Федеральное агентство связи

Ордена трудового Красного Знамени федеральное государственное бюджетное

образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра Математической кибернетики и информационных технологий

Лабораторная работа № 3

«Изучение архитектуры микропроцессора i8086/88»

вариант 1

Выполнил: студент группы БВТ1904 Абакаров Г. Г.

Проверила: Анастасия Изотова

Москва, 2020

**Цель работы:** изучение архитектуры, способов адресации и системы команд микропроцессора i8086/88. Составление программы на языке Ассемблера, использующей команды пересылки данных для вывода сообщения на экран терминала.

**Задание:**

1. изучить архитектуру, способы адресации и систему команд микропроцессора;
2. написать программу на языке Ассемблера для вывода строки с фамилиями студентов данной бригады на экран терминала в текстовом режиме;
3. скомпилировать программу с помощью Ассемблера;

**Указания по выполнению работы:**

Программа, предназначенная для трансляции в EXE-формат исполняемого файла, должна содержать:

1. объявление сегмента данных оператором «DATASG SEGMENT PARA», внутри которого с помощью оператора DB должен быть определен массив байт, содержащий ASCII-коды символов строки с фамилиями студентов команды, а также байты атрибутов цвета символов согласно варианту задания. Описание сегмента данных завершается оператором «DATASG ENDS»;
2. объявление сегмента кода оператором «CODESG SEGMENT PARA»;
3. объявление процедуры оператором «BEGIN PROC FAR»;
4. оператор «ASSUME CS:CODESG,DS:DATASG», устанавливающий сегментные регистры CS и DS при запуске программы;
5. команды инициализации, характерные для любой программы EXE-формата:

push ds

mov ax, 0

push ax

mov ax, DATASG

mov ds, ax

1. команды, выполняющие «полезную работу» по выводу в текстовом режиме сообщения на экран терминала:

- запись в регистр дополнительного сегмента адреса сегмента видеобуфера (0B800h);

- запись в индексный регистр источника смещения массива из сегмента данных программы;

- запись в индексный регистр приемника начального смещения в видеобуфере согласно варианту задания;

- сброс флага направления;

- загрузка в регистр-счетчик числа пересылаемых байт;

- Пересылка строки байт в видеопамять (команда movsb с префиксом повторения);

- завершение программы (команда ret)

1. операторы, завершающие описание процедуры «BEGIN ENDP», сегмента кода «CODESG ENDS» и программы в целом «END BEGIN».

В текстовом режиме видеобуфер имеет размер в 4000 байт. При этом каждый четный байт содержит ASCII-код символа в соответствующем знакоместе экрана, а каждый нечетный сегмент – соответствующий байт атрибутов символа. Отдельные биты байта атрибутов задают цвет переднего плана и цвет фона знакоместа, на котором изображается символ:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Разряд | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Функция | Цвет фона | | | | Цвет переднего плана | | | |
| Значение | Мерцание | R | G | B | I | R | G | B |

Здесь R – включение красного цвета;

G – включение зеленого цвета;

B – включение синего цвета;

I – включение повышенной яркости символов.

Задавая определенную комбинацию бит в байте атрибутов, можно использовать 16 различных цветов для переднего плана и 8 цветов для фона знакоместа.

Экран организован как прямоугольное поле, состоящее из 25 рядов знакомест по 80 символов в каждом ряду. При этом в видеобуфере байты первого символа (крайнего слева) следующей строки следуют непосредственно за байтами последнего (крайнего справа) символа предыдущей строки.

|  |  |  |  |
| --- | --- | --- | --- |
| № варианта | Строка экрана | Начальная позиция в строке | Цвет символов |
| 1. | 1 | 56 | Ярко-красный по черному |

Код программы

.model **small**

.**stack** 100h

DATASG **SEGMENT** PARA

color **equ** 12 *; цвет ярко красный на черном фоне*

string **db** "A", **[**color**]**, "b", **[**color**]**, "a", **[**color**]**, "k", **[**color**]**, "a", **[**color**]**, "r", **[**color**]**, "o", **[**color**]**,\

"v", **[**color**]**, " ", **[**color**]**, "G", **[**color**]**, "a", **[**color**]**, "s", **[**color**]**, "a", **[**color**]**,\

"n", **[**color**]**, ",", **[**color**]**, " ", **[**color**]**, "B", **[**color**]**, "a", **[**color**]**, "s", **[**color**]**,\

"k", **[**color**]**, "a", **[**color**]**, "e", **[**color**]**, "v", **[**color**]**, "a", **[**color**]**, " ", **[**color**]**,\

"M", **[**color**]**, "a", **[**color**]**, "r", **[**color**]**, "i", **[**color**]**, ",", **[**color**]**, " ", **[**color**]**,\

"B", **[**color**]**, "a", **[**color**]**, "t", **[**color**]**, "u", **[**color**]**, "e", **[**color**]**, "v", **[**color**]**,\

" ", **[**color**]**, "K", **[**color**]**, "i", **[**color**]**, "r", **[**color**]**, "i", **[**color**]**, "l", **[**color**]**,\

"l"

strlen **equ** **$**-string

DATASG ENDS

CODESG **SEGMENT** PARA

ASSUME **DS**:DATASG

ASSUME **CS**:CODESG

BEGIN PROC PARA

**push** **ds**

**mov** **ax**, 0

**push** **ax**

**mov** **ax**, DATASG

**mov** **ds**, **ax**

**mov** **ax**, 0B800h

**mov** **es**, **ax**

**lea** **si**, string

**mov** **di**, 80+56

**cld**

**mov** **cx**, **[**strlen**]**

**rep** **movsb**

**ret**

BEGIN ENDP

CODESG ENDS

END BEGIN