Министерство образования Республики Беларусь

Учреждение Образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Лабораторная работа №4

по дисциплине «Программирование на языке ассемблера»

на тему «Создание видеоигры»

вариант №5

|  |  |  |
| --- | --- | --- |
| Выполнил  студент гр. 250501  Лукьянов Е.О. |  | Проверил  Туровец Н.О. |

Минск 2023

**Цель работы:** Ознакомиться в рамках создания видеоигры с обработкой нажатий кнопок клавиатуры, рассмотреть прямой доступ к видеопамяти с целью формирования игрового поля и информации для пользователя.

**Теоретические сведения**

Для выполнения работы требуется рассмотреть следующие элементы языка ассемблера и операционной системы:

1. Прямой доступ к видеопамяти.

Кроме использования прерываний DOS, описанных в лабораторной работе №2, программа может выводить текст на экран с помощью пересылки данных в специальную область памяти, связанную с видеоадаптером – видеопамять. Этот вариант вывода более быстр, чем при выводе символов через прерывания, а также позволяет формировать в консоли определенные эффекты, часто не используемые в режиме вывода в позицию курсора. В большинстве текстовых видеорежимов под видеопамять отводится специальная область памяти, начинающаяся с абсолютного адреса B800h:0000h и заканчивающаяся адресом B800h:FFFFh. Все, что программа запишет в эту область памяти, будет пересылаться в память видеоадаптера и отображаться на экране.

В текстовых режимах для хранения каждого изображенного символа используются два байта:

-- байт с ASCII-кодом символа;

-- байт атрибута символа (указывает цвет символа и фона, мигание).

Байт атрибута символа имеет следующий формат (биты):

-- 7 – символ мигает (по умолчанию) или фон яркого цвета (если его действие было переопределено прерыванием 10h).

-- 6 – 4 – цвет фона.

-- 3 – символ яркого цвета (по умолчанию) или фон мигает (если его действие было переопределено прерыванием 11h).

-- 2 – 0: цвет символа.

Для установки требуемого программе видеорежима используется прерывание 10h (видеосервис) BIOS. Видеорежимы отличаются друг от друга разрешением (для графических) и количеством строк и столбцов (для текстовых), а также количеством возможных цветов. В данной лабораторной работе использование графических режимов видеоадаптера не требуется, поэтому в описании прерываний эта информация будет опущена.

-- Прерывание BIOS 10h, функция 00 – установить видеорежим:

Ввод: AH = 00,

AL = номер режима в младших 7 битах:

-- 00 – 40х25 черно-белый текстовый режим;

-- 01 – 40х25 стандартный 16-цветный текстовый режим;

-- 02 – 80 х 25 черно-белый текстовый режим;

-- 03 – 80 х 25 стандартный 16-цветный текстовый режим;

-- 07 – 80 х 25 черно-белый стандартный монохромный. Видеорежим номер 3 используется в DOS по умолчанию.

Если старший бит AL не установлен в 1, то экран очищается.

Вывод: Обычно отсутствует, но некоторые BIOS помещают в AL 30Н для текстовых режимов и 20h для графических

-- Прерывание BIOS 11h – конфигурация оборудования:

Ввод: AH = 11

Вывод: AX = состояние оборудования (биты 5 и 4 указывают текущий видеорежим):

-- 00 – не используется,

-- 01 – 40х25 цветной режим,

-- 10 – 80х25 цветной режим,

-- 11 – 80х25 черно-белый режим.

Курсор не является символом из набора ASCII-кодов. Компьютер имеет собственное аппаратное обеспечение для управления видом курсора. Обычно символ курсора похож на символ подчеркивания и всегда мерцает. Для работы с курсором используются следующие функции BIOS:

-- Прерывание BIOS 10h, функция 01 – установить размер курсора:

Ввод: AH = 01,

CH = номер верхней линии (20H – подавить курсор),

CL = номер нижней линии. Можно установить следующий размер курсора по вертикали:

-- от 0 до 13 – для монохромных и EGA мониторов;

-- от 0 до 7 для большинства цветных мониторов. Курсор сохраняет свой вид, пока программа не изменит его. Стандартные размеры курсора для монитора:

-- монохромный – 12/13;

-- цветной – 6/7.

-- Прерывание BIOS 10h, функция 02 – установить положение курсора: Ввод: AH = 02,

BH = номер страницы,

DH = номер строки (считая от 0, установка на строку 25 делает курсор невидимым),

DL = номер столбца (считая от 0).

Отсчет номера строки и столбца ведется от верхнего левого угла экрана (символ в левой верхней позиции имеет координаты 0, 0). Номера страниц 0 – 3 (для видеорежимов 2 и 3) и 0 – 7 (для видеорежимов 1 и 2) соответствуют области памяти, содержимое которой в данный момент отображается на экране. Можно вывести текст в неактивную в настоящий момент страницу, а затем переключиться на нее, чтобы изображение изменилось мгновенно.

-- Прерывание BIOS 10h, функция 03 – получить положение и размер курсора (каждая страница использует собственный независимый курсор):

Ввод: АН = 03,

ВН = номер страницы.

Вывод: DH = строка текущей позиции курсора (считая от 0),

DL = столбец текущей позиции курсора (считая от 0),

СН = первая строка размера курсора,

CL = последняя строка размера курсора.

-- Прерывание BIOS 10h, функция 05 – установить активную страницу (доступно для цветных текстовых режимов):

Ввод: АН = 05,

AL = номер страницы (обычно 0).

Для режима 40х25 можно устанавливать до 8 страниц (от 0 до 7), а для режима 80х25 – до 4 страниц (от 0 до 3).

Вывод символов на неактивную страницу, а затем установка ее активной позволяют формировать более естественный и плавный вывод данных.

-- Прерывание BIOS 10h, функция 06 – прокрутка экрана вверх (вставка чистых строк снизу):

Ввод: АН = 06,

Если AL = 00 – прокрутка всего экрана с заполнением пробелами (очистка), иначе AL = число строк для прокрутки вверх,

BH = атрибут вставляемого символа,

CH = строка верхнего левого угла окна (считая от 0),

CL = столбец верхнего левого угла окна (считая от 0),

DH = строка нижнего правого угла окна (считая от 0),

DL = столбец нижнего правого угла окна (считая от 0).

-- Прерывание BIOS 10h, функция 07 – прокрутка экрана вниз (вставка чистых строк сверху):

Ввод: АН = 07,

Если AL = 00 – прокрутка всего экрана с заполнением пробелами (очистка), иначе AL = число строк для прокрутки вниз,

BH = атрибут вставляемого символа,

CH = строка верхнего левого угла окна (считая от 0),

CL = столбец верхнего левого угла окна (считая от 0),

DH = строка нижнего правого угла окна (считая от 0),

DL = столбец нижнего правого угла окна (считая от 0). Прерывание 10h также обеспечивает функции вывода данных на уровне BIOS:

-- Прерывание BIOS 10h, функция 08 – считать символ и атрибут символа в текущей позиции курсора:

Ввод: АН = 08, ВН = номер страницы.

Вывод: АН = атрибут символа,

AL = ASCII-код символа.

-- Прерывание BIOS 10h, функция 09 – вывести символ с заданным атрибутом на экран:

Ввод: АН = 09,

ВН = номер страницы,

AL = ASCII-код символа,

BL = атрибут символа,

СХ = число повторений символа.

Выводит на экран любой символ, включая специальные символы (например, CR).

-- Прерывание BIOS 10h, функция 0Ah – вывести символ с текущим атрибутом на экран:

Ввод: АН = 0Ah,

ВН = номер страницы,

AL = ASCII-код символа,

СХ = число повторений символа.

Выводит на экран любой символ, включая специальные символы (например, CR).

-- Прерывание BIOS 10h, функция 0Eh – вывести символ в режиме телетайпа:

Ввод: АН = 0Eh,

ВН = номер страницы,

AL = ASCII-код символа.

Символы CR (0Dh), LF (0Ah), BEL (07h) интерпретируются как управляющие символы.

Если текст при выводе выходит за пределы нижней строки, то экран прокручивается вверх.

-- Прерывание BIOS 10h, функция 13h – вывести строку символов с заданными атрибутами:

Ввод: АН = 13h

AL = режим вывода (биты):

-- 0 – переместить курсор в конец строки после вывода,

-- 1 – строка содержит не только символы, но также и атрибуты, так что каждый символ описывается двумя байтами: ASCII-код и атрибут,

-- 2 – 7 – зарезервированы.

СХ = длина строки (только число символов),

BL = атрибут, если строка содержит только символы,

DH = строка, начиная с которой будет выводиться строка символов,

DL = столбец, начиная с которого будет выводиться строка символов,

ES:BP = адрес начала строки в памяти.

Символы CR (0Dh), LF (0Ah), BEL (07h) интерпретируются как управляющие символы.

1. Обработка нажатия кнопок клавиатуры.

Обработка нажатий на клавиатуру может производиться различными способами:

-- с помощью прерываний ввода символов DOS;

-- с помощью прерываний ввода символов BIOS;

-- с помощью прямого доступа к буферу клавиатуры

-- с помощью доступа к портам ввода-вывода клавиатуры.

Ввод символов с помощью функций прерывания DOS 21h рассмотрен ранее в лабораторной работе №2. По сравнению с функциями DOS, прерывание BIOS 16h предоставляет больше возможностей для считывания данных и управления клавиатурой и такой доступ практически эквивалентен по производительности прямому доступу к буферу клавиатуры.

Каждой клавише на клавиатуре соответствует уникальный код, называемый скан-код. Этот код посылается клавиатурой при каждом нажатии и отпускании клавиши и обрабатывается BIOS – записывается в кольцевой буфер клавиатуры.

Функции прерывания 16h:

-- Прерывание BIOS 16h, функция 00h (10h, 20h) – чтение символа с ожиданием:

Ввод: АН = тип клавиатуры:

-- 00h – 83/84-клавиши,

-- 10h – 101/102-клавиши,

-- 20h – 122-клавиши. Тип клавиатуры можно определить с помощью функции 09h прерывания 16h.

Вывод: Если нажатой клавише соответствует ASCII-символ, то в АН возвращается код этого символа, а в AL – скан-код клавиши.

Если нажатой клавише соответствует расширенный ASCII-код, то в AL возвращается префикс скан-кода (например, Е0 для серых клавиш) или 0, если префикса нет, а в АН – расширенный ASCII-код.

-- Прерывание BIOS 16h, функция 01h (11h, 21h) – проверка наличия символа в буфере:

Ввод: АН = тип клавиатуры:

-- 01h – 83/84-клавиши,

-- 11h – 101/102-клавиши,

-- 21h – 122-клавиши.

Вывод: Если буфер пуст, то флаг ZF = 1;

Если в буфере присутствует символ, то флаг ZF = 0, а дополнительная информация содержится в регистре AX:

AL = ASCII-код символа, 0 или префикс скан-кода;

АН = скан-код нажатой клавиши или расширенный ASCII-код. Проверяемый символ остается в буфере клавиатуры.

-- Прерывание BIOS 16h, функция 02h (12h, 22h) – получить состояние клавиатуры:

Ввод: АН = тип клавиатуры:

-- 02h – 83/84-клавиши,

-- 12h – 101/102-клавиши,

-- 22h – 122-клавиши.

Вывод: AL = байт состояния клавиатуры 1,

АН = байт состояния клавиатуры 2 (только для функций 12h и 22h).

Байт состояния клавиатуры 1 (расположен в памяти DOS по адресу 0000h:0417h или 0040h:0017h):

-- бит 7 – Ins включена,

-- бит 6 – CapsLock включена,

-- бит 5 – NumLock включена,

-- бит 4 – ScrollLock включена,

-- бит 3 – Alt нажата (любая Alt для функции 02h, и только левая Alt для 12h/22h),

-- бит 2 – Ctrl нажата (любая Ctrl),

-- бит 1 – левая Shift нажата,

-- бит 0 – правая Shift нажата.

Байт состояния клавиатуры 2 (расположен в памяти DOS по адресу 0000h:0418h или 0040h:0018h): -- бит 7 – SysRq нажата,

-- бит 6 – CapsLock нажата,

-- бит 5 – NumLock нажата,

-- бит 4 – ScrollLock нажата,

-- бит 3 – правая Alt нажата,

-- бит 2 – правая Ctrl нажата,

-- бит 1 – левая Alt нажата,

-- бит 0 – левая Ctrl нажата.

Т.к. эти байты расположены в памяти по фиксированному адресу, то вместо вызова прерывания удобнее просто считывать и даже перезаписывать значения этих байт напрямую, что изменит состояние клавиатуры.

-- Прерывание BIOS 16h, функция 05 – поместить символ в буфер клавиатуры:

Ввод: АН = 05h,

СН = скан-код (можно поместить 0 вместо скан-кода, если функция, которая будет выполнять чтение из буфера, будет использовать только ASCII-код),

CL = ASCII-код.

Вывод: Если операция выполнена успешно, то AL = 00,

Иначе, если буфер клавиатуры переполнен, то AL = 01h.

К буферу клавиатуры также можно обратиться напрямую – буфер находится по адресу 0000h:041Eh и занимает 16 слов, по 0000h:043Dh включительно. Каждый символ хранится в буфере в виде слова, в таком же виде, как возвращает функция 01h прерывания INT 16h.

По адресу 0000h:041Ah находится адрес (ближний) по которому будет расположен следующий введенный символ (указатель на начало буфера), а по адресу 0000h:041Ch лежит адрес конца буфера. Т.к. буфер клавиатуры – закольцован, то если эти адреса начала и конца буфера равны, то буфер пуст.

Иногда буфер клавиатуры размещается в другой области памяти, тогда адрес его начала хранится в области данных BIOS по адресу 0480h, а конца – по адресу 0482h.

1. Доступ к системным часам.

Персональный компьютер содержит два устройства для управления процессами:

-- часы реального времени (RTC) – имеют автономное питание, используются для чтения/установки текущих даты и времени, установки будильника и для вызова прерывания IRQ8 (INT 4Ah) каждую миллисекунду;

-- системный таймер – используется одновременно для управления контроллером прямого доступа к памяти, для управления динамиком и как генератор импульсов, вызывающий прерывание IRQ0 (INT 8h) 18,2 раза в секунду.

Для видеоигры, создаваемой в данной лабораторной работе, указанные выше устройства лучше всего использовать на уровне функций DOS или BIOS как средство для определения текущего времени, организации задержек и формирования случайных чисел. Управление часами RTC и внутренними часами операционной системы средствами DOS:

-- Функция DOS 2Ah (INT 21h) – считать дату:

Ввод: AH = 2Ah

Вывод: СХ = год (1980 – 2099),

DH = месяц,

DL = день,

AL = день недели (0 – воскресенье, 1 – понедельник и т.п.).

-- Функция DOS 2Bh (INT 21h) – установить дату:

Ввод: АН = 2Bh,

СХ = год (1980 – 2099),

DH = месяц, DL = день.

Вывод: Если введена несуществующая дата, то АН = FFh,

Если дата установлена, то АН = 00h.

-- Функция DOS 2Ch (INT 21h) – считать время:

Ввод: AH = 2Ch

Вывод: СН = час,

CL = минута,

DH = секунда,

DL = сотая доля секунды.

-- Функция DOS 2Dh (INT 21h) – установить время:

Ввод: АН = 2Dh,

СН = час,

CL = минута,

DH = секунда,

DL = сотая доля секунды.

Вывод: Если введено несуществующее время, то AL = FFh,

Если время установлено, то AL = 00h.

BIOS позволяет управлять часами PTC напрямую:

-- Прерывание BIOS 1Ah, функция 02h – считать время RTC:

Ввод: АН = 02h

Вывод: CF = 1 – если часы не работают или попытка чтения пришлась на момент обновления,

CF = 0 – если время успешно считано, то:

СН = час (в формате BCD),

CL = минута (в формате BCD),

DH = секунда (в формате BCD),

DL = 01h – если действует летнее время,

DL = 00h - если не действует летнее время.

-- Прерывание BIOS 1Ah, функция 03h – установить время RTC:

Ввод: АН = 03h,

СН = час (в формате BCD),

CL = минута (в формате BCD),

DH = секунда (в формате BCD),

DL = 01h - если используется летнее время,

DL = 00h - если не используется летнее время.

-- Прерывание BIOS 1Ah, функция 04h – считать дату RTC:

Ввод: АН = 04h

Вывод: CF = 1 – если часы не работают или попытка чтения пришлась на момент обновления,

CF = 0 – если дата успешно считана, то:

СХ = год (в формате BCD, например, 1998h для 1998-го года),

DH = месяц (в формате BCD),

DL = день (в формате BCD).

-- Прерывание BIOS 1Ah, функция 05h – установить дату RTC:

Ввод: АН = 05h,

СХ = год (в формате BCD),

DH = месяц,

DL = день.

BIOS отслеживает каждый отсчет системного таймера с помощью своего обработчика прерывания IRQ0 (INT 8h) и увеличивает на 1 значение 32- битного счетчика, который располагается в памяти по адресу 0000h:046Ch, причем при переполнении этого счетчика байт по адресу 0000h:0470h увеличивается на 1. Программа может считывать значение этого счетчика в цикле (например, просто командой MOV) и таким образом организовывать задержки, (например, пока ждать пока счетчик не увеличится на 1 (минимальная задержка будет равна приблизительно 55 микросекундам)). Для работы со счетчиком времени в BIOS есть функции:

-- Прерывание BIOS 1Ah, функция 00h – прочитать значение счетчика времени:

Ввод: АН = 00h.

Вывод: CX:DX = значение счетчика,

AL = байт переполнения счетчика.

-- Прерывание BIOS 1Ah, функция 01h – установить значение счетчика времени:

Ввод: АН = 01h,

CX:DX = значение счетчика.

Для изменения частоты работы таймера, BIOS имеет специальные функции:

-- Прерывание BIOS 15h, функция 86h – формирование задержки таймера:

Ввод: АН = 86h,

CX:DX = длительность задержки в микросекундах.

Вывод: Если таймер был занят, то CF = 1,

Если задержка выполнена, то CF = 0:

AL = маска, записанная обработчиком в регистр управления прерываниями.

-- Прерывание BIOS 15h, функция 83h – управление работой счетчика:

Ввод: АН = 83h,

AL = 1 – прервать счетчик,

AL = 0 – запустить счетчик:

CX:DX = длительность задержки в микросекундах,

ES:BX = адрес байта, старший бит которого по окончании работы счетчика будет установлен в 1.

Вывод: Если таймер был занят, то CF = 1,

Если задержка выполнена, то CF = 0:

AL = маска, записанная обработчиком в регистр управления прерываниями.

**Код программы**

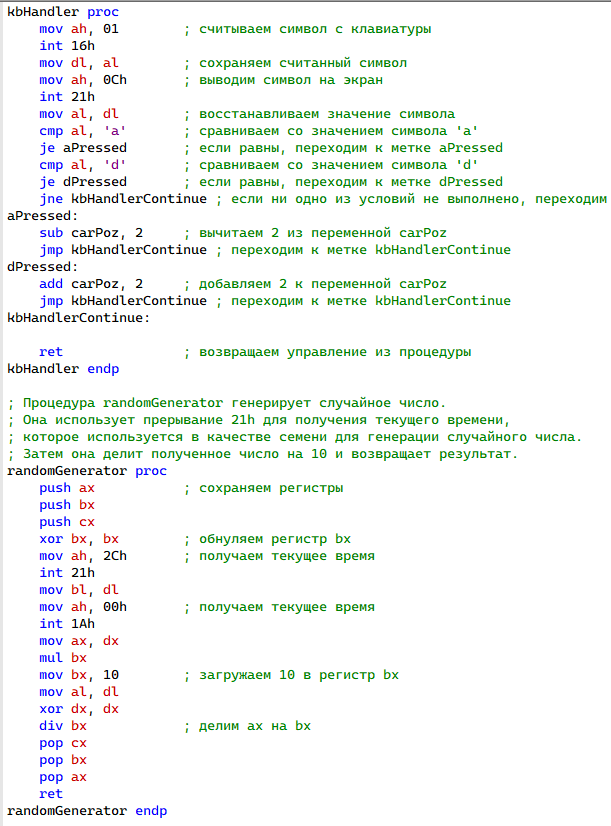
****

Рисунок 1 – код программы



Рисунок 2 – код программы

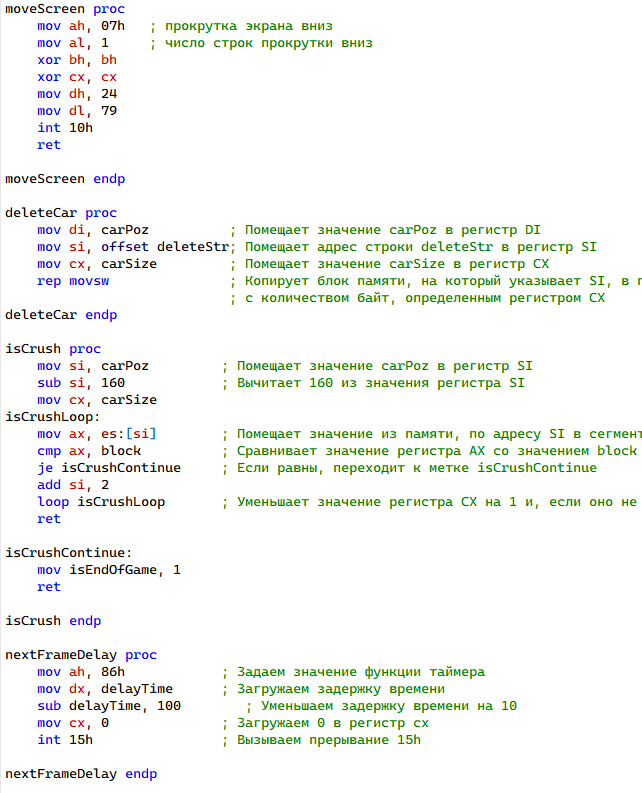


Рисунок 3 – код программы

**Вывод программы**

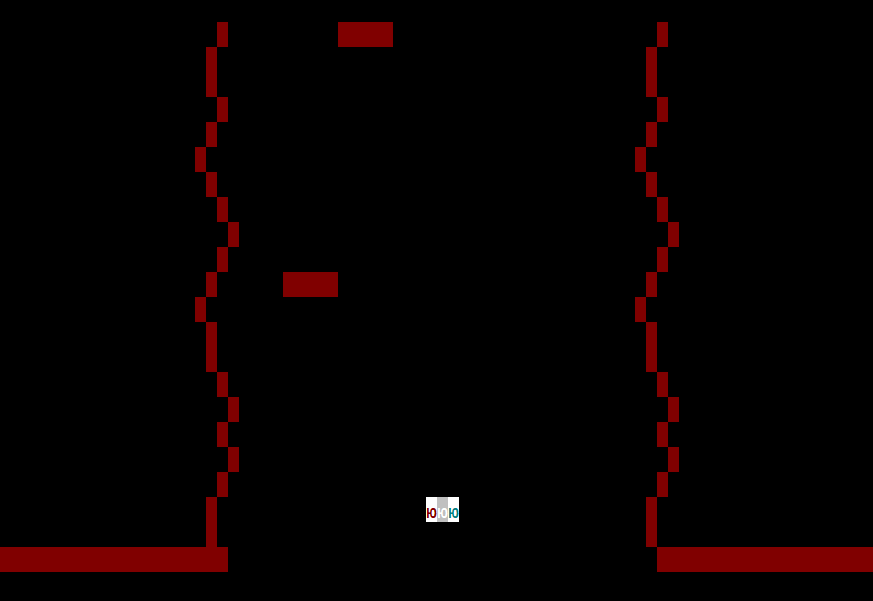
****

Рисунок 4 – Результат работы программы