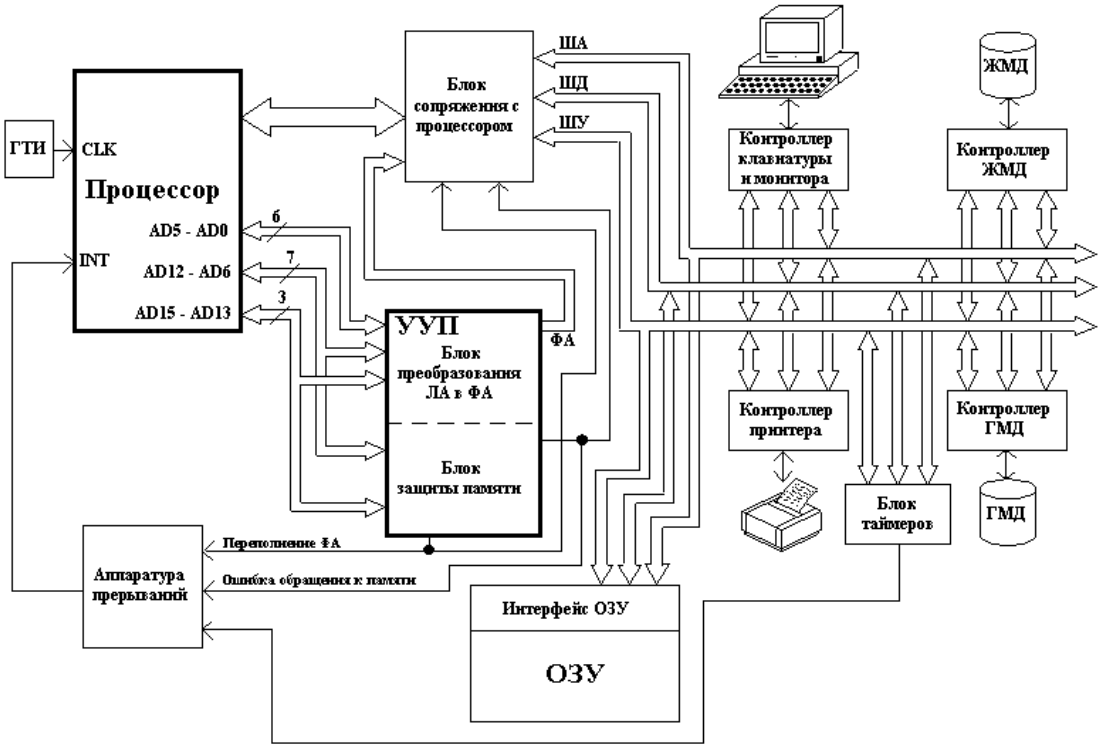
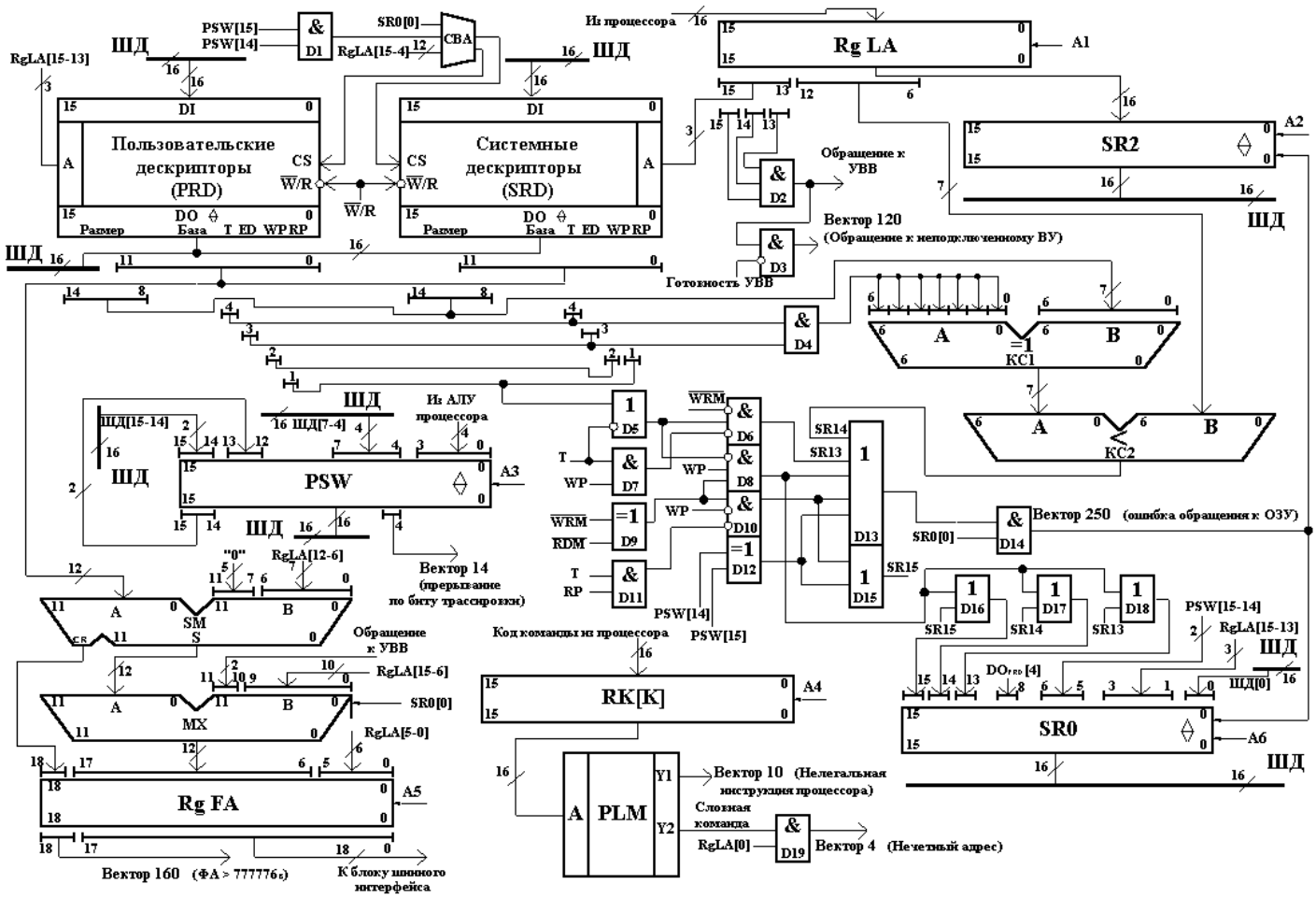
$KСтруктурная схема лабораторной установки#



$ФункциональKная схема устройства управления памятью (УУП)#K



$Размещение регистров в адресном пространстве#

Все регистры УУП имеют свои адреса в адресном пространстве сегмента ввода вывода.

Адреса регистров базовых адресов и регистров дескрипторов:

Пользовательский режим			Системный режим		
№ регистра	PAR	PDR	№ регистра	PAR	PDR
0	177640	177600	0	172340	172300
1	177642	177602	1	172342	172302
2	177644	177604	2	172344	172304
3	177646	177606	3	172346	172306
4	177650	177610	4	172350	172310
5	177652	177612	5	172352	172312
6	177654	177614	6	172354	172314
7	177656	177616	7	172356	172316

Адреса регистров ошибок и регистра слова состояния:

Регистр	Адрес
SR0	177572
SR2	177576
PSW	177776

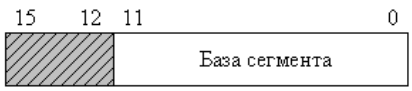
Примечание: регистр ошибок SR2 доступен только для чтения.

Адреса подключенных внешних устройств:

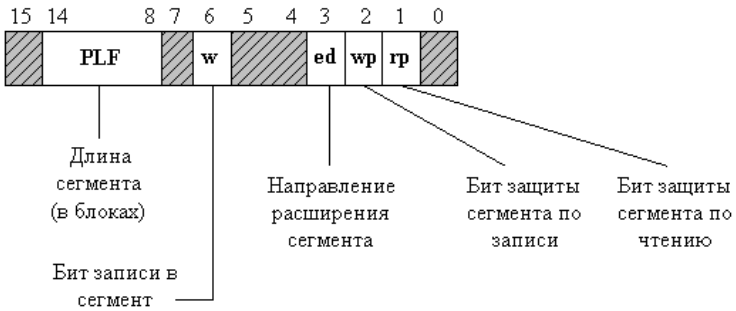
Адреса ВУ	Тип внешнего устройства
777170-777172	Регистры гибкого магнитного диска
777440-777476	Регистры жесткого магнитного диска
777546	Таймер
777514-777516	Регистры печатающего устройства
777560-777566	Регистры дисплея и клавиатуры
777724-777726	Регистры ОЗУ

$Форматы регистров базовых адресов и регистров дескрипторов#

Формат регистров базовых адресов PAR:



Формат регистров дескрипторов PDR различен в зависимости от режима работы лабораторной установки.
В режиме без кодовых сегментов (см. режимы работы лабораторной установки) формат регистров дескрипторов следующий:



Биты **WP** и **RP** определяют способ доступа в сегмент:

wp	rp	Способ доступа в сегмент
0	0	Сегмент не доступен ни для чтения ни для записи
0	1	Сегмент доступен только для чтения
1	0	Не используется
1	1	Сегмент доступен как для чтения так и для записи

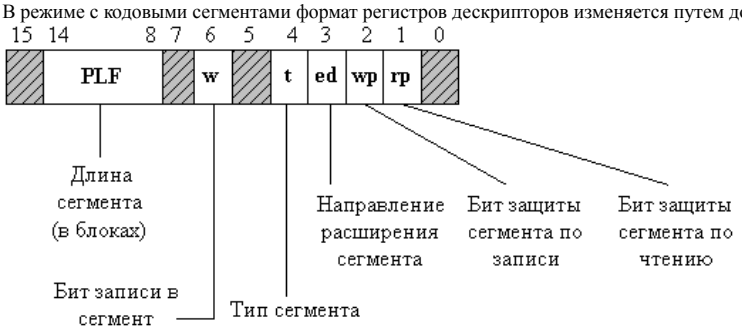
Бит **ED** определяет направление расширения сегмента:

ED	Направление расширение сегмента
0	Сегмент расширяется в сторону увеличения адресов
1	Сегмент расширяется в сторону уменьшения адресов

Бит **W** является флагом записи в сегмент и устанавливается только аппаратно при записи в сегмент.

Поле **PLF** является 7 разрядным и может принимать значение от 0 (минимальная длина сегмента 1 блок) до 177 в 8 с.с. (максимальная длина сегмента 200 блоков).

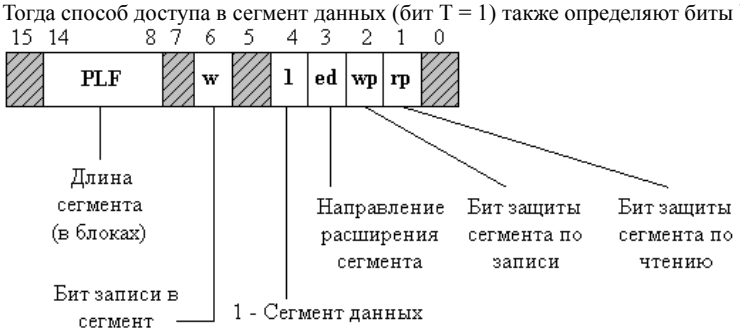
В режиме с кодовыми сегментами формат регистров дескрипторов изменяется путем добавления бита **T**, определяющего тип сегмента:



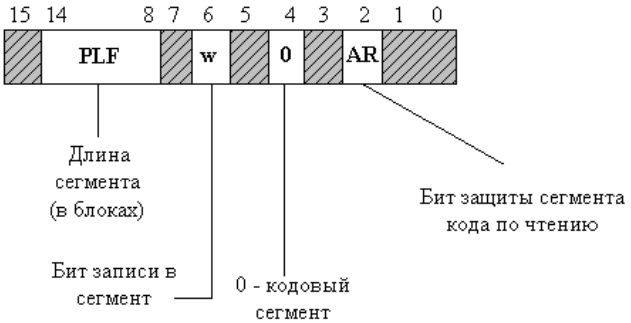
Бит **T** определяет тип сегмента:

t	Тип сегмента
0	Кодовый сегмент
1	Сегмент данных

Тогда способ доступа в сегмент данных (бит T = 1) также определяют биты **WP** и **RP** как и в режиме без кодовых сегментов:



Если же это сегмент кода (бит T = 0), то способ доступа в сегмент определяет лишь бит **AR** (allow read разрешить чтение). Сегменты кода всегда расширяются в сторону увеличения адресов, поэтому нет необходимости учитывать бит ED и он не имеет значения для сегмента кодов:

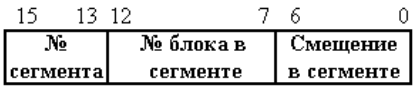


В этом случае доступ в сегмент кода определяется следующим образом:

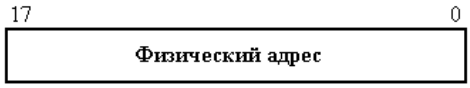
AR	Способ доступа в сегмент
0	Разрешено только выполнение
1	Разрешено выполнение и чтение

$Форматы регистров логического и физического адреса#

Регистр логического адреса разделен на три поля:
- номер сегмента;
- номер блока в сегменте;
- смещение.

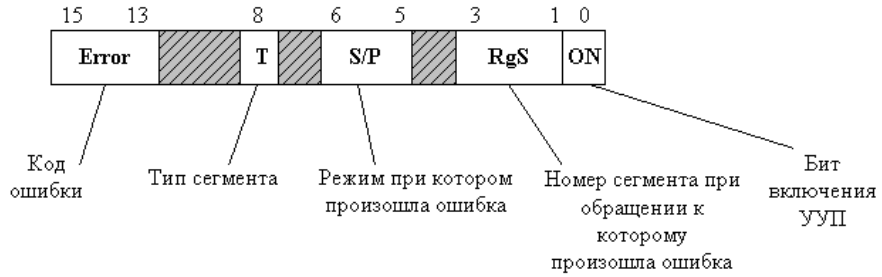


Регистр физического адреса является 18 разрядным и позволяет адресоваться к 256 Кбайтам.



$Форматы регистров ошибок#

В состав устройства управления памятью входят два регистра ошибок SR0 и SR2, которые используются операционной системой для анализа кода ошибки и причины прерывания по защите памяти.
Регистр SR0 указывает характер ошибки, состояние УУП и другую информацию, используемую при выполнении подпрограммы обработки прерывания:



Биты SR0 [15-13] указывают причину прерывания только для вектора 250 и устанавливаются аппаратно.
Для режима работы **без кодовых сегментов**:

- бит SR0[15] устанавливается при обращении к неактивному сегменту (поля wp и rp регистров дескрипторов равны 0 запрет по чтению и записи) или в случае установки в PSW[15-14] недопустимого режима работы (PSW[15-14] = 01 или 10);
- бит [14] устанавливается при нарушении длины (размера) сегмента;
- бит [13] устанавливается при попытке выполнить запись в сегмент, доступный только для чтения, т.е. биты wp и rp регистра дескриптора равны 0 и 1 соответственно (см. форматы регистров базовых адресов и регистров дескрипторов);
- биты [15-13] устанавливаются, если в полях wp и rp дескриптора установлены атрибуты защиты 1 и 0 соответственно.

Таким образом, все возможные сочетания битов SR0[15-13] в режиме без кодовых сегментов могут быть представлены следующей таблицей:

SR0[15-13]	Причина прерывания
100	Защита по записи и чтению
010	Нарушение границ сегмента
001	Защита по записи
110	Защита по записи и чтению и нарушение границ сегмента
011	Защита по записи и нарушение границ сегмента
100	Несуществующий режим работы
111	Несуществующий атрибут защиты

Для режима работы **с кодовыми сегментами**:

- бит SR0[15] устанавливается при обращении к сегменту кода, защищенному по чтению (поле **ar** регистров дескрипторов равно 0 запрет чтения из сегмента

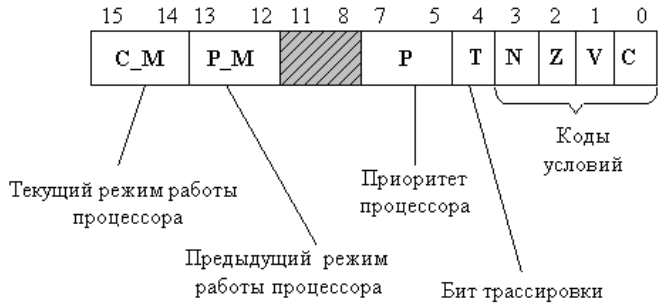
кода) или в случае установки в PSW[15-14] недопустимого режима работы (PSW[15-14] = 01 или 10);
- бит [14] устанавливается при нарушении длины (размера) сегмента кода;
- бит [13] устанавливается при попытке выполнить запись в сегмент кода.

Таким образом, все возможные сочетания битов SR0[15-13] в режиме с кодовыми сегментами могут быть представлены следующей таблицей:

SR0[15-13]	Причина прерывания
100	Защита по чтению из сегмента кода
010	Нарушение границ сегмента кода
001	Защита по записи в сегмент кода
110	Защита по чтению из сегмента кода и нарушение его границ
011	Защита по записи в сегмент кода и нарушение его границ
101	Защита по чтению из сегмента кода и по записи в него
111	Защита по чтению из сегмента кода, по записи в него и нарушение его границ

В регистре SR2 фиксируется логический адрес, при обработке которого обнаружена ошибка, поэтому запись в регистры SR0 и SR2 производится только по сигналу сегментной ошибки.

$**Формат слова состояния процессора**#



Коды условий:

N флаг отрицательного результата;
Z флаг нулевого результата;
V флаг переполнения;
C флаг переноса.

$**Общая информация о системе команд**#

Команды могут включать следующую информацию: мнемонику, операнд источник и операнд приемник.

Мнемоника. Определяет выполняемую операцию. Если для команда может работать с байтами то существует байтовый эквивалент команды, представляющий собой добавление к мнемонику символа **B**.

Например: **MOV (R1), @#124** команда работает со словами
MOVB (R2), @#125 команда работает с байтами

Операнд источник. Используется для двухадресных команд. Определяет операнд, который является источником для операции.
Операнд приемник. Используется для двухадресных и одноадресных команд. Определяет операнд, который является приемником для операции.

Например: **ADD (R3)+, R2**
Содержимое адреса в R3 источник операции
Регистр R2 приемник операции

Примечание. Все числовые значения в командах записываются в восьмеричной системе счисления.

Например: **INC @#25353**
25352 число, в восьмеричной системе счисления, равное **10 986** в десятичной системе счисления

Существует 4 вида команд:

Двухадресные команды
Одноадресные команды
Команды управления ходом программы
Операции над кодами условий

$Двухадресные команды

Двухадресные команды представлены в следующей таблице

Примечание 1:

(B) означает то, что команда может работать с байтами. Например:
MOV R1, R2 команда работает со словами
MOVB R1, R2 команда работает с байтами

Примечание 2:

(исх) первый операнд, указанный в команде источник
(назн) второй операнд, указанный в команде приемник

$Одноадресные команды#

Одноадресные команды представлены в следующей таблице

Примечание 1:

Команда **MTPS** #K перенос байта в слово состояния. Значение K (младший байт) записывается в младшие разряды PSW.

Внимание! Данной командой нельзя установить бит трассировки T (PSW[4]) , то есть ее действие можно определить следующим образом:

PSW[7-5]:=K[7-5] и PSW[3-0]:=K[3-0]

Примечание 2:

(B) означает то, что команда может работать с байтами. Например:
DEC (R1) команда работает со словами
DECB (R1) команда работает с байтами

Примечание 3:

(назн) операнд, указанный в команде приемник

$Команды управления ходом программы#

Команды управления ходом программы представлены в следующей таблице

Описание некоторых важных для выполнения лабораторной работы команд:

EMT #V - командное прерывание для системных программ
(SPS):=PSW, (SPS):=PC, PC:=(IDTS[D7-D0].0)

#V - переход по номеру вектора, V - номер вектора ППОП

TRAP #V - командное прерывание для пользовательских программ
(SPP):=PSW, (SPP):=PC, PC:=(IDTP[D7-D0].0)

#V - переход по номеру вектора, V - номер вектора ППОП

RTI - возврат из системного прерывания
PC:=(SPS) +, PSW:=(SPS) +

RTT - возврат из пользовательского прерывания
PC:=(SPP) +, PSW:=(SPP) +

JSR @#A - обращение к подпрограмме (SPP):=PC

RTS - возврат из подпрограммы PC:=(SPP) +

JMP @#A - безусловный переход PC:= A

Выше используются следующие обозначения:

(SPS) операция записи в системный стек
(SPP) операция записи в пользовательский стек
(SPS) + операция извлечения из системного стека
(SPP) + операция извлечения из пользовательского стека
@#A абсолютная адресация

$Операции над кодами условий#

Операции над кодами условий представлены в следующей таблице

$Общая информация о видах адресаций#

В разработанной лабораторной установке существует две группы адресаций:

Адресации, использующие регистры общего назначения
Дополнительные виды адресаций.

Первые адресации используют какой либо регистр общего назначения (РОН) для обращения к операндам. Содержимое регистра в зависимости от адресации может быть использовано в качестве операнда, базового адреса массива или, наоборот, индексного смещения относительно базового адреса и т.п. Например, в РОНе может находиться база массива, а смещение, заданное в команде будет определять номер элемента массива.
Во многих случаях бывает удобнее использовать непосредственный операнд или его абсолютный адрес в команде. Поэтому в лабораторной установке реализована вторая группа адресаций дополнительные виды адресаций (без использования РОНов).

$Адресации, использующие регистры общего назначения#

Ниже подробно рассмотрены все виды адресаций, использующие регистры общего назначения. В примерах будет использоваться двухадресная команда MOV, которая выполняет пересылку содержимого первого операнда (источника) по адресу второго операнда (приемника). Если происходит пересылка байта, то мнемоника команды MOVB.
В примерах будет приводится содержимое используемых регистров и ячеек памяти до и после операции. Все численные значения приводятся в *восьмеричном виде.*
Для отображения 16 разрядного слова используется 6 цифр, а для байта - 3 цифры.
Примечание. Для выполнения лабораторной работы необходимо использовать лишь некоторые виды адресаций, в зависимости от варианта. Некоторые адресации, например все косвенные, вообще не встречаются ни в одном из вариантов, а реализованы с целью ознакомления с ними тех студентов, которые хотят дополнительно изучить этот материал.

Адресации, использующие регистры общего назначения:

Регистровая адресация
Регистровая косвенная адресация
Автоинкрементная адресация
Косвенная автоинкрементная адресация
Автодекрементная адресация
Косвенная автодекрементная адресация
Индексная адресация
Косвенная индексная адресация

$Дополнительные виды адресаций

Дополнительные виды адресаций не используют регистры общего назначения. Всего в лабораторной установке реализовано 2 вида:
Непосредственная адресация
Абсолютная адресация

$Регистровая адресация#

При такой адресации операндом является содержимое регистра. Обозначение на Ассемблере: **Rn**

Пример пересылка содержимого регистра R2 в регистр R3. Содержимое регистров R2 и R3 до выполнения команды равно 100 и 350 соответственно:

Команда: **MOV R2,R3**

$Регистровая косвенная адресация#

Содержимое регистра является адресом операнда. Обозначение на Ассемблере: **(Rn)**

Пример переслать содержимое регистра R2 по адресу, находящемуся в R3.

Команда **MOV R2,(R3)**

$Автоинкрементная адресация#

Эта адресация аналогична регистровой косвенной, но после выполнения операции содержимое регистра увеличивается на размер операнда b (b=1, если операнд представлен в виде байта, b = 2, если в виде слова), где b размер операнда. Обозначение на Ассемблере **(Rn)+**

Пример 1 переслать содержимое регистра R2 (слово) по адресу, находящемуся в R3. После выполнения операции содержимое R3 увеличивается на 2 (так как операция со словом).

Команда: **MOV R2, (R3)+**

Пример 2 переслать содержимое регистра R2 (младший байт) по адресу, находящемуся в R3. После выполнения операции содержимое R3 увеличивается на 1 (так как операция с байтом).

Команда: **MOVB R2, (R3)+**

$Косвенная автоинкрементная адресация#

При такой адресации содержимое регистра является адресом адреса операнда. Затем содержимое регистра увеличивается на два независимо от размера операнда. Это определяется тем, что регистр содержит адрес адреса (операнда), а адрес всегда занимает слово. Обозначение на Ассемблере: **@(Rn)+**

Пример переслать содержимое регистра R2 по адресу, находящемуся в ячейке памяти на которую указывает R3. После выполнения операции содержимое R3 увеличивается на 2.

Команда: **MOV R2, @(R3)+**

$Автодекрементная адресация#

Эта адресация является обратной по отношению к автоинкрементной. Здесь содержимое регистра сначала уменьшается на размер операнда, а затем используется в качестве адреса операнда. Обозначение на Ассемблере: **(Rn)**

Пример 1 переслать содержимое регистра R2 (слово) по адресу, находящемуся в R3, уменьшенному на 2. Перед выполнением операции содержимое R3 уменьшается на 2 (так как операция со словом).

Команда: **MOV R2, (R3)**

Пример 2 переслать содержимое регистра R2 (младший байт) по адресу, находящемуся в R3, уменьшенному на 1. Перед выполнением операции содержимое R3 уменьшается на 1 (так как операция с байтом).

Команда: **MOVB R2, (R3)**

$Косвенная автодекрементная адресация#

Эта адресация аналогична автодекрементной, но содержимое регистра является адресом адреса операнда и уменьшается всегда на 2 независимо от размера операнда. Обозначение на Ассемблере: **@(Rn)**

Пример переслать содержимое регистра R2 по адресу, находящемуся в ячейке памяти на которую указывает R3 2. Перед выполнением операции содержимое R3 уменьшается на 2:

Команда: **MOV R2, @(R3)**

$Индексная адресация#

При этой адресации содержимое выбранного регистра складывается с содержимым слова, следующим непосредственно за командой. Полученная сумма является адресом операнда. Обозначение на ассемблере: **X(Rn)**
Здесь X содержимое слова, следующего непосредственно за командой. Обычно содержимое регистра используется в качестве базового адреса массива, а смещение указывает на конкретный элемент этого массива.

Пример 1 переслать слово из регистра R2 в ячейку памяти, адрес которой определяется суммой 2004+R3:

Команда: **MOV R2, 2004(R3)**

Пример 2 переслать младший байт из регистра R2 в ячейку памяти, адрес которой определяется суммой 2001+R3

Команда: **MOVB R2, 2001(R3)**

$Косвенная индексная адресация#

Эта адресация аналогична индексной, но сумма, полученная в результате сложения содержимого регистра и смещения является адресом адреса операнда. Эта сумма всегда должна быть четной, так она является адресом (операнда). Обозначение на ассемблере: **@X(Rn)**

Пример переслать слово из регистра R2 в ячейку памяти, адрес которой находится в ячейке памяти по адресу 2002+R3:

Команда: **MOV R2, @2002(R3)**

$Непосредственная адресация#

Данная адресация позволяет непосредственно в команде указать операнд для операции. Следует отметить, что непосредственный операнд находится сразу после кода команды и для его чтения происходит обращение к ОЗУ (в ячейку расположенную сразу после команды).

Обозначение на ассемблере: **#K**
где K непосредственный операнд.

Пример переслать непосредственный операнд 100 в регистр R2:

Команда: **MOV #100, R2**

$Абсолютная адресация#K

Данная адресация позволяет непосредственно в команде указать адрес операнда для операции. Следует отметить, что адрес операнда находится сразу после кода команды и для его чтения происходит обращение к ОЗУ (в ячейку расположенную сразу после команды).

Обозначение на ассемблере: **@#K**
где @#K адрес операнда.

Пример 1 переслать содержимое ячейки с адресом 100 по адресу, находящемуся в R2:

Команда: **MOV @#200, (R2)**

Пример 2 переслать содержимое ячейки с адресом 100 по адресу 200:

Команда: **MOV @#100, @#200**

$Вектора прерываний, реализованные в ЛУ#

В лабораторной установке реализованы следующие вектора прерываний:

Вектор	Причина прерывания
004	Исход времени, нечетный адрес, переполнение стека
010	Нелегальные или резервные инструкции процессора
014	Внутреннее прерывание по биту трассировки T PSW
120	Адрес неподключенного внешнего устройства
160	Физический адрес внешнего устройства больше 777776
250	Ошибка диспетчера памяти

$Порядок выполнения лабораторной работы#

Цель

работы

Целью лабораторной работы является:

" изучение методов и средств преобразования логического адреса в физический, реализованных в УУП;
" изучение методов и средств по защите памяти, реализованных в УУП;
" изучение архитектуры процессора и УУП (регистров процессора, слова состояния, регистров ошибок и т.д.)
" приобретение навыков в программировании системы защиты памяти, использовании команд программных прерываний для системного и пользовательского режимов работы.

Примечание:

Система команд процессора и виды адресаций практически полностью совпадают с системой команд и видами адресаций фирмы DEC (см. общая информация о системе команд и общая информация о видах адресаций).
Устройство управления памятью, реализованное в лабораторной установке, выполняет те же функции, что и УУП фирмы DEC. Но есть некоторые отличия:

" Формат регистров дескрипторов изменяется при режиме работы с кодовыми сегментами (см. форматы регистров базовых адресов и регистров дескрипторов);
" Изменен формат регистра SR0 в одном бите (вместо бита D, отвечающего за режим диагностики, введен бит T тип сегмента, см. форматы регистров ошибок).

Выполнение лабораторной работы складывается из домашней подготовки, экспериментальных исследований на лабораторной установке и оформления отчета.

Домашняя подготовка

Домашняя подготовка включает повторение лекционного материала по теме: "Изучение принципов организации памяти: преобразование ЛА в ФА и средства защиты памяти" по литературным источникам и методическим указаниям, знакомство с составом программного обеспечения лабораторного практикума и разработку текстов:

" программ инициализации устройства управления памятью;
" подпрограмм обработки прерываний по векторам ошибок;
" текстов программ пользователей в соответствии с вариантом задания.

Экспериментальные исследования

Экспериментальные исследования включают в себя: ввод, отладку и выполнение разработанных программ. После загрузки программы-модели для лабораторных исследований на экран дисплея выводится форма, включающая редактор для ввода команд, систему меню и выходное окно, отображающее состояния регистров процессора и УУП.
Лабораторные исследования включают ввод, отладку и выполнение разработанных программ.
Отладку программ рекомендуется осуществлять в пошаговом режиме, а демонстрацию результатов - в автоматическом, с установкой точек останова в тех местах, где требуется изменять входную информацию.
Методика исследований состоит из следующих этапов:
ввод начальных адресов векторов прерываний, используемых пользователем (окно "Просмотр таблицы IDT");
инициализация УУП путем загрузки регистров-дескрипторов и регистров базовых адресов, вычисленных при домашней подготовке;
ввод текстов подпрограмм, в соответствии с вариантом заданий;
анализ работы УУП по преобразованию ЛА в ФА и средств защиты путем выполнения исследовательской части разработанного программного обеспечения в соответствии с вариантом задания в пошаговом режиме;
установка точек останова на необходимых командах;
запуск программы в автоматическом режиме и получение результатов.
Результаты исследований автоматически формируются в окне результатов (пункт меню "Просмотр Просмотр результатов"), которое записывается в файл отчета по лабораторной работе при выборе пункта меню "Файл Генерация отчета".
Для останова выполнения программы исследований можно вставить точки останова с помощью пункта меню "Запуск точка останова".
Для выработки вектора 160 необходимо использовать сегментный регистр номер 7 (PARS[7]).

Внимание!

При получении вектора 160 (физический адрес внешнего устройства более 777776) каждый раз при обращении к сегменту ввода вывода (PAR[7]) процессор будет выполнять неправильную адресацию. Для выхода из этой ситуации необходимо перед самым выходом из подпрограммы обработки прерываний вектора 160 нажать клавишу <F4>. При этом в процессоре автоматически восстанавливается некорректное значение регистра PAR7.
При получении вектора вектора 250 (только для несуществующего режима работы) необходимо **принудительно** восстановить значение режима работы в слове состояния PSW путем нажатия клавиши <F4> сразу после выхода из подпрограммы обработки прерываний вектора 250. Программно изменять содержимое регистра PSW не рекомендуется, так как при этом каждый раз будет возникать прерывание по несуществующему режиму работы.

Требования к отчету

Отчет по лабораторной работе должен содержать: титульный лист, задание, содержимое регистров-дескрипторов, таблицы IDT, листинг входного окна (тексты программ) и окно результатов исследований с комментариями (с указанием причины прерывания), а также выводы.

$Задание на лабораторную работу#

1. В основной программе установить значения регистров базовых адресов и регистров-дескрипторов для системного режима.
Задания приведены в пункте Варианты заданий на лабораторную работу.

Помните !!! Для сегмента данных, значение, записанное в поле РАЗМЕР дескриптора, равное 0, означает, что размер сегмента равен 1 му блоку. Например, если в поле РАЗМЕР дескриптора записано значение 33 (8 с.с.), то это означает, что фактически размер сегмента равен 34 (8 с.с.) блокам, если в поле РАЗМЕР дескриптора записано значение 177 (8 с.с.), то это означает, что фактически размер сегмента равен 200 (8 с.с.) блокам.
Для сегмента стека тоже самое: Если в поле РАЗМЕР дескрптора для сегмента стека находится значение 0, то фактически это максимальный размер стека 200 (8.с.с.) блоков. Если в поле РАЗМЕР дескриптора для сегмента стека находится значение 177 (8 с.с.), то это минимальный размер стека 1 блок, дальше 176 (8 с.с.) 2 блока, 175 (8 с.с.) 3 блока&..и т.д.
В задании, в столбце РАЗМЕР, указаны те значения которые должны быть в дескрипторах, а не сами размеры сегментов. Фактически размеры самих

сегментов больше на 1 тех размеров, которые указаны в задании.

Поэтому для загрузки дескрипторов необходимо пользоваться следующими правилами для заполнения поля РАЗМЕР:

1) В поле РАЗМЕР дескриптора для сегмента данных необходимо записывать то число, которое указано в задании (а не уменьшенное на 1);
1) В поле РАЗМЕР дескриптора для сегмента стека необходимо записывать инверсию (т.е. обратный код) того числа, которое указано в задании (а не дополнитльный код числа).

Например:

RgS	Баз.адр.	Размер	Атр.защ.	RgS	Баз.адр.	Размер	Атр.защ.
0	501600	33		4	604500	57	Зп
1	443200	121*	Чт и Зп	5	667200	72	Зп
2	556000	24		6	425100	11	
3	543000	71		7	760000	177	

* означает расширение сегмента в сторону уменьшения адресов (сегмент стека)

```
00 MOV #5016,@#172340 ; загрузка баз.адреса в PAR0
01 MOV #4432,@#172342 ; загрузка баз.адреса в PAR1
. . . . . .
07 MOV #7600,@#172356 ; загрузка баз.адреса в PAR7
10 MOV #15406,@#172300 ; загрузка дескриптора в PDR0
11 MOV #27010,@#172302 ; загрузка дескриптора в PDR1
. . . . . .
17 MOV #77406,@#172316 ; загрузка дескриптора в PDR7
```

Примечание: предварительно значение дескриптора необходимо вычислить, т.е. распределить по полям и перевести из 2 с.с. в 8 с.с.:

X. 001.101.1 XX.XX 0 . 11 X = 015406 в 8 с.с. PDR0
Размер ED Разрешено чтение и запись

X. 010.111.0 XX.XX 1 . 00 X = 027010 в 8 с.с. PDR1 (размер в обратном коде).
X. 111.111.1 XX.XX 0 . 11 X = 077406 в 8 с.с. PDR7

2.Включить устройство управления памятью.

```
20 MOV #1,@#177572 ; включение УУП
```

Выполнить загрузку регистров базовых адресов и регистров- дескрипторов для пользовательского режима. Помните: **УУП включено! Процессор находится в системном режиме работы.** Задания приведены в пункте Варианты заданий на лабораторную работу.

3. Установить пользовательский режим работы процессора.

```
42 MOV #140000,@#177776 ; установка пользовательского режима работы.
```

4. Составить подпрограммы:

а) нахождения суммы N-ых элементов сегментов с 0 по 7 с накоплением суммы в M-ой ячейке сегмента 0. Адреса указаны в 8 с.с. По первому и второму адресам **абсолютная адресация**.
Номера ячеек N и M приведены в пункте Варианты заданий на лабораторную работу.

Примечание:

Пользовательские программы должны начинаться с адреса, отстоящего от последней команды инициализации УУП на 20 номеров команд.

б) выполнения заданной двухадресной команды с операндами из ячеек N и M, находящихся в сегментах со смежными номерами. Первый операнд команды (ячейка N) должен быть из сегмента с четным номером, а второй операнд (ячейка M) из сегмента с нечетным номером (то есть всего 4 команды с сочетанием операндов N-M из сегментов 0 1, 2 3, 4 5, 6 7). Команда, виды адресаций к операндам, номера ячеек N и M приведены в пункте Варианты заданий на лабораторную работу.

5. Составить две подпрограммы для заданных одноадресных команд для тысячных элементов (8 с.с.) для сегментов с 0 по 7 для заданной адресации.
Команды и виды адресаций к операндам приведены в пункте Варианты заданий на лабораторную работу.

6. Выполнить обращение к подпрограмме, разработанной в пункте 4 а) по команде JSR.

7. Выполнить изменения содержимого приведенных регистров базовых адресов и дескрипторов для пользовательского режима через вектор #K после чего повторить выполнение пункта 6.

Начальные адреса подпрограмм обработки векторов #K определяются в свободном адресном пространстве между программой инициализации УУП и началом пользовательских программ. Начальный адрес подпрограммы в таблицах векторов задается номером команды, а не содержимым программного счетчика.
Данные для перезагрузки приведены в пункте Варианты заданий на лабораторную работу.

8. Выполнить обращения к разработанным подпрограммам:

" в пункте 4 б) по команде TRAP
" в пункте 5 по командам JSR.

9. Составить тестовые программы для выработки всех типов векторов прерываний, реализованных в ЛУ, включая вектор 250 для заданных вариантов заполнения таблицы дескрипторов и базовых адресов для следующих сочетаний причин прерываний:

Вектор	Причина прерывания
004	Нечетный адрес
010	Нелегальные или резервные инструкции процессора
014	Внутренне прерывание по биту трассировки T слова состояния PSW
120	Обращение к неподключенному внешнему устройству
160	Физический адрес внешнего устройства больше 777776_8
250	Ошибка диспетчера памяти

Для вектора 250 необходимо выявить все следующие ситуации:

Регистр ошибок SR0[15-13]	Причина прерывания
100	Защита по записи и чтению
010	Нарушение границ сегмента
001	Защита по записи
110	Защита по записи и чтению и нарушение границ сегмента
011	Защита по записи и нарушение границ сегмента
100	Несуществующий режим работы
111	Несуществующий атрибут защиты

10. Установить в окне НАСТРОЙКИ флажок КОДОВЫЕ СЕГМЕНТЫ. Произвести изменение типа для двух заданных сегментов на кодовые сегменты для пользовательского режима с заданными атрибутами через вектор #К. Атрибуты сегментов приведены в таблице из пункта Варианты заданий на лабораторную работу.

ПОМНИТЕ! Формат регистров дескрипторов изменяется при установке флажка "Кодовые сегменты" (см. форматы регистров базовых адресов и регистров дескрипторов).

11. Составить тестовые программы для выработки вектора 250 при обращении к кодовым сегментам. Базовые адреса и дескрипторы сегментов оставить такими, какими они были после выполнения 9-го задания.

Необходимо получить следующие сочетания битов SR0[15-13]:

Регистр ошибок SR0	Причина прерывания
100	Защита по чтению из сегмента кода
010	Нарушение границ сегмента кода
001	Защита по записи в сегмент кода
110	Защита по чтению из сегмента кода и нарушение его границ
011	Защита по записи в сегмент кода и нарушение его границ
101	Защита по чтению из сегмента кода и по записи в него
111	Защита по чтению из сегмента кода, по записи в него и нарушение его границ

ПОМНИТЕ! Выработку этих причин прерывания необходимо осуществлять в режиме работы с разрешением кодовых сегментов. Для этого в настройках программы должен быть установлен флажок "Кодовые сегменты" перед выполнением задания 9.

Примечание:
В подпрограммах обработки прерываний для векторов 4, 10, 120 и 160 в регистр R0 последовательно загрузить номер вектора прерывания и содержимое программного счетчика, сохраненного в стеке, на котором выработан сигнал прерывания. Для вектора 250 в ППОП в регистр R0 последоватлельно загрузить номер вектора, содержимое регистра SR0, содержимое регистра SR2, содержимое PC .
Регистры R1-R5 можно использовать при адресациях, вычисляющих логический адрес с помощью регистров. Регистр R0 не рекомендуется использовать для этих целей, т.к. в ППОП его содержимое необходимо предварительно сохранять в стеке, а перед командой возврата - восстанавливать из стека.

```
160 MOV (R6),R0 ; загрузка PC из стека в R0
161 RTI
```

$Варианты заданий на лабораторную работу#К

Варианты заданий базовых адресов и атрибутов защиты системных сегментов

Вариант 1

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	5016	33*	
1	4432	101	
2	5560	24	З и Ч
3	5430	37*	
4	6040	57	
5	6670	112	З
6	4250	11	
7	7600	177	

Вариант 2

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	6016	24*	
1	6545	36	
2	6732	57*	
3	6060	72	З
4	5430	11	З и Ч
5	5070	33	
6	4640	101	
7	7600	177	

Вариант 3

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	6716	37	
1	6432	57*	
2	5560	62	З и Ч
3	6330	11	
4	6140	33*	З
5	4770	101	
6	6230	24	
7	7600	177	

Вариант 4

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	4016	14*	
1	5245	47	
2	6432	57	З
3	0430	72	
4	6330	11*	
5	4770	43	
6	6140	111	З и Ч
7	7600	177	

Вариант 5

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	5616	33*	
1	5432	101	З и Ч
2	4560	54	З
3	4367	27*	
4	5140	57	
5	4770	62	
6	6010	13	
7	7600	177	

Вариант 6

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	6716	34*	
1	5760	57	
2	5432	67	
3	5560	112	З
4	6330	14	З и Ч
5	4750	36*	
6	5140	51	
7	7600	177	

Вариант 7

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	6216	33*	
1	5432	71	
2	5560	144	
3	6330	56*	
4	6140	77	
5	5600	72	З
6	5010	15	З и Ч
7	7600	177	

Вариант 8

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	6516	64*	
1	5245	27	З
2	6432	47	З и Ч
3	4560	76	
4	5100	12	
5	5770	36*	
6	6140	51	
7	7600	177	

Вариант 9

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	4716	23*	
1	4432	111	З
2	6560	34*	
3	6330	57	
4	5140	77	З и Ч
5	4770	112	
6	5560	11	
7	7600	177	

Вариант 10

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	5616	24*	
1	6245	105	З и Ч
2	4750	67	З
3	4560	72	
4	5330	13*	
5	5770	53	
6	6140	31	
7	7600	177	

Вариант 11

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	4250	33*	
1	5016	101	З и Ч
2	4432	24	
3	5560	37*	
4	5430	57	З
5	6040	112	
6	6670	11	
7	7600	177	

Вариант 12

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	4640	24*	
1	6016	36	
2	6545	57*	З
3	6732	72	З и Ч
4	6060	11	
5	5430	33	
6	5070	101	
7	7600	177	

Вариант 13

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	6230	37	
1	6716	57*	З и Ч
2	6432	62	
3	5560	11	З
4	6330	33*	
5	6140	101	
6	4770	24	
7	7600	177	

Вариант 14

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	4770	14*	
1	4016	47	З
2	5245	57	
3	6432	72	
4	0430	11*	
5	6330	43	З и Ч
6	6140	111	
7	7600	177	

Вариант 15

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	6010	33*	
1	5616	101	З
2	5432	54	
3	4560	27*	
4	4367	57	
5	5140	62	
6	4770	13	З и Ч
7	7600	177	

Вариант 16

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	5140	34*	
1	6716	57	
2	5760	67	З
3	5432	112	З и Ч
4	5560	14	
5	6330	36*	
6	4750	51	
7	7600	177	

Вариант 17

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	5010	33*	
1	6216	71	
2	5432	144	
3	5560	56*	
4	6330	77	З
5	6140	72	З и Ч
6	5600	15	
7	7600	177	

Вариант 18

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	6140	64*	
1	6516	27	З и Ч
2	5245	47	
3	6432	76	
4	4560	12	
5	5100	36*	
6	5770	51	З
7	7600	177	

Вариант 19

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	5560	23*	
1	4716	111	
2	4432	34*	
3	6560	57	
4	6330	77	З и Ч
5	5140	112	
6	4770	11	З
7	7600	177	

Вариант 20

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	6140	24*	
1	5616	105	З
2	6245	67	
3	4750	72	
4	4560	13*	
5	5330	53	
6	5770	31	З и Ч
7	7600	177	

Примечания:

- В столбце "Размер" * означает расширение сегментов в сторону уменьшения адресов;
- В столбце "Атрибуты защиты" **З** означает атрибут защиты по записи (сегмент доступен только для чтения); **З и Ч** означает атрибуты защиты по записи и чтению (сегмент не доступен ни для чтения ни для записи);

" В столбце "Размер" указана величина, меньше фактического размера на 1.

Варианты заданий базовых адресов и атрибутов защиты для пользовательских сегментов

Вариант 1

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	1045	23	
1	2432	111	З и Ч
2	1560	34	З
3	3430	47*	
4	4000	67*	
5	1670	72	З
6	3600	12	З и Ч
7	7600	77	

Вариант 2

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	0540	44	
1	1545	56	
2	2732	67	З и Ч
3	3060	52	З
4	4000	11*	
5	2070	23	З и Ч
6	3640	111*	
7	7600	77	

Вариант 3

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	2245	26*	
1	2432	57	З
2	1560	42	З и Ч
3	1330	14	
4	4000	53*	
5	3770	121	З
6	3123	44	З и Ч
7	7600	77	

Вариант 4

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	1210	14	
1	0245	55	З
2	0432	67	З и Ч
3	1140	72	З и Ч
4	4000	11*	
5	2770	73	З
6	3140	111*	
7	7600	77	

Вариант 5

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	3245	63	
1	3432	121	
2	1560	44	З и Ч
3	1020	65*	
4	4000	47*	
5	1770	32	З
6	2010	13	З и Ч
7	7600	77	

Вариант 6

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	1010	24*	
1	0540	37	З
2	1432	67	
3	2560	112	З и Ч
4	4000	12*	З
5	0750	34	
6	3140	61	
7	7600	77	

Вариант 7

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	2245	53*	
1	3432	171	З
2	1560	34	
3	1330	67	З и Ч
4	4000	77*	
5	2523	42	З
6	3010	15	З и Ч
7	7600	77	

Вариант 8

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	1010	44	
1	0245	27*	
2	0432	157	З и Ч
3	0560	66	З
4	4000	12*	
5	1770	76	З
6	2140	51	З и Ч
7	7600	77	

Вариант 9

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	1245	63	
1	0432	111	З и Ч
2	1560	44	
3	0030	37*	З
4	4000	77	
5	2770	52	З и Ч
6	3563	11*	
7	7600	77	

Вариант 10

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	0110	74	
1	0245	115	З и Ч
2	1410	27*	
3	2560	72	З
4	4000	13*	
5	0770	113	
6	1140	41	З и Ч
7	7600	77	

Вариант 11

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	3430	23	
1	1045	111	З
2	2432	34	
3	1560	47*	
4	4000	67*	З
5	1670	72	З и Ч
6	3600	12	З и Ч
7	7600	77	

Вариант 12

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	3060	44	
1	0540	56	З и Ч
2	1545	67	З
3	2732	52	
4	4000	11*	З и Ч
5	2070	23	
6	3640	111*	
7	7600	77	

Вариант 13

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	1330	26*	
1	2245	57	З и Ч
2	2432	42	
3	1560	14	
4	4000	53*	З
5	3770	121	З и Ч
6	3123	44	З
7	7600	77	

Вариант 14

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	1140	14	
1	1210	55	З и Ч
2	0245	67	З и Ч
3	0432	72	
4	4000	11*	З
5	2770	73	
6	3140	111*	З
7	7600	77	

Вариант 15

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	1020	63	
1	3245	121	З и Ч
2	3432	44	
3	1560	65*	
4	4000	47*	З
5	1770	32	З и Ч
6	2010	13	
7	7600	77	

Вариант 16

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	2560	24*	
1	1010	37	
2	0540	67	З и Ч
3	1432	112	З
4	4000	12*	
5	0750	34	
6	3140	61	З
7	7600	77	

Вариант 17

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	1330	53*	
1	2245	171	
2	3432	34	З и Ч
3	1560	67	
4	4000	77*	З
5	2523	42	З и Ч
6	3010	15	З
7	7600	77	

Вариант 18

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	0560	44	
1	1010	27*	З и Ч
2	0245	157	З
3	0432	66	
4	4000	12*	З
5	1770	76	З и Ч
6	2140	51	
7	7600	77	

Вариант 19

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	0030	63	
1	1245	111	
2	0432	44	З
3	1560	37*	
4	4000	77	З и Ч
5	2770	52	
6	3563	11*	З и Ч
7	7600	77	

Вариант 20

№ сегмента	Базовый адрес	Размер	Атрибуты защиты
0	2560	74	
1	0110	115	
2	0245	27*	З
3	1410	72	
4	4000	13*	
5	0770	113	З и Ч
6	1140	41	З и Ч
7	7600	77	

Примечания:

" В столбце "Размер" * означает расширение сегментов в сторону уменьшения адресов;
" В столбце "Атрибуты защиты" **З** означает атрибут защиты по записи (сегмент доступен только для чтения); **З и Ч** означает атрибуты защиты по записи и чтению (сегмент не доступен ни для чтения ни для записи);
" В столбце "Размер" указана величина, меньше фактического размера на 1.

Варианты заданий номеров используемых ячеек памяти

Номер варианта	N	M	Номер варианта	N	M
1	2000	500	11	3000	700
2	2050	600	12	3050	650
3	2100	700	13	3100	400
4	2200	750	14	3200	650
5	2400	550	15	3400	450
6	2450	650	16	4450	350
7	2500	400	17	4500	600
8	2700	450	18	4700	750
9	3000	300	19	5000	400
10	3500	350	20	5500	550

Варианты заданий мнемоник и адресаций для двухадресной команды

Номер варианта	Команда	Адресация для первого операнда	Адресация для второго операнда
1	MOVB	Индексная	Автодекрементная
2	BIC	Индексная	Регистровая косвенная
3	BISB	Регистровая косвенная	Абсолютная
4	SUB	Автоинкрементная	Индексная
5	MOV	Автодекрементная	Абсолютная
6	BICB	Абсолютная	Автоинкрементная
7	BIS	Автоинкрементная	Регистровая косвенная
8	XOR	Автодекрементная	Автоинкрементная
9	MOVB	Абсолютная	Индексная
10	BIC	Регистровая косвенная	Автодекрементная
11	BISB	Абсолютная	Регистровая косвенная
12	SUB	Автоинкрементная	Автодекрементная
13	MOV	Индексная	Автоинкрементная
14	BIC	Индексная	Абсолютная
15	BIS	Регистровая косвенная	Автоинкрементная
16	XOR	Автодекрементная	Индексная
17	MOVB	Абсолютная	Автодекрементная
18	BICB	Регистровая косвенная	Индексная
19	SUB	Автодекрементная	Регистровая косвенная
20	BIS	Автоинкрементная	Абсолютная

Варианты заданий мнемоник и адресации для одноадресных команд

Номер варианта	Команда	Адресация	Команда	Адресация
1	TSTB	Абсолютная	CLR	Автодекрементная
2	TST	Регистровая косвенная	COMB	Индексная
3	TSTB	Автоинкрементная	INC	Абсолютная
4	TST	Автодекрементная	DECB	Автоинкрементная
5	TSTB	Индексная	NEG	Регистровая косвенная
6	TST	Абсолютная	ASRB	Автодекрементная
7	TSTB	Регистровая косвенная	ASL	Индексная
8	TST	Автоинкрементная	RORB	Абсолютная
9	TSTB	Автодекрементная	ROL	Автоинкрементная
10	TST	Индексная	SWAB	Регистровая косвенная
11	TST	Индексная	ROLB	Регистровая косвенная
12	TSTB	Автодекрементная	SWAB	Абсолютная
13	TST	Автоинкрементная	NEGB	Автодекрементная
14	TSTB	Регистровая косвенная	ASR	Индексная
15	TST	Абсолютная	ASLB	Автоинкрементная
16	TSTB	Индексная	ROR	Регистровая косвенная
17	TST	Автодекрементная	INCB	Абсолютная
18	TSTB	Автоинкрементная	DEC	Автодекрементная
19	TST	Регистровая косвенная	CLRB	Индексная
20	TSTB	Абсолютная	COM	Автоинкрементная

Варианты заданий для изменения пользовательских дескрипторов

Номер Варианта	Номер регистра	Базовый адрес	Атрибуты	Номер регистра	Базовый адрес	Атрибуты
1	1	2134	З	5	0122	-
2	5	3434	З	6	0174	З и Ч
3	1	4534	-	5	0156	З
4	1	0143	З и Ч	2	2534	-
5	4	6444	З и Ч	6	0073	З
6	1	5344	З	6	0056	З и Ч
7	1	1234	З и Ч	6	0034	-
8	2	0134	З	5	0023	-
9	2	1434	З и Ч	5	0162	З
10	4	1534	З и Ч	6	0273	З
11	1	3122	З и Ч	6	0161	-
12	4	4174	З	6	2222	З и Ч
13	1	0121	З и Ч	5	0527	-
14	1	0534	-	4	6344	З и Ч
15	1	1073	З	2	2444	З и Ч
16	1	4056	З	6	1443	З и Ч
17	5	2034	-	6	4321	З и Ч
18	2	1123	З и Ч	5	3134	-
19	1	0062	З и Ч	4	3431	З
20	3	6273	З и Ч	5	4001	З

Примечание: В столбце "Атрибуты" **З** означает атрибут защиты по записи (сегмент доступен только для чтения); **З и Ч** означает атрибуты защиты по записи и чтению (сегмент не доступен ни для чтения ни для записи);

Варианты заданий изменения типов сегментов на кодовые

Номер варианта	Номер регистра	Атрибут защиты	Номер регистра	Атрибут защиты
1	1	Вп и Чт	5	Вп
2	2	Вп	5	Вп и Чт
3	1	Вп	6	Вп и Чт
4	3	Вп и Чт	5	Вп
5	1	Вп	4	Вп и Чт
6	3	Вп	4	Вп и Чт
7	1	Вп и Чт	5	Вп
8	2	Вп	6	Вп и Чт
9	1	Вп	5	Вп и Чт
10	3	Вп и Чт	6	Вп
11	1	Вп	5	Вп и Чт
12	2	Вп	4	Вп и Чт
13	1	Вп и Чт	6	Вп
14	4	Вп	6	Вп и Чт
15	1	Вп	2	Вп и Чт
16	2	Вп	3	Вп и Чт
17	2	Вп и Чт	4	Вп
18	1	Вп	6	Вп и Чт
19	2	Вп	6	Вп и Чт
20	3	Вп и Чт	5	Вп

Примечание: В столбце "Атрибуты защиты" **Вп** означает то, что разрешено только выполнение (чтение из кодового сегмента запрещено); **Вп и Чт** означает то, что разрешено и выполнение и чтение из кодового сегмента.

$Описание меню#

Ниже представлено описание всех пунктов меню лабораторной установки.

Меню **Файл** содержит следующие подпункты:

Открыть Загрузить файл с программой
Сохранить Сохранить программу
Сохранить как Сохранить программу под другим именем
Калькулятор Вызвать встроенный калькулятор Windows
Генерация отчета Сохранение результатов в текстовый файл для отчета
Выход Выход из программы

Меню **Правка** содержит следующие подпункты:

Копировать	Копировать выделенный блок в буфер обмена
Вставить	Вставить блок из буфера обмена
Вырезать	Очистить выделенный фрагмент программы
Вставить строку	Вставить пустую строку (подробнее см. Настройка системы)
Удалить строку	Удалить строку (подробнее см. Настройка системы)
Размер окна	Распахнуть/Свернуть окно команд

Меню **Просмотр** содержит следующие подпункты:

Просмотр стека	Просмотр системного и пользовательского стеков
Просмотр таблицы IDT	Просмотр и редактирование таблицы IDT
Просмотр ОЗУ	Просмотр ОЗУ
Просмотр результатов	Просмотр окна результатов

Меню **Запуск** содержит следующие подпункты:

Один шаг	Пошаговое выполнение программы
Автоматически	Запустить программу на автоматическое выполнение
Точка останова	Установить точку останова на строку под курсором
Убрать все точки останова	Удалить все точки останова из программы
Сброс	Проинициализировать установку
"Мягкий" сброс	Сбросить некорректные значения PSW[15..14] и PARS[7]
Отладка	Выполнение команды без изменения счётчика команд

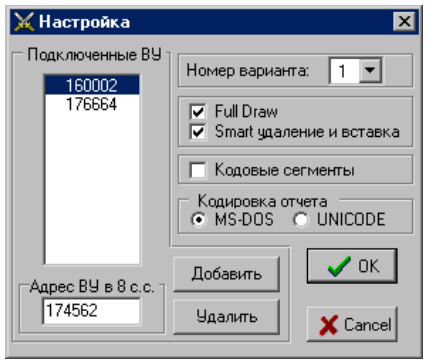
Меню **Настройка** Для настройки лабораторной установки (подробнее см. Настройка системы)

Меню **Помощь** содержит следующие подпункты:

Помощь	Вызов справочной системы
Задание	Посмотреть задание на лабораторную работу
Результаты теста	Посмотреть результаты входного теста
О программе	Информация о программе и авторе программы

$Настройка системы#

Для указания параметров работы лабораторной установки служит пункт меню **Настройка.**



В нем можно указать следующую информацию:

- Номер варианта в поле "**Номер варианта**", определяющийся номером подгруппы и машины в лаборатории.

- Степень прорисовки выходных данных. Если снять флажок **Full Draw**, то при запуске программы в автоматическом режиме поле программ и содержимое регистров перерисовываться не будет, что позволяет увеличить быстродействие в несколько раз.

- Формат вставки строк. Если установить флажок **Smart удаление и вставка**, то при редактировании удаление и вставка строк будут действовать до первой пустой строки, иначе - на весь текст программы.

- Режим работы лабораторной установки. При установке флажка "**Кодовые сегменты**" лабораторная установка начинает работать в режиме с возможностью использования кодовых сегментов, и соответственно с возможностью выработки векторов прерываний связанных с нарушением атрибутов защиты кодовых сегментов. Формат регистров дескрипторов в данном случае изменяется (см. форматы регистров базовых адресов и регистров дескрипторов). При отсутствии установки флажка лабораторная установка работает в режиме без кодовых сегментов соответственно с другим форматом регистров дескрипторов.

- Кодировку отчета. В поле "**Кодировка отчета**" можно установить формат символов, записываемых в отчет. При установке кнопки "**MS DOS**" отчет будет записан в кодировке DOS, а при установке кнопки "**UNICODE**" в кодировке UNICODE.

- Дополнительные внешние устройства. В поле **Адрес ВУ в 8 с.с.** вводится адрес подключаемого внешнего устройства (ВУ). При нажатии на кнопку "**Добавить**" происходит подключение нового внешнего устройства. Подключение происходит только в том случае, если адрес введен в восьмеричной системе счисления, является четным и лежит в пределах 160000 177776, иначе появляется сообщение об ошибке. При добавлении нового ВУ, его адрес отображается списке "**Подключенные ВУ**". Для удаления ВУ из списка его необходимо выделить из списка и нажать кнопку "**Удалить**".

$Режимы работы лабораторной установки#K

Лабораторная установка может работать в двух режимах:
- без использования кодовых сегментов;
- с использованием кодовых сегментов.

Режим без использования кодовых сегментов

Данный режим является основным для выполнения лабораторной работы. Он используется для выполнения всех заданий, кроме последнего. В этом режиме нет разделения на кодовые сегменты и сегменты данных.
Сегменты данных могут разделяться на просто сегменты данных и на сегменты стека, в зависимости от соответствующего бита в регистрах дескрипторах (см. форматы регистров базовых адресов и регистров дескрипторов).

Режим с использованием кодовых сегментов

Особенностью данного режима является то, что сегменты разделяются на сегменты кодов, данных и стеков. Чтобы включить режим с использованием кодовых сегментов необходимо установить соответствующий флажок в окне настроек (см. Настройка системы), а чтобы отключить флажок необходимо сбросить.
При работе лабораторной установки в данном режиме формат регистров дескрипторов УУП изменяется: вводится дополнительный бит, разделяющий сегменты на сегменты кодов и сегменты данных (см. форматы регистров базовых адресов и регистров дескрипторов).

$Замечания#

" При вводе программы возможен быстрый переход на строку с нужным номером. Для этого необходимо нажать клавишу Ctrl и, не отпуская ее, ввести нужный номер. После ввода клавиша Ctrl отпускается и происходит мгновенный переход на набранный номер команды. Такая же возможность существует в окне ОЗУ, где можно переходить на необходимый адрес таким же способом.
" При сохранении программ также происходит сохранение точек останова, таблица IDT, настройки программы, номер варианта;
" Выполнение программы можно прервать путем нажатия клавиши ESC;
" Для выполнения лабораторной работы часто бывает удобно пропустить ряд кодманд, не выполняя их. Это осущесвляется с помощью комментирования. Перед командой ставиться точка с запятой или любой другой символ. Например: **;jsr @#100**. Таким образом каждый раз при выполнении команды происходит прерывание по вектору 10 несуществующая команда. Обрабатывается 10 вектор и происходит возврат на следующую команду (то есть фактически команда не выполняется). Чтобы разраешить выполнение команды достаточно убрать комментирующий символ.
