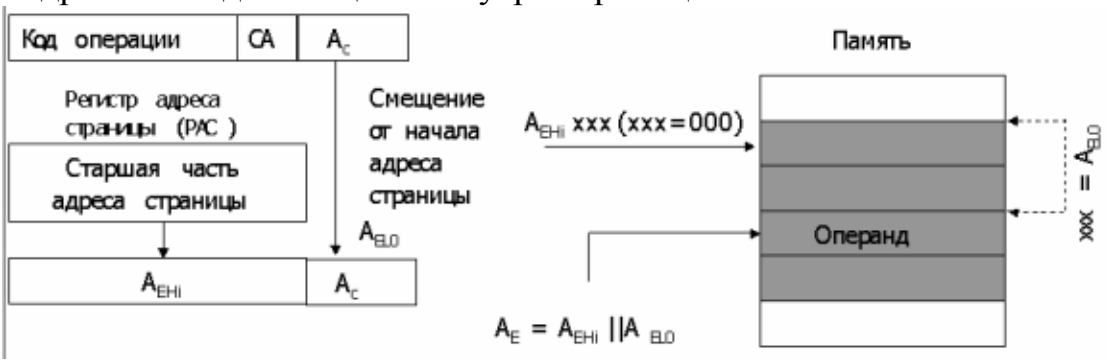


Рис. 1.9. Страничная адресация

2. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

В процессе работы над лабораторной работой можно использовать наработки с предыдущих лабораторных работ.

На выходе должна получится система, состоящая из блока памяти, блока управления, блока РОНов и шин данных, адреса и управления.

Для выполнения задания необходимо выполнить следующие этапы:

1. Ввести шину данных (ШД), шину адреса (ША) и шину управления (ШУ).

2. Разделить память на память данных (блок RAM) и память команд (блок ROM). На адресные входы памяти завести ША. Ввод и вывод данных осуществлять через ШД. Объединить два типа памяти в один блок.

3. Ввести блок управления, в котором производить дешифрацию команд и выдавать управляющие сигналы и сигналы синхронизации памяти и регистров в зависимости от выполняемой команды.

4. Ввести блок регистров общего назначения (РОН) и управляющую логику для него.

5. Ввести специальные регистры, разрядность которых определяется разрядностью шины данных. Физически разместить их в блоке управления.

Специальные регистры:

1) **Счетчик команды IP** (instruction Pointer), который служит для фиксации и формирования адресов команд, располагающихся в ячейках памяти. Необходимо обеспечить автоинкремент этого регистра.

2) **Регистр команд IR** (Instruction Register), который предназначен для хранения кода команды или его части в процессе выполнения команды после ее приема из памяти.

3) **Регистр адреса операнда AR** (Address Register), в который записывается исполнительный адрес операнда.

4) **Регистр данных DR** (Data Register), в который помещаются непосредственно сами операнды.

5) **Регистр признаков результата или регистр флагов FR** (Flag Register), в котором необходимо задать биты для переполнения, признака нуля, признака знака, переноса.

Интерфейс с внешней средой у верхнего блока иерархии должен ограничиваться старт/стоповым сигналом и сигналом внешнего тактирования **clk.**

Выполняемую команду можно разбить на несколько этапов. Для разделения этих этапов во времени можно использовать блок опорных сигналов **DCa** или блок генерации сигнала специальной формы, разработанные на лабораторной работе 2.

Необходимо выполнить цикл заданных по варианту команд. Цикл состоит из следующих фаз:

- выборки команды и формирования адреса следующей команды;
- декодирования команды;
- формирования исполнительных адресов операндов;
- выборки операндов;

- исполнения операции;
- записи результата.

Некоторые шаги цикла команд могут отсутствовать, это зависит от выполняемой команды.

Если также при анализе команды вы приходите к выводу, что вам не нужны некоторые специальные регистры в виду отсутствия действия для этого регистра, то в операции вы их не используете, но сама их реализация должна присутствовать.

Семейство ПЛИС для реализации - Flex10K (изменить семейство ПЛИС(Family) можно в настройках проекта в ветке Device или в пункте меню Assignments->Device).

ВАРИАНТЫ ЗАДАНИЙ К ЛАБОРАТОРНОЙ РАБОТЕ № 4

№	Разр. ША	Разр. ШД	Команда 1	Адресация операндов команды 1	Команда 2	Адресация операндов команды 2
1	8	8	R->M	Прямая регистровая	JMP adr	Непосредственная
2	4	8	M->R	Прямая	JZ adr	Относительная со смещением
3	8	4	const->R	Непосредственная	JS adr	Относительная со смещением
4	4	4	const->M	Неопсредственная	JC adr	Непосредственная
5	8	8	M->R	Косвенная	JMP adr	Относительная со смещением
6	4	8	R->M	Неявная	JZ adr	Непосредственная
7	8	4	R+C->M	Со смещением	JS adr	Непосредственная
8	4	4	R->M	Косвенная регистровая	JC adr	Относительная со смещением

СОДЕРЖАНИЕ ОТЧЕТА ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

1. Задание.
2.1. Схема блока, самого верхнего в иерархии проекта.
2.2. Схемы входящих в него блоков
3. Результаты моделирования.