МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Запорізький національний технічний університет

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт здобувачами першого (бакалаврського) рівня вищої освіти з англійською мовою навчання
за спеціальністю 141 – «Електроенергетика, електротехніка та електромеханіка»
(освітня програма «Електричні та електронні апарати»)
при вивченні навчальної дисципліни
«Основи мікропроцесорної техніки»

2024

Методичні вказівки до виконання лабораторних робіт здобувачами першого (бакалаврського) рівня вищої освіти з англійською мовою навчання за спеціальністю 141 – «Електроенергетика, електротехніка та електромеханіка» (освітня програма «Електричні та електронні апарати») при вивченні навчальної дисципліни «Основи мікропроцесорної техніки» / Укл.: Л. Б. Жорняк, С. В. Войтенко. – Запоріжжя: НУ «Запорізька політехніка», 2024. – 78 с.

Укладачі: Л. Б. Жорняк, доцент, к.т.н.
С. В. Войтенко, ст. викладач

Рецензент: В. В. Василевський, доцент, к.т.н.

Відповідальний за випуск: Р. Е. Мохнач, завідувач лабораторією кафедри ЕЕА

Затверджено
на засіданні кафедри
«Електричні та
електронні апарати»
Протокол № _7_
від «_26_» грудня 2023р.

Затверджено НМК ЕТФ
Протокол № _5_
від «_25_» січня 2024 р

CONTENT

INTRODUCTION

The method of laboratory works is widely used in the system of work on students' perception and assimilation of new material.

Laboratory work is a method of learning in which students work under the guidance of a teacher and according to a pre-prepared plan experiments (testing) or perform certain practical tasks. And during their implementation, students can perceive and understand new educational material.

To perform laboratory work, it is necessary to understand the new curriculum and material containing the following methodological techniques:

– determination of the topic and purpose of classes and tasks of laboratory work;

– brief theoretical information;

– determination of the procedure for performing laboratory work and its individual stages;

– direct performance of laboratory work by students and teacher control over the progress of classes and monitoring of safety equipment;

– summing up the results of laboratory work and formulating the main conclusions.

The purpose of these laboratory works is to develop students' skills in practical work with a microprocessor, consolidate and concretize the acquired theoretical knowledge, and fully realize the connection between theory and practice in teaching.

Methodical instructions for the organization and conduct of laboratory work were developed in accordance with the work program of the educational component of ВК 08 "Fundamentals of Microprocessor Technics" and are intended for the performance of laboratory work by students.

Laboratory works on the academic discipline are aimed at assimilating knowledge, mastering skills and forming elements of general competencies provided for by the work program of the academic discipline. As a result of mastering the academic discipline, the student should

be able:

– to work with microprocessor systems;

– to program microprocessor systems;

to know:

– purpose, functions, characteristics and composition of microprocessor systems;

– command systems, features of the organization of the interrupt system of microprocessor systems;

– MP memory organization and access to it.

The work program involves students completing

practical classes, including practical training as a mandatory component tasks using a personal computer.

The main function performed by laboratory work is to teach students practical skills in working with computer equipment.

Analyzing the content of laboratory work on the basics of microprocessor technics, it is easy to notice that the methods act as generalized skills. To perform any laboratory work on basics of microprocessor technics, it is necessary to study the features functioning of the microprocessor, carry out calculations and analyze operation modes.

Planning of laboratory work is carried out using methodological recommendations for conducting laboratory work on the basics of microprocessor technology.

The methodological recommendations include:

– topic of laboratory work from the program on the subject "Fundamentals of Microprocessor technics";

– the purpose of the laboratory work (it must be taken into account that the formulation of the goals are often vague and do not focus students on specific activities);

– brief theoretical principles (in this part, the manual for laboratory work duplicates the content of the textbook);

— list of equipment and devices for laboratory research operation;

– basic (installation) diagram of laboratory research;

– the order of execution, a brief description of the methods of activity of students, forms of presentation of measurement results (tables, diagrams, graphic arts);

– conclusions from the work;

– self-test questions.

This structure of methodological recommendations is currently used in all types of educational institutions (colleges, technical schools, universities). At the same time, it must be admitted, she organizes the activities of students, but does not reveal the logical sequence of operations and laboratory research methods.

Currently, students are performing laboratory work on educational microprocessor set (УМПК-80) based on the КР580ВМ80А microprocessor (Intel8080). The laboratory setup consists of a training microprocessor set (УМК), a set of modules connected to its system bus and various peripheral devices. УМК presents is an educational microcomputer intended for studying programming, design and configuration of microprocessor devices and systems, made on MP КР580ВМ80. The development of information technology has led to the emergence of the concept "Virtual Laboratory Workshop" (VLW), which is based on computer simulation. The main ways of using VLW in the educational process:

– as a computer "simulator" for preparing for practical work in a real laboratory (in this case, the programs of computer and physical experiments are, as a rule, the same);

– as an addition to a real workshop, providing for such computer experiments that for various reasons (technical, financial, organizational, etc.) cannot be implemented on physical equipment.

Using VLW as a computer "simulator" allows it is better for the student to prepare for conducting a physical experiment, to better understand the effects being studied, and to acquire skills in working with measuring instruments (if the virtual workshop includes computer models of measuring instruments, similar in their properties to the properties of real instruments). Typically, this approach can be recommended for distance learning students, since it not only contributes to better assimilation of the material being studied, but also allows them to reduce the duration of classes in real life laboratories during their stay within the walls of the educational institution.

The development of computer network technologies has led to the emergence of a laboratory workshop, implemented in remote access mode to any equipment. Considering that providing remote access to any equipped premises is associated with many problems (the need connecting a laboratory prototype to a PC, for reliable protection consequences of emergency situations, low efficiency of equipment use due to the inability in some cases to implement collective access, etc.), intelligent technologies have a sufficient number a lot of secrets. However, it also has a right to exist and in some cases has obvious advantages over VLW.

An important question is whether VLW is an alternative to real laboratory practice. On the one hand, modern computer simulation

technologies make it possible to create virtual interfaces of real laboratory equipment that reproduce both the appearance and its parameters with very low deviations. On the other hand, maintaining and timely updating laboratory equipment, including measuring instruments, requires significant financial resources. However, any, even the highest quality, VLW, in most cases does not replace a teacher, who has work with real equipment.

One of these VLWs is the software model of the УМПК-80 stand emulator, for which laboratory work topics were developed.

Criteria for laboratory work evaluating are as follows: when assessing acquired skills when performing practical work the pass/fail grading scale is used.

Assessment of practical exercises/laboratory work is carried out in accordance with the following rules:

−regulations on constant monitoring of academic performance and intermediate student certification;

−regulations on planning, organizing and execution of laboratory research work and practical classes.

1 LABORATORY WORK No. 1
EDUCATIONAL УМПК-80 MICROPROCESSOR SET

The purpose of work is to study the device of an educational microprocessor set УМПК-80 and bodies of its control, input, indication.

Subject of a research: an educational microprocessor set УМПК-80, its opportunities and control devices.

1.1. Assignment and characteristics of УМПК 80 set

УМПК represents the completed microcomputer and is intended for preparation of the experts in the field of microprocessor engineering by studying of structure and bases of programming of the microprocessor КР580ВМ80А. УМПК can be used as the controlling computer at creation and research of operation of control systems of electrical devices and processes. It is an easily Make ed and convenient tool for debugging small (up to 2 Kilobytes) programs of the user. The means of indication on a front panel allow to observe processes of transformation and information transfer during operation of УМПК.

Table 1.1–Technical characteristics of УМПК set

Technical characteristics of УМПК-1	Parameter
Type of the used microprocessor	КР580ВМ80А
Volume of the random-access memory	2 Kbytes
Volume of the read-only memory	2 Kbytes
Possibility of interruption	1 vector
The software	System program "Monitor"
Voltage of power supply	220 V ± 22 V with the frequency 50 Hz ± 1 Hz
Input and output signal levels are compatible with TTL IC levels	

1.2 УМПК 80 set configuration

УМПК set consists of the following components (Figure 1.1):
– microcomputer;
– operator board;
– supply unit.

Microcomputer is one of the main parts and it controls the operation of the УМПК set, performing memory access, input, output and information display. The basis of a microcomputer is an operating unit (OU), which consists of a microprocessor (MP) and a clock frequency generator (CFG).

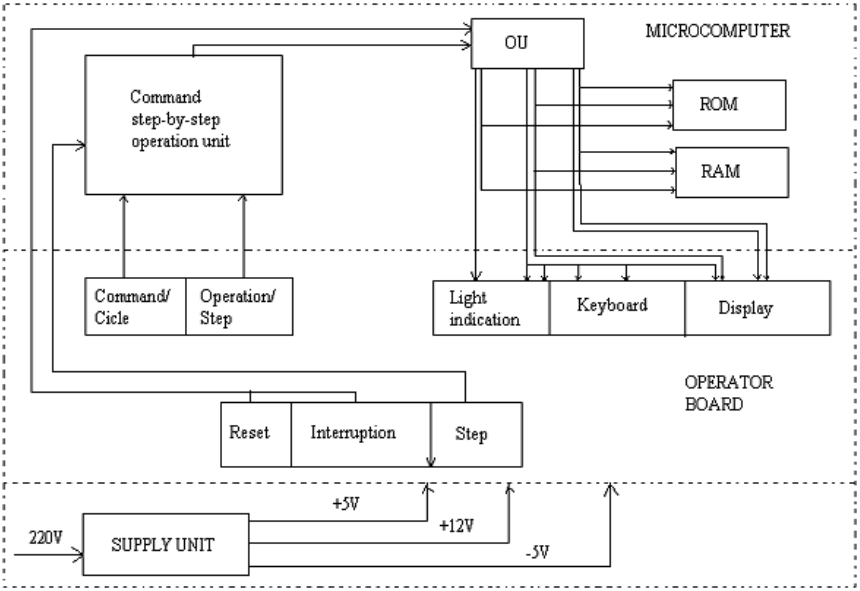


Figure 1.1 – Components of the УМПК set

The KP580BM80A microprocessor is used to execute a specific set of instructions and is implemented on a single IC, which includes a storage device and several general purpose registers (RGP). A register should be understood as a minimum storage element consisting of 8 digits (bits). The information stored in such a register is called a word. RAM registers are usually called cells to which a specific number-address is assigned. These registers are used to store data and user commands and are designed as a separate BIC. The microprocessor can address them

using special commands. Accumulator (A) is the most important MP recorder. The operation of УМПК consists in the data exchange between registers and is described by the system of commands, each of which defines a certain type of data exchange and conversion.

1.2.1. Architecture of УМПК set memory

Let's consider the structure of the УМПК registers available to the user based on MP KP580BM80A (Figure 1.2). The address of all registers inside the УМПК is carried out in the binary numbering system, while it is convenient for the user to use the reserve hexadecimal system to shorten the record. Next, we will use the reserved hexadecimal system for the address label and register contents. The word size of MP KP580 is 8 bits (1 Byte), so the bit size of all abbreviated registers, except for the SP (Stack Pointer) index and the PC (Program Counter) command counter, is also 1 Byte. MP KR580 has one register (accumulator) A and six RGPs: B, C, D, E, H, L. RGPs can be combined in two-byte registers BC, DE, HL. At the same time, registers B, D, H contain the high Byte, and registers C, E, L contain the low Byte of the hexadecimal number (word). The MP also contains the condition flag register F, which uses 5 of the 8 digits.

The execution of programs requires, as a rule, access to the base memory and buffer registers of the input block (IB). Buffer registers are used to coordinate the operation of the MP and peripheral devices (printer, display, transmitters, measuring devices, etc.). The memory access to all registers occurs because a large amount of data cannot be located in the RGP, and the program must have communication devices with the "outside world" to receive and receive information. Therefore, it is necessary to imagine the memory access amount and IU registers.

The memory address space (field) is a set of memory cells that can be accessed by the MP. The MP KP580 address buffer register is 16-bit, so the memory field contains 2^{16} = 65536 cells (64 Kilobytes, since 1 Kilobyte = 1024 Bytes) capacity of each 1 Byte with the corresponding address from 0000 up to FFFF. This real microcomputer set can use not the complete address field, but only a part of it. The set of memory cells actually present in this microcomputer forms an operating (physical) memory space (field). In УМПК set, the operating memory space is 4 Kilobytes, of which 2 Kilobytes are occupied by Read Only Memory (ROM) and 2 Kilobytes – by Read Access Memory (RAM). Information

can only be read from the ROM, while the RAM allows you to both read the contents of its cells and write new data into them. The IU address space allows a microcomputer to contain up to $2^8 = 256$ individually addressable input and output devices (ports). Port addresses can be in the range 00…FF.

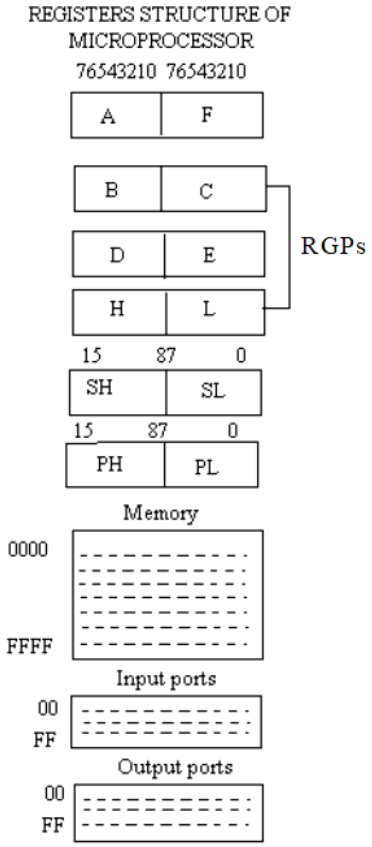


Figure 1.2 – УМПК-80 registers and memory spases

In УМПК, 1 Kilobyte of the ROM (address 0000…03FF) is occupied by the "Monitor" program and 1 Kilobyte of ROM (address 0400…07FF) is access to the user. RAM is used to store various programs and user data. Other 54 cells of RAM (the addresses FFCA…FFFF) are occupied by the program "Monitor" for storage of

operational information and should not be used by the operator. Address 0800 – is the starting address from which all user programs begin.

1.2.2. The operating unit

The operating unit (OU) performs all information processing operations. Its initial state is to read information at ROM address zero every time after pressing the RESET control button on the remote control. In this way, the "Monitor" program is called, which ensures the output of information from the tablet and its display on the indicator. On-Line Storage (OLS) state information is written to the state register at the beginning of each machine cycle. Table 1.1 shows the possible states of OLS. State 0 in the table corresponds to a low level of potential, and state 1 – high. Depending on the state of this register, signals are generated that control the operation of the complete microcomputer. Table 1.2 shows the definition of each bit of the flag register. Line over the control signal WO in Tables 1.1 and 1.2 defines the active state of the signal, i.e. logic 0.

Table 1.1 – Possible states of the OU

OU state	The flag register degrees OU states							
	D7 MEMR	D6 INP	D5 M1	D4 OUT	D3 HLTA	D2 STACK	D1 WO	D0 INTA
Computer instruction choice	1	0	1	0	0	0	1	0
Storage reading	1	0	0	0	0	0	1	0
Storage Writing	0	0	0	0	0	0	0	0
Stack Reading	1	0	0	0	0	1	1	0
Writing of stack	0	0	0	0	0	1	0	0
Input	0	1	0	0	0	0	1	0
Output	0	0	0	1	0	0	0	0
Interruption	0	0	1	0	0	0	1	1
Halt	1	0	0	0	1	0	1	0
Interruption of a Halt	0	0	1	0	1	0	1	1

Table 1.2 – Bits of the flag register of the OU

Name	Control digit	Comment
INTA	D0	Shows the signal of the Interruption
WO	D1	Shows that in the current machine cycle there is writing to memory or output operation
STACK	D2	The address bus has the data of stack
HLTA	D3	Shows the signal of the command “NO”
OUT	D4	Shows that in the current machine cycle there is output command working
M1	D5	Shows that in the current machine cycle for reading of the 1 command byte
INP	D6	Shows that in the current machine cycle the input command is working
MEMR	D7	Shows that in the current machine cycle there will be storage reading

1.3 УМПК set control, input and indication devices

On the face side of УМПК there are:

– on/off button;
– reset and interruption button;
– step-by-step mode control button;
– functional keyboard;
– data input keyboard;
– data bus, address bus state, MP control signals light indicators.

On/off button is placed in the lower left part of the face panel. Pressed button corresponds to ON position. Over the button there are 3 light indicators: +5V; -5V; +12V. During overloading the safety works and the correspondent indicator lights on. In this case the УМПК should be turned off. Button RESET (СБ) is used for the initialization of the system program "Monitor". After pressing it the program "Monitor" start is made and in the left position of the display symbol "-" appears. It means that УМПК is ready for receiving programs. Button INTERRUPTION (ПР) is placed under the СБ. After pressing this button the signal INTERRUPTION CALL OF 7-TH LEVEL is made and if the interruption is allowed (command EI (Enabled Interrupt) was made) the

current program execution will stop, after that the control will be transferred to the address 38H. If ПР is pressed during the program "Monitor" execution the sign "?" will appear on the screen. In the opposite case the address of interruption point will be on the screen. Buttons OPERATION/STEP (РБ/ШГ), COMMAND/CICLE (КМ/ЦК), STEP (ШГ), control the execution of step-by-step mode. Those buttons set one of the two modes of step-by-step operation. First – command. For setting this mode button РБ/ШГ should be pressed. Each pressing of the button ШГ causes the execution of the current command. During this on the light indicators of data bus, address bus condition, МР control signals address, code of executed command, control signals will be lit in the binary code.

Second mode – operation by command cycles. For putting this mode buttons <РБ/ШГ> and <КМ/ЦК> should be pressed. In this the way of command execution can be traced. After pressing ШГ the next machine cycle will be executed. Light indicators there will be information corresponding to each machine cycle. Keyboard of УМПК is divided into two parts. In the left there are functional buttons. Definite function of "Monitor" program is assigned to that buttons:

<П> – read and edit memory cells;

<РГ> – read and edit register content;

<СТ> – program start;

<КС> – calculation of control sum;

<ЗК – filling of memory array by a constant;

<ПМ> – moving memory array;

<__> -- divider;

<ВП> – execute.

The right part is used for entering parameters in hexadecimal system.

<РН> – higher byte of command counter;

<PL> – lower byte of command counter;

<SH> – higher byte of stack pointer;

<SL> – lower byte of stack pointer;

<A>, <B>, <C>, <D>, <E>, <F>, <H>, <L> – registers.

Six-segment display is used for data displaying in the hexadecimal system. Four left segments show address and register identification, two right segments – data.

1.4 Getting ready for work

It's necessary to execute following:

−put <On/Off> button to no pressed position;

−connect УМПК to the circuit with 220V;

−press buttons <РБ/ШГ> and <КМ/ЦК>;

−turn on the УМПК. Light indicators +5 V, -5 V, +12 V should not be lit;

– press button <СБ> and sign "–" should appear on the display. It means УМПК is ready.

The next turning on of the УМПК should be no less than in 20 sec after the turning off. In the other case the supply unit safety will work and the indicators will be on. In this case turn off the УМПК and wait until the indicators will go down.

1.5 Task

1.5.1 Study the structure, basic technical characteristics and operating controls of УМПК.

1.5.2 Make the basic operations with keyboard, setting of different modes and using of the indication.

1.6 Methodical guide

1.6.1 Define the location of basic controls on the УМПК, and their meaning.

1.6.2 Define the location and the type of indication.

1.6.3 Study characteristics and structure of basic УМПК parts.

1.6.4 Make practically the order of turning on and off.

1.6.5 Make the way of calling and reading of content of registers, memory cells.

1.6.6 Make the work with keyboard.

1.6.7 Study the way of setting of different modes, role of lone indicators.

1.7 Content of the report

1.7.1 Topic, purpose, review of the done work.

1.7.1 Practical aspects of considered questions.

1.7.3 Conclusions.

1.8 Self-test questions

1.8.1 Basic characteristics of УМПК, its functional possibilities.
1.8.2 Structure of УМПК, basic parts, their characteristics.
1.8.3 Parameters and memory organization.
1.8.4 Register structure of microprocessor.
1.8.5 Characteristic of the possible states of operating unit.
1.8.6 Characteristic of the controls, indication.
1.8.7 Basic requirements for the proper operation of the УМПК.

2 LABORATORY WORK No. 2 STUDING THE INTERFACE AND MODE OPERATION OF THE SOFTWARE STAND OF THE УМПК-80

The purpose of the work is to study the interface of the software model of the УМПК-80 stand, also the capabilities of the "Monitor" system program emulator and to acquire the skills of working with the basic commands of MP based on the K580.

2.1 General information about working with the K580 emulator

To run the proposed programs, you can use the K580 emulator, which is located in the aud. 226 a and from the system administrator on the server or on the Internet for independent work of students in studying this educational discipline are in the repository НУ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА» by link http://eir.zp.edu.ua/handle/123456789/488 and also in Distance learning systems by link https://moodle.zp.edu.ua/course/view.php?id=2089. This is a software model (emulator) of a laboratory stand of an educational microprocessor set УМПК-80 (a microcomputer with a basic with a microprocessor type K580, U880, Intel 8080, 8085, Z80 and a built-in display. In the УМК set, the program is built into the storage device (ROM) that can be used in educational process for studying various microcomputers together with УМК, which allows students to prepare laboratory work at home using computing devices.

This model was designed to reduce students' time on regular operations (installation, dismantling, etc.). It gives more extensive and

convenient possibilities for typing and developing programs (for example, it may be possible to check all registers, memory cells, enter commands in mnemonic symbols (operators) and in hexadecimal codes, assembly and disassembly commands, etc.).

The software model of the УМПК-80 stand is a means which provides much wider and more convenient opportunities for typing and debugging programs (for example, it may be possible to simultaneously view all registers, memory, enter commands in mnemonics and in hexadecimal codes, assembly and disassembly of instructions etc.).

Obtaining competencies in the program should begin with studying the work window. To do this, open the file with the software model of walls and УМПК-80 on your computer. A program dialog box will appear on the monitor screen (Figure 2.1).

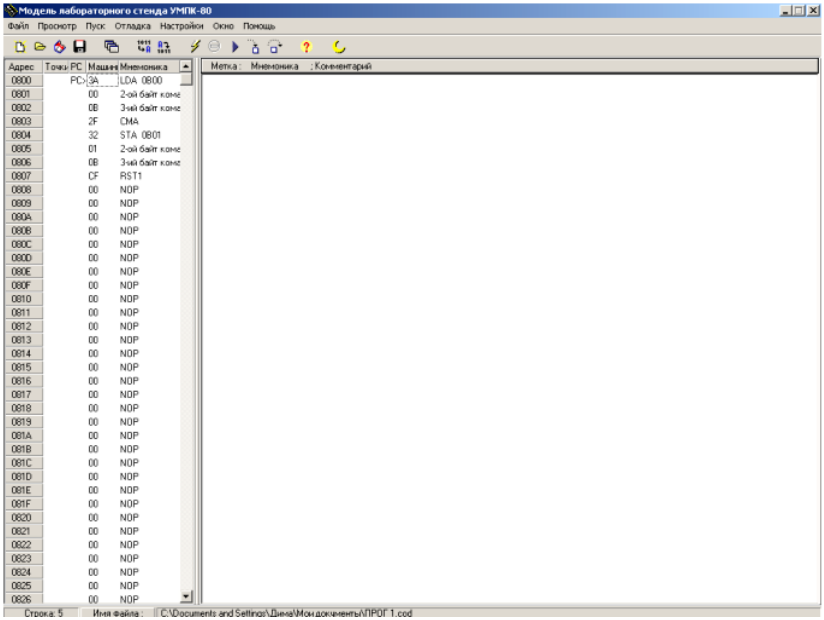


Figure 2.1 – View of the computer screen when working with the software model of the УМПК-80 stand. The main model window of the K580 set emulator

The model menu is located at the top of the main model window. It contains all the commands necessary to work with the model (Figure 2.2).

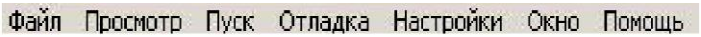


Figure 2.2 – View of the model menu

It includes:

– the “File (Файл)” submenu contains commands for working with files. The “File (Файл)” submenu is located in the main menu of the model and is called up by the “Alt+F” key combination. It contains commands for working with files (Table 2.1);

– “View (Просмотр)” submenu, contains commands for controlling output information. The “View (Просмотр)” submenu is located in the main menu of the model and is called keyboard shortcut “Alt+П”. It contains commands for working with visual objects of the model (windows, panels, etc.) (Table 2.2);

– the “Start (Пуск)” submenu contains commands for executing and processing the program under study. The “Start” submenu is located in the main menu of the model and is called up by the “Alt+ У” key combination. It contains the commands necessary to launch the API in automatic mode, as well as its translation into various code formats (hex codes or mnemonic codes);

– the “Debugging (Отладка)” submenu contains commands for debugging the program under study. The “Debugging (Отладка)” submenu is located in the main menu of the model and is called keyboard shortcut “Alt+O”. It contains the commands necessary to debug the API in step-by-step modes;

– the “Settings (Настройки)” item opens the model settings window.. The “Settings (Настройки)” command is located in the main menu of the model and is called keyboard shortcut “Alt+H”. This command calls up the model settings window.

The toolbar (Figure 2.3) is designed for quick access to the most frequently used commands in the menu and contains:

– creating a new file;
– opening a file;
– closing the file;
– saving the file to disk;
– opening all windows;
– disassembling the program under study;

– assembly of the program under study;
– processor reset;
– stop the execution of the program under study;
– execution of the program under study in automatic mode
– execution of the program under study according to command cycles;
– execution of the program under study in machine cycles;
– call up help information;
– exit from the model.

Table 2.1 – Table of commands for the “File (Файл)” submenu

Command	Description of actions when selecting a command
New Новый	The current program is closed. If in there was no saved data then It is proposed to save them. Created empty program
Open Открыть	The current program is closed. If in there was no saved data, then it is suggested that you save them. A file with an program that was previously saved opens on disk
Reopen Открыть вновь	The model can remember up to 10 names of the last opened files. In this case, by the command “Open again” a list of them opens, from which you can select the one you need. After what happens is the following: the current program is closed, and if it there was no saved data, then it is proposed to write them down; the selected file is opened if found
Save Сохранить	The current program is saved to a file on disk. If the program has not previously been recorded, you will be prompted to enter a file name
Save as *.asm Сохранить как *.asm	Mnemonic codes of current program are saved in asm file
Save as *.cod Сохранить как *.cod	Store machine codes in cod file
Close Закрыть	The current program is closed. If in there was no saved data, then It is suggested that you write them down
Выход	The current program is closed. If there was no saved data in the system, then it is proposed to save it. The model completes its work

The toolbar can be disabled (connected) by selecting “View / Toolbar (Просмотр / Панель инструментов)” in the main menu of the program. When a checkmark is placed to the right of a command, the toolbar is visible/

Figure 2.3 – Toolbar

Table 2.2 – Table of commands for the “View (Просмотр)” submenu

Command	Description of actions when selecting a command
Registers and flags Регистры и флаги	A window opens and activates "Registers and Flags (Регистры и флаги)"
Stand RAM ОЗУ стенда	A window opens and activates "Stand RAM (ОЗУ стенда)"
Stack Стек	A window opens and activates "Stack (Стек)"
Ports Порты	A window opens and activates "Ports (Порты)"
Stand Tools Средства стенда	A window opens and activates " Stand Tools (Средства стенда)"
All Все	Windows open and become active: “Registers and flags (Регистры и флаги)”, “Stand RAM ОЗУ стенда)”, “Stack («Стек)”, “Ports (Порты)”, “Stand facilities «Средства стенда)”
Toolbar Displays Панель инструментов	Toolbar Displays (hides) the toolbar. With a check mark to the right of commands toolbar is visible
Decodering of codes Дешифрация кодов	If there is a checkmark to the right of the command, then in the machine codes section hex codes are disassembled, starting from the address in the commands counter
Viewing of messages Просмотр сообщений	If there is a checkmark to the right of the command, the messages section is visible

The machine code section of the main model window is intended for entering and editing current programs in machine codes and is a table, each row of which corresponds to one RAM cell. The table columns have the following purpose (Table 2.3)

To enter a value into a cell, use the control keys or move the cursor (dotted frame) over it with the mouse and enter a new value, then confirm the entry with the “enter” key. It should be noted that you must always enter both digits of the number, i.e. to enter a value, for example, “7” or “F”, you need to dial “07”, “0F” respectively.

Table 2.3 – Machine code section table

Column header	Column Purpose
Address	Contains addresses of memory cells
Breakpoints	Displays set breakpoints
PC	Displays program counter in "PC" view
Machine code	Contains the value of the memory cell, the address specified in the “address” column.
Mnemonics	Contains a mnemonic description of cell values.

The machine code section has a local menu - the machine code section menu, which is called up by right-clicking the mouse.

Additional window “Registers and flags (Регистры и флаги)” called by the command "view / registers and flags (просмотр / регистры и флаги)" in the main menu of the model or by the key combination "Ctrl+R". It contains the registers of the K580BM80 microprocessor. The window is divided into three parts - sections: “character registers («регистр признаков)”, “indicators (указатели)”, “RGP registers (регистры РОН)” (Figure 2.4).

Moving through sections and their elements is carried out using the “Tab” (forward) and “Shift+Tab” (backward) keys or using the mouse. If you move the mouse pointer over a field while entering input and hold it for several seconds, the tooltip that appears will contain the decimal and binary representations of the value of this field.

The “Stand RAM (ОЗУ стенда)” window is called up with the command “view / Stand RAM («просмотр / ОЗУ стенда)” in main menu of the model or the keyboard shortcut “Ctrl+M”. You can view any

fragment of memory in it. The contents of the memory are displayed as table, each line of which shows sixteen bytes, so the address of the cell lying at the intersection of row “0820h” and column “D” is equal to "082Dh" (Figure 2.4). The main function of the window is to view memory, however, when You can edit the value of any cell if you wish. To do this, use the cursor keys or mouse to move the cursor (dotted frame) to the memory cell whose value you want to correct and press the “enter” key or double-click on it with the left mouse button. This activates the RAM editor, in which you need to enter a new cell value, after which the editor must be closed.

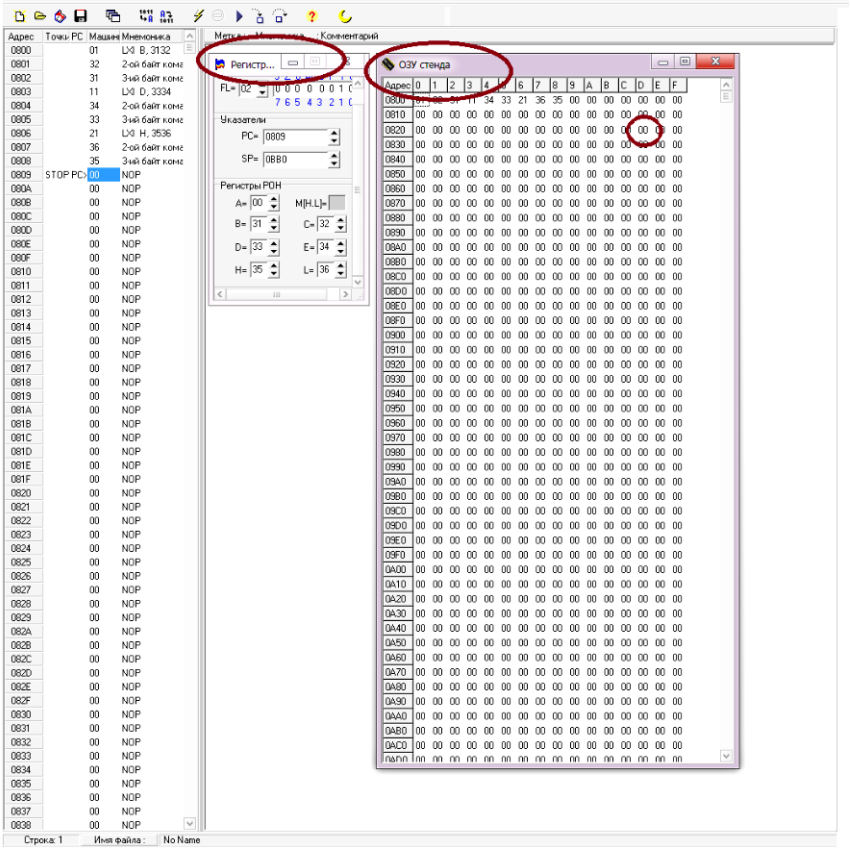


Figure 2.4 – Window of model view with RAM and registers and flags

The “RAM editor” window is called from the “Stand RAM (Редактор ОЗУ)” and “Stack (Стек)” windows, since their main function is to view memory. The editor is designed for editing memory. Editing is done one byte at a time. The address of the current cell is displayed to the left of the inscription “RAM address”. The editor has four buttons (Table 2.4).

Table 2.4 – Command table of the RAM editor window

Button	Description of actions when pressing a button
Rec.+Zoom. Зап.+Увел.	The edited cell value is written to the current address, after which the current address is increased by one
Forward Вперед	The current address is incremented by one without writing a new cell value
Back Назад	The current address is decreased by one without writes a new cell value
Close Закрыть	The RAM editor closes

To edit a cell, you need to set a new value for the memory cell in the input field, and then confirm the entry by pressing the “Write+Increase («Зап.+Увел.»)” button.

To get skills of working with the emulator, you need to try it work as an example. So follow this sequence to program in emulator**.**

For example, execute the program according to table 2.5 that implements the equation: HL = (B OR C) AND DE AND F0FF H.

To run the emulator, it is necessary activate the K580 program directly. On the screen you can see the main window along with the context menu (Figure 2.1).

The next step is to write the machine codes into memory. To do this, you need to move the cursor to "Machine code", as shown in Figure 2.2. The emulator will automatically write the mnemonic (or build commands) that you downloaded.

If it is necessary to set the beginning of the program, that is, the beginning memory address, you need to double-click with the left key next to the corresponding address until the "PC" sign appears, as shown in Figure 2.5.

If you need to write down the code into the last memory cell, you need to left-click next to the address for the word to appear "STOP", as shown in Figure 2.6.

If necessary, the dialed program can be saved in personal "Student" folders at each student's address.

Table 2.5

Adress	Command	Code	Comment
800	MVI B, 00	06 00	Download into B← 00 (content)
802	MVI C, 01	0E 01	Download into C ← 01 (content)
804	MVI D, 10	16 10	Download into D ← 10 (content)
806	MVI E, 23	1E 23	Download into E ← 23 (content)
808	MOV A, C	79	Move the content of C to A (A ← C)
809	ORA B	B0	Comparison A = A OR B
80A	ANA E	A3	Comparison A = A AND E
80B	ANI FF	E6 FF	Comparison A = A AND FF H
80D	MOV L, A	6F	Move the content of A to L (L ← A)
80E	MOV A, D	7A	Move the content of D to A (A ← D)
80F	ANI F0	E6 F0	Comparison A = A AND F0 H
811	MOV H, A	67	Move the content of A to H (H ← A)

To directly execute the program itself, it is necessary to push the cursor at the beginning of the program (that is, at the first address), go to "Start" context menu and click "Run" as shown in Figure 2.7 while the emulator should stop at the last one the address of the executable program.

To check the obtained result, it is necessary to go to "Просмотр" and in the window select " Регистры и флаги" (see Figure 2.8).

To check the recorded program or the received result located in the memory cell, it is necessary to go to "Просмотр" and "ОЗУ стенда" (Figure 2.6).

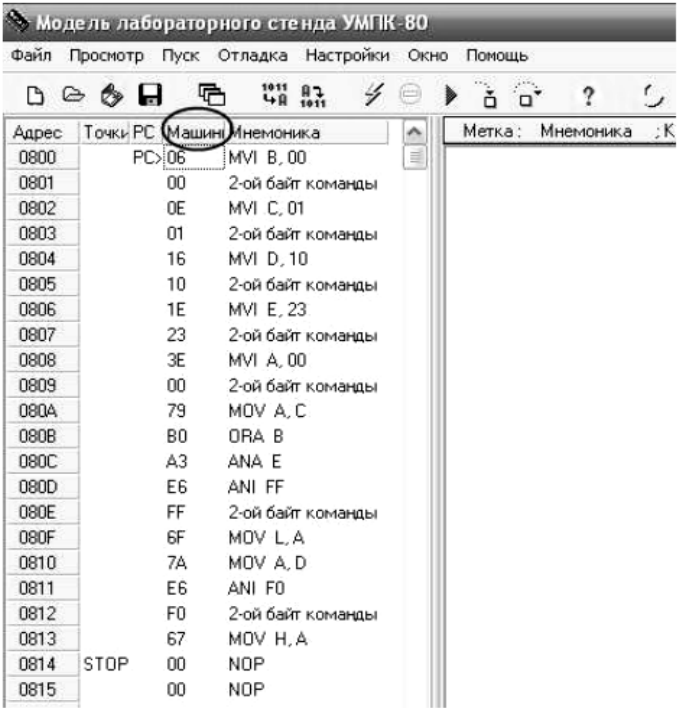


Figure 2.5 – Writing down command codes to the emulator memory

2.2 Content of the report

The report must contain:

– name and purpose of work;

– algorithms for executing each command according to the following task: create and execute a program that has as initial data your individual height in the hexadecimal system;

– conclusions.

2.3 Self-test questions

2.3.1 Purpose and capabilities of the "Monitor" program emulator. Characteristics of its systemic features.

2.3.2 Characteristics of the main commands and operations, their connection with functional keyboard and display.

2.3.3 Basic operations with memory arrays.

2.3.4 Concept of buffer, features of its use.

2.35 Procedure and features of operating with overlapping memory areas.

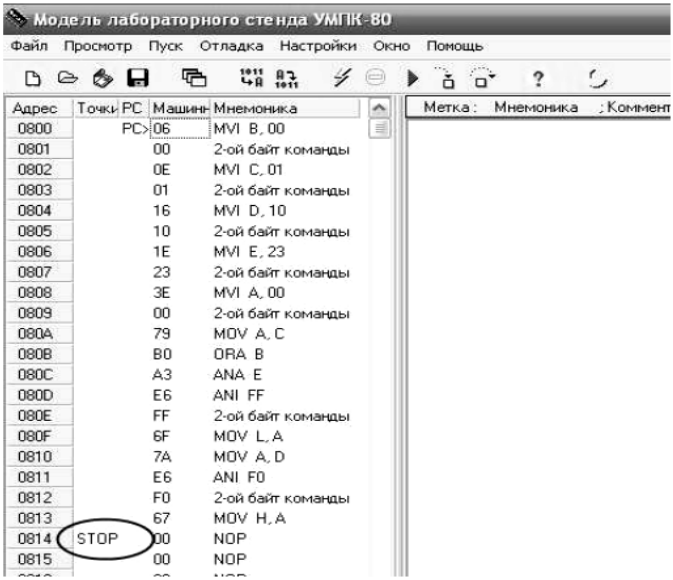


Figure 2.6 – Setting the last program breakpoint

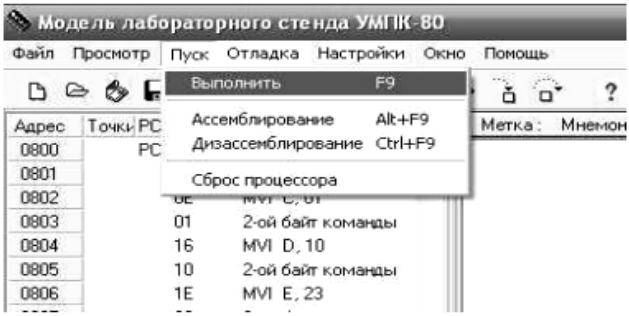


Figure 2.7 – Start of program execution

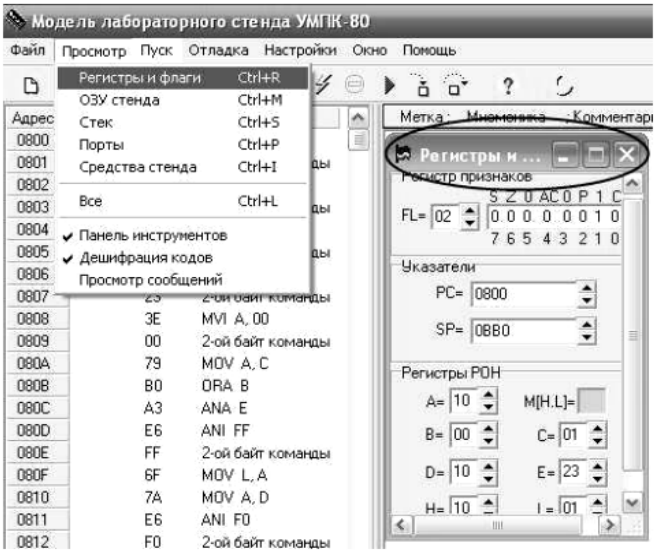


Figure 2. 8 – Checking the contents of the registers

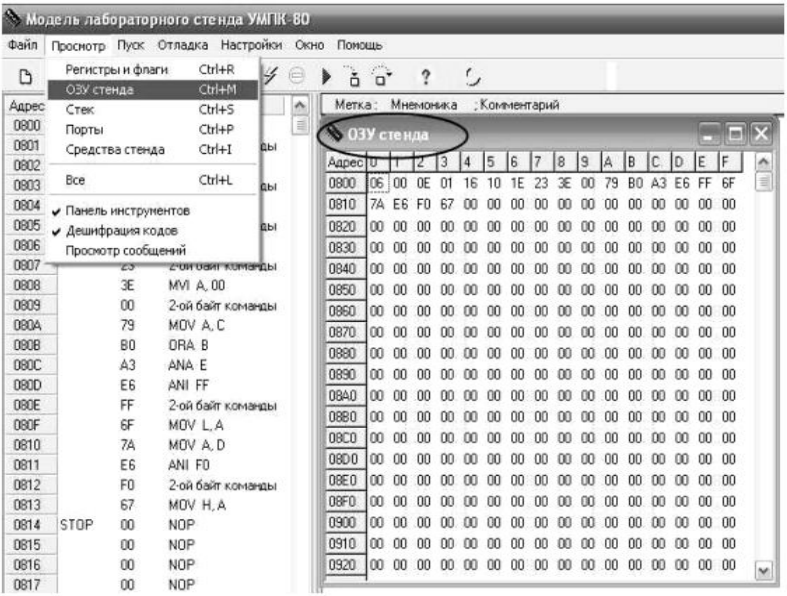


Figure 2.9 – Checking the contents of memory cells

3 LABORATORY WORK No. 3 ARCHITECTURE OF THE KP580BM80A MICROPROCESSOR. THE REGISTERS OF THE MICROPROCESSOR. COMMANDS OF REGISTERS LOAD. COMMANDS OF TRANSFER

The purpose of work is to study the structure of KP580BM80A MP and commands of operation with programmable accessible registers of this microprocessor.

3.1. Architecture of the KP580BM80A microprocessor

The structural diagram of the studied MP is presented in Figure 3.1. MP KP580BM80A includes the following devices:

- six 8-bit general purpose registers (RGP) -B, C, D, E, H, L;
- 8-digit accumulator (A);
- four 8-digit registers for temporary storage of the information BP1, BP2, W and Z;
- 8-digit register of attributes (flags) RA (F);
- 8-bit parallel arithmetic-logic device (arithmetic-logic unit);
- circuit (scheme) of a decimal correction (СДС);
- 8-bit command register (RC);
- 16-digit counter of commands (РК);
- 16-digit stack index (SP);
- 16-digit address register (RA);
- command decoder and machine cycle control circuit (schemes) (DC and MCCC);
- multiplexer (M);
- register selection circuit (scheme) (SCR);
- control unit (CU);
- buffer of data (BD);
- address buffer (AB).

For the programmer, 10 registers are available in KP580BM80A MP, these are: six RGP, accumulator, command counter, stack index and attribute register.

RGPs are used to store the numbers involved in the operation to indicate the address of a memory cell or 16-digit data. In the latter case, 8-bit registers B, C, D, E, H, L are combined into register pairs BC, DE,

HL. In the first registers of the pairs - B, D, H, the older bytes are located, and in the second - C, E, L - the younger bytes. The call is made by the name of the first register. So, the listed pairs can be called B, D, H, respectively. In addition to the name, each RGP, as well as register A, has a three-digit binary code:

Name of the register.....B C D E H L M A.
Binary code..…….... 000 001 010 011 100 101 110 111.

ALU is designed to perform arithmetic and logical operations on eight-bit data and operands.

The accumulator A is the main part of the MP arithmetic device. All arithmetic and logical operations are performed in the ALU using the storage device. For each of these operations, it is assumed that one operand is placed in accumulator and the other is placed in memory or RGP. The arithmetic and logic unit is intended for execution of arithmetical and logic operations on 8-digit data and operands. The result of the operation is placed in the accumulator. Stack write operations and reading from the stack are shown in Figure 3.2.

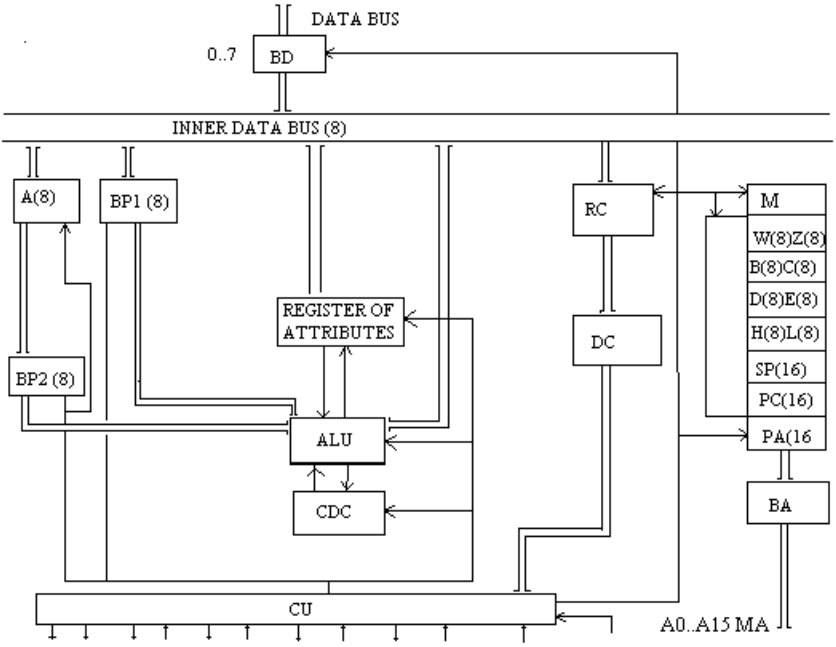
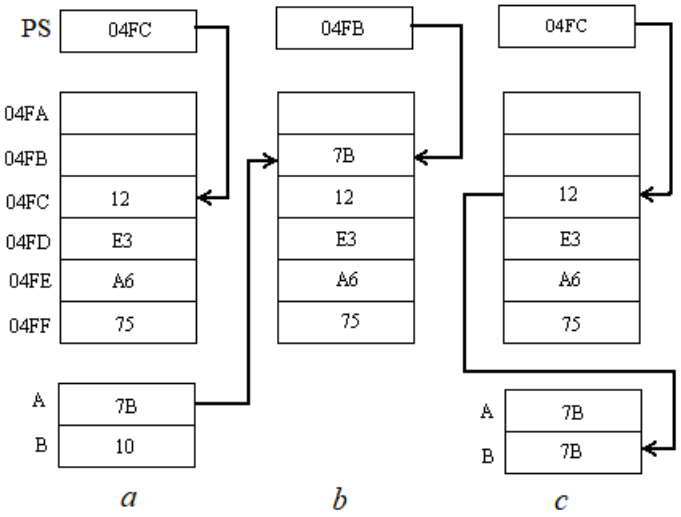


Figure 3.1 – Structural diagram of KP580 BM80 MP

M – is the memory cell, which address contains in register pair HL.

The command counter is used to access the memory cells in which the program is stored. It always stores the address of the command to be executed. After the next command is executed, its content is incremented by 1, 2, or 3, depending on the length of the command in bytes.

SP stack index (IS). The stack is a memory space chosen by the programmer, into which data is written and from which data is read in a strictly defined order according to the LIFO (Last-In, First-Out) algorithm, last in, first out. With the help of 16-bit SP, the programmer determines which memory space is allocated for the stack. SP contains the address of the stack cell that is available for reading information. After a read is performed, the content of PS is automatically incremented by 1. Before writing to the stack, the content of PS is decremented by 1, after which a write is performed. The operations of writing and reading from the stack are shown in Figure 3.2.



a – is initial condition of stack; *b* – is stack after execution of the command "write to stack code from the accumulator";
c – is stack after executing command "read code from stack to register B".

Figure 3.2 – The operations of record and reading from stack

The feature register F is designed to store five features (flags) that appear when performing some operations. These signs (bits conditions) are stored in the corresponding FR degrees (Figure 3.3).

THE REGISTERS of ATTRIBUTES (FR)

7	6	5	4	3	2	1	0
S	Z	0	AC	0	P	1	C

Figure 3.3 – The register of attributes

S (Sign) – attribute of the sign of the result stored in the accumulator A. With a positive sign, S = 0, with a negative sign, S = 1;

Z (Zero) – if the contents of the accumulator A = 0, then attribute Z = 1, otherwise Z = 0.

C (Carry) – transfer attribute. C = 1, if there is a carry-over unit from the higher bit (overflow) when executing commands.

AC (Auxiliary Carry) – is an additional carry attribute. It is set in a degree, if when executing commands there is a unit from the third digit of the number (from the lower tetrad into high).

P (Parity) – parity attribute. P=1 if the number of units in the accumulator A discharges will be parity. A score of zero also falls to even. In the opposite case, P=0.

The numbers 1, 3 and 5 in the attribute register are not used as attributes. The full description of each MP command must include information about which attributes in the F register are affected by the command. The contents of the register pair A and F, called the program condition word and stands for PSW.

3. 2. Structure of commands

The binary numbers chosen from memory and indicating execution of defined operation, refers to as a command. The binary number subjected to processing refers to as an operand. The system of commands executes by MP contains 78 base commands (common number of commands - 244).

Usually command of MP KP580BM80A consists of two fields: fields of an operation code (КОП) and field of the address of an operand (address part). In MP KP580 some ways of setting the address of operands - ways address are used: a direct address, indirect register address, direct address and immediate address.

Different address ways need different number of digits in a format of a command. Therefore commands of MP have different length - one, two or three bytes.

At direct register address an operand (one or two) is in RCA, which address is showed in an address part of a command (in a binary system of the address have a code 000-111, and in 16 - 0-7). Such commands have length of 1 byte. In a fig. 2.4,a there is the binary representation of a command of transfer of contents of the accumulator to the register B. On the right hex code of a command is specified.

At direct addressing the address of a memory cell is showed in a field of the address of a command. Such commands have length in three bytes: 1st byte – is operation code, 2nd byte – is lower digits of the address, 3-rd byte – is higher digits of the address. In a fig. 2.4, b the command of a load of the accumulator by contents of a cell 7AC3 is shown.

At immediate addressing the operand is a part of the command and depending on its digit occupies the second or second and third bytes of a command. In fig. 2.4 c there is a command of transfer of a code 4C to the accumulator, and in fig. 2.4, d — is loading of register pair BC by a code 1F5A.

Any command has mnemonic (symbolical) exposition facilitating spelling of the program for the programmer. For example, ADD (to combine), MOV (to send), XCHG (exchange of contents of register pairs D and H) etc. Field of a mnemonics should be separated from operands even by one blank. The operands are numbers in hexadecimal system. If the number begins with the letter, before the letter the insignificant zero is put, differently this number will be recognized as a name. For example, 0FAH, 0B7H, 0DEH etc.

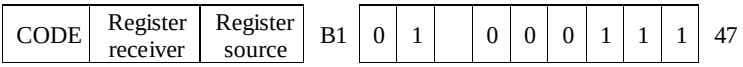
The operands disjoint by a coma, if there is more than one. The blanks between operands inadmissible. It is necessary to remember, that:

1) Maximum 8-digit number 0FFH, and 16-digit - 0FFFFH.

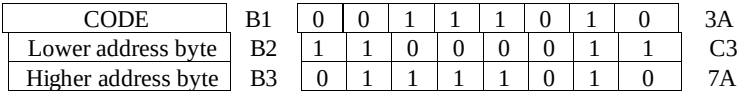
2) It is necessary whenever possible to use shorter (smaller number of bytes) command.

3) If the command influences on definite bits of the register of attributes, other bits of this register do not change the condition.

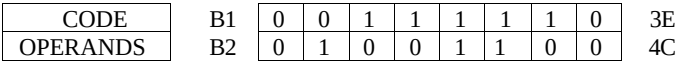
Command encoding for the K580 VM 80A emulator is presented in Appendix A (page 76).



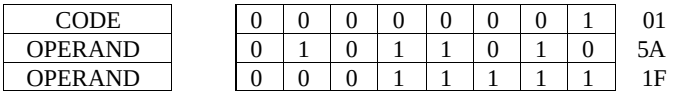
a



b



c



d

a – direct register addressing; *b* – direct addressing;
c, *d* – immediate addressing.

Figure 3.4 – Types of addressing

All commands can be divided into 7 groups:

1) Command of transfer of the information.
2) Command of transmission of control or transition.
3) Command of arithmetical and logic operations.
4) Command of organizing of subprograms.
5) Command of input and output.
6) Command of operation with stack.
7) Command of management of a condition of MP. Further we shall use the following conditional labels:

R, Rl, R2. – one of 8-digit RGP or accumulator;
RP – register pair B, D or H, and also index of stack;
data – 8-digit operand or its name;
data l6 – 16-digit operand or its name;
addr – address of a cell of memory;
port – 8-digit address IU or its name.

3. 3 Commands of RGP loading

3.3.1. The command MVI (Move Immediate) serves to load of an RGP by an 8-digit binary operand. Its length is 2 bytes, executed in 7 steps. Format of a command MVI R, data.

For example: MVI A, FFH – load the accumulator by number FFH; MVI C, FFH – to load the register C by the name FFH. The load command of each register has hex code. In first byte КОП is put and the receiver register, in second – operand. If as the register R in a command the conditional register M is specified, second byte of a command (operand) loads into a cell of memory, which address previously was written down in a register pair HL.

Table 3.1– Codes of first byte of a command MVI

The register the receiver	A	B	C	D	E	H	L	M
Code of first byte of a command MVI	3E	06	OE	16	1E	26	2E	36

The task.

1) Write down to memory, from address 800H the following sequence of commands.

It is necessary to remember, that the command MVI has 2-bytes, therefore for each command it is assigned two cells of memory. The machine code of a command is entered at one byte. So, in a cell 800 the code 3E, and in a cell 801 -– 00 etc.

The address	Command	Machine code	The comment
800	MVI A, 00	3E 00	To load the register A by a code 00H
802	MVI B, 01	06 01	To load the register B by a code 01H
804	MVI C, 02	0E 02	To load the register C by a code 02H
806	MVI D, 03	16 03	To load the register D by a code 03H
808	MVI E, 04	1E 04	To load the register E by a code 04H
80A	MVI H, 05	26 05	To load the register H by a code 05H
800	MVI L, 6	2E 06	To load the register L by a code 06H

2) Execute this sequence with the help of a command <СТ 800 _80E ВП>. On the display the stopping address 80E will appear.

3) Check up contents of the registers, using a command SCANNING AND MODIFICATION of CONTENTS of the REGISTERS.

3.3.2 The command LXI (Load register pair Immediate) serves for loading of register pair, specified in a command, by 16- digit data. Length of a command is 3 bytes; it is executed in 10 steps.

In first byte there is operation code and register, in second and third – 16-digit operand is put. And third byte of a command is loaded into the first register of pair, and second byte – in the second register of pair.

Format of a command LXI RP, data 16.

As RP the register pairs B, D, H, SP are used.

For example: LXI D, FF00 H - in the register D higher byte – FF, in the register E lower – 00 records.

The codes of a command LXI are shown in the table 3.2.

Table 3. 2 – Codes of a command LXI

Addressed register pair	B	D	H	SP
Code of first byte of a command LXI	01	11	21	31

The task.

1) Write down in memory, starting from addresses 800H, codes of the following sequence of commands:

The address	Command	Machine code	The comment
800	LXI B, 3132H	01 32 31	To load RP BC by a code 3132H
803	LXI D, 3334H	11 34 33	To load RP DE by a code 3334H
806	LXI H, 3536	21 36 35	To load RP HL by a code 3536

2) Execute this sequence of commands with the help of a command START of the PROGRAM.

3) Check up contents of all registers, which should coincide with set values.

3.3.3 Command of loading of the register of the index of stack. The command of an immediate load of the register SP looks like LXI SP, data 16.

The command of an indirect load of the register SP looks like SPHL. Command has 1-byte. A code of this command is F9. On a command SPHL in the index of stack contents of pair HL are loaded. Therefore register pair HL previously was loaded by the necessary number.

The task.

1)Write down to memory to the address 800H codes of a command:

The address	Command	Machine code	The comment
800	LXI SP, 0B10H	31 10 0B	To load the index of stack SP by a code 0B10H

2) Execute this command.

3) Check up contents of the register SP.

4)Write down in memory to the address 800H codes of the following commands:

The address	Command	Machine code	The comment
800	LXI H 0B30H	21 30 0B	To load RP HL by a code 0B30H
803	SPHL	F9	To load the register SP. by a code from RP HL

5) Execute this command: <СТ 800_ 804 ВП>.

6) Check up contents of the register SP byte-by-byte.

All commands of RGP loading do not form the attributes, i.e. condition of the register F doesn't change.

3.4 Commands of transfer

The given command is needed for an information transfer from the register to the register, from the register to memory and from memory to the register. A general view of a command:

MOV Rl, R2;

R1 – Register – receiver;

R2 – Register – source.

The command occupies 1 byte and is executed in 5 steps. Doesn't form attributes.

For example, MOV A, B – to send contents of the register B to the accumulator, MOV M, A – to send contents of the accumulator to a cell of memory, which address is stored in pair HL. At execution of all these commands contents of the register – source R2 is saved.

The task

1) Write down to memory, starting from address 800H, codes of the following sequence of commands:

The address	Command	Machine code	The comment
800	MVI A, 00H	3E 00	Zero the register A
802	MOV B, A	47	To send contents A to B
803	MOV C, B	48	To send contents B to C
804	MOV D, C	51	To send contents C to D
805	MOV E, D	5A	To send contents D to E
806	MOV H, L	63	To send contents E to H
807	MOV L, H	6C	To send contents H to L

2) Execute this sequence of commands.

3) Check up contents of the registers.

4) Compare an amount of memory occupied by this program, and duration of its execution to volume and duration of the similar program zeroing of the same registers executed with the help of commands MVI for each register (see the task to section 3.3.1).

3.5 A command of loading of the command counter

On this command in the counter of commands contents of register pair HL record. The command looks like: PCHL. Code of this command is E9.

For example, to load to the counter of commands the address 900 it is necessary previously to load it to RP HL, and then to execute a command PCHL. This command has no operands and executes unconditional transition with indirect addressing. Length of a command is 1 byte, duration 4 steps. The command PCHL does not form any attributes.

The task

1) Write down to memory to the address 800H codes of the following commands:

The address	Command	Machine code	The comment
800	LXI H, 0900H	21 00 09	To load RP HL by a code 0900H
803	PCHL	E9	To load PC by a code RP HL. To proceed to the address 0900H

2) Execute these commands <СТ 800_ 804 ВП>. On the display the address 0900 will appear. It will mean, that the address 0900H is loaded in the counter of commands and the transition to this address was made.

The task.

1) Write and execute the program of RGP load with codes corresponding to the your own height in cm.

2) Write and execute the program of loading of register pairs with codes corresponding to your height in cm.

3) Write and execute the program of a load of the register SP:

4) Write and execute the program of transfer for all RGP, previously having loaded one of RGP your own height in cm.

3.6 The content of the report

3.6.1 Theme, purpose, basic information on this work.

3.6.2 Results of the executed tasks.

3.6.3 Conclusions of the work.

3.7 Self-test questions

3.7.1 Different types of buses.

3.7.2 Programmable accessible parts of microprocessor, their short characteristic.

3.7.3 Command system, command group, command description.

3.7.4 Peculiarities of addressing.

3.7.5 Commands of registers and register pairs loading.

3.7.6 Transition commands.

4 LABORATORY WORK No. 4. ADDRESSING TO MEMORY. METHODS AND COMMANDS OF WORKING WITH MEMORY

The purpose of work is to study addressing methods and commands of operation with memory.

Subject of a research: address space of memory, methods of the addressing and operation with УМК memory.

4.1 Brief theoretical information

Memory is the device that stores commands and data. There is fixed (long-time) and operative memory. Accordingly the devices realizing each type of memory in abbreviated form called ROM and RAM. Accessible to the user practically in all basic modes of УМК operations is the RAM, and RAM operation further should be understood under the term memory.

The device of memory consists of blocks of the identical size - cells of memory, size of everyone is one byte, used for storing of one word of the information in a binary code. In its turn each cell consists of elements of memory, the condition of which corresponds to one binary digit - bit of the information. Cells of memory are numbered by numbers from 0 up to 65535, which called address. Frequently for convenience hexadecimal values of addresses are used, then the address range of УМК is 0000H-FFFFH.

To write down to memory a word, it is necessary to give on the bus of the address of memory signals appropriate to the address of a cell, in which it is necessary to place a recorded word, and to give necessary signals appropriate to a value of a word, on the bus of record.

The memory is arranged in such a manner that the given word will be transferred in a cell with the specified address and can be stored there as long as necessary. In the RAM it is stored as long as the supply voltage is on. At any moment, having addressed the memory, is possible to receive contents of a stored word (a copy). While reading, contents of a cell of memory do not vary. The time of access does not depend on addresses of a cell of memory. In УМПК accessible to the user are the cells with addresses from 0800 up to FFC9 are. The addresses input/output ports can be in limits 00H-FFH, the program monitor occupies the space FFCAH - FFFFH.

The registers and register pairs consisting of a set of storing elements, numbered with digits are applied for intermediate operative storage of words.

In a microprocessor system two methods of addressing to memory: immediate and indirect. The following types of addressing are distinguished: register, indirectly - register, immediate and direct.

At immediate addressing, a cell of memory is put in the second and third bytes of a command.

Format of a command in a general view: КОП AHAL,
where КОП – is operation code;

AHAL – is address of a cell of memory (AH – is address higher byte, AL – is address lower byte).

Whereas the specified command is 3-byte, it will be placed in three cells of memory. After byte of an operation code put down lower byte of the address, and after – higher:

n КОП
n + 1 AL
n + 2 AH

There are two commands of immediate record to memory.

STA AHAL - record to memory by the immediate address AHAL of contents of the register A.

SHLD AHAL - record to memory of contents of register pair HL.

On the given command to the address AHAL contents of the register L, and to the address AHAL + 1 - contents of the register H will be recorded.

Example: Write down codes of commands from the table 4.1:

Table 4.1

Address	Command	Code	Comment
800	STA 880H	32 80 08	Record to memory from reg. A to the address 880H
803	SHLD 890H	22 90 08	Record to memory: From reg. L to the address 890H; From reg. H to the address 891H.

Initial values of the registers A, H and L must have your growth.

Execute by a command: <СТ 800 _ 806 ВП>.

Checkup result on values of contents of cells 880H, 890H, 891H.

Indirect addressing assumes, that the address of a cell of memory located in register pairs HL, BC, DE. For each concrete command of operation with memory register pair is fixed. Therefore before to set a command it is necessary to specify the address in appropriate register pair.

For example:

LXI H, 800H; record of the address in the register HL.

MOV M, A; record to memory from the register A to the address HL.

LXI D, 900H; record of the address in the register DE.

STAX D; record to memory from the register A to the address DE.

The commands of reading of memory are distinguished by a type of addressing on immediate and indirect.

Commands of immediate reading of memory:

LDA AHAL - reading of memory by immediate address AHAL in the register A.

LHLD AHAL - reading of memory on immediate address AHAL in register pair HL. Thus in the register L will be recorded content of a cell AHAL, and in the register H – will be contents of a cell with address AHAL + 1. Example: Write down codes of commands according to the table 4.2.

Table 4.2

The address	Command	Code	The comment
800	LDA 880H	3A 80 08	Reading in the register A from a cell 880H.
803	LHLD 890H	2A 90 08	Reading in register L from a cell 890H, in register H of contents cell 891H

To execute the specified commands: <СТ 800_806 ВП>. To checkup contents of the registers A, H and L. They must have your growth.

The commands of read – is record of memory at address through a register pair HL looks like:

MOV M, R – record in memory of contents of the register;

MOV R, M – load of the register from memory;

where R – register of common use A, B, C, D, E, H, L.

Example:

1) Write down the memory codes of the program, shown in the table 3.3. Execute the specified sequence with a command: <СТ 800 _ 826 ВП>. Checkup contents of cells of memory: 900 H = AAH, 901 = CCH, 902 = BBH, 903 = EEH, 904 H = DDH, 905 H = 09H, 906 H = 06H.

2) Write down codes of the program shown in the table 4.4.

Execute the specified sequence of commands: <СТ 800_81С ВП>. Checkup contents of the registers: A = DDH, B = EEH, C = BBH, D = CCH, E = AAH, H = 09H, L = 06H.

The commands of reading – record of addressing with use of register pairs BC and DE look like:

STAX B – record contents of the register A to memory to the address specified in register pair BC;

STAX D – record from the register A to memory to the address from register pair DE;

LDAX D – reading of contents of memory in the register A to the address specified in register pair DE;

LDAX B – reading from memory in the register A to the address from register pair BC.

Example: Write down to memory codes of the program from the table 4.5.

Table 4.3

The address	Command	Code	The comment
800	MVI A, AAH	3E AA	Load the register A with AA
802	MVI B, BBH	06 BB	Load BB the register B
804	MVI C, CCH	OE CC	Load CC the register C
806	MVI D, DDH	16 DD	Load DD the register D
808	MVI E, EEH	1E EE	Load the register E
80A	LXI H, 900H	21 00 09	Load HL with address 900H
80D	MOV M, A	77	Record A to the address HL
80E	LXI H, 901H	21 01 09	Load HL with address 901H
811	MOV M, C	71	Record A to the address HL
812	LXI H, 902H	21 02 09	Load HL with address 902H
815	MOV M, B	70	Record A to the address HL
816	LXI H, 903H	21 03 09	Load HL with address 903H
819	MOV M, E	73	Record A to the address HL
81A	LXI H, 904H	21 04 09	Load HL with address 904H
81D	MOV M, D	72	Record A to the address HL
81E	LXI H, 905 H	21 05 09	Load HL with address 905H
821	MOV M, H	74	Record A to the address HL
822	LXI H, 906 H	21 06 09	Load HL with address 906H
825	MOV M, L	75	Record A to the address HL

Table 4.4

The address	Command	Code	The comment
800	LXI H, 900 H	21 00 09	Load HL = 900, address M
803	MOV E, M	5E	Reading E=M, to the address HL
804	LXI H, 901H	21 01 09	Load HL = 901, address M
807	MOV D, M	56	Reading D=M, to the address HL
808	LXI H, 902H	21 02 09	Load HL = 902, address M
80B	MOV C, M	4E	Reading C=M, to the address HL
80C	LXI H, 903H	21 03 09	Load HL = 903, address M
80F	MOV B, M	46	Reading B=M, to the address HL
810	LXI H, 904H	21 04 09	Load HL = 904, address M
813	MOV A, M	7E	Reading A=M, to the address HL
814	LXI H, 905H	21 05 09	Load HL = 905, address M
817	MOV H, M	66	Reading H=M, to the address HL
818	LXI H, 906 H	21 06 09	Load HL = 906, address M
81B	MOV L, M	6E	Reading L=M, to the address HL

Examples:

1) Write down to memory codes of the program from the table 4.5:

Execute the specified sequence with command:<СТ 800_80C ВП>. Checkup contents of cells 900H and 910H: 900H = 0FH, 910H = F0H.

Table 4.5

The address	Command	Code	The comment
800	LXI B, 900H	01 00 09	Load BC = 900H
803	MVI A, 0FH	3E 0F	Load A =0FH
805	STAX B	02	Record M =A to the address BC
806	LXI D, 910H	11 10 09	Load DE = 910H
809	MVI A, F0H	3E F0	Load A =F0H
80B	STAX D	12	Record M =A to the address DE

2) Write down to memory codes of the program from the table 4.6.

Table 4.6

Address	Command	Code	Comment
800	LXI D, 900H	11 00 09	Load DE = 900H
803	LDAX D	1A	Reading A = M to the address DE
804	MOV L, A	6F	Transfer L ← A
805	LXI B, 910H	01 10 09	Load BC = 910H
808	LDAX B	0A	Reading A = H to the address BC
809	MOV H, A	67	Transfer H ← A

To execute the specified sequence of commands: <СТ 800_80А ВП>. Checkup contents of the registers H, L: H = F0H, L = 0FH.

Register address is used in commands of an aspect: MOV R1, R2; ADD R; DCHI, etc.

The task

1) To study the device of УМПК memory, basic ways of addressing and command of call.

2) To write down and to execute the programs of data record to memory from the register, register pair.

3) To write and to execute the programs of a loading the registers from memory, using commands of reading of memory to the register, register pair to the immediate address and command of transfer.

4) To write and to execute the programs of record to memory and reading of its contents at addressing through a register pair and two register pairs.

5) To accept the decisions and to execute any variants of the tasks, in those cases when they are shown. To make and to solve examples with register addressing.

4.2 Methodical guide

4.2.1 Study structure of memory, its functional features, ways of recording and writing of contents, basic commands, their formats, ways of exposition. To execute examples.

4.2.2 To write and to execute the programs of data recording to memory with your own height.

4.3 Content of the report

4.3.1 Specification and requirements of operation, brief review on the investigated questions.

4.3.2 Programs written on the requirements of language of the Assembler.

4.3.3 Results of execution of the programs, conclusions.

4.4 Self-test questions

4.4.1 Physical fundamentals of memory, its technical realization. Types of memory.

4.4.2 Ways and types of addressing.

4.4.3. Feature and types of register addressing.

4.4.4 Feature of call to cells of memory.

4.4.5 Distinction between direct and immediate address.

4.4.6 Difference of register addressing from indirect – register addressing.

5 LABORATORY WORK No.5. BINARY ARITHMETICS OF THE MICROPROCESSOR. OPERATIONS AND COMMANDS

Purpose of work is to study arithmetical operations and commands of the microprocessor KP 580.

Subject of a research is way of realization and technique of arithmetical operations with binary numbers realized by the microprocessor.

5.1 Brief theoretical information

Independently of the way of introducing numbers in the initial programs, MP operates only with binary numbers, which has its own specificity. Binary arithmetic's reflect specifics of arithmetical operations with binary numbers. Basic arithmetical operations of MP are addition and subtraction of binary numbers, that implies from principles of architecture of computing processes.

Addition of binary numbers, as well as decimal, is made on digit from lower to the higher, but it is much easier in realization, as the numbers have only two values. The greatest conformity to natural representation of addition gives tabulated form, in which members form lines and columns, and the result of addition of numbers of the same name digits is on intersection of the appropriate line and column. With reference to addition of binary digits the specified table looks like the table 5.1.

Table 5.1 – Addition of binary numbers

Number	Number		
	1	0	1
	0	1	1
	1	1	0 *

* The carry of unit to the higher digit takes place.

The binary addition of pair of numbers is evaluated by the following rules:

– the addition of two units, gives zero, in lower digit of result and carry of unit to the higher digit (1 + 1 = 0 *);

– the adding of unit to zero of lower digit will give in result unit without carry to the higher digit (1 + 0 = 1);

– the result of adding zeros is zero (0 + 0 = 0).

At all simplicity of binary addition as procedures, big size of record of the large numbers in the binary type causes a set of carries from one digit to another, that creates definite inconveniences.

A binary subtraction also in many respects similar to decimal. The tabulated form of record is given in table 5.2.

Table 5.2 – A subtraction of binary numbers

Reduced	Subtraction		
	1	0	1
	0	1*	1
	1	1	0

* Means a loan of unit from the next higher digit

Fundamentals of a subtraction are:

– a subtraction from unit of unit and from zero of zero gives zero (1 – 1 = 0, 0 – 0 = 0);

– the subtraction of zero from unit gives unit (1 – 0 = 1);

– the subtraction of unit from zero requires a loan of unit from the higher digit also gives result unit (0 – 1 = 1).

Arithmetical operations also are shift of a binary number on one digit to the left or to the right, comparison of two binary numbers, and reduction or increasing of number by 1.

Complicated arithmetical operations are such, which can be presented as a combination of several simple. So, the multiplication can be presented as a combination of addition and shift, and division as a sequence of a subtraction and shift.

In a microprocessor system МР580, which variant is УМПК the realization of the following commands of binary arithmetic is possible.

1) Addition of eight-digit numbers.

2) Addition 16 - digit numbers.

3) Subtraction of eight-digit numbers.

4) Increment.

5) Decrement.

All arithmetical operations with eight-digit operands in a system KP580 are grounded on the notion has one of operands situated in the accumulator (register A), and another – in the register (RGP) or in memory (M). The address of a cell of memory (M) should be specified in register pair HL. The second operand can be also number given immediately to the command. The result of arithmetical operation is recorded to the accumulator. The operation of subtraction has that feature that it is always made from content of the accumulator, i.e. reduced number should be an operand placed in the accumulator.

By the result of arithmetical operation of addition and the subtraction the following types of attributes are established:

C – carry, Z – zero, S – sign, P – parity, AC – auxiliary carry

Addition of 16-digit binary numbers in the specified system is referred to as double addition. It's supposed that one of operands is in register pair HL, and second - either in DE, or in BC. The result records to HL. By result of operation bit of carry C is established or reset.

The operation of increment consists in increase of content of the registers, register pairs, cell of memory at the address HL by unit. Increment of register and memory changes bits of attributes: Z, S, P, AC. Increment of register pair does not effect on bits of attributes.

The operation of decrement consists in reduction of contents of the registers, of register pairs, cell of memory at the address in HL on unit. The changed bits of attributes are similar to a command increment.

Commands of addition of eight-digit numbers:

ADD R – addition with the register A, B, C, D, E, H, L;

ADD M – addition with a cell of memory (address in HL);

ADI B – addition with immediate number, B – byte;

ADC R – is addition with the registers: A, B, C, D, E, H, L plus bit of carry C;

ADC M – is addition with cell of memory (address in HL) plus bit of carry C;

ACJ B – is addition with immediate number, B – byte, plus bit of carry C.

Example:

1) Execute the program from table 5.3 execute the operation program of adding: A = A+B+M+1

Table 5.3

The address	Command	Code	The comment
800	ADD B	80	A=A+B
801	LXI H, 900H	21 00 09	Load address H
804	ADD M	86	HL=900 H (A=A+M)
805	ADI 01	C6 01	A=A+1

Execute the program, previously taking the initial data of your own height

2) Input to the memory the program of addition of 16-digit numbers with the basis of commands for 8-digit addition from the table 5.4 for the equation: HL = DE + BC.

Execute the program, previously taking the initial data of your own height

Table 5.4

Address	Command	Code	Comment
800	MOV A, C	79	Transfer A ⟵ C
801	ADD E	83	Add lower byte
802	MOV L, A	6F	Lower byte into L
803	MOV A, B	78	Transfer A ⟵ B
804	ADC D	8A	Add higher byte, taking carry into account
805	MOV H, A	67	Transfer H ⟵ A

The commands of a subtraction 8-digit numbers present like these:

SUB R – subtraction of the register: A, B, C, D, E, H, L;

SUB M– subtraction of a cell of memory (address HL);

SUI B – subtraction of immediate number, B – byte;

SBB R – subtraction of the register: A, B, C, D, E, H, L minus bit of carry C;

SBB M – subtraction of a cell of memory (address HL), minus bit of carry C;

SBI B – subtraction of immediate number, and minus bit of carry.

Examples:

1) Execute the program from the table 4.5 made the operation of A = A – B – M – 01H

Table 5.5

Address	Command	Code	The comment
800	SUB B	90	A = A - B
801	LXI H, 900H	21 00 09	Load HL=900H, address M
804	SUB M	96	A=A - M
805	SBI 1	DE 01	A=A - 1

Execute the program previously taking the initial data of your own height. Checkup obtained results.

2) Input to the memory the program from the table 5.6 of a subtraction of 16-digit numbers HL=DE – BC/

Table 4.6

Address	Command	Code	Comment
800	MOV A, E	7B	Load A from the register E
801	SUB C	91	Subtraction of data bytes E - C
802	MOV L, A	6F	Load L from the register A
803	MOV A, D	7A	Load A from the register D
804	SBB B	98	Subtraction of higher byte counting carry D, B - bit of carry
805	MOV H, A	67	Load H from the register A

Execute the program previously taking the initial data of your own height. Checkup obtained results.

The commands of double addition look like:
DAD H – addition HL=HL+HL.
DAD B – addition HL=HL+BC .
DAD D – addition HL=HL+DE.
Examples:

1) Execute the program from table 5.3 made the operation from the table 5.7 made the operation HL = BC + DE.

Table 5.7

Address	Command	Code	Comment
800	MOV H, B	60	Transfer H < -B
801	MOV L, C	69	Transfer L < - C
802	DAD D	19	HL=HL + DE

Execute the program previously taking the initial data of your own height. Checkup obtained results.

2) Input to the memory the program of subtraction of the address content of an array cell by its serial number and reading of an array cell in the register A, number of an element (0…256) – in the register C, base address of an array - in HL, shown in the table 5.8.

Table 5.8

Address	Command	Code	Comment
800	LXI H, 900H	21 00 09	Load base addresses of an array
803	MVI B, 0	06 00	Load B=0 - higher byte of mot.
805	DAD B	09	HL=HL + BC, address. element array
806	MOV A, M	7E	Reading of an array in register A

Execute the program previously taking the initial data of your own height. Checkup obtained results.

The commands of increment look like:

INR R – increase by unit content of the register: A, B, C, D, E, H, L.

INR M – increase by unit content of a cell of memory. The address M is in HL.

INX RP – increase at unit of contents of pair registers: BC, DE, SP. In the given command the identifier of the higher register (INX B) should be given.

Examples:

1) Write down to memory to the address 800H a command:

INR E: E = E + 1

Execute the program previously taking the initial data of your own height. Checkup obtained results.

2) To write down in memory codes of commands

```
800     21 00 09     LXI H. 900H: load. HL = 900H, address M
803        34          INR M        : M = M + 1
```

Execute the program previously taking the initial data of your own height. Checkup obtained results.

3) Write down to memory a code of a command

800 13 INX D: DE = DE + 1.

Execute the program previously taking the initial data of your own height as contents of register pair DE. Checkup obtained results.

The commands of decrement look like:

DCR R – reduction by unit of contents of the register A. B. C. D. E. H. L:

DCR M – reduction by unit of contents of a cell of memory - address M in HL

DCX RP – reduction by unit of contents of pair registers: BC, DE, HL. SP.

In a command the identifier of the higher register is specified. For example DCX B.

Examples:

1) Write down to memory the following command:

DCR C: C=C - 1.

Execute the program previously taking the initial data of your own height for the register C.

2) Write down to memory codes of commands:

800 21 00 09 LXI H, 900H: Load HL = 900H, address M;
803 35 DCR M: M = M – 1.

Execute the program previously taking the initial data of your own height for initial values of memory cell contents M. Checkup obtained results.

3) Write down in memory a code of a command:

DCX H and execute formula HL = HL – 1.

Execute the program previously taking the initial data of your own height as contents of register pair HL. Checkup obtained results.

The note: the command decrement of register pair does not effect on contents of bits of attributes.

5.2 The task

5.2.1 Write and execute the programs of addition and subtraction of contents of the registers, register pairs, cells of memory agrees of the methodical guide.

5.2.2 Write and execute the program of filling of an array on the given index of an array cell, duplication of contents of arrays, using commands of double addition.

5.2.3 Write and to execute the program of filling of an array of memory by data using commands increment of pair of registers and register.

5.2.4 Write and to execute the program of filling of an array of memory by data, using commands of decrement of pair registers and register.

5.3 Self-test questions

5.3.1 Characteristic of elementary operations of binary arithmetic.

5.3.2 Feature and type of introducing of operations of addition and subtraction.

5.3.3 Essences and assignment of operations of double addition, subtraction.

5.3.4 Operations of shift, their practical value.

5.3.5 Principles of filling and carry of arrays.

6 LABORATORY WORK No.6 LOGIC OPERATIONS. ARCHITECTURE AND COMMANDS

Purpose of work is to study principles of architecture and commands of logic operations. Subject of the research: logic operations and commands of data processing.

6.1 Brief theoretical information

Application of microprocessor devices assumes logic processing of information handling. Therefore they are able to replace a set of logic circuits, to execute complicated logic functions on the basis using elementary logic operations.

In a system of commands of МП КР580 for executing of logic operations the following commands are used:

- logic addition;
- logic multiplication;
- excluding or;
- inversion.

The commands of logic addition realize logic operation OR. Result is equal to 1, if even one of the appropriate bytes equal to 1, and is equal to 0, if both are equal to zero. For example:

10101001

OR

<u>00110010</u>

10111011

where OR – a label of operation OR.

The commands of logic multiplication realize logic operation AND. The result is equal 1, if both appropriate bits are equal to 1, and is equal 0, if even one of them is equal to 0. For example:

```
          1010100
AND
          00110010
          00100000,
```

where AND – a label of operation AND.

The commands of excluding OR realize logic operation ADDITION BY MODULE TWO. The result is equal to 1, if the appropriate bits are opposite (1 and 0), and is equal to 0, if they are identical (1 and 1, 0 and 0). For example:

```
          10101001
XOR
          00110010
          10011011
```

where XOR - label of operation ADDITION BY MODULE TWO.

The command of inversion realizes operation DENYING of contents of the accumulator. For example:

```
          10101001
NOT       ________
          01010110
```

where NOT — a label of operation DENYING.

All logic commands are executed bit-by-bit with eight digit operands. Thus one of operands should be in the accumulator, and second – or in one of the registers of common assignment, or in a cell of memory – or is set in second byte of a command. The result of execution of a command is recorded to the accumulator. Thus bit of carry is established to zero, and other bits are established according to result of execution of a command.

6.1.1 Commands of logic addition

ORA R – with the register: A, B, C, D, E, H, L;
ORA M – with a cell of memory, which address is in HL;
ORI B – with an immediate operand, B – byte.
Examples:
1) Write down in memory codes of the program, according to the table 6.1. executing the formula : A = (A OR C OR M OR 80H)

Table 6.1

Address	Command	Code	Comment
800	ORA C	B1	A = A OR C
801	LXI H, 900H	21 00 09	Load HL = 900 H, address M
804	ORA M	B6	A = A OR M
805	ORI 80H	F6 80	A = A OR 80H

Execute the program with the command <СТ> 800__807 <ВП> previously taking the initial data of your own height. Checkup obtained results.

2) Write down in memory codes of the program from table 5.2 realizing formula HL = (BC OR DE)

Table 6.2

Address	Command	Code	Comment
800	MOV A, C	79	Transfer A ←- C
801	ORA E	B3	A = A OR E
802	MOV L, A B	6F	Transfer L ← A lower byte of result
803	MOV A, B	78	Transfer A ← B
804	ORA D	B2	A = A OR D
805	MOV H, A	67	Transfer H ← A, highest byte of result

Execute the program with the command: <СТ> 800__806 <ВП> previously taking the initial data of your own height. Checkup obtained results.

6.1.2 Commands of logic multiplication

ANA R – with the register A, B, C, D, E, H, L;
ANA M – with a cell of memory, which address is in HL;
ANA B – with an immediate operand. B – byte.
Examples:
1) Write down to memory from table 5.3 codes of the program realizing formula : A = (A AND D AND M AND 7FH)

Table 6.3

Address	Command	Code	Comment
800	ANA D	A2	A = A AND D
801	LXI H, 900H	21 00 09	Load HL = 900 H, address M
804	ANA M	A6	A = A ANA M
805	ANI 7FH	F6 7F	A = A AND 7FH

Execute the program with a command <CT> 800__807 <ВП> previously taking the initial data of your own height. Checkup obtained results.

2) Write down to memory codes of the program, according to the table 5.4 realizing formula HL = (B OR C) AND DE AND F0FFH

Table 6.4

Address	Command	Code	Comment
800	MOV A, C	79	Transfer A ← C
801	ORA B	B0	A = A OR B
802	ANA E	A3	A = AND E
803	ANI FFH	E6 FF	A = A AND FFH
805	MOV L, A	6F	Transfer L ← A, lower byte of result
806	MOV A, D	7A	Transfer A ← D
807	ANI FOH	E6 F0	A = A AND F0H
809	MOV H, A	67	Transfer H ← A, higher byte of result

Execute the program with a command: <СТ> 800__80A <ВП> previously taking the initial data of your own height. Checkup obtained results.

6.1.3 Commands of addition by module two

XRAR – with a register: A, B, C, D, E, H, L;
XRAM - with a cell of memory, which address is in HL;
XRIB - with an immediate operand, B - byte.
Examples:
1) Write down to memory codes of the program, according to the table 6.5, realizing formula A = A XOR A XOR E XOR M XOR AAH

Table 6.5

Address	Command	Code	Comment
800	XRA A	AF	A = A XOP A, A = 0
801	XRA E	AB	A = A XOR E
802	LXI H, 900H	21 00 09	Load. HL =900 H, address M
805	XRA M	AE	A = A XOR M
806	XRI AAH	EE AA	A= A XOR AAH, res.

Execute the program with a command: <СТ> 800__808 <ВП> previously taking the initial data of your own height. Checkup obtained results.

2) Write down to memory, according to the table 5.6 codes of the program realizing formula HL = (A OR B) AND DE XOR (HL XOR C)

Table 6.6

Address	Command	Code	Comment
800	ORA B	B0	A = A OR B
801	ANA E	AE	A = A AND E
802	MOV E, A	5F	Transfer E←A
803	MOV A, C	79	Transfer A ← C
804	XRA L	A	A = A XOR L
805	XRA E	AB	A = A XOR E

Continuation of Table 6.6

Address	Command	Code	Comment
806	MOV L, A	6F	Transfer L, A, lower byte of result
807	MOV A, H	7C	Transfer A ← H, higher byte of result
808	XRA D	AA	A = A XOR D
809	MOV H, A	67	Transfer H ← A, higher byte of result

Execute the program with a command: <СТ> 800__80A <ВП> previously taking the initial data of your own height. Checkup obtained results.

6.1.4 A command of inversion

CMA – inversion of the accumulator.

Examples:

1) Write down in memory the program from in the table 6.7 realizing formula A = NOT ((NOT B) AND (NOT C))

Table 6.7

Address	Command	Code	The comment
800	MOV A, B	78	Transfer A ← B
801	CMA	2F	F = NOT A
802	MOV B, A	47	Transfer B ← A (NOT B)
803	MOV A, C	79	Transfer A ← C
804	CMA	2F	A = NOT A (NOT C)
805	ANA B	A0	A = A ANA B
806	CMA	2F	A = NOT A, result

Execute the program with a command: <СТ> 800__807 <ВП> previously taking the initial data of your own height. Checkup obtained results.

2) Write down to memory the program according to the table 5.8 realizing formula : M3 = NOT (NOT Ml) OR (NOT M2)

Addresses of cells of memory: Ml =900 H, M2 = 901 H, M3 = 902 H.

Execute the program with a command: <СТ> 800__80D <ВП> previously taking the initial data of your own height. Checkup obtained results.

6.2 Steps of work execution

1) Using command of logic addition writes and execute the programs of realization of the given formula s.

2) On the basis of commands of logic multiplication do the programs of the given formula s.

3) Using the command excluding OR write and execute the programs of realization of the specified formula s.

4) On the basis of inversion of the accumulator make and execute the programs of realization of the given formulas.

Table 6.8

Address	Command	Code	Comment
800	LXI H, 900 H	21 00 09	Load HL = 900 H, address H
803	MOV A, N	7E	Reading A = M1
804	CMA	2F	A= NOT A (NOT M1)
805	MOV C, A	4F	Transfer C ← A, subresult
806	INX H	23	HL = HL + 1, address M2 (901)
807	MOV A, N	7E	Reading A = M2
808	CMA	2F	A = NOT A (NOT M2)
809	ORA C	B1	A = A OR C
80A	CMA	2F	A = NOT A, result
80B	INX H	23	HL = HL + 1, address M3
80C	MOV M, A	77	Record M3 = A, result

6.3 Methodical guide

6.3.1 Execute item 1 of the task for formula HL = (B OR C OR DE) OR 8800H.

Fill in the table for values B, C, DE, HL, F. Initial values set arbitrary.

6.3.2 Execute the task 1 for the formula : M3 = Ml OR M2.

Addresses of cells of memory accordingly Ml = 900H, M2 = 901H, M3 = 902H.

Input data Ml and. M2 take arbitrary. Results present in the table: Ml, M2, M3, F.

6.3.3. Execute task 2 for formula HL = (HL AND BC OR DE) AND 03FFH.

Initial value take arbitrary. Fill in the table: HL(in.), BC, DE, HL(res.), F.

6.3.4 Execute formula M2 =A AND Ml OR C AND D.

Addresses of cells of memory M1= 900H, M2 = 901H. Initial values take arbitrary. In the table reflect values A, C, D, Ml, M2, F.

6.3.5 According to the task 3 realize formula H = ((L XOR D) OR E AND H) XOR (B OR C) XOR 0FH.

Initial values take arbitrary. In the table reflect contents of registers: HL(in.), D, E, B, C, HL (res.), F.

6.4 Contents of the report

6.4.1 Features of logic operations.

6.4.2 Essence of elementary logic operations.

6.4.3 Technique and commands of logic addition.

6.4.4 Technique and commands of logic multiplication.

6.4.5 Features and commands of addition by module two.

6.4.6 Sense and practical value of inversion.

7 LABORATORY WORK No. 7
OPERATIONS AND COMMANDS OF SHIFT

Purpose of work is to study operations and commands of simple and cyclical shift.

Subject of a research: features and abilities of information processing of the increased formats on the basis of operations and commands of simple and cyclical shifts in a system KP580.

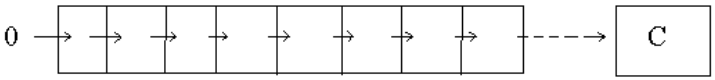
7.1 Brief of theoretical information

The commands of shift allow on the basis of elementary arithmetical operations to do a necessary set of operations on numbers. Besides do the completed operations on data processing. Are used mainly at bit testing and comparison.

The commands of shift of MP K580 (Intel 8080) operate only with data of the accumulator and do not require other operands located in memory or the registers. The addressing in this case is named implicit, and frequently the name is not specified. Many MP have others, than given MP, types of commands of shift. For example, MP Motorola 68000 is doing the shift of contents not only of accumulator, but also of memory cells, and command of operations of shift influence not only an attribute of carry, as it is in given MP, but also on all other attributes.

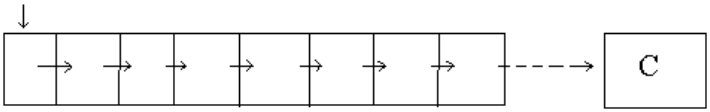
Let's consider the pictures with possible types of shift:

a) shift to the right:



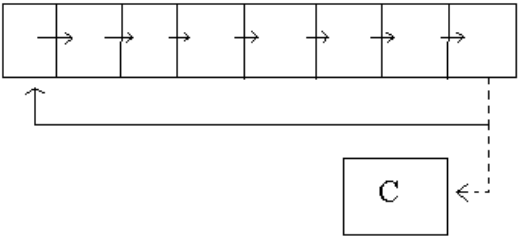
1) extreme right bit is transferred to C (register of carry);
2) all other bits are moved together to the right;
3) in extreme left digit is transferred 0.

b) Arithmetical shift to the right:



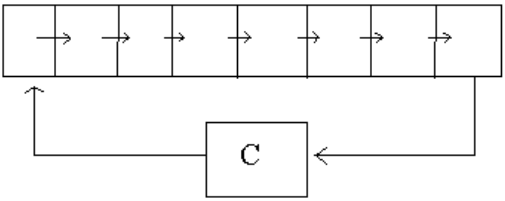
1) extreme right bit is transferred to C;
2) other bits are moved together to the right;
3) extreme left bit remains without change.

c) Cyclical shift to the right:



1) extreme right bit is transferred to C;
2) other bits are moved together to the right;
3) to extreme left digit initial contents of extreme right digit are transferred.

d) Cyclical shift to the right with carry:



1) extreme right bit is transferred to C;
2) other bits are moved together to the right;
3) to extreme left digit initial contents of the register of carry C are transferred;

The shifts to the left have identical technique, but are made in the opposite direction. Their diagrams and algorithms are offered to construct by yourself.

In a system of commands of MP K580 the following commands of shift are used:

1) cyclical shift to the left;
2) cyclical shift to the right;

3) shift to the left through carry;

4) shift to the right through carry.

The commands of shift are executed in the accumulator above 8-digit operands. The result is stored to the accumulator.

7.2 Commands of cyclical shift

The command of cyclical shift to the left displaces each bit in limits of byte on one digit to the left. Thus contents of the higher digit are recorded in lower digit and to a bit of carry.

The command of cyclical shift to the right will move each bit in limits of byte on one digit to the right. Thus contents of lower digit are recorded to the higher digit and a bit of carry.

RLC – command of cyclical shift to the left;

RRC – command of cyclical shift to the right;

Examples:

1) Write down to memory codes of the program realizing cyclical shift of byte on 4 digits

Table 7.1

800	RLC	07	Cyclical shifts to the left on 1
801	RLC	07	Cyclical shifts to the left on 1
802	RLC	07	Cyclical shifts to the left on 1
803	RLC	07	Cyclical shifts to the left on 1

Execute the program, previously set initial values <СТ> 800 __ 804 <ВП>.

2) Write down in memory codes of the program realizing operation of john of the higher tetrads of two bytes, contained in the registers B and C in one, using a command RRC

Table 7.2

800	MOV A, C	79	Transfer A ← C
801	RRC	0F	Migration of the higher tetrad of 1 byte
802	RRC	0F	on a place of lower
803	RRC	0F	Migration of the higher tetrad of 1 byte
804	RRC	0F	on a place of lower
805	ANI 0FH	E6 0F	Selection of the higher tetrad of 1 byte
807	MOV C, A	4F	Transfer C ← A
808	MOV A, B	78	Transfer A ← B
809	ANI F0H	E6 F0	Selection of the higher tetrad of 2 byte
80B	ORA C	B1	Join of two bytes in one

Execute the program, previously set initial values <СТ> 800 __ 80С <ВП>. Check up obtained results.

7.3 Commands of shift by carry

The command of shift to the left through carry displaces contents of each bit in limits of byte on one digit. Thus content of bit of carry is recorded to lower digit, and the content of higher bit is recorded to bit of carry.

The command of shift to the left through carry will move contents of each digit in limits of byte to the right on one digit. Thus in the higher digit of byte the value of byte of carry is recorded, and to it content of lower digit of byte will be recorded, using the given command, it is possible to realize a command of division on numbers, multiple 2.

RAL – command of shift to the left through carry;

RAR – command of shift to the right through carry.

Examples:

1) Write down to memory codes of the program realizing cyclical shift to the left on one digit of contents of register pair HL.

Table 7.3

800	ORA A	B7	Reset of carry in 00
801	MOV A, L	7D	Shift to the left of L on 1 digit through carry
802	RAL	17	0 to lower digit of L
803	MOV L, A	6F	L ← A
804	MOV A, H	7C	Shift to the left of H on 1 digit through carry
805	RAL	17	Taking into account carry from L
806	MOV H, A	67	Higher digit of H to carry
807	MOVA, L	7D	Carry to lower digit of L
808	ACI 0	CE 00	
80A	MOV L, A	6F	Transfer L ← A

Execute the program, previously set initial values <СТ> 800 __ 80B <ВП>. Checkup obtained results.

2) Write down to memory codes of the program realizing operation of multiplication by 4 of contents of the register C. B = C × 4

Table 7.4

800	MOV A, C	79	Transfer A ← C
801	ORA A	B7	Reset of bit of carry
802	RAL	17	Multiplication by 2
803	RAL	17	Multiplication by 2
804	MOV B, A	47	Transfer B ← A

Execute the program, previously set reference values <СТ> 800 __ 805 <ВП>. Check up obtained results.

3) Write down to memory codes of the program of division by 8 contents the register H. The whole part of result should be placed in D, rest in E.

Table 7.5

800	MOV A, M	7C	A ← M
801	ORA A	B7	Reset carry to 0
802	RAR	1F	A = A/2
803	ORA A	B7	The carry is equal 0
804	RAR	1F	A = A/2
805	ORA A	B7	The carry is equal 0
806	RAR	1F	A = A/2
807	MOV D, A	57	D = H/8 - whole part
808	MOV A, H	7C	Transfer A ← H
809	ANI 07H	E6 07	Selection of the result rest
80B	MOV E, A	5F	The rest to E

Execute the program, previously set initial values <СТ> 800 __ 80C <ВП>. Checkup obtained results.

7.4 Steps of work execution

7.4.1 Make by a program way joining of separate tetrads into byte.

7.4.2 Execute logic multiplication of the given bits of one byte by the given bits of another;

7.4.3 Make by a program way - cyclical shift on the given number of digits of contents of registers pair.

7.4.4 Multiply contents of registers pair by the given number.

7.4.5 Divide contents of registers pair by the given number.

7.5 Methodical guide

7.5.1 Write and execute the program of joining of lower tetrads of two bytes into one. A lower tetrad of 2nd byte place to the higher tetrad of resulting byte. Use: E – 1st byte, D – 2nd byte, A – result. Initial values of combined bytes take arbitrary.

7.5.2 Write and execute the program executing the operation of logic multiplication of 3 bits (5,6,7) of one byte and 3 bits (2,3,4) of the second byte, previously those bit should be moved to the lower digits and

other positions zeroed. Take the next values: L – 1st byte, H – 2nd byte and C – result. Initial values take arbitrary.

7.5.3 Write and execute the program of cyclical shift on 3 digits of contents of registers pair DE. Input data take arbitrary.

7.5.4 Write and execute the program of multiplication of contents of registers pair HL by 4. Input data take arbitrary.

7.5.5 Write and execute the program of division of contents of registers pair BC by 8. BC = BC/8. Input data take arbitrary.

7.6 Contents of the report

7.6.1. Fundamentals of the theoretical information.
7.6.2 The initial programs, data and results of the tasks.
7.6.3 Conclusions on the done work.

7.7 Self-test questions

7.7.1 Characteristics of shift to the right or to the left.
7.7.2 Features arithmetical shifts to the right or to of the left.

7.7.3 Algorithms of cyclical shifts.
7.7.4 Cyclical shifts with carry. Their features.
7.7.5 Operations and commands of shift of МПК 580
7.7.6 Practical use of operations of shift, their usefulness.

8 LABORATORY WORK № 8 OPERATIONS AND COMMANDS OF COMPARISON

The purpose of work is to study operations and commands of comparison of microprocessor КР 580.

Subject of the research: principles, technique of data monitoring during its processing.

8.1 Brief theoretical information

In practice there are situations, when it is a necessity to compare binary numbers. They are most typical at diagnostics of correctness of MP operation (MPS) and decision making during data processing.

The operation of comparison consists in comparing of any two data elements.

The fact of equality of two numbers can be revealed with the help of operations «Excluding OR" and "Subtraction". However the bad side of those methods is part or complete loss of input data during execution of the specified operations. The operation COMPARISON doesn't have those shortcomings. It based on a subtraction of contents of the register or cell of memory from contents of the accumulator without change of input data, i.e. contents of compared operating elements (cell of memory, register, accumulator) after execution of the specified operation.

In a system of commands of MP KP 580 three types of commands of comparison are used:

– comparison of contents of the accumulator and register,

– comparison of contents of the accumulator and cell of memory,

– comparison of contents of the accumulator with an immediate operand.

The commands of comparison are executed by means of an internal subtraction from contents of the accumulator accordingly, contents of the register, cell of memory or immediate operand. Contents of the accumulator thus do not vary. Result of comparison is setting of bits of attributes. The possible variants of set of attributes are shown in the table 8.1.

Table 8.1

Result of Comparison	Attributes	
	Zero	Carry
Equally	1	0
It is more	0	0
It is less	0	1

Except for the listed attributes bits of parity and sign are established by results of an internal subtraction. Bit of parity is equal 1 when an amount of units in the result is even, and is equal 0, if an amount of units is odd.

Bit of the sign is set equal to a value of the higher digit of result.

It allows process of execution of the program to be dependent on a value of result of execution of the previous operation. For calling to the information on results of evaluations in MP there is a set of flips making the program-accessible register (Flags) F, which are set to unit or are reset to zero depending on result of executed evaluations. The attributes are formed by results of arithmetical or logic operations. Each flip stores one of condition bits. The values of four bits of conditions can be checked with a help of a program:

1) S – sign of result (1 – minus, 0 - plus);
2) Z – zero result (1 – zero, 0 – not zero);
3) C – carry (1 – there is carry, 0 - the carry is not present);
4) P – parity, (1 – number of units is even, 0 – odd).

Fifth attribute is the additional carry AC from a lower tetrad in the higher during the processing of binary-decimal numbers, which is used together with a command DAA.

Arrangement of attributes in digits of RP the following:

Table 8.2

S	Z	0	AC	0	P	1	C
7	6	5	4	3	2	1	0

The check of attributes of a result is carried out in the binary form of contents of the register F.

8.2 Commands of comparison with contents of the register

Commands of comparison with contents of the register CMP are following: A (BF), CMP B (B8), CMP C (B9), CMP D (BA), CMP E (BB), CMP H (BC), CMP L (BD).

Example.

Write down to memory codes of the program of comparison of contents of the registers C and B.

Table 8.3

800	MOV A, C	79	Transfers A ← C
801	CMP B	B8	Comparison with register B

Execute the program, previously set initial values <СТ> 800 __ 802 <ВП>.

For checking the attributes, specified in the table, the content of F should be translated from hex to binary form and contents of required bits is checked.

Command of comparison with contents of a cell of memory CMP M (code BE) – comparison of contents of the register A and cell of memory, which address is given in pair HL.

Example:

Execute the program, previously set the initial values.

Table 8.4

800	LXI H, 900H	21 00 09	Loads HL, address
803	CMP M;	BE	Comparison A and M

8.3 Command of comparison with an immediate operand

Command of comparison with an immediate operand (code FE) are following:

CPI B - comparison of contents of the register A with number given in second byte of a command, where B - byte.

Example:

Write down to memory codes of a command of comparison with an immediate operand.

Table 8.5

800	CPI 7FH;	FE 7FH	Comparison of 7FH with A

Execute the program, previously set initial values <СТ> 800 __ 802 <ВП>

8.4 Steps of work execution

8.4.1 Make and execute the program of comparison of contents of the registers.

8.4.2 Make and execute the program of comparison of contents of the register and cell of memory.

8.4.3 Make and execute the program of comparison of contents of the register with an immediate operand.

8.5 Methodical guide

8.5.1. Execute item 1 of the task for the registers H and L. Make and fill in the table of input data and results. Input data take arbitrary. Specify condition of basic attributes.

8.5.2. Execute item 2 of the task for the register B and cell of memory with address 906H. Fill in the table of input data and results of attributes. Initial values of compared values take arbitrary.

8.5.3. Execute item 3 of the task for the register H and immediate operand 5AH. Fill in the table of input data and results of a condition of bits of attributes. Initial values of the register take arbitrary.

8.6 Content of the report

8.6.1 Basic information on an investigated question.

8.6.2 Programs of executing the tasks in language of Assembler and machine codes.

8.6.3 Tables of input data and results, conclusions on the done work.

8.7 Questions for self-verification

8.7.1 Features of comparison operations, their practical value.

8.7.2 Equivalent operations of comparison and their deficiencies.

8.7.3 Results of comparison and their definition.

8.7.4 To list attributes with possible values of bits and their brief performance.

8.7.5 Structure of the register of attributes and its practical use.

LIST OF REFFERENCES

1. https://uk.wikipedia.org/wiki/Арифметико-логічний_пристрій
2. https://led-stars.com.ua/ua/g5876122-155-seriya-mikroshem
3. https://org2.knuba.edu.ua/mod/book/tool/print/index.php?id=32482
4. https://uk.wikipedia.org/wiki/Комп%27ютер
5. https://uk.wikipedia.org/wiki/Лічильник_імпульсів
6. https://uk.wikipedia.org/wiki/Логічні_елементи
7. https://uk.wikipedia.org/wiki/Мікроконтролер
8. https://uk.wikipedia.org/wiki/Мікропроцесор
9. https://uk.wikipedia.org/wiki/Мікросхема
10. https://uk.wikipedia.org/wiki/ПЛІС
11. https://uk.wikipedia.org/wiki/Регістр_(цифрова_техніка)
12. https://uk.wikipedia.org/wiki/Суматор
13. https://uk.wikipedia.org/wiki/Тригер
14. https://uk.wikipedia.org/wiki/Цифровий_компаратор
15. https://uk.wikipedia.org/wiki/Шифратор
16. Lecture ОМРТ.pdf [Електронний ресурс] – Режим доступу: https://moodle.zp.edu.ua/mod/assign/view.php?id=96396
17. Бойко, В.І. Основи схемотехніки електронних систем підручник [Текст]:/ Бойко В.І., Гуржій А.М., Жуйков В. Я. та ін. – К.: Вища шк., 2004. – 527 с.: іл.
18. Болюх, В. Ф. Основи електроніки та мікропроцесорної техніки: навч. посібник / В. Ф. Болюх, В. Г. Данько. – Харків : НТУ "ХПІ", 2011. – 257 с. [Електронний ресурс] – Режим доступу: https://repository.kpi.kharkov.ua/items/b8236fc7-4f0c-4ab2-b925-f8f0eafd9f3d
19. Бондаренко, І.М. Мікропроцесорні системи контролю та керування: Навч. посібник для студентів ЗВО / І. М. Бондаренко, О. В. Бородін, В. П. Карнаушенко. − Харків: ХНУРЕ, 2020. – 244 с.
20. Грищук, Ю. С. Мікропроцесорні пристрої: Навчальний посібник. – Харків НТУ «ХПІ», 2007. – 280 с. [Електронний ресурс] – Режим доступу: http://web.kpi.kharkov.ua/ea/wp-content/uploads/sites/25/2013/04/Mikroprotsesorni-pristroyi.pdf
21. Електронний посібник з дисципліни «Мікропроцесорні системи». [Електронний ресурс] – Режим доступу: https://ms.ptngu.com/index.html

22. Колонтаєвський, Ю. П. Конспект лекцій з дисципліни «Мікропроцесорна техніка» (для студентів, які навчаються за напрямом 6.050701 – Електротехніка та електротехнології всіх форм навчання) / Ю. П. Колонтаєвський. – Харків: ХНУМГ ім. О. М. Бекетова, 2016. – 78 с.
23. Лекції ОМРТ 2022.pdf [Електронний ресурс] – Режим доступу: https://moodle.zp.edu.ua/mod/assign/view.php?id=96396
24. Методичні вказівки до виконання лабораторних робіт студентами всіх форм навчання при вивченні дисципліни «Основи мікропроцесорної техніки» для підготовки бакалаврів за спеціальністю 141 «Електроенергетика, електротехніка та електромеханіка» з подальшим навчанням за освітніми програмами «Електричні та електронні апарати» та «Електромеханічне обладнання енергоємних виробництв». Частина 1 / уклад.: Л. Б. Жорняк, М. В Антонова. Запоріжжя: ЗНТУ, 2019. – 72 с. [Електронний ресурс] – Режим доступу: http://eir.zntu.edu.ua/handle/123456789/4107
25. Методичні вказівки до виконання лабораторних робіт студентами всіх форм навчання при вивченні дисципліни «Основи мікропроцесорної техніки» для підготовки бакалаврів за спеціальністю 141 «Електроенергетика, електротехніка та електромеханіка» з подальшим навчанням за освітніми програмами «Електричні та електронні апарати» та «Електромеханічне обладнання енергоємних виробництв». Частина 2 / Л. Б. Жорняк, М. В Антонова. Запоріжжя: ЗНТУ, 2019. – 38 с. [Електронний ресурс] – Режим доступу: http://eir.zntu.edu.ua/handle/123456789/4108
26. Методичні вказівки до виконання лабораторних робіт студентами з англійською мовою навчання при вивченні дисципліни «Основи мікропроцесорної техніки» для підготовки бакалаврів за спеціальністю 141 «Електроенергетика, електромеханіка та електромеханіка» з подальшим навчанням за освітньою програмою «Електричні машини і апарати » / Л. Б. Жорняк, Г. А. Рябенко - Запоріжжя : ЗНТУ, 2016. – 66 с. [Електронний ресурс] – Режим доступу: http://eir.zntu.edu.ua/handle/123456789/3330
27. Мікропроцесор [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Мікропроцесор

28. Мікропроцесорна техніка: Навчальний посібник з дисципліни для всіх форм навчання та студентів іноземців напряму підготовки 6.050701 «Електротехніка та електротехнології»/ В. В. Кирик.-К.: ІВЦ «Видавництво «Політехніка», 2014.- 183 с.
29. Огородник, К. В. Мікропроцесорна техніка: навчальний посібник / К. В. Огородник, Б. П. Книш. – Вінниця: ВНТУ, 2018. – 106 с.
30. Основи інформатики. Складові частини мікропроцесорних систем [Електронний ресурс] – Режим доступу: https://elprivod.nmu.org.ua/ua/interesting/components_mp_systems.php
31. Основи мікропроцесорної техніки. [Електронний ресурс] – Режим доступу: http://vozom.ho.ua/MP/page11.html
32. Основні терміни. Структура та функціонування мікропроцесорної системи [Електронний ресурс] – Режим доступу: https://studfile.net/preview/7075014/page:3/
33. Поджаренко, В. О. Основи мікропроцесорної техніки. Навчальний посібник. / В. О. Поджаренко, В. Ю. Кучерук, В. М. Севастьянов – Вінниця: ВНТУ, 2006. – 226 с. [Електронний ресурс] – Режим доступу: https://core.ac.uk/download/pdf/52159035.pdf
34. Структура і принципи роботи мікропроцесорної системи [Електронний ресурс] – Режим доступу: https://stud.com.ua/28301/tovaroznavstvo/struktura_printsipi_roboti_mikroprotsesornoyi_sistemi
35. Терлецький, А. І. Будова та програмування 8-розрядного мікропроцесора. Методичні рекомендації до виконання лабораторних робіт з дисципліни «Архітектура комп'ютерів» (2-й семестр) для студентів напряму «Комп'ютерна інженерія» / А. І. Терлецький, О. Б. Фрик. − Івано-Франківськ, 2012. − 88 с.
36. Ткачов, В.В. Мікропроцесорна техніка: навч. посібник/В.В. Ткачов, Г. Грулер, Н. Нойбергер та ін. – Д.: Національний гірничий університет, 2012. – 188 с. [Електронний ресурс] – Режим доступу: https://library.kre.dp.ua/Books/2-4%2.pdf

APPENDIX A

FIRST TETRAD	SECOND TETRAD																
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	NOP	LXI BC	STAX BC	INX BC	INR B	DCR B	MVI B	RLC	--	DAD BC	LDAX BC	DCX BC	INR C	DCR C	MVI C	RRC	0
1	--	LXI DE	STAX DE	INX DE	INR D	DCR D	MVI D	RAL	--	DAD DE	LDAX DE	DCX DE	INR E	DCR E	MVI E	RAR	1
2	--	LXI HL	SHLD HL	INX HL	INR H	DCR H	MVI H	DAA	--	DAD HL	LHLD HL	DCX HL	INR L	DCR L	MVI L	CMA	2
3	--	LXI SP	STA adr	INX SP	INR M	DCR M	MVI M	STC	--	DAD SP	LDA adr	DCX SP	INR M	DCR M	MVI A	CMC	3
4	MOV B, B	MOV B, C	MOV B, D	MOV B, E	MOV B, H	MOV B, L	MOV B, M	MOV B, A	MOV C, B	MOV C, C	MOV C, D	MOV C, E	MOV C, H	MOV C, L	MOV C, M	MOV C, A	4
5	MOV D, B	MOV D, C	MOV D, D	MOV D, E	MOV D, H	MOV D, L	MOV D, M	MOV D, A	MOV E, B	MOV E, C	MOV E, D	MOV E, E	MOV E, H	MOV E, L	MOV E, M	MOV E, A	5
6	MOV H, B	MOV H, C	MOV H, D	MOV H, E	MOV H, H	MOV H, L	MOV H, M	MOV H, A	MOV L, B	MOV L, C	MOV L, D	MOV L, E	MOV L, H	MOV L, L	MOV L, M	MOV L, A	6
7	MOV M, B	MOV M, C	MOV M, D	MOV M, E	MOV M, H	MOV M, L	HLT	MOV M, A	MOV A, B	MOV A, C	MOV A, D	MOV A, E	MOV A, H	MOV A, L	MOV A, M	MOV A, A	7
8	ADD B	ADD C	ADD D	ADD E	ADD H	ADD L	ADD M	ADD A	ADC B	ADC C	ADC	ADC E	ADC H	ADC L	ADC M	ADC A	8
9	SUB B	SUB C	SUB D	SUB E	SUB H	SUB L	SUB M	SUB A	SBB B	SBB C	SBB D	SBB E	SBB H	SBB L	SBB M	SBB A	9
A	ANA B	ANA C	ANA D	ANA E	ANA H	ANA L	ANA M	ANA A	XRA B	XRA C	XRA D	XRA E	XRA H	XRA L	XRA M	XRA A	A
B	ORA B	ORA C	ORA D	ORA E	ORA H	ORA L	ORA M	ORA A	CMP B	CMP C	CMP D	CMP E	CMP H	CMP L	CMP M	CMP A	B
C	RNZ	POP BC	JNZ adr	JMP adr	CNZ adr	PUSH BC	ADI data	RST0	RZ	RET	JZ adr	--	CZ adr	CALL adr	ACI data	RST1	C
D	RNC	POP DE	JNC adr	OUT port n	CNC adr	PUSH DE	SUI data	RST2	RC	--	JC adr	IN port n	CC adr	--	SBI data	RST3	D
E	RPO	POP HL	JPO adr	XTHL	CPO adr	PUSH HL	ANI data	RST4	RPE	PCHL	JPE adr	XCHG	CPE adr	--	XRI data	RST5	E
F	RP	POP psw	JP adr	DI	CP adr	PUSH psw	ORI data	RST6	RM	SPHL	JM adr	EI	CM adr	--	CPI data	RST7	F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	