

## Виртуальные машины.

### *Требования к реализации*

0. Реализовать интерпретатор виртуальной машины как консольное приложение.

1. Объектно-ориентированный подход.

1.1 Процессор – отдельный класс, разделение на интерфейс и реализацию;

Выполнение команд – вызов функтора (не использовать оператор-переключатель!)

1.2 Память – поле в классе. Память – отдельный независимый класс; сама память – динамический массив;

1.3 Команды процессора – иерархия классов-функторов; базовый класс – абстрактный класс Command с перегруженной операцией operator()

2. Разработать и описать в пояснительной записке:

– коды команд

– методы адресации операндов

– множество флагов-результатов

– форматы команд

– форматы операндов

Обязательно должны быть реализованы команды:

2.1 Пересылки/загрузки/сохранения

2.2 Арифметика (целые и дробные)

2.3 Сравнения (целые и дробные)

2.4 Битовые операции

2.5 Ввод-вывод (целые и дробные)

2.6 Переходы

– безусловный

– условный

– к подпрограмме

– возврат из подпрограммы

Допускается разработать и реализовать любые другие команды.

3. Загрузчик программ для виртуальной машины – независимая функция, вызываемая в функции main() интерпретатора.

3.1 Код программы для виртуальной машины должен быть записан в текстовом файле.

3.2 Формат представления кода программы в файле должен быть разработан и описан в пояснительной записке.

3.3 Загрузчик должен быть способен загружать код программы по любому адресу памяти

3.3 Главная программа должна получать имя файла как параметр командной строки, и передавать загрузчику как параметр при вызове.

Название темы курсового проекта:

**Реализация виртуальной машины VMxx**

xx – номер варианта.

## 1. Одноадресная, с сумматором и адресным регистром

PSW – 32 бита = 16 + 16 = IP + Flags

Память – байтовая, размер адреса = 16 бит.

**сумматор** – 4 байта

**адресный регистр** – 2 байта

Типы данных:

Целые знаковые – 4 байта

Дробные – 4 байта

Структура команды, 24 бита:

Код операции – 7 бит, b – 1 бит

b = 0 – адрес (абсолютная адресация)

b = 1 – адрес + регистр (индексная/базовая)

Адрес – 16 бит

Загрузка адресного регистра: схема та же

b = 0 – адрес (константа) в команде (константа = адрес)

b = 1 – регистр + константа в команде

Арифметика в адресном регистре, сохранение адресного регистра

Загрузка-сохранение сумматора

Арифметика дробная на сумматоре

Арифметика целая знаковая на сумматоре

Переходы:

Безусловный прямой: IP = адрес (константа в команде);

бит b работает по схеме загрузки адресного регистра:

b = 0 – адрес (константа) в команде (константа = адрес)

b = 1 – регистр + константа в команде

Если адрес = 0, то это косвенный переход по адресному регистру

Условный – то же самое, проверяет флаги;

Вызов подпрограммы, адрес возврата сохраняется в адресном регистре

Возврат – безусловный переход прямой: b = 1, регистр + 0

## 2. Одноадресная, стековая с адресным регистром

PSW – 32 бита = 16 + 16 = IP + SP + Flags

Память – байтовая, размер адреса = 16 бит.

**Стек** вычислений – 16 ячеек по 4 байта – в процессоре  
**адресный регистр** – 2 байта

Типы данных:

Целые знаковые – 4 байта

Дробные – 4 байта

Структура команды, 24 бита:

Код операции – 7 бит, b – 1 бит

b = 0 – адрес (абсолютная адресация)

b = 1 – адрес + регистр (индексная или базовая)

Адрес – 16 бит

Безадресные команды занимают 1 байт = 8 бит

Загрузка адресного регистра: схема та же

b = 0 – целая константа в команде (константа = адрес)

b = 1 – регистр + константа в команде

Арифметика в адресном регистре, сохранение адресного регистра

Загрузка стека, копирование вершины стека, извлечение из стека.

Арифметика дробная в стеке, *безадресная*, сохранение в стеке

Арифметика целая знаковая в стеке, *безадресная*, сохранение в стеке

Переходы:

Безусловный прямой: IP = адрес (константа в команде);

бит b работает по схеме загрузки адресного регистра:

b = 0 – адрес (константа) в команде (константа = адрес)

b = 1 – регистр + константа в команде

Если адрес = 0, то это косвенный переход по адресному регистру

Условный – то же самое, проверяет флаги;

Вызов подпрограммы, адрес возврата сохраняется в адресном регистре

Возврат – безусловный переход прямой: b = 1, регистр + 0

### 3. Двухадресная, без Регистров Общего Назначения, с адресными регистрами

$PSW = IP + Flags = 16 + 16 = 32$  бит.

Память – байтовая; размер адреса – 16 бит.

**2 адресных регистра** – 16 бит

Типы данных:

Целые знаковые – 4 байта

Дробные – 4 байта

Структура команды: 3 байта (24 бита), 4 байта (32 бита); результат по первому адресу.

32 бита: КОП – 6 бит, bb – 2 бита, смещение1 – 12 бит, смещение2 – 12 бит  
bb – смещение + регистр

Загрузка адресных регистров:

24 бита: КОП – 6 бит, bb – 2 бита, адрес (константа) – 16 бит  
bb = 00 – непосредственная константа = адрес  
bb = 01 – первый – адрес, второй = смещение + регистр  
bb = 10 – первый = смещение + регистр, второй – адрес  
bb = 11 – оба = смещение + регистр

Пересылки

Арифметика в адресном регистре, сохранение адресного регистра

Арифметика дробная, арифметика целая

Переходы:

Безусловный – 1 слово = 16 бит (использование bb отличается от стандартного):

КОП – 6 битов, bb – 2 бита, смещение (со знаком) – 8 бит

bb = 00 – относительный по смещению:  $IP = IP + \text{смещение}$  (не далее, как на 127-128 слов)

bb = 10 – прямой по первому регистру:  $IP = a1 + \text{смещение}$

bb = 01 – прямой по второму регистру:  $IP = a2 + \text{смещение}$

bb = 11 – запрещено (резерв)!

Условный – аналогично с проверкой флагов

Вызов подпрограммы – 32 бит:

обычная форма команды с использованием bb;

первый адрес – переход, второй – запоминание адреса возврата

Возврат – безусловный прямой переход по второму регистру:  $a2 + 0$

#### 4. Двухадресная, с Регистрами Общего Назначения, универсальная адресация операндов

PSW = IP + Flags, 16+16 = 32 бит.

Память – слова 16 бит, объединяются в двойные слова. Размер адреса – 16 бит.

Данные:

Целые знаковые, беззнаковые – 1 слово

Целые знаковые – 2 слова

Дробные – 2 слова

РОН – 8 штук, 16 бит; содержимое – целое со знаком, целое без знака (адрес)

Они же – 4 регистра, 32 бит, номера – четные; содержимое – целое со знаком

Они же – 4 регистра с плавающей точкой по 32 бита, номера – четные.

Структура команд: 1 слово, 2 слова, 3 слова; результат по второму адресу

КОП – 7 бит, *s* – 1 бит, *dd* – 2 бита, *r1* – 3 бита, *r2* – 3 бита, *o1* – 16 бит, *o2* – 16 бит

*s* – **размер операнда (для целых)**

*s* = 0 – 1 слово

*s* = 1 – 2 слова

*dd* – **формат операнда (первый и второй)**

*dd* = 00 – операнды в регистрах; *o1*, *o2* – отсутствуют;

*dd* = 01 – регистр – адрес

*dd* = 10 – адрес – регистр

*dd* = 11 – адрес – адрес

Адрес = регистр + смещение (если регистр используется)

Адрес = смещение (если регистр не используется)

Особые случаи: *s*=1, *d* = 0 (аргумент в регистре) и номер регистра – нечетный

Это формат пересылок, арифметики целой, арифметики дробной;

Пересылка:

Регистр-регистр, Регистр-память, Память-регистр, Память-память

арифметика дробная, арифметика целая – то же самое

Переходы:

Безусловный (использование битов *s*, *dd* – отличается от стандартного)

– *s* = 0: прямой, IP = адрес

– *s* = 1: относительный, IP = IP + адрес

*dd* = 00: адрес = *r1* – это косвенный переход

*dd* = 10: адрес = *r2* – это косвенный переход

*dd* = 11: адрес = *r2*+*o2*

*dd* = 01: адрес = *o2*

Условный — то же самое, только проверяются флаги

Вызов подпрограммы — адрес возврата запоминается в *r1*

Возврат – безусловный прямой переход (по любому варианту)

Если регистр в команде не используется, то можно использовать для дополнительных кодов операций

## 5. Двухадресная, с Регистрами Общего Назначения, один операнд в регистре

PSW = IP + Flags, 16+16 = 32 бит.

Память – байтовая, объединяются в 2 и 4 байта. Размер адреса – 16 бит.

Данные:

Короткие целые знаковые, беззнаковые – 2 байта

Длинные целые знаковые, беззнаковые – 4 байта

Дробные – 4 байта

РОН – 8 штук, 16 бит; содержимое – короткое целое со знаком, короткое целое без знака (адрес)

Они же – 4 регистра для длинных целых по 32 бита, номера – четные

Они же – 4 регистра с плавающей точкой по 32 бита, номера – четные.

Структура команд: 2 байта, 4 байта; результат по первому

16 бит: КОП – 7 бит,  $s - 1$  бит, r1 – 4 бита, r2 – 4 бита

32 бита: КОП – 7 бит,  $s - 1$  бит, r1 – 4 бита, r2 – 4 бита, off – 16 бит

$s$  – размер целого операнда

$s = 0$  – 2 байта

$s = 1$  – 4 байта

Адрес = регистр + смещение (если регистр используется)

Адрес = смещение (если регистр не используется)

Смещение = адрес = константа в команде

Особые случаи:  $s=1$ , номер регистра – нечетный

Наличие или отсутствие смещения определяется кодом операции.

Это формат пересылок, арифметики целой, арифметики дробной;

Пересылка:

Регистр-регистр, Регистр-память, Память-регистр

арифметика дробная, арифметика целая – то же самое

Переходы: формат 16 бит

Безусловный (использование бита  $s$  – нестандартное)

–  $s = 0$ : прямой, IP = адрес

–  $s = 1$ : относительный, IP = IP + адрес

Безусловный косвенный.

Условный — то же самое, только проверяются флаги

Вызов подпрограммы: адрес возврата запоминается в r1

Возврат – безусловный прямой переход

Если регистр в команде не используется, то можно использовать для дополнительных кодов операций!

## 6. Трехадресная, с регистрами общего назначения, все операнды в регистрах

PSW = IP + Flags, 16 + 16 = 32 бит

Память: слова по 16 бит, размер адреса – 16 бит

Типы данных:

Целые знаковые, беззнаковые – 2 слова

Дробные – 2 слова

РОН – 8 штук, 32 бит – целые

Они же 8 штук, 32 бита – дробные

Структура команды: 1 слово, 2 слова; результат в первом операнде

16 бит: КОП – 7 бит, r1 – 3 бита, r2 – 3 бита, r3 – 3 бита

32 бита: КОП – 7 бит, r1 – 3 бита, r2 – 3 бита, r3 – 3 бита, адрес (константа) – 16 бит

Пересылка:

Память-регистр, регистр-память, регистр-регистр;

r2 = xxx – биты пересылки

000 – регистр-регистр

001 – регистр-память, адрес (константа) в команде

011 – регистр-память, адрес = r3+константа (в команде)

101 – память-регистр, адрес (константа) в команде

111 – память-регистр, адрес = r3+константа (в команде)

010 – загрузка адреса в регистр, адрес (константа) в команде

110 – загрузка адреса в регистр, адрес = r3+константа (в команде)

100 – резерв

Арифметика целая – только на регистрах

Арифметика дробная – только на регистрах

r1 = r2 \$ r3

Переходы:

Безусловный прямой, r1 – не используется

r2 = xxx – тип перехода

001: прямой, IP = адрес (константа) в команде

011: прямой, IP = r3+адрес (константа) в команде

Безусловный относительный: команда формата-16: смещение = <r1r2r3>; IP = IP+смещение

Безусловный косвенный.

Условный — то же самое, только проверяются флаги

Вызов подпрограммы — адрес возврата запоминается в r1

Возврат – безусловный прямой переход

Если регистр в команде не используется, то можно использовать для дополнительных кодов операций!

## 7. Двухадресная, с Регистрами Общего Назначения, операнды в регистрах

PSW = IP + Flags, 16 + 16 = 32 бит

Память: слова 32 бита, размер адреса – 16 бит

Типы данных:

Целые знаковые, беззнаковые – 1 слово

Дробные – 1 слово

РОН – 32 штуки, 32 бита – целые

Они же, 32 бита – дробные

Структура команды: короткие 16 бит (две команды в слове), длинные 32 бит

16 бит: КОП – 6 бит, r1 – 5 бит, r2 – 5 бит

32 бита: КОП – 6 бит, r1 – 5 бит, r2 – 5 бит, адрес (константа) – 16 бит

Пересылка регистр-регистр

Пересылка память-регистр, регистр-память – r2 не используется

Загрузка константы-адреса в регистр – r2 не используется

Арифметика целая – только на регистрах

Арифметика дробная – только на регистрах

Результат – в r2

Переходы

Безусловный прямой: команда формата-32

r2 = xxxxx – тип перехода

00001: прямой, IP = адрес (константа) в команде

00011: прямой, IP = r1+адрес (константа) в команде

Безусловный относительный: команда формата-16; смещение = <r1r2>; IP = IP+смещение

Безусловный косвенный прямой: команда формата-16; IP = r1

Условный — то же самое, только проверяются флаги

Вызов подпрограммы — адрес возврата запоминается в r2

Возврат – безусловный прямой переход

Если регистр в команде не используется, то можно использовать для дополнительных кодов операций!



## 8. Смешанная, с адресными регистрами и стеком

PSW = IP + Flags, 16 + 16 = 32 бита

Память: байтовая, размер адреса – 16 бит

Типы данных:

Целые знаковые, беззнаковые – 2, 4 байта

Дробные – 4 байт

Адресные регистры – 8 штук, 16 бит

A7 = SP – указатель стека; стек – в любом месте памяти

Структура команды: 1 байт, 2 байта, 3 байта, 4 байта

16 бит: КОП – 7 бит, a1 – 3 бита, a2 – 3 бита, a3 – 3 бита

24 бита: КОП – 7 бит, b – 1 бит, адрес (константа) – 16 бит

32 бита: КОП – 7 бит, a1 – 3 бита, a2 – 3 бита, a3 – 3 бита, адрес (константа) – 16 бит

Загрузка адресного регистра: формат-16, формат-32

КОП – 7 бит, m – 3 бита, a2 – 3 бита, a3 – 3 бита, адрес (константа) – 16 бит

m – xxx: тип адреса

001: a2 = a3, формат-16

010: a2 = адрес, формат-32; a3 – резерв

011: a2 = a3 + адрес

Арифметика целая короткая (адресная – целая беззнаковая)

– только в адресных регистрах: формат-16, трехадресная

Арифметика целая длинная – только в памяти;

– адреса в адресных регистрах; формат-16, двухадресная: [a1] = [a1] \$ [a2+a3]

Арифметика дробная – только в стеке; однобайтовые команды

Загрузка-сохранение стека: формат-24; бит b – резерв

Переходы

Безусловный: формат-24

b – тип перехода

0: прямой, IP = адрес (константа) в команде

1: косвенный, IP = [адрес]

Безусловный: формат-16

Относительный: смещение = <r1r2r3>; IP = IP+смещение

Безусловный: формат-32

– прямой, косвенный

b – 1 бит: резерв

m – 2 бита

x = 1 – регистр участвует в формировании адреса

x = 0 – регистр не участвует в формировании адреса

Условный — то же самое, только проверяются флаги

Вызов подпрограммы – адрес возврата запоминается в a1 (в стеке)

Возврат – безусловный прямой переход

## 9. Смешанная, с адресными регистрами

PSW = IP + Flags, 16 + 16 = 32 бита

Память: слова – 32 бита, размер адреса – 16 бит

Типы данных:

Целые знаковые, беззнаковые – 1 слово

Дробные – 1 слово

Адресные регистры – 256 штук, 16 бит

Структура команды: 1 слово

32 бита: КОП – 8 бит, r1 – 8 бит, r2 – 8 бит, r3 – 8 бит

Загрузка адресного регистра: 1 слово

КОП – 8 бит, r1 – 8 бит, адрес (константа) – 16 бит

Арифметика целая короткая (адресная – целая беззнаковая)

– только в адресных регистрах, трехадресная

Арифметика целая длинная – только в памяти;

– адреса в адресных регистрах, трехадресная: [r1] = [r2] \$ [r3]

Переходы

Безусловный:

r1 – тип перехода

0: прямой, IP = адрес (константа) в команде

1: прямой косвенный, IP = [адрес (константа) в команде]

2: прямой косвенный регистровый, IP = r2 + r3

3: относительный, IP = IP+смещение; смещение = константа в команде

Условный — то же самое, только проверяются флаги

Вызов подпрограммы – адрес возврата запоминается в r1

Возврат – безусловный прямой переход

## 10. Смешанная, два операнда в регистрах, один – в памяти

PSW = IP + Flags, 16 + 16 = 32 бита

Память: слова – 32 бита, размер адреса – 16 бит

Типы данных:

Целые знаковые, беззнаковые – 1 слово

Дробные – 1 слово

РОН – 16 штуки, 32 бита – целые

Они же, 32 бита – дробные

Структура команды: 16 бит (2 команды в слове), 32 бита

16 бит : КОП – 8 бит, r1 – 4 бит, r2 – 4 бит

32 бита: КОП – 8 бит, r1 – 4 бит, r2 – 4 бит, адрес (константа) – 16 бит

Результат – в r2.

Пересылка регистр-регистр

Пересылка память-регистр, регистр-память

Загрузка константы-адреса в регистр

Арифметика целая – двухадресная и трехадресная

Арифметика дробная – двухадресная и трехадресная

Переходы:

Безусловный:

– прямой

– относительный

– косвенный прямой

Условный – то самое, только с проверкой флагов

К подпрограмме – адрес возврата в r2

Возврат из подпрограммы

## 11. Смешанная, один операнд в регистрах, один – в памяти

PSW = IP + Flags, 16 + 16 = 32 бита

Память: байтовая, размер адреса – 16 бит

Типы данных:

Целые знаковые, беззнаковые – 4 байта

Дробные – 4 байт

РОН – 16 штуки, 32 бита – целые

Они же, 32 бита – дробные

Структура команды: 2 байта, 3 байта

КОП – 7 бит, d – 1 бит, r1 – 4 бит, r2 – 4 бит, смещение (константа) – 8 бит

d = 0: смещения нет

d = 1: смещение есть

Результат – в r1.

Пересылка регистр-регистр

Пересылка память-регистр, регистр-память

Арифметика целая

Арифметика дробная

Переходы:

Безусловный:

– прямой

– относительный

– косвенный прямой

Условный – то самое, только с проверкой флагов

К подпрограмме – адрес возврата в r1

Возврат из подпрограммы

## 12. Стековая, один операнд в памяти, другой в стеке

PSW = IP + SP+Flags, 16 + 5 + 11 = 32 бита

Стек: 32 слова по 32 бита, указатель стека SP – в PSW

Память: байтовая, размер адреса – 16 бит

Типы данных:

Целые знаковые, беззнаковые – 4 байта

Дробные – 4 байта

Структура команды: 3 байта

24 бит: КОП – 8 бит, смещение (константа) – 16 бит

Результат – в стеке или в памяти

Пересылки/загрузки/сохранения

Арифметика целая

Арифметика дробная

Переходы:

Безусловный:

– прямой

– относительный

– косвенный прямой

Условный – то самое, только с проверкой флагов

К подпрограмме – адрес возврата в стеке

Возврат из подпрограммы – косвенный прямой переход по адресу в стеке

### 13. Трехадресная, два операнда в регистрах, один – в памяти

PSW = IP + Flags, 16 + 16 = 32 бита

Память: байтовая, размер адреса – 16 бит

Типы данных:

Целые знаковые, беззнаковые – 4 байта

Дробные – 4 байт

РОН – 8 штуки, 32 бита – целые

Они же, 32 бита – дробные

Структура команды: 3 байта

КОП – 8 бит, r1 – 3 бит, r2 – 3 бит, смещение (константа) – 10 бит

Результат – в r1.

Пересылка регистр-регистр, память-регистр, регистр-память

Арифметика целая

Арифметика дробная

Переходы:

Безусловный:

– прямой

– относительный

– косвенный прямой

Условный – то самое, только с проверкой флагов

К подпрограмме – адрес возврата в r1

Возврат из подпрограммы

#### **14. Трехадресная, два операнда в регистрах, один – в памяти**

PSW = IP + Flags, 16 + 16 = 32 бита

Память: слова по 32 бита, размер адреса – 16 бит

Типы данных:

Целые знаковые, беззнаковые – 4 байта

Дробные – 4 байт

РОН – 16 штуки, 32 бита – целые

Они же, 32 бита – дробные

Структура команды: 1 слово

32 бита: КОП – 8 бит, r1 – 4 бит, r2 – 4 бит, адрес (константа) – 16 бит

Результат – в памяти.

Пересылка регистр-регистр

Пересылка память-регистр, регистр-память

Арифметика целая

Арифметика дробная

Переходы:

Безусловный:

– прямой

– относительный

– косвенный прямой

Условный – то самое, только с проверкой флагов

К подпрограмме – адрес возврата в r1

Возврат из подпрограммы

## 16. Виртуальная БЭСМ-6

PSW – 32 бита = 16 + 16 = IP + Flags

Память – слова по 48 бит, размер адреса = 16 бит.

**сумматор** – 48 бит

**адресный регистр** – 16 бит

Типы данных:

Целые знаковые – 48 бит

Дробные – 48 бит: знак – 1 бит, порядок – 7 бит, мантисса – 40 бит; код прямой.

Структура команды, 24 бита:

Код операции – 7 бит, b – 1 бит

b = 0 – адрес (абсолютная адресация)

b = 1 – адрес + регистр (индексная или базовая)

Адрес – 16 бит

Две команды в слове

Загрузка адресного регистра: схема та же

b = 0 – адрес (константа) в команде (константа = адрес)

b = 1 – регистр + константа в команде

Загрузка-сохранение сумматора

Арифметика дробная на сумматоре

Арифметика целая знаковая на сумматоре; реализация операций «вручную»

Переходы:

Безусловный прямой: IP = адрес (константа в команде);

бит b работает по схеме загрузки адресного регистра:

b = 0 – адрес (константа) в команде (константа = адрес)

b = 1 – регистр + константа в команде

Если адрес = 0, то это косвенный переход по адресному регистру

Условный – то же самое, проверяет флаги;

Вызов подпрограммы, адрес возврата сохраняется в адресном регистре

Возврат – безусловный переход прямой: b = 1, регистр + 0



**20. Виртуальная машина RISC – Н. Вирт**

Вирт Н. Построение компиляторов. – М.: ДМК Пресс, 2010. – 192 с.  
Глава 9. RISC-архитектура как цель. – с. 75.

**21. Виртуальная машина MMIX – Д.Кнут**

Кнут Д. Искусство программирования, том 1, выпуск 1. MMIX – RISC- компьютер нового тысячелетия. – М.: ООО «И.Д. Вильямс», 2007. – 160 с.

**22. Виртуальная машина – Ч. Уэзерелл**

Уэзерелл Ч. Этюды для программистов. – М.: Мир, 1982. – 282 с.  
Глава 25. Уча – учимся, или Моделирование большого компьютера. – с. 159.

**23. Виртуальная машина О-машина – С.З. Свердлов**

Свердлов С.З. Языки программирования и методы трансляции. – СПб.: Питер, 2007. – 638 с.  
Генерация кода.  
Виртуальная машина – с. 352  
Архитектура виртуальной машины – с. 353  
Программирование в коде виртуальной машин – с. 358  
Реализация виртуальной машины – с. 362

**24. Виртуальная машина Y86 – Р. Брайант**

Брайант Р., О'Халларон Д. Компьютерные системы: архитектура и программирование. – СПб.: БХВ-Петербург, 2005. – 1104 с.  
Глава 4. Архитектура процессора – с. 301.

**25. Виртуальная машина УУМ/ДС – Л. Бек**

Бек Л. Введение в системное программирование. – М.: Мир, 1988. – 448 с.  
Глава 1. Основные понятия.  
1.3.2. Структура УУМ/ДС – с. 17

**26. Виртуальная машина IBM/370 – Л. Бек (см. также Джермейн)**

Бек Л. Введение в системное программирование. – М.: Мир, 1988. – 448 с.  
Глава 1. Основные понятия.  
1.4. Структура IBM/370 – с. 21

**27. Виртуальная машина CYBER – Л. Бек**

Бек Л. Введение в системное программирование. – М.: Мир, 1988. – 448 с.  
Глава 1. Основные понятия.  
1.6. Структура CYBER – с. 33

**28. Виртуальная машина JVM – Э. Таненбаум**

Таненбаум Э. Архитектура компьютера. – 4 издание. – СПб.: Питер, 2002. – 704 с.  
Глава 5. Уровень архитектуры команд – с. 334.

**29. Виртуальная машина Motorola 68000 – К. Хамахер**

Хамахер К., Вранешич З., Заки С. Организация ЭВМ. – 5 издание. – СПб.: Питер; Киев: Издательская группа BHV, – 2003. – 848 с.  
Глава 3. Системы команд процессоров АРМ, Motorola, Intel – с. 154.

**30. CLR – intiut'овская книжка.**