Laboratory No 6

Programming of the x86 family microprocessors by MS-DOS software tools

1. The aim of the work

To acquaint with MS-DOS programs *Debug* and *Code View* and to learn how to use them studying the instruction set of the x86 family microprocessors, in creating and debugging simple assembler programs.

2. Short description of programs

2.1. Program *Debug*

MS-DOS program *Debug* is intended for error check in the program execution and object files (∗.exe, ∗.obj). The program also allows the revision and change of contents of memory and registers, writing simple programs in the assembler language and using the unassembler, i.e. changing hexadecimal program codes into mnemonic ones.

The program can execute 19 user commands (see Table 1). The following syntax was applied to their description:

[ ] – angle brackets between which the written syntax element is not necessary.

Address – the memory cell address “segment:displacement”. If the segment is not indicated, its address is taken from register **CS**.

Range – the memory field is indicated by the initial address and the field length in bytes. For example, **DS:100 L10H** means the memory field of 16 byte length beginning with address **DS:100**.

Drive – a hexadecimal number showing a disk drive: 0 = A:, 1 = B:, etc.

List – one or more bytes written by a hexadecimal system, separated by spaces or the symbol sequence in quotation marks.

Register – the register name, e.g. **AX**.

First sector – a hexadecimal number showing the number of the initial sector in the disk.

Number – a hexadecimal number showing the number of sectors or commands.

Value – a hexadecimal number.

Table 1. Commands of the program *Debug*

Command	Purpose	Format
?	Help	–
Assemble	Changes the assembler mnemonic code to the hexadecimal code	**A** [address]
Compare	Checks differences between two memory ranges	**C** range address
Dump	Dumps the memory range content. If the range is not indicated, dumps 128 bytes from the last previously shown position	**D** [range]
Enter	Information input (byte list or text between quotation-marks) to memory beginning with the indicated address. If the list is not formed, it is entered in the memory edition mode, i.e. by pressing space key the memory cell content is taken out and it is allowed to change it	**E** address [list]
Fill	Fills the memory range with the byte list or by a symbol line written between quotation-marks	**F** range list
Go	Starts the program for execution beginning with the indicated address after setting the interrupt point	**G** [=address] [addresses]
Hex	Sum and difference of two hexadecimal numbers	**H** value1 value2
Input	Information input through input port	**I** port address
Load	Information input from a disk drive not indicating the file name	**L** [address] [drive] [firstsector] [number]

Table 1 continuation.

Load	From a disk drive reads file indicated by command **N** enters it to the random access memory beginning with the indicated address	**L** [address]
Move	Moves the indicated information amount from one memory place to another place	**M** range address
Name	Names the file and indicates a disk drive, which further applies command **L** or **W**. By default a disk drive A is taken	**N** [drive:] file name, e.g., N b:\test. txt
Output	Information output through a port	**O** port address
Proceed	Executes the selected number of commands beginning with the indicated address. If address is not indicated, the command whose address is **CS:IP** is executed. If the number of executed commands is not indicated, only one command is executed. (Interrupt and subroutine call commands are regarded as one)	**P** [=address] [number]
Register	Reads the register content. If a concrete register name is indicated, the content of that register is read and it is allowed to change it	**R** [register]
Search	Searches for the byte list or the text indicated between quotation-marks in the indicated memory range	**S** range list
Trace	Executes the indicated number of commands beginning with the selected address. If the address is not indicated, the command which address is **CS:IP** is executed. If the number of executed commands is not indicated, only one command is executed	**T** [=address] [value]
Unassemble	Unassembler. Conversion of hexadecimal codes of the indicated memory range to the assembler mnemonic codes	**U** [range]

Table1 end.

Write	Writing of information into a disk drive without indication of the file name	**W** [address] [drive] [firstsector] [number]
Write	Writing of information, which quantity in bytes is indicated in register **CX**, beginning with the selected address, into a file indicated in command **N**.	**W** [address]
Quit	Quit from the program	**Q**

Marking of the flag register **FL** states:

CY (C = 1) – carry from (to) bit 7 occurred;

NC (C = 0) – there was no carry;

OP (P = 0) **–** in the operation result the uneven number of equal bits;

PE (P = 1) – in the operation result the even number of equal bits;

NA (A = 0) – there was no carry from (to) bit 4;

AC (A = 1) – there was carry from (to) bit 4;

NZ (Z = 0) – the operation result is not equal to zero;

ZR (Z = 1) – the operation result is equal to zero;

PL (S = 0) – the operation result is positive;

NG (S = 1) – the operation result is negative;

DI (I = 0) – interrupts are unallowable;

EI (I = 1) – interrupts are allowable;

UP (D = 0) – in operations with data lines the content of registers **SI** and **DI** is increased;

DN (D = 1) – in operations with data lines the content of registers **SI** and **DI** is decreased;

NV (O = 0) – there was no overflow during the arithmetical operation;

OV (O = 1) – there was overflow during the arithmetical operation.

2.2. Program *CodeView*

MS-DOS program *CodeView* alongside with the main purpose to debug compiled program files and search for errors can be used in the investigation of instruction set of the x86 family microprocessors as well in the creation and debugging of simple assembler programs. Contrary to program *Debug*, program *CodeView* is applied to instructions and registers not only of 16, but also 32 bits (of Intel® 80386 and later generation microprocessors). It has a more convenient user interface. The main *CodeView* commands: **A** – assembler, **U** – unassembler, **D** – memory review, **R** – register review, **T** – execution of instructions by steps, **E** – data input, etc. are the same as in program *Debug*, but the program also has additional possibilities. The edited or created program can be executed not only by steps or nonstop but also in the animation mode by selecting the animation speed and the program work can be observed in some windows simultaneously; in later version programs the register contents can be modified not only from the command line but also in the register window using a mouse and keyboard; the possibility to put the program interrupt points or to execute the program up to the site shown by cursor, etc. is foreseen.

CodeView has some horizontal menu items. The desired *CodeView* operating mode can be chosen from vertical lists of each item. Further information about *CodeView* and its commands is presented in the help (key F1).

3. Task

1. To type-in the command *Debug* in MS-DOS command line and start the program of the same name.

2. To test the execution of the main commands of *Debug* program. If necessary, use the help (key “?”).

3. Using command **E** (Enter) beginning with address CS:100, to enter hexadecimal codes of programs created in laboratory No 5 and

in a step mode to check the results of their execution (command **T** – Trace).

4. To process the hexadecimal codes of assembler commands of item 3 by unsassembler (command **U** – Unassemble). To write down the information output by the unassembler and at each command to give comments about the modes used in the operand addressing in random access memory.

5. To create and execute the assembler program to solve the arithmetical expression presented by the lecturer.

6. To start program *CodeView* (cv.bat in folder CODEVIEW). By means of command **N** to select the desirable calculation system (8, 10, 16).

7. To get acquainted with the purpose of the program and its main commands.

8. To prepare *CodeView* for work with the 32-bit instructions and registers.

9. By means of command **A** (Assembler), beginning with address CS:100, to enter the program obtained in item 4 of the task by changing names of registers in it by corresponding names of 32-bit registers.

10. To execute the program created according to item 9 of the task in a step mode (command **T** – Trace), by applying, where possible, 32-bit operands.

11. To write down the program of item 9 of the task or to save it in a floppy. At each command to give comments about the modes used in the operand addressing in random access memory.

12. To compare programs of items 4 and 10 of the task and draw conclusions.

13. To create the program for the Intel® 80386 processor which would periodically add a unit to the accumulator content until the sum reaches the selected number of cycles. To execute the program in the animation (Animate) mode by choosing the proper rate of animation.

4. Contents of the report

1. The aim of the work.
2. Text of the assembler program with 16-bit instructions and comments.
3. Text of the assembler program with 32-bit instructions and comments.
4. Texts of the assembler programs created by themselves.
5. Conclusions.

5. Test questions

1. The purpose of programs *Debug* and *CodeView*.
2. Enumerate the main differences of programs *Debug* and *CodeView*.
3. Concepts of the assembler and unassembler.
4. How do commands of microprocessors Intel® 8086 and Intel® 80386 differ?
5. Write the instruction presented by the lecturer mnemonically.
6. What does command **N** mean in programs *Debug* and *CodeView*?
7. Enumerate commands having the same mnemonics and meaning of programs *Debug* and *CodeView*.

References

1. Berger A. S. Hardware and Computer Organization. USA, Burlington: Newnes; Book & DVD Edition. May 6, 2005. 512 p.

2. Brey B. B. The Intel Microprocessors 8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, Pentium Pro Processor, Pentium II, Pentium III, Pentium 4. Architecture, Programming and Interfacing. USA, New Jersey: Pearson Prentice Hall; 7th Edition. March 23, 2005. 912 p.

3. Triebel W. A. The 8088 and 8086 Microprocessors: Programming, Interfacing, Software, Hardware and Applications. USA, New Jersey: Pearson Prentice Hall; 3rd Edition. August 29, 2002. 1040 p.

4. Uffenbeck J. The 80x86 Family: Design, Programming and Interfacing. USA, New Jersey: Pearson Prentice Hall; 3rd Edition. February 14, 2001. 678 p.

5. Antonakos J. L. Introduction to the Intel Family of Microprocessors: A Hands-On Approach Utilizing the 80x86 Microprocessor Family. USA, New Jersey: Pearson Prentice Hall; 3rd Edition. June 3, 1998. 768 p.

6. Gražulevičius A. Mikroprocesoriai. Laboratorinių darbų užduotys ir metodikos nurodymai. Vilnius: Technika, 2000. 60 p.