ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ЭКОНОМИКИ И СЕРВИСА

Микропроцессорные системы

Программирование INTEL 8086
Системная программа Debug

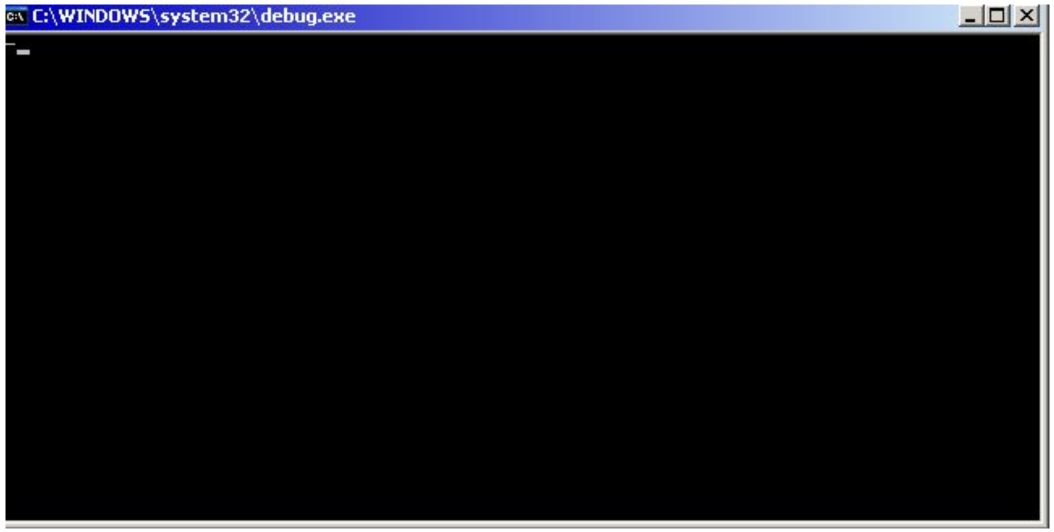
Запуск программы Debug

Для запуска программы Debug

В DOS - набрать debug в командной строке

В Windows – Нажать Пуск -> Выполнить -> набрать debug в командной строке

Для перехода из оконного режима в полноэкранный и обратно нажимайте Alt+Enter



```
-_
```

Окно программы Debug в среде Windows

Команды Debug: a

- Перевод мнемонических машинных кодов в числовой код(ввод программы в память)**(a)**ssembler

- **a[нач.адрес]**

  нач.адрес указывает адрес, с которого программа располагается в памяти

Команды Debug: d

- Вывод на экран содержимого участка памяти(**d**)unp

- **d[нач.адрес] L [число]**

  нач.адрес указывает на первый байт сегмента памяти, который надо вывести на экран;

  число задает размер участка памяти, который надо просмотреть

Команды Debug: e

- Ввод данных в оперативную память компьютера**(e)**nter

- **e[нач.адрес] [список значений]**

  нач.адрес указывает на первый байт сегмента памяти; список значений задает вводимые значения в шестнадцатерчином виде через пробел

Команды Debug: t

- Выполнение одной (нескольких) команд в пошаговом режиме**(t)**race

- **t=[Нач.адрес] [число]**

нач.адрес указывает адрес команды.

число задает количество команд которые требуется выполнить

Hex: **H value1 value2** Простой калькулятор для сложений и вычитаний шестнадцатиричных чисел. Можно ввести два числа, каждое не более чем по 4 цифры. Сначала debug выведет сумму этих чисел, а затем разность.

Search: **S range list**
Searches within a range of addresses for a pattern of one or more byte values given in a list. The list can be comprised of numbers *or character strings enclosed by matching single or double quote marks.* [NOTE: In the examples below, if you do find the same data on your computer, the locations could easily vary from ours!]

Compare**: C range address**
Compares two blocks of memory. If there are no differences, then DEBUG simply displays another prompt (-).

Fill: **F range list** This command can also be used to *clear* a whole segment of Memory as well as *filling* smaller areas with a continuously repeating phrase or single byte.

Go: G [=address] [addresses]
Go is used to run a program and set *breakpoints* in the program's code.

Unassemble:* U [range]** ***Disassembles machine instructions into 8086 Assembly code. **Without the optional [range]**, it uses **Offset 100** as its starting point, disassembles about 32 bytes and then remembers the next byte it should start with if the command is used again.

Input: I port
Output: O port byte
The use of I/O commands while running Windows™9x/Me is *just plain* unreliable! This is especially true when trying to directly access hard disks! Under Win NT/2000/XP, the I/O commands are only an ***emulation***; so don't trust them. Though the example below still works under Win2000/XP, it's most likely using some WinAPI code to show what's in the Windows clock area; *not* directly from an RTC chip.

Load: L [address] [drive] [firstsector] [number]
or program!* (See the N command for more on this)** This command will LOAD the selected number of sectors from any disk's Logical Drive under the control of MS-DOS or Windows into Memory. The **address** is the location in Memory the data will be copied to (use only 4 hex digits to keep it within the memory allocated to DEBUG), the **drive** number is mapped as: 0=A:, 1=B:, 2=C:, etc., **firstsector** counts from ZERO to the largest sector in the volume *and finally* **number** specifies **in hexadecimal** the ***total **number** of sectors that will be copied into Memory (so a floppy disk with 0 through 2,879 sectors would be: 0 through **B3F** in Hex).

Move: M range address This command should really be called: COPY (not Move) as it actually *copies* all the bytes from within the specified **range** to a new **address**.

Name: N [pathname] [arglist]
This command can be used to load files into DEBUG's Memory *after* you have started the program, but it's main function is to create a new file under control of the Operating System which DEBUG can WRITE data to.

Register: R [register] Entering ' r ' all by itself will display *all* of the 8086 register's contents *and* the next instruction which the IP register points to in both machine code and an unassembled (Assembly Language) form.
Proceed: P [=address] [number] *Proceed* acts exactly the same as Debug's T (Trace) command for most types of instructions... EXCEPT: *Proceed* will immediately execute ALL the instructions (rather than stepping through each one) inside any Subroutine CALL, a LOOP, a REPeated string instruction or any software INTerrupts. This means that you do not have to single-step through any of the code contained in a Subroutine or INT call if you use the Proceed (P) command.

This means *Proceed* will be the command you use most often to debug programs, and Trace will only be used to step into a Subroutine or possibly check the logic of the first few iterations of a LOOP or REP instruction.
Write: W [address] [drive] [firstsector] [number]
The WRITE (W) command is often used to save a program to your hard disk from within DEBUG. But the only safe way to do so, especially under Windows, is by allowing the OS to decide where to physically create that file on the disk. This is done by first using the Name (N) command to set up an *optional path* and **filename** for the new file (or to overwrite one that already exists).