JPRS L/10318

11 February 1982

Translation

UNIFIED COMPUTER SYSTEM HARDWARE AND SOFTWARE (YES-II)

By

V.V. Przhiyalkovskiy and Yu.S. Lomov

FBIS FOREIGN BROADCAST INFORMATION SERVICE

NOTE

JPRS publications contain information primarily from foreign newspapers, periodicals and books, but also from news agency transmissions and broadcasts. Materials from foreign-language sources are translated; those from English-language sources are transcribed or reprinted, with the original phrasing and other characteristics retained.

Headlines, editorial reports, and material enclosed in brackets [] are supplied by JPRS. Processing indicators such as [Text] or [Excerpt] in the first line of each item, or following the last line of a brief, indicate how the original information was processed. Where no processing indicator is given, the information was summarized or extracted.

Unfamiliar names rendered phonetically or transliterated are enclosed in parentheses. Words or names preceded by a question mark and enclosed in parentheses were not clear in the original but have been supplied as appropriate in context. Other unattributed parenthetical notes within the body of an item originate with the source. Times within items are as given by source.

The contents of this publication in no way represent the policies, views or attitudes of the U.S. Government.

JPRS L/10318

11 February 1982

UNIFIED COMPUTER SYSTEM HARDWARE AND SOFTWARE (YES-II)

Moscow TEKHNICHESKIYE I PROGRAMMNYYE SREDSTVA YEDINOY SISTEMY EVM (YES EVM-2) in Russian 1980 (signed to press 1 Aug 80) pp 1-232

[Book "Unified Computer System Hardware and Software (YeS-II)" by Viktor Vladimirovich Przhiyalkovskiy and Yuriy Sergeyevich Lomov, Izdatel'stvo "Statistika", 20,000 copies, 232 pages]

CONTENTS

[Text] Annotation

This book deals with the questions of development of the architecture, internal structure and principles of operation of the family of software-compatible computers in the Unified System (YeS EVM). The main problems of development of functional properties, technical characteristics and software for the Unified System are reflected.

This book is intended for specialists using Unified Computer System hardware and software, for computer center workers, students in VUZ's and people interested in the problems of development of computers.

Foreword

Equipping the national economy with computers is now a major factor determining the rate of development of science, growth in industrial production and enhancement of managerial efficiency. Much attention was paid in the decisions of the 24th and 25th CPSU congresses to the development of computers and introduction of them in the various spheres of the national economy.

A decisive role in the development of computers continues to be played by general-purpose computers used to solve scientific and technical problems and to solve problems within information and logic systems and automated control systems at various levels. The emergence at the end of the sixties of minicomputers and in recent years even microcomputers has not only not reduced the field of application of general-purpose computers, but rather, conversely, has expanded it through the emergence of combined systems in which information undergoes initial processing on micro and minicomputers and is then sent for centralized processing and storage to a central computer.

In 1969, an intergovernmental agreement was concluded between the countries in the socialist community and the Intergovernmental Commission on Cooperation in Computer

Technology was formed. Because of this agreement, the rate of development, production and introduction of modern computer hardware into the national economy of the countries in the socialist community has increased considerably. Within a relatively short period, the Unified System of Computers was developed in the community countries. The system includes several computers with various throughputs that have full program and information compatibility, an extensive set of peripherals and common technological design principles of construction.

The results of cooperation on the program for the first phase of the Unified System (YeS EVM-1) were presented at the international exhibition "YeS EVM--73" held in Moscow. By that time, development had been completed on six computers with throughput ranging from 5000 to 500,000 operations/second, about 100 types of peripherals and four operating systems supporting efficient operation of the hardware.

In 1979, the second international exhibition, "YeS EVM and SM EVM--79," was held successfully. Presented at it were the results of cooperation among the countries in the socialist community on the program for the second phase of the Unified System (YeS EVM-2). YeS EVM-1 was replaced by the new, more modern YeS EVM-2 system that consists of seven computers with throughput ranging from 20,000 to 5 million operations/second.

The YeS EVM-2, production of which began several years ago, is an essential evolution of the YeS EVM-1 in many directions. The most essential features of the new system are:

the increase in throughput in the YeS EVM-2 computers, achieved in the majority of cases without increasing cost;

the increase in precision of computations (appearance of operations on operands having quadruple length);

the introduction of special control operations to expand computer operating modes, in particular to facilitate operation within multimachine and multiprocessor complexes;

further improvement of the input/output system, appearance of new peripherals with higher throughput, increase in channel throughput, and the capability of simultaneous operation of several high-speed storage units;

the emergence of supplementary hardware to raise the efficiency of software system operation in the modes of multiprogramming, time sharing, teleprocessing, multimachine and multiprocessor operation;

the introduction of hordware and software to support organization of the virtual storage mode;

improvement of the monitoring system, introduction of facilities for correcting single and detecting double errors. Introduction and development of facilities for diagnosing malfunctions and recovery of system after failure;

further development of means of microprogram control, introduction of reloadable storage of microprograms; and

development of the software system to support the new computer operating modes and new hardware.

It should be noted that the series of YeS EVM-2 machines provides for software compatibility between the models of this series and with the YeS EVM-1 machines. In addition, the YeS EVM-2 system, just as the YeS EVM-1, maintains full compatibility at the user program level with widespread foreign computer systems, i.e., joins in fact the world sta.dard, formed in the last 10-15 years, of external structure of general-purpose computers.

This book covers a broad range of questions of architecture, internal structure and organization of computer systems, as well as questions of software for the YeS EVM-2 machines. It is the first publication to elucidate comprehensively the principles of operation of the new system.

Academician V. S. Semenikhin.

Chapter 1. Unified System Architectural Development

1.1. Unified System of Electronic Computers

The family of program-compatible third-generation computers, called the Unified System of Electronic Computers (YeS EVM), includes a series of models with a speed ranging from several thousands to several millions of instructions per second. All models in the Unified System are united by common principles of operation, uniformity of control procedures and a uniform method of organization of links between the individual system modules. Thanks to the large nomenclature of peripherals and the standard method for connecting them, computers systems with various configurations can be set up. The software and hardware ensure efficient operation of the Unified System of Computers in various modes for solving a wide range of scientific, technical, economic, managerial and other problems and offer the capability of using these computers both in computer centers of varying function and in various types of automated systems for data management and processing.

Two phases of work on the Unified System program have now been completed: the first phase, YeS EVM-1, and the second, YeS EVM-2.

The main goal in developing the first phase was to develop a family of third-generation computers meeting the modern requirements imposed on the architecture, software, design and technology. In addition, as a result of the rapidly developing field of computer applications, the need arose for computer hardware with high operating qualities and intended for mass production.

This goal could not have been met without the solution to the problems of further evolution of the theory of organization of structures and computational processes of computers, and the development of related fields of science and technology that support creation of the new element base with integrated circuits, new materials and equipment. In the process, methods of automating the design and production of computers were extensively developed and implemented.

The structure and principles of operation of phase-one computers are defined by the following basic concepts:

computer software compatibility;

standard set of instructions, forms and formats for data representation;

standard set of operating units;

standard procedures for the input/output control system [IOCS];

extensive nomenclature of peripherals connectable through a standard input/output interface; and

unity of design principles that support a high degree of unification.

The YeS EVM-1 includes several dozens of types of peripherals, four operating systems and seven computer models: YeS-1010 (VNR [Hungarian People's Republic]), YeS-1020 (USSR), YeS-1021 (ChSSR [Czechoslovak Socialist Republic]), YeS-1030 (USSR), YeS-1032 (PNR [Polish People's Republic]), YeS-1040 (GDR) and the YeS-1050 (USSR).

In subsequent years, new models were developed: YeS-1012 (VNR), YeS-1022 (USSR), YeS-1033 (USSR) and the YeS-1052 (USSR) which are modifications of the YeS-1010, YeS-1020, YeS-1030 and the YeS-1050 computers developed earlier.

The main technical characteristics of the YeS EVM-1 were widely published earlier.

Thus, 11 models of the family of the Unified System have been developed and are in operation within the YeS EVM-1.

In the YeS EVM-2, which is a further evolution of the YeS EVM-1, all the merits of the preceding system have been kept, and in addition, new functional capabilities have been added. The latest achievements in the microelectronic component base were made use of in developing it. The improved technical and economic characteristics of the YeS EVM-2, which put it on a qualitatively higher level of development and meet the enhanced user requirements, were obtained as a result of:

raising the throughput of the central processing units and the overall efficiency of the system while at the same time observing the requirements of economy and adaptability to manufacture;

expanding hardware and software functional capabilities;

developing the input/output system;

increasing the capacity of main storage and improving the organization of data storage;

raising system reliability, developing efficient checking and diagnosing facilities;

further developing the software system;

providing the capability of setting up multiprocessor and multimachine computer systems; and

developing a new complex of peripherals.

The YeS EVM-2 now includes a large nomenclature of hardware and software for the Unified System which supports operation of seven models: YeS-1015 (VNR), YeS-1025 (ChSSR), YeS-1035 (USSR), YeS-1045 (USSR), YeS-1055 (GDR), YeS-1060 (USSR) and the YeS-1065 (USSR).

1.2. Basic Directions in Development of Principles of Operation of the Unified Computer System

Combining the efforts of a large number of teams of developers, including those from the different countries, to implement the Unified Computer System is possible given the common logic structure of the entire system that defines the set of interrelated requirements and links between the hardware and software components, i.e., with the common principles of system operation.

The YeS EVM-2 operating principles were developed thanks to including in the logic structure:

an expanded instruction set;

expanded control facilities in the processor;

indirect addressing of data in the channels;

the block-multiplex mode of channel operation;

new facilities for multiprocessor operation;

facilities for raising precision of floating-point operations;

an expanded system of interrupts;

new facilities for time readout;

facilities for recording program events;

facilities to support monitor programs; and

facilities for raising the efficiency of checking and diagnosing.

Instruction System. In contrast to the YeS EVM-1 (computing functions of which are implemented by the universal instruction set), the YeS EVM-2 has a more developed instruction set. The additional instructions are intended to provide for new functions of the processor and channels, provide multiprocessor facilities and facilities for time readout, and to expand the capabilities of scientific and technical computer applications. The instruction set used to execute system control functions has also been increased considerably.

Control Modes in the Processor. There are two modes of operation of the processor in the YeS EVM-2: basic control mode and extended control mode.

The basic control mode provides for functioning of the computer in accordance with the YeS EVM-1 operating principles. The extended control mode is intended to make use of the new YeS EVM-2 functions and facilities. To this end, control registers that can be addressed by a program have been incorporated in the processor structure. Stored in the control registers are information on system status and information needed when implementing YeS EVM-2 capabilities.

Dynamic Address Translation. Computer system efficiency is largely determined by the method of organizing storage of data, addressing system adopted and structure of distribution of information flows. In the YeS EVM-2, these questions have been resolved by introducing the facilities of dynamic address translation and indirect addressing of data, which with the corresponding support of the operating system and in the aggregate with it allow making use of a 16-megabyte extent of virtual storage.

The concept "virtual" has become widespread in recent years and literally means "apparent." It is used in terms such as "virtual storage," "virtual machines," etc. In this case it means that the complete illusion of using a several-fold greater size of main storage is created for a programmer working with the physically limited size of main storage, or the illusion is created that each user has been allocated his own installation with all resources when several users are working simultaneously with one computer. This illusion is achieved through special organization of control of the corresponding processes. This is the purely external aspect of the virtual concept. The internal significance is that virtual organization allows making more efficient use of system reources, for instance in the examples cited, through reducing the limitations on the size of main storage or, respectively, limitations on the joint use of the equipment.

The introduction of dynamic address translation facilities reduces the limitations associated with the necessity of assigning fixed areas of real main storage for programs. Dynamic address translation facilities allow placing programs or parts of them in external storage units (i.e., outside main storage) with subsequent introduction of them into the free space of main storage upon request under control of the processor. These relocations, as well as relocations of information from main storage to external are performed automatically by the system without programmer intervention. This allows making efficient use of computer resources, including main storage resources, in the process of computations.

Thus, dynamic address translation facilities offer the user working storage, the size of which exceeds the size of storage physically connected to the computer.

Indirect Addressing of Data. While dynamic address translation allows translation of addresses of instructions and data formed in the processor, the mechanism of indirect addressing of data serves the same purpose during input/output [IO] operations in a channel. Indirect addressing facilities optimize execution of IO operations, allowing one channel command to control transmission of data located in noncontiguous areas of real main storage.

Dynamic address translation in the processor and indirect addressing of data in a channel are used together to organize the virtual storage mode.

Block Multiplex Mode of Channel Operation. In the block multiplex mode, blocks of data from various IO devices are multiplexed, and in the process, each data block is transmitted in the burst mode. Implementation of block multiplexing raises considerably the efficiency of channel operation through parallel operation of several high-speed external IO devices.

Facilities for Multiprocessor Operation. Multiprocessor operation presumes organization of access of two (multi in the general case) processors to a storage area defined as common in a system configuration. There may also be storage areas belonging to the individual processors in the system. In addition to programs and the resident part of the operating system, the common extent of main storage holds information on the status of the processor and operations necessary for control of it. This information is stored in the starting area of storage with the addresses 0-4096, which is called the permanently allocated (fixed) area.

Multiprocessor facilities are intended for operation of several processors with a common extent of main storage. These facilities provide for partitioning of main storage, address prefixing, interprocessor signaling and synchronization of astronomical time clocks.

Facilities for Raising Precision of Operations with Floating Point. For floating-point addition, subtraction and multiplication operations, facilities have been provided in the YeS EVM-2 that enable working with a mantissa consisting of 28 hexidecimal numbers. In addition, necessary operations are performed to round off the result and convert numbers from the long to the short format and vice versa.

The introduction of these facilities stems from the fact that with increased computer throughput, it has become possible to solve complex and laborious scientific and technical problems with high precision. These problems, as a rule, are characterized by a large number of computational iterations, as a result of which a considerable increase in rounding errors could be expected with ordinary precision of computations.

The system of interrupts makes it possible to expeditiously react and change the status of the computer under certain conditions that arise in the system or outside it. These conditions include requirements on the part of control facilities and external units, incorrect results of execution of operations, improper addressing as well as results of operation of apparatus monitoring circuits.

The interrupt system has been enhanced by the introduction of program debugging facilities, improvement of the apparatus for error detection and correction, and expansion of the functional properties of the processor and capabilities of multiprocessor organization.

Time Readout Facilities. In the YeS EVM-1, the sole hardware for time readout was the interval timer. The increase in processor throughput required development of new facilities whose resolution would be commensurate with the instruction processing rate. In addition, software facilities for time readout using just the interval timer are rather inconvenient, require large storage size and take up considerable processor time for their operation.

In addition to the interval timer, a time-of-day [TOD] clock and a CPU timer have been incorporated in the YeS EVM-2. They have been implemented in the form of hardware-controlled 52-bit counters with a resolution of 1 microsecond. Their values are set by the instructions SET CLOCK and SET CPU TIMER.

The CPU timer does not change when the processor is in the stopped state. Once set, the TOD clock runs even when the CPU is in the wait or stopped state. The clock measures elapsed time. It can be used to determine the date and time of day.

A CPU timer interruption occurs each time a negative quantity is in the timer. To interrupt the CPU at a given time indicated by the TOD clock, there is a comparator in the CPU in which a specified value is set by the instruction SET COMPARATOR.

New instructions have been introduced to store readouts from the clock, CPU timer and comparator in storage: STORE CLOCK, STORE CPU TIMER and STORE CLOCK COMPARATOR.

Program Event Recording. Facilities for program event recording and providing monitor programs give the user the capability of debugging new programs at the same time other problems are being solved.

These facilities interrupt processor operation and write to a defined storage area information needed by a user when the following events occur in a program:

successful completion of a transfer instruction;

change in contents of specified general-purpose registers;

instruction is fetched from specified area of main storage; or

change in contents of specified area in main storage.

Operation of the facilities for program event recording reduces processor speed; therefore, to maintain the rate of execution of programs not needed for these facilities, masking of them has been provided for.

Provision of Monitor Programs. Facilities for support of monitor programs permit organizing transfer of control to a program that effects specific control functions (the monitor). This occurs when the instruction CALL MONITOR is encountered in a program being executed and transfer of control is permitted (not masked).

Checking and Diagnostics. Capabilities for raising the efficiency of checking and diagnostics in the YeS EVM-2 are provided by various software and hardware facilities for detecting, localizing and correcting errors, as well as facilities to restore the computing process when malfunctions occur.

1.3. Basic Properties of Unified Computer System Architecture

In making an overall evaluation of all the changes in the logic structure of the YeS EVM-2, it can be noted that they are aimed at evolving the main features of the Unified System Architecture: efficiency, general-purposeness, compatibility and reliability.

Efficiency. The new models of the second phase of the Unified Computer System, from an architectural point of view representing one computer, cover a broad range of throughput that increases from the bottom to the top of the line. In addition, the computational capacity of each model can be enhanced by uniting two processors by means of multiprocessor facilities. As a rule, within the bounds of one system, the throughput of each successive model is higher than a dual processor system built on the basis of the preceding model.

YeS EVM-2 models have better technical and economic characteristics than YeS EVM-1 models. In particular, the throughput/cost ratio has been increased two to threefold; the absolute value of throughput for processors in the top models also has been increased: from 0.7 (YeS-1052) to 5 million instructions/second (YeS-1065); and size of main storage has been increased. Storage size ranges from 1 to 8 megabytes as a function of model throughput; IO system throughput has been raised both through the capability of connecting a larger number of channels and the increase in their throughput to 3 megabytes/second.

In addition, along with the operating systems developed for phase-one models, software is used in phase-two models that takes the new functions and capabilities into account.

All this taken together gives a wide choice of hardware and software while ensuring efficiency in solving specific user problems.

General-purposeness. General-purposeness means the capability of solving a wide range of problems in different classes with approximately the same efficiency. In second-phase models, this capability is supported primarily by the expansion of the general-purpose instruction set, which consists of the standard set and instructions for economic and scientific applications.

The standard set includes instructions for fixed-point arithmetic, control, exchange, IO, logic operations and storage protection instructions. The instruction set for economic applications contains the standard instruction set plus instructions that allow operation of facilities for processing decimal data with variable wordlength. Instructions for scientific applications include floating-point operations in addition to the standard set.

Versatility in using Unified System Computers is also achieved by the modular hardware principle that allows connecting peripherals for various functions and developing a software system that includes modes for multiprogramming, time sharing, interaction and teleprocessing.

Thus, the appropriate components can be selected to build a specific computer complex most suited to a given application considering the requirements for throughput, functional capabilities and set of peripherals.

Compatibility. Second-phase architecture ensures software compatibility both between second-phase models and with phase-one models. The different second-phase models are compatible downwards and upwards. This provides for convenience in maintenance, use of unified operating systems and application program packages developed earlier, and simplicity in training and mastering the new models.

However, the principle of compatibility does have limitations: programs must use identical hardware and must not depend on the duration of execution of instructions by a processor and on functions peculiar to a given model (model-dependent). In addition, a program must not use areas assigned or that could be assigned for special hardware functions.

Phase two is compatible with phase one from "bottom up." To execute programs on phase-two models, that were written for phase-one models, not only the above cited limitations have to be taken into account, but also those associated with the features of the new system, which consist in:

proper use of the bits in the program status word that determine the codes for output of results;

proper use of permanently allocated storage areas in the YeS EVM-2; and

taking into account the new capabilities of the IO channel for pre-fetching of information and the new capabilities for retry of instructions and specifics of the new instructions.

Reliability of the YeS EVM-2 models is ensured both by the technology of manufacture and by the use of special hardware and software facilities (more flexible system for processing machine errors, circuits for correction of main storage errors, mechanism for retry of instructions in the processor and channels, and the system for microdiagnostic procedures).

In addition, the capabilities for preventing erroneous writes have been increased in the YeS EVM-2 through dynamic address translation which permits isolating one program from the other during joint use of the same resources.

1.4. Structure of YeS EVM-2 Models

The base structure of the YeS EVM-2 models (fig. 1.) is described by the presence of the obligatory (standard) functional devices of the five levels: the central processing unit (processor), main storage, channels, peripheral control units and the peripheral equipment. The internal organization of each of these devices may vary in the different models, but in doing so, the common principles of operation that ensure compatibility and the unified methods of system control are maintained.

Peripherals are connected through control units to channels by a standard IO interface. Properly, this interface can be considered the sixth obligatory level of the system. The channels have a link with the central processor since the latter coordinates and controls the entire system as a whole, and a direct link to main storage, thanks to which independent operation of the channels and the processor is achieved. The system permits connection of different types of channels: byte-multiplexer, selector and block-multiplexer.

The structure of the YeS EVM-2 permits the capability of connecting additional (nonstandard) functional units (for example, array processor, channel-to-channel adapter, logic repeater)and various units that support communication between two processors at the level of external storage units or a common area of main storage.

The central processor is the nucleus of the system. Its functions include execution of arithmetic and logic operations, organization of main storage access, fetching and decoding of instructions, initiation of operation of executive units and IO procedures, processing of interrupts and others. In accordance with these functions, the processor includes a central control unit, an arithmetic and logic unit and a storage control unit.

The internal organization of the processor structure (by which is meant the interaction of the functional assemblies and units for organization of the computational process) may vary from model to model, but in all models the processor performs the same logic functions, i.e. the result of execution of an instruction will be identical irrespective of the model.

As a function of requirements for throughput and economy, the internal structure permits parallel or serial processing of information, a different width of data streams, varying number of levels of overlap of instructions, and various algorithms for execution of operations.

At the same time, unity of principles of operation is achieved by standard facilities: program status word register, general registers, registers with floating-point and control registers which form the internal storage for the processor. Stored in these registers are control information, information on system status, addresses and operands. In addition, adherence to the principles of operation is achieved by unity of formats and forms of representation of instructions and data.

The program status word (SSP) [PSW], which is stored in a 64-bit register, is intended for instruction sequence control. The PSW contains the address of the current instruction and information on the status of equipment needed by the program being executed at the given time. The set of this information varies as a function of the control mode.

Sixteen 32-bit general-purpose registers are used as index registers in operations for addressing and as data and result registers for logic and fixed-point operations.

The general-purpose registers are addressed from 0 to 15 by using a four-bit code placed directly in the instruction. For work with 64-bit numbers, adjacent registers may be used in pairs--an even and an odd register.

There are four 64-bit floating-point registers intended for storing operands during floating-point operations. The first two and last two registers may be combined when the expanded data format (128-bit) is used.

Sixteen 32-bit control registers (RU) are used in the processor for performing the new YeS EVM-2 functions. They are addressed by using a four-bit code in the instructions LOAD CONTROL and STORE CONTROL. It is possible to access a group of control registers.

Operating principles are adhered to also by the unity of formats and forms of data representation.

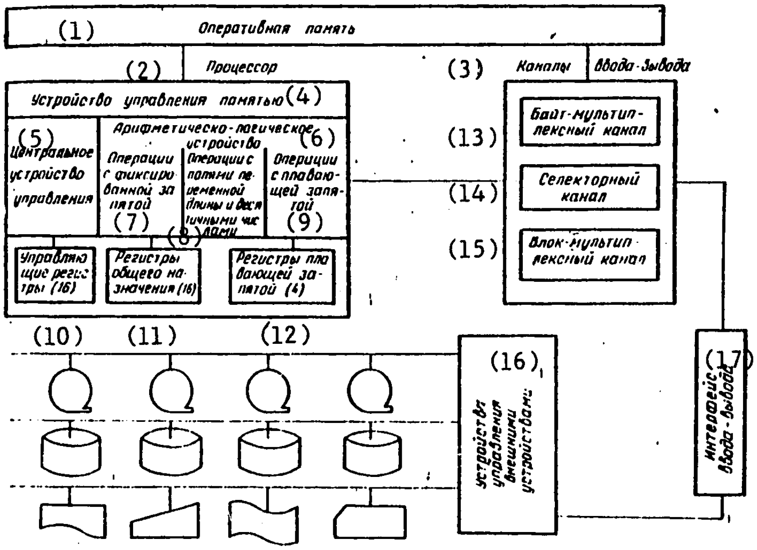

(1) Оперативная память
(2) Процессор
(3) Каналы ввода-вывода
Устройство управления памятью (4)
(5) Центральное устройство управления
Арифметическо-логическое устройство (6)
Операции с фиксированной запятой (7)
Операции с полями переменной длины и десятичными числами (8)
Операции с плавающей запятой (9)
Управляющие регистры (16)
Регистры общего назначения (16)
Регистры плавающей запятой (4)
(10)
(11)
(12)
(13) Байт-мультиплексный канал
(14) Селекторный канал
(15) Блок-мультиплексный канал
(16) устройства управления внешними устройствами
(17) Интерфейс ввода-вывода

Fig. 1. YeS EVM-2 Base Structure

Key:

1.	Main Storage	9.	Floating-point operations
2.	Processor	10.	16 control registers
3.	IO channels	11.	16 general-purpose registers
4.	Storage control unit	12.	4 floating-point registers
5.	Central control unit	13.	byte-multiplexer channel
6.	Arithmetic and Logic unit	14.	selector channel
7.	Fixed-point operations	15.	block-multiplexer channel
8.	Operations with variable-length fields and decimal numbers	16.	peripheral control units
		17.	IO interface

All instructions (fig. 2) have a length of two, four or six bytes and one of six formats: RR, RX, RS, SI, S or SS. The instruction format reflects the location of the operand taking part in the operation.

In the RR instruction format, both operands are placed in the general registers or the floating-point registers, depending on the type of operation. When an RX format instruction is executed, one operand is selected from the general registers or floating-point registers, and the second one from main storage. For this format, the address of the second operand (memory address) is formed by using an index. The RS instruction format differs from the RX only in that indexation of the address of the second operand is not required for it. For the SI format instruction, the first operand is selected from main storage and the second directly from the instruction. In executing SS format instructions, both operands are selected from main storage. For S format instructions, the first operand is specified in implicit form, and the second is placed in main storage.

Execution of instructions in the processor begins with fetching of the instruction whose address is placed in the program status word register. Then this address is incremented by the number of bytes in the instruction, forming the address of the next instruction.

The address by which the data is accessed in main storage is contained in the register indicated by the field R of the instruction, or is formed from the base, index and displacement, defined by the fields B, X and D of the instruction (index 1 or 2 defines the number of the operand).

The base is placed in general-purpose registers and is a 24-bit number, by the use of which main storage can be addressed. The base address is used for addressing an area of a program or data files and can also be used as an index.

The index is also placed in general-purpose registers and is used only in RX format instructions for addressing an individual element in a file by the use of a 24-bit number.

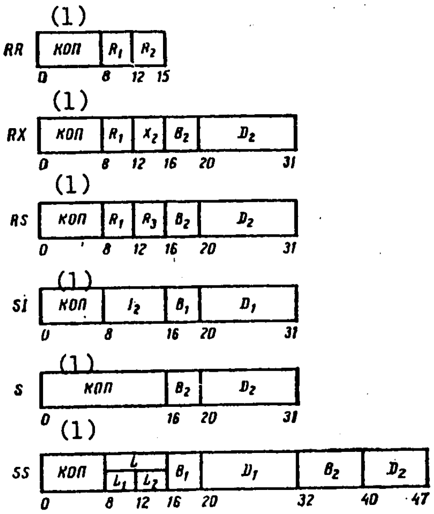


Fig. 2. Instruction Formats

Key:

1. KOP [operation code]

The displacement is a 12-bit number, specified directly in the instruction, that addresses individual sections of a file with a size of 4096 bytes.

In forming the address, the base, index and displacement are added together as positive binary numbers and overflow is ignored.

In SS and S format instructions, length of operands in bytes is specified by field L.

Development of the YeS EVM-2 processor structure is aimed at increasing the efficiency of the computational process. In this plan, besides the new functions determined by the principles of operation, the most important are the further development and introduction of microprogram control, and the introduction of buffer main storage that supports fetching of data at a rate corresponding to the processor cycle of operation.

Main storage is intended for storing data files and must ensure fast access with direct addressing to information for the processor and channels. The largest main storage size is 16,277,216 bytes and is defined by a 24-bit address that always points to the extreme left byte in fetching some block of information. In the process, the length of the memory block addressed is determined by two methods: explicitly and implicitly. In the first case, the address is accompanied by a length code that indicates the number of bytes addressed. In the second, the length of the data field is defined by the instruction code. This is the case for fixed-length information since it can equal only 1, 2, 4 or 8 bytes.

To properly define the start of blocks of fixed-length information, the specific integral boundary rule has been established for placement of data in main storage. In accordance with this rule, fixed-length data are placed in storage beginning at the integral boundary for a given block of storage. The boundary for some block of information is called integral if its address is a multiple of the number of bytes in the block. Thus, the binary code of the address for reference to main storage for a halfword, word or doubleword of information must have one, two or three low-order bits, respectively, equal to zero. For the majority of unprivileged instructions in YeS EVM-2 processors, the restriction on placement of operands at the integral boundary has been removed. In certain cases, this makes it possible to reduce the size of memory taken up by data, however, in doing so, processor speed may be considerably reduced, especially for high-throughput computers. Therefore, placing operands at an arbitrary byte boundary is recommended only in exceptional cases.

The reduction in speed is associated with the fact that for fetching an operand located at a nonintegral boundary, there may be required, first, two references to main storage instead of one, and second, alignment of the operand prior to its processing in the arithmetic unit. Placing an operand at a nonintegral boundary for write instructions also requires two accesses to storage.

Development of the principles of virtual storage which expand the capabilities of a computer system led to the formation of concepts of logical, real and virtual addresses.

Logical addresses are those used by a program that are translated into real by the address translation facilities.

The different levels of storage linked together by information transmission channels form the aggregate of computer storage units called virtual storage. The highest level of storage, main storage or part of it, is called real storage. Addresses used to access real or virtual storage are called real or virtual, respectively.

Depending on size, main storage may be implemented both in the form of an individual unit and integrated in the processor. In both cases, access is provided to the processor and channels by priority, and channels have the highest priority.

The IO channels organize and service the process of data exchange between peripherals and main storage, freeing the central processor from these functions. Data is exchanged with peripherals in two modes: burst and multiplexer.

In accordance with operating modes, two types of channels are used in the YeS EVM-1: the selector which services one unit at a given time until completion of the data transmission in the burst mode, and the multiplexer that operates in both the burst and multiplexer modes, permitting simultaneous execution of several IO operations for data transmission in small portions (bytes, for instance) at time intervals less than 100 microseconds.

Used in the YeS EVM-2 is a new type of channel, a block-multiplexer which makes it possible to combine the operations for making the peripherals ready for data transmission with IO operations.

The channel functional capabilities were also developed further: facilities were introduced for indirect addressing during IO operations, supporting facilities of dynamic address translation of the processor in the organization of virtual storage, and facilities for repetition of instructions in a channel and the two-byte interface. The aggregate of these facilities has made it possible to increase IO efficiency to a considerable extent.

From the point of view of specific implementation, channels can be made both in the form of individual independent units and with partial use of processor equipment. In any case, uniformity of IO procedures defined by the YeS EVM-2 principles of operation is maintained.

The broad nomenclature of peripherals for the Unified Computer System includes devices with various principles of operations, functional purpose and rate capabilities: perforated card and tape units, tape and disk storage units, plotters and CRT units, etc.

The technical characteristics of facilities already in existence have been improved and new ones developed in the YeS EVM-2 peripherals. New disk storage units with large capacity and tape storage units with a high density of recorded information have been developed. The peripheral nomenclature has been supplemented with displays and systems based on them.

Control units are intended to control the peripherals. Physically, they can be placed in the peripherals themselves, in the processor or channels, or made as individual units, but in any case, realization of all capabilities supporting operation of channels with the peripherals is maintained.

The control units are connected to a channel through the IO interface which makes the control signals device-independent.

The YeS EVM-2 control units have been improved in accordance with the evolution of the IO peripherals and channels.

Chapter 2. Raising Efficiency of Computations

Enhancement of computational efficiency in the YeS EVM-2 is primarily linked to the development of the structural organization and to the expansion of the functional capabilities of the central processing unit, the processor, and is achieved through:

improvement in the throughput/cost ratio as the basic parameter of the general-purpose computer;

increase in the absolute value of throughput for models of each class;

development of structures with fundamentally new qualities: virtual organization, automation of program debugging, event recording and others;

availability of advanced facilities for building multiprocessor and multimachine systems, and special-purpose systems;

improvement of operating characteristics--reliability and validity of computations, reduction of hardware maintenance time and others.

Discussed in this chapter are the basic structural concepts and hardware features of the processor by which the new capabilities for data processing in the YeS EVM-2 are realized.

2.1. Main Concepts of Processor Structure

The criterion of efficiency of processor computations for all models in the Unified Computer System is the throughput/cost ratio, i.e. achieving a given throughput with the least outlays for equipment. However, this problem is solved differently for different models. While a given throughput is achieved for low and medium speed machines by relatively uncomplicated structural methods and, in connection with this, the determining factor is reducing equipment cost by using a cheaper element-design and technological base, for high-throughput machines that, as a rule, use the latest achievements in microelectronics and technology, special complexity is presented by the organization of the computer structure and optimization of the computing process to obtain a given rate of calculations. From this point of view, processor structure efficiency is largely determined by:

width of the paths for processing and transfer of data;

number of levels of instruction processing overlap;

structure and algorithms for operation of the executive units;

organization of execution of instructions for transfer of control;

organization of CPU internal storage; and

degree of joint use of equipment by the CPU a ıd channels.

In addition, improvement in the organization of CPU internal structure in all YeS EVM-2 models is related to the development of the principles of microprogram control and realization of the capabilities of connecting special-purpose units optimized for execution of a particular class of problems.

Width of Paths for Data Processing and Transfer. In the Unified System of Computers, the methods selected for addressing main storage and the byte form of data representation make it possible to use a different width of paths for data processing and transfer in different models as a function of economic expediency and the capability of achieving given parameters on throughput.

The width of the path for processing operands differs as applied to the CPU, i.e. the width of the arithmetic unit, and the width of the path for data exchange between the CPU and main storage (instructions and operands)

The width of the path for data exchange affects both the rate of processing and the size of the CPU equipment. This stems primarily from the fact that the width of the path for data exchange largely determines the organization and effective time of main storage operation.

As a rule, main storage is divided into independent logical blocks that makes possible access to some blocks before completion of an operation begun in other blocks. Used in the process is the principle of interleaving of addresses or the so-called interleaving of main storage, which consists in the fact that the addresses of related addressed storage cells are located in various logical blocks. Usually, a two-, four- or eight-fold interleaving of storage is used, i.e. there are two, four or eight independent blocks of storage in the system. The width of the path for data exchange with one logical block is two, four or eight bytes. With interleaving of storage, each logical block has its own independent main paths for data exchange.

The width of the path for data exchange in the majority of cases clearly defines the width of the path for processing of operands in the arithmetic and logic units.

By the width of the arithmetic unit is meant the width of the adder and the main registers for the operands. And when there are several adders in the CPU, as for example in high-throughput CPU's, the width is determined by the adder in which operations on operands with fixed-point are executed.

Since arithmetic unit equipment makes up from 40 to 70 percent of the total CPU equipment, the selection of the width for data processing has a considerable effect on the total equipment. At the same time, it is obvious that when the width of the processing path is reduced, the time for execution of operations will be increased for operands whose size exceeds the processing path width. Taking this into account, two, four and eight bytes are usually selected for the exchange and processing path width for low, medium and high-throughput machines, respectively.

Number of Levels of Instruction Processing Overlap. To increase the rate of instruction processing in the CPU's, the concurrent processing method is used: the whole processing process is subdivided into stages and execution of the different stages of several serial instructions coincides in one CPU operation cycle. To achieve concurrent processing of instructions in a CPU, functional and independent blocks are separated out; each of them executes only one of the instruction processing stages. For example, for the case of three-level concurrency, the processing process is subdivided into three stages: the stage of preparing the instruction for execution, i.e. fetching the instruction from main storage and generating the addresses of the operands, the operand fetching stage, and the stage of direct execution of the operation in the arithmetic unit. Subdividing into stages is usually based on equal time intervals for execution of the individual stages, since only in this case is maximum efficiency achieved for making use of the individual functional blocks of the CPU that afford concurrent processing, i.e. there is no idle time in waiting for the completion of operation of preceding or subsequent blocks.

For three-level concurrency in a CPU, it is necessary to assign a block for fetching instructions and generating addresses, a block for fetching operands and an executive block or arithmetic unit.

In addition, characteristic of YeS EVM-2 CPU's are stages for prefetching of instructions and operands and buffering them for subsequent processing, and overlap of execution of the various operations in the executive units of the arithmetic and logic unit, which increases even more the number of processing stages and overlap levels.

Although the efficiency of the method of overlapped processing of instructions, from the point of view of increasing CPU throughput, is a function of the number of overlap levels, this dependency is not proportional. The full use of efficiency of the overlap method is rather rarely achieved in solving problems, since any successful branch instruction in the program or use by a following instruction of results obtained during execution of a preceding instruction causes stoppages of instruction processing.

In fact, time T_c for processing S instructions with K levels of overlap can be represented by the following expression:

$$T_c = T\left(\frac{K+S-1}{K}\right),$$

where T is the time for execution of one instruction without overlap.

It is easy to see that when one instruction is executed (S=1), T_c=T, and with a full CPU load over a long time ($S \to \infty$), the rate of instruction processing tends to the value equal to T/K. Thus, the overlap method does not reduce the execution time for one instruction; it does make it possible to increase device throughput. Therefore, if dependent instructions (instructions, upon sequential entry of which to input of the unit, processing of the next instruction can be started only after completion of execution of the preceding one) are encountered among the instructions S, CPU throughput is reduced to the value defined by the order of dependency of the overlap levels compared to the case when there is no dependency of instructions. Levels of processing of instructions with numbers m and n ($m < n$) are dependent, if to start processing of the current instruction at level m, it is necessary and sufficient that processing of the preceding instruction at level n be completed. The value N=n-m is called the order of dependency of the overlap levels. It can be shown that in this case, the CPU instruction processing rate is defined by the expression

$$T_c = T\left(\frac{K_1+S-1}{K_1}\right),$$

where K_1=K/N. Physically, this means that in the case of dependency of instructions, it is as if the number of overlap levels is reduced and, consequently, the CPU throughput is reduced. Thus, if execution of an instruction at the first level cannot be started until the instruction at the last level has been executed (for example, the result of the preceding instruction changes the content of the following instruction), the order of dependence of the levels of overlap N equals K, which is equivalent to operating without overlap (K_1=1).

The relationship between instruction execution overlap efficiency and instruction interrelationships presented above is considered in the most general form. In fact, this relationship is far more complex and, in addition, instruction execution overlap efficiency is affected by a large number of other factors. It should also be taken into consideration that overlap of processing increases the size of equipment and complicates the control circuit.

All these circumstances have to be taken into account in selecting the number of levels of instruction processing overlap in each specific case to achieve given parameters and first of all the throughput/cost ratio. Experience in developing the Unified System computers and studies made indicate that overlap processing is technically and economically expedient for a class of machines in which up to five instructions are executed simultaneously. Therefore, the CPU's in the different models of the YeS EVM-2 generally use two- and five-level overlapping as a function of throughput. At the same time, sequential processing of instructions is provided for in the smaller models, and in those machines where overlapping is used, modes of operating without overlap have been provided for to facilitate search for and localization of malfunctions in diagnostic procedures.

Structure and Algorithms for Executive Unit Operation. These units include the traditional CPU arithmetic and logic units [ALU] and specialized hardware (for example, a high-speed multiplier) that was developed in the YeS EVM-2. Executed in these units are all the arithmetic and logic operations in the universal instruction set in the YeS EVM-2, which includes operations on numbers with fixed-point and with floating-point with conventional and extended precision, logical operations and decimal arithmetic operations. Operations are executed with fixed and variable length fields.

Further raising of the efficiency of executive units in the YeS EVM-2 compared to the YeS EVM-1 is determined by the:

expansion of functional capabilities;

improvement of structural organization; and

use of more efficient algorithms.

The functional capabilities of YeS EVM-2 CPU executive units have been expanded by the introduction of facilities for executing extended-precision floating-point operations. In this case, CPU efficiency is increased in solving scientific and technical problems and primarily those that have a large number of computational iterations. Using conventional-precision floating-point operations in solving these problems in some cases does not provide the necessary precision of the result because of rounding errors. The introduction of facilities that permit operating with a mantissa consisting of 28 hexidecimal numbers reduces the criticality of this factor. Seven extended-precision floating-point arithmetic instructions have been introduced in the YeS EVM-2: ADD NORMALIZED, LOAD ROUNDED (extended to long), LOAD ROUNDED (long to short), MULTIPLY (extended), MULTIPLY (long to extended--RR instruction format), MULTIPLY (long to extended--RX instruction format) and SUBTRACT NORMALIZED.

Executive unit structure in the various YeS EVM-2 models determines the following trend:

minimal size of equipment through universal arithmetic units with a small width of operating units (8-16 bits) for the small models;

use of additional equipment which speeds up execution of individual operations with an increase in the width of the operating assemblies to 32 bits for models in the medium class;

achievement of maximum speed in the large models through specialization in operations of arithmetic units of large width (64 bits); and use of microprogram control to the full extent of operations for small and medium-size models, and for large models to the extent in which achievement of the given throughput is possible.

With the introduction into the CPU of buffer storage that supports fetching data at the rate equal to the CPU operating cycle, the time for execution of operations in the operating units of the executive units has become decisive in determining the computer speed. CPU computation rate depends to the highest extent on execution of addition/subtraction and multiplication/division operations with fixed and floating point. Based on this, the structure of the operating units must in the first place be oriented to rapid execution of these operations.

An efficient and widely employed method for structural organization of executive units that results in increased throughput is the use of the principle of local parallelism which consists in parallel processing in time of the algorithm for execution of an individual operation by splitting it into a series of independent sections. Such parallel processing is possible when the result of operation of one section of the algorithm does not depend on the result of operation of another. Thus, in executing a floating-point operation, operations on exponents are executed at the same time as operations on the mantissas.

Concrete realization of this principle requires incorporating several processing functional assemblies that operate concurrently in time: mantissa adder, exponent adder, operating shifter and analysis circuits. The principle of local parallelism leads to increased speed not only through parallel processing of the operands proper but also through parallel analysis of the operands and execution of the functions of checking and control.

Used in the large YeS EVM-2 models also is the well known method of increasing speed: the method of conveyor processing. To implement conveyor processing, the entire processing apparatus is subdivided into steps, operating sequentially one after the other, that perform elementary operations. The sequence of elementary operations being performed while passing through all the steps determines the execution of the full operation. The high speed of the conveyor method of processing stems from the short fixed time for execution of an elementary operation and the concurrency of operation of all steps. Intermediate results of processing in these steps are stored in registers which makes it possible to avoid the effect of asynchronization. Selection of the number of steps and clock time for their operation is determined by the capability of achieving a given throughput taking into account limitations on equipment size and the convenience of obtaining the required clock frequency from the basic frequency for synchronization, and is also based on the functional composition and physical properties of the element system selected.

Speed of executive units can be increased also through their specialization. Specialization, i.e. orientation of unit functions to execution of one type of operation, allows optimal achievement (for a specific unit) of maximum possible speed with existing technology, although it should be noted (as a disadvantage) that a considerable amount of equipment is required to implement the whole set of operations. Therefore, a basic problem is the optimal selection of the number and types of units. Since specialized units are used for the highest throughput models where it is necessary to obtain the maximum possible speed while disregarding outlays for equipment, in selecting types of units, i.e. their organization, the orientation is on speeding up execution of operations which are processed in more than one machine cycle in the universal units. These operations include primarily multiplication, division and addition with floating-point. Based on this, the following types of executive specialized units are used in the CPU for the large YeS EVM-2 model: unit for fixed-point operations, unit for floating-point operations, unit for multiplication and division, and a unit for decimal arithmetic.

A further development of this method for increasing the speed of executive units is the organization of parallel operation of the operating units. However, for high efficiency of parallel operation, special software is required that allows distributing instruction streams with regard to specialization of the units.

The problem of optimal correlation between speed and economy for medium-size computers, as a rule, is solved by speeding up only certain operations. "Accelerators" are used for these purposes, i.e. supplementary equipment operating at an increased clock frequency and oriented to speeding up a certain section of the algorithm for execution of certain operations.

Thus, used is the YeS-1045 computer is an accelerator for executing a number of operations based on the tabular method of obtaining a result. The acclerator is implemented with high-speed microcircuits of programmable read-only memory (PPZU) [PROM] with a 256 X 4 organization. A table is stored in the PROM, and the appropriate result is generated at the PROM output as a function of the input signal. In other words, the various Boolean functions of the input set are output as a function of the program embedded in the PROM. Also undergoing acceleration are operations that have a substantial effect on throughput (multiplication, transfer, packing) and operations that, on the one hand, are conveniently implemented by using the tabular method, and on the other, do not require additional outlays for equipment. The accelerator is controlled by a special accelerator control memory with a size of 512 48-bit words, the cycle of which is half the cycle of the CPU control memory. Use of the accelerator has made it possible to increase the throughput of the CPU in the YeS-1045 computer by 15 percent on the average with a 6 percent increase in equipment.

Executive unit throughput is largely determined by the efficiency of the algorithms for execution of multiclock operations (addition, multiplication, division with floating-point, translation of codes, and a number of operations for processing variable-length fields). In this case, by efficient algorithms are meant algorithms that make optimal use of the structure of the executive units. It is obvious that these algorithms will vary for each computer class in the Unified System.

Organization of Execution of Instructions for Transfer of Control. Used in rating internal CPU speed are so-called mixtures of instructions that take into account execution time and recurrence (weight) of individual instructions in programs for the most typical classes of problems. Statistics show that CPU computation rate is most affected by only about 10 types of instructions which require raising the execution rate. These instructions include (in addition to those discussed in the preceding section) transfer of control instructions too which affect not only the rate, but to a considerable extent also the CPU structure.

In the Unified System of Computers, two methods of raising the execution rate of transfer of control instructions are widespread. The first is based on a statistical account of the probability of successful execution of an instruction for conditional transfer of control, and the second, on a statistical account of the most probable length of program cycles realized by using instructions for conditional transfer of control.

Implementation of the first method requires rapid prefe:ching of instructions from main storage, overlap processing and incorporation of additional buffers with a size of 8-16 bytes into the CPU structure. One buffer is used to store the basic sequence of instructions in executing the program. When a transfer of control instruction is encountered, an estimate of the most probable outcome is made and fetching is organized for the instructions of that branch of the program that will most probably be executed after completion of the branch instruction. These instructions are placed in the second buffer and processing of them is started even before the transfer of control instruction is executed. When the estimate is correct, the high rate of instruction processing is maintained; in the opposite case, fetching of the other branch is not required since it is stored in the first buffer and can be processed right after completion of execution of the transfer of control instruction.

In the fastest computers, the CPU structure has a third buffer for preselected instructions. It is used to store instructions selected as a result of processing of the next transfer of control instruction in the most probable branch of the program before completion of execution of the preceding one.

Implementation of the second method for speeding up execution of conditional trans fer of control instructions is organized by a buffer for processed instructions, in which usually from 64 to 128 doublewords of instructions are kept, i.e. in executing the basic sequence of instructions in a program, the processed instructions are not erased, but stored in a the special buffer. In this case, a special algorithm has to be provided to replace processed instructions of old information in the buffer by new. The main idea of the second method is this: Since conditional transfer of control instructions are usually placed at the end of a cyclic section of a program when programs are written, in the majority of cases, this conditional transfer will point to a section of the program already selected. This is precisely the section of the program with high probability that is stored in the buffer of processed instructions, which makes it possible to reduce the time for preparing instructions of the new branch for execution.

Shown in fig. 3 are the structural schemes for paths of instructions in speeding up execution of transfer of control instructions by the method of estimating the probability of a transfer (a) and the method of probable length of a cycle (b).

Both of these methods require additional equipment outlays for implementation and result in considerable complication of the control algorithms in the CPU; however, they are used in the large YeS EVM-2 models because they produce a considerable increase in the CPU speed, while reducing execution time by a factor of two to three for transfer of control instructions.

Organization of CPU Internal Storage. CPU internal storage consists of a large number of resiters and buffer storage for temporary storing of intermediate results obtained during execution of operations, storage of data files during exchange,

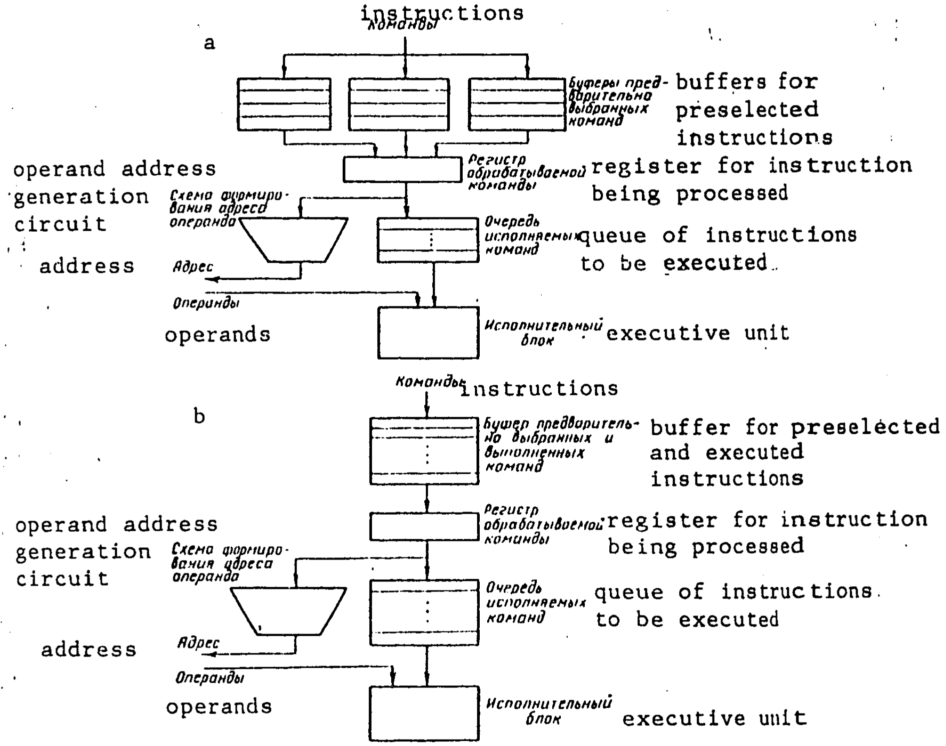

instructions
Команды
a
Буферы предварительно выбранных команд
buffers for preselected instructions
Регистр обрабатываемой команды
register for instruction being processed
operand address generation circuit
Схема формирования адреса операнда
Очередь исполняемых команд
queue of instructions to be executed
address
Адрес
Операнды
operands
Исполнительный блок
executive unit
Команды
instructions
b
Буфер предварительно выбранных и выполненных команд
buffer for preselected and executed instructions
Регистр обрабатываемой команды
register for instruction being processed
operand address generation circuit
Схема формирования адреса операнда
Очередь исполняемых команд
queue of instructions to be executed
address
Адрес
Операнды
operands
Исполнительный блок
executive unit

Fig. 3. Scheme for execution of transfer of control instructions by the method of estimating transfer probability (a) and the method of estimating loop length (b)

storage of control attributes and information on computer status during program execution and information that controls the sequence of instruction execution. CPU computational efficiency is largely determined by the structure and speed of internal storage, since certain of its components directly determine or affect to a considerable extent the CPU operating cycle.

A major characteristic of a CPU and computer is the CPU and IO channel data exchange rate with main storage. As noted earlier, selection of the optimal exchange path width and use of interleaved memory make it possible to increase the data exchange rate; however, a substantial factor in increasing this rate is the reduction of fetch time and cycle time of main storage. CPU buffer storage and channel buffer storage are used in YeS EVM-2 CPU's to reduce the effective cycle of main storage.

Main buffer storage is an intermediate block of storage (between the basic main storage and the CPU) that operates with the CPU cycle and provides for storing instruction and operand files most frequently used by the processor. Buffer storage usually holds from 8 to 64 bytes, which permits a reduction in main storage access time. Buffer storage is filled as data is fetched from main storage during program execution. Since programs have a limited amount of instructions and operands and usually operate in the cyclic mode for processing a data file, after execution of the first cycle, about 90-95 percent of needed information will be in buffer storage. Consequently, during subsequent operation of the program, instructions and operands will be fetched not from main, but from buffer storage. In this case, the effective main storage cycle will be close to the buffer storage cycle. Algorithms for replacing information in buffer storage provide for holding the most frequently used data files. Exchange between main and buffer storage occurs in blocks containing from 16 to 64 bytes, which permits making effective use of interleaved main storage.

Processor buffer storage may be used for operation of IO channels, but this leads to the emergence of conflict situations during a simultaneous call from the channels and the processor and, consequently, to a reduction in the efficiency of the operation of buffer storage. Therefore, to raise the rate of data exchange between channels and main storage, used more often is special buffer storage for channels, which is broken down into groups of registers allocated for each IO channel. As a rule, four to eight double words of information each are stored in these registers. With this organization of exchange, the capability emerges of making efficient use of interleaved storage and reducing data exchange time, because the channel effects its exchange not with main storage, but with the buffer storage for the channel. Information is exchanged between main storage and channel buffer storage also in blocks of four to eight double words each.

The introduction of buffer storage units, in addition to reducing the effective cycle of main storage, also leads to a reduction in the number of conflicts during a simultaneous call for storage from channels and the processor. Both the former and the latter are especially important for high-throughput computers, which has dictated the use of buffer memories in the YeS EVM-2 medium and large models. Use of these memories in small models is inexpedient because of the relatively high cost of the additional equipment needed to implement them and the considerable complication of the organization of the processor control structure.

The method of storing intermediate results of computations also largely determines processor structure. In essence, in the Unified System of Computers, this method is determined by the instruction set, form of data representation and addressing structure. In the general case, there are three ways of organizing storage of intermediate results obtained during performance of calculations.

The first is based on sending any results to main storage immediately after completion of the calculations without intermediate storage in the processor. This is one of the most uneconomical methods since repeated access to main storage is required in the process of executing one instruction.

The second method calls for using an intermediate register through which the results are transferred to main storage. Here the number of transfers is reduced, since the information in the register can be used during execution of the operations. But in this case too, rather frequent access to main storage is required to free the register.

The third method uses several registers that form a small amount of storage, and the results of the various groups of operations are kept in their own registers. In this case, the number of registers or storage size is selected in such a way that as a rule, only final results are stored in main storage.

In essence, all three methods are used in Unified System processors, but the third one is the main one. For this purpose, the processor structure has 16 general-purpose registers and four registers for storing results of floating-point operations (see chapter 1). The general-purpose registers store results of fixed-point operations and are also used as index registers in modifying an address and executing instructions associated with addressing, which eliminates the need for additional registers for these purposes. Besides this, a program status word [PSW] register and a group of control registers are used to store control features and information on the current status of the system during program execution.

Direct writing of results to main storage is provided for only for "storage-to-storage" instruction format, during execution of which variable-length operands are processed. This is because storage of intermediate results would require a buffer too large in size and in addition, these instructions are not decisive for computer speed.

For writing information to main storage in large models, in addition to the methods discussed, individual registers are used that are intended for coordinating operation of the CPU and main storage with store instructions.

The microprogram method of control is used to one or another extent in all YeS EVM-2 models. Organization of the control storage for microprograms is a major feature determining the efficiency of CPU operation. In each specific case, it is selected on the basis of the computer purpose, throughput and structure. From a physical point of view, control storage for microprograms comes in two varieties: read-only storage [ROS] and writeable control storage [WCS]. ROS is used only for reading of information and, as a rule, is faster and simpler for control than WCS. At the same time, WCS offers additional capabilities for raising CPU efficiency through continuous improvement of algorithms for executing operations.

Joint Use of Equipment by CPU and Channels. To reduce the total amount of equipment in some models in the Unified System, the principle of so-called integrated channels has been implemented. It is based on the fact that processing of instructions in the CPU during program execution and processing of IO operations in a channel are largely similar and require the same type of both executive and control assemblies and units. Based on this and on the fact that information exchange between a channel and an external device takes considerably more time than that for data exchange between a channel and main storage, it is possible to use a part of the CPU equipment to perform channel functions. This permits a substantial reduction in channel equipment and designing it into the CPU. The common equipment is used primarily by the CPU. After completion by a channel of an exchange of a routine portion of data with an external device, CPU operation is halted temporarily and the necessary operations for a channel exchange with main storage are performed.

In each specific case, depending on the organization of the computational process, the channels can use various equipment of the CPU, but, as a rule, this equipment includes the control storage for the microprograms and the storage control unit (exchange path). Microprogram control provides for data exchange between the channels and main storage and organization of processing of control information, and the microprogram storage unit is used by the channels and CPU on a time-sharing principle. In addition, in many cases, the CPU ALU is designated for processing control information for a channel. Joint use of equipment is provided for by using hardware-controlled priority circuits that permit transfer of control to a specific unit only at certain points of the microprogram sequence of the CPU and channels. Since channel operation can be interrupted upon CPU request, each channel must be provided with its own address register for storing the current address when control storage is transferred to the CPU. After transfer of microprogram control to a channel, it will continue its operation at the address stored in the register.

In the general case, from the viewpoint of equipment used, channel functions are divided as follows: data is exchanged with IO peripherals under control of channel hardware in parallel with CPU operation; data exchange between channels and main storage, as well as processing of control information is performed by CPU facilities under control of microprograms and in this case the CPU is not performing its own operations.

The principle of integrated channels is used only in the small and sometimes in medium-size YeS EVM-2 models, since use of this principle in large models is not warranted because of limitations that it imposes on CPU speed.

2.2. System Control Facilities

CPU system control facilities provide for the set of necessary actions for monitoring system states, specifying modes of operation, protecting programs from destruction, expeditious linking of hardware facilities together and synchronization of their operation in time, external intervention, etc. Some of these facilities are well known from their use in YeS EVM-1 (facilities of direct control, external interrupts, storage protection and initial program loading). A number of new facilities have been introduced in the YeS EVM-2 that raise control capabilities to an even greater extent:

facilities of extended control and control registers;

facilities for expansion of the system of interrupts;

facilities for program event recording;

facilities providing for monitor programs; and

facilities for time readout.

Extended Control Mode. The PSW defines the status of the computer and controls operation of hardware and software.

In the YeS EVM-2, there are two control modes that determine the function of the PSW fields and bits and use of permanently assigned locations in main storage.

Operation of computer hardware while maintaining full program compatibility with the YeS EVM-1 is defined as the basic control mode (BC mode).

Operation of the new hardware and software under which additional functions are performed and greater capabilities emerge is defined as the extended control mode (EC mode).

The modes are specified by the value in PSW bit 12: 0 for the BC mode or 1 for the EC mode.

In the EC mode, the location of certain control fields in the PSW is changed, some PSW fields have been removed and additional ones introduced. In particular, the interruption code and the instruction-length code have been assigned permanent main storage locations, masks for interruptions have been expanded and placed in the control registers, and additional control fields have been introduced in the PSW: the program event recording mask and the translation mode.

Table 1 shows the allocation of PSW fields that fully determine the state of the computer hardware at a given time.

For the EC mode, PSW bit 1 has been allocated for the program event recording mask, and bit 5 defines the translation mode. Unused PSW bits in the EC mode must contain zeros.

Table 1. Allocation of PSW Bits

	PSW Bits	
Function of PSW Fields	BC	EC
Channel masks 0-5	0-5	*
IO mask	6	6
External interrupt mask	7	7
Protection key	8-11	8-11
Control mode	12	12
Machine-check mask	13	13
Wait state	14	14
Problem state	15	15
Interruption code	16-31	**
Instruction-length code	32-33	**
Condition code	34-35	18-19
Program mask	36-39	20-23
Instruction address	40-63	40-63

* Channel masks are stored in control register 2.

** Interruption and instruction-length codes are stored in permanently assigned locations in main storage.

Control Registers. Sixteen 32-bit control registers have been provided for storing additional control information that expands PSW information in the CPU.

Control information is loaded into the registers and their contents stored in main storage by execution of the instructions LOAD CONTROL and STORE CONTROL.

Control registers support CPU operation in the EC mode, hold the expanded masks for external interruptions and IO channel interruptions, control information for the facilities of virtual organization of the system and other new facilities in the Unified System.

Control register 0 includes the fields for control of the block multiplex mode for the IO channels (bit 0) and control of suppression of set system mask [SSM] (bit 1). Facilities for organization of multiprocessor systems use the clock synchronization control field--bit 2 of control register 0, and masks for malfunction alert, emergency signal, external call and clock synchronization check--bits 2, 16-19 of control register 0. Dynamic address tranaslation facilities use a control field for page and segment size control--bits 8, 9 and 11 of control register 0, as well as fields for segment table length and segment table address--bits 0-7 and 8-25 of control register 1.

Channel masks which determine the CPU accessibility for IO interruptions in the BC mode are stored directly in the PSW (bits 0-5). In the EC mode, these masks are located in control register 2 (bits 0-31). Control register 8 contains the monitor masks in bits 16-31.

Program event recording facilities contain individual event masks in bits 0-3 and 16-31 of control register 9; the starting and ending addresses of the main storage area monitored by the recording are stored in bits 8-31 of control registers 10 and 11.

Machine error processing facilities and recovery facilities use control registers 14 and 15 for their purposes.

Unused bits of control registers and unused registers must contain zeros.

Interrupt System. As a function of the interrupt source and cause, there are six classes of interrupts: supervisor call, program, external, IO, restart and machine check.

During execution of the SUPERVISOR CALL instruction, an interrupt signal is generated whose main purpose is the switching of the CPU from the problem to the supervisor state. Program interrupts occur when an instruction is executed incorrectly, and when operands and computer devices are used incorrectly. External interrupts provide for CPU response to signals from time readout facilities, interrupt signals from the operator's console and signals from six external sources. Requests for an IO interrupt come from a channel after completion of an IO operation in a channel or external device control unit, as well as after CPU intervention when certain situations occur in the IO system. The restart interruption is initiated by activating the restart key on the system console.

A machine check interruption results from a system malfunction and is calssified as either hard or soft. A hard interrupt is caused by a nonrecoverable machine error. A soft interrupt is caused by a recoverable error.

Each class of interrupts has been allocated two fixed locations in main storage. Stored in the first location is the current PSW transferred from the PSW register at the instant of interruption. The double word stored in the first location is called the old PSW. It is stored again in the PSW register after execution of the interrupt subroutine. Stored in the second location is the new PSW that after storage of the current status in the first location is transferred to the PSW register for initialization of the interrupt subroutine. Bits 16-31 of the old PSW contain an interruption code that specifies the cause or source of the interruption.

With the emergence of the new hardware facilities (dynamic address translation, program event recording, monitor, multiprocessor systems and new time readout), it was necessary to introduce new types of interruptsto support the interaction of this hardware with the software system. This in turn required the development of control of interrupts through inclusion of masks for the new hardware facilities in the control registers.

Expansion concerned the class of program interrupts and the class of external interrupts. Program interrupts were introduced in operation of dynamic address translation facilities associated with use of a segment (interrupt code 10) and a page (interrupt code 11), as well as with specification during translation (code 12). Interrupts supporting operation of the monitor and program event recording also pertain to the program class and have codes 40 and 80, respectively.

Signals of external interrupts were introduced that are associated with operation of a multiprocessor system (malfunction alert, emergency signal and external call: interrupt codes 1200, 1201 and 1202, respectively), as well as with operation of the new time readout facilities (time-of-day clock synchronization check, clock comparator and CPU timer: codes 1003, 1004 and 1005).

In addition to expansion of the interrupt system, with the introduction of the new hardware facilities the need arose too for storing additional information during the interruption. This in turn required new fields in permanently allocated main storage that were intropduced as well for machine check handling facilities and recovery facilities.

Table 2 gives the location of the new fields in permanently allocated storage.

Monitoring Facilities. Monitor facilities were introduced for basically two reasons. First, raising the capabilities of the multiprogramming and time sharing modes required development of facilities that would allow selective storage of information at a certain time during program execution. Second, there was a requirement for statistics that permit monitoring the course of execution of programs and analyzing the efficiency of CPU operation. These facilities can be used to track which programs were executed and at what times, and also how often they were used.

Access to the monitor program that implements execution of the necessary functions is effected by using interrupts. For these purposes, the special instruction

Table 2. Assigned Main Storage Locations

Address hexa- decimal	decimal	Function of field
0	0	restart new PSW
8	8	restart old PSW
18	24	old PSW for external interrupts
20	32	old PSW for supervisor call interrupts
28	40	old PSW for program interrupts
30	48	old PSW for machine check interrupts
38	56	old PSW for IO interrupts
40	64	channel status word
48	72	channel address word
50	80	interval timer
58	88	new PSW for external interrupts
60	96	new PSW for supervisor call interrupts
68	104	new PSW for program interrupts
70	112	new PSW for machine check interrupts
78	120	new PSW for IO interrupts
84	132	processor address during external interrupt in EC mode
86	134	external interruption code in EC mode
88	136	instruction-length code and supervisor call interrupt code in EC mode
8C	140	instruction-length code and program interrupt code in EC mode
90	144	translation exception address during program interrupt in EC mode
94	148	monitor class number during program interruption due to a monitor event
96	150	program event recording code during program event interrupt
98	152	address of instruction that caused interrupt for program event
9C	156	monitor code during monitor event interrupt
A8	168	channle ID during execution of STORE CHANNEL ID instruction
AC	172	IO extended logout address
B0	176	limited channel logout information
B8	184	IO address during an IO interruption in the EC mode
D8	216	machine check interruption code and expanded information on the machine check interruption

MONITOR CALL has been introduced that is placed at certain points within the program being executed. As soon as program processing reaches this point, the MONITOR CALL instruction is fetched and a program interrupt occurs that is serviced by the monitor subroutine.

The MONITOR CALL instruction has the SI format, i.e. the operand is placed directly in field I_2 (bits 8-15). In this case, bits 12-15 specify one of the 16 possible monitor classes, and bits 16-31 (fields B_1 and D_1) the monitor code. In essence, the monitor code performs the role of an interrupt code identifying the function that must be performed. Within the bounds of each monitoring class, 24-bit addressing can be used that is defined by the monitor code.

Depending on circumstances, performance of a particular function defined by the monitor class may be suppressed. For this in bits 16-31 of control register 8 are stored masks corresponding in ascending order of the numbers to all 16 monitoring classes. For those monitor classes with a mask bit equal to one, an interrupt is permitted. If the mask bit is zero, no interrupt is initiated.

During an interrupt, the interrupt code (monitor call) is placed in the old PSW, and the monitor class and code are placed in permanently allocated main storage at addresses 148 and 157-159, and in the process, zeros are placed in the bytes with addresses 149 and 156.

Program Event Recording Facilities. Computer program debugging is a laborious, but necessary operation that cannot be solved by another method. Therefore, the availability in the CPU of special facilities that facilitate the process of program debugging and reduce machine time in the process is a mandatory part of modern computers. In the YeS EVM-2, these facilities include program event recording facilities. Program events to be processed are recorded by using the mechanism of program interrupts. The interrupt code identifies the program events that cause the interruption.

Program event recording facilities operate only in the EC mode and control information for this is stored in control registers 9-11. Bits 0-3 of control register 9 contain the event masks and bits 16-31 contain the general register masks. Event masks specify which events are monitored and the bits are assigned as follows:

0 -- successful branch;
1 -- instruction fetch;
2 -- storage alteration; and
3 -- general register alteration.

General register masks specify which general registers are monitored for alteration of their contents. For this, each of the 16 mask bits in ascending order corresponds to a general register number.

The starting address of the monitored main-storage area is stored in bits 8-31 of control register 10. The ending address of the monitored main-storage area is stored in bits 8-31 of control register 11. The starting and ending addresses specify a storage area for two events in a program: instruction fetching and storage alteration. When the starting address is equal to the ending address, only the location designated by that address is monitored. When the starting address is larger than the ending address, the monitored storage area consists of two zones. One zone covers the area from the starting address to the largest address in the system, and the other, from location 0 to the starting address. Thus, the maximum possible amount of main storage, defined by a 24-bit address, may be allocated for recording of these events.

During an interrupt caused by program event recording, additional information on the cause of the interrupt is placed in the permanently allocated storage area (locations 150-155). The specific event code that caused the interrupt is placed in bits 0-3 of location 150. The values of these bits correspond exactly to the values of the event mask bits in bits 0-3 of control register 9. The addresses of instructions that caused a given event in a program are stored in bits 8-31 of locations 153-155. Zeros are placed in all other bits of locations 150-155.

Time readout facilities include the time-of-day clock, CPU timer, clock comparator and the interval timer.

The time-of-day clock continuously measures elapsed time and is a 52-bit binary counter, in which information is represented in the form of an unsigned fixed-point number. The clock is incremented by adding a one in bit position 51 every microsecond following the rules for unsigned fixed-point arithmetic. A carry into bit position 0 is ignored, and counting continues from zero on.

The time-of-day clock operates in all CPU states: wait/running, problem/supervisor and stopped/operating. Its operation is not affected by any operations for CPU and system reset. Time-of-day [TOD] clock operation can be stopped only by a clock malfunction, by disconnection of power to the CPU or clock itself and when it is in the STOPPED state.

The STOPPED state of the TOD clock is set each time before its contents has to be altered by the SET CLOCK instruction, by which the current number stored in the counter is replaced by the operand specified by the instruction. Transition from the STOPPED state to a new state and vice versa is defined by bit 2 of control register 0. When this bit is one, the TOD clock remains in the STOPPED state. A new clock value is set by the SET CLOCK instruction only when this bit is zero.

The TOD clock value can be stored in main storage by the instruction STORE CLOCK. When this instruction, as well as the instruction SET CLOCK, is executed, facilities have been provided to ensure synchronization of clocks when there is more than one in a multiprocessor system organization.

When it is necessary to cause an external interrupt at a certain TOD clock value, the clock comparator is used for these purposes. The value specified in a program is stored in the comparator by the instruction SET CLOCK COMPARATOR; this value is continuously compared to the TOD clock value. An interrupt signal is generated at the moment these values coincide. Comparator contents are stored in storage by the instruction STORE CLOCK COMPARATOR.

The CPU timer provides a means for measuring elapsed CPU time and for causing an interruption when a prespecified amount of time has elapsed. Just as the TOD clock, the CPU timer is a binary counter with the same format, except that a one is not added, but subtracted from the 51st bit. In the process, bit 0 in the timer counter is used as the sign of a fixed-point number. A request for a CPU-timer interruption exists whenever the value in the CPU timer is negative (bit 0 is one).

When both the CPU timer and TOD clock are running, the stepping rates are synchronized such that both are stepped at the same rate. In contrast to the TOD clock, the CPU timer does not change its state when the CPU is in the STOPPED state. The timer reading can be stored in storage by using the instruction STORE CPU TIMER. The instruction SET CPU TIMER is used to change the value of the timer.

The interval timer, in association with a program, can serve both as a real-time clock and as an interval timer. It is in location 80 of main storage. The 32-bit number is treated as a signed fixed-point number. An interrupt occurs when this number becomes negative (bit 0 is one). Interval timer contents are reduced by one in bit position 23 at 300 cycles per second between the execution of instructions.

2.3. Dynamic Address Translation

The multiprogram mode has become the main mode of computer operation. Efficient use of this mode presupposes the availability of a large amount of main storage as well as hardware and software facilities for dynamic storage allocation in the process of program processing.

The largest main storage size for the Unified System of Electronic Computers is limited to the address capacity adopted and may be 16M bytes. However, development of main storage of this size with the required time parameters presents considerable technical difficulty. In addition, the size of system, standard and control programs far exceeds the size of real main storage. Based on this, only active sections of system programs are in main storage, which restricts the capability of processing them efficiently. The availability of a large amount of external storage and programming in symbolic addresses with use of the virtual principle makes it possible to get around this limitation. The programmer appears to have the maximum permissible amount of main storage and in the process, storage is reallocated for programs dynamically withput programmer participation.

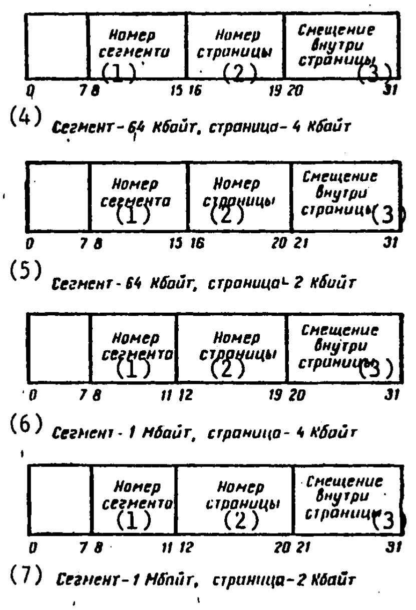
In the YeS EVM-1, used for dynamic storage allocation was the method of base registers, in which a real address of main storage was formed as the sum of a symbolic address and the base. Base addresses for the various system programs are stored in register local storage of the CPU.

Dynamic storage allocation using base registers does not have sufficient flexibility since the system program has to be brought into main storage completely, even if this is not necessary. In addition, any input of a new program requires physical reallocation of storage size, which is time-consuming.

The dynamic storage allocation method with page and segment organization is used in the YeS EVM-2. This method assumes subdividing the entire extent of virtual storage into blocks called segments and pages. Symbolic (logical) addresses are translated into real ones by special translation tables. Dynamic address translation is possible only when operating in the EC mode.

Logical Address Structure. Segments and pages are used as movable blocks of data in the dynamic address translation mode. A segment may be 64K or 1M byte; a page may be 2K or 4K bytes. Sizes of a segment and a page are controlled by the values of bits 11,12 and 8,9 respectively of control register 0. Data in each block are addressed by sequential logical addresses. A logical address consists of a page index (number) field, a segment index (number) field and a byte index field (displacement within a page). Fig. 4 shows the formats of the logical address for the different segment and page sizes.

To translate the logical into real addresses, translation tables are used for segments and pages. These tables reside in main storage and determine the current allocation of storage actually installed. The address and length of the segment table are defined by the appropriate bits of control register 1. The entry fetched from the segment table designates the length, availability and origin of the corresponding page table. The entry fetched from the page table indicates the availability of the page and contains the high-order bits of the real address. A zero value in the availability bits contained in the entries of the segment and page tables indicates the given entry is available for use.


Номер сегмента (1)
Номер страницы (2)
Смещение внутри страницы (3)
0 7 8 15 16 19 20 31
(4) Сегмент-64 Кбайт, страница-4 Кбайт
Номер сегмента (1)
Номер страницы (2)
Смещение внутри страницы (3)
0 7 8 15 16 20 21 31
(5) Сегмент-64 Кбайт, страница-2 Кбайт
Номер сегмента (1)
Номер страницы (2)
Смещение внутри страницы (3)
0 7 8 11 12 19 20 31
(6) Сегмент-1 Мбайт, страница-4 Кбайт
Номер сегмента (1)
Номер страницы (2)
Смещение внутри страницы (3)
0 7 8 11 12 20 21 31
(7) Сегмент-1 Мбайт, страница-2 Кбайт

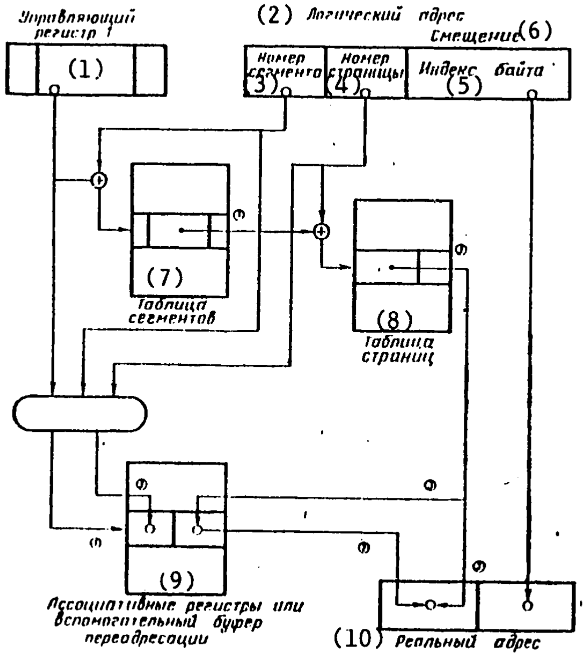


Fig. 4. Formats of the logical address

Key:
1. segment index
2. page index
3. byte index
4. for 64K-byte segments and 4K-byte pages
5. for 64K-byte segments and 2K-byte pages
6. for 1M-byte segments and 4K-byte pages
7. for 1M-byte segments and 2K-byte pages

Fig. 5. Diagram of dynamic address translation process

Key:
1. control register 1
2. logical address
3. segment index
4. page index
5. byte index
6. displacement
7. segment table
8. page table
9. translation-lookaside buffer
10. real address

Translation. The translation process (fig. 5) is as follows: The segment-index portion of the logical address is used to select an entry from the segment table, the starting address and length of which are specified by the contents of control register 1. This entry designates the page table to be used. The page-index portion of the logical address is used to select an entry from the page table. This entry contains the high-order bits of the real address. The byte-index field of the logical address is used for the low-order bit positions of the real address.

A translation buffer is used to speed up the translation process. Capacity and sizes of the translation buffer are model-dependent parameters. The translation buffer includes the high-order portion of the logical addresses and their corresponding real addresses, by which the CPU accesses main storage. Thus, considering that the CPU accesses main storage usually by sequential addresses and with a high probability within the bounds of one page, table entries are fetched from main storage only once. The information obtained in the first reference subsequently remains in the buffer, and all subsequent references to storage that use translation table entries from the same area of storage are performed by using the buffer. Buffer size ranges from 8 words for the YeS-1035 to 128 words for the YeS-1060 model.

There are certain conditions under which information may be placed in the buffer and used for dynamic address translation. The concepts of a valid, attached and active entry are introduced. An entry is valid when the segment or page valid bit in this entry is zero. A segment table entry is attached when the dynamic address translation mode is specified, the entry is within the segment table designated by control register 1, and it is designated by a logical address with regard to segment size. A page-table entry is attached when it is within the page table designated by the page table address and page table length in an attached and valid segment table entry. An entry is active when it may remain recorded in the translation buffer. An entry may be placed in the buffer when it is valid and attached. Information on the state of translation table entries and their use is given in table 3.

Table 3. Use of Translation Tables

State of table entry:			Can copy of entry be in buffer?	Can table entry be fetched for translation?	Can table entry be used for translation?	Can buffer copy be fetched for translation?
active	attached	valid				
yes	yes	yes	yes	yes	yes	yes
yes	yes	no	yes	yes	no	yes
yes	no	yes	yes	no	no	no
yes	no	no	yes	no	no	no
no	yes	no	no	yes	no	no
no	no	yes	no	no	no	no
no	no	no	no	no	no	no

For efficient processing of the algorithm for page replacement in main storage, two types of recording are used: recording of references to a main storage block location during storing or fetching of data, and recording of changes which reflects information on which pages in main storage had data stored in them.

In the dynamic address translation mode, the storage protection key code is extended with two additional bits. The reference bit is set to one each time a location in the corresponding storage block is referred to either for storing or fetching of information. The change bit is set to one each time information is stored in the corresponding storage block.

Reference and change recording takes place for both CPU and IO channel accesses for 2048-byte blocks and does not depend onpage size invoked.

2.4. Microprogram Control

Selection of the CPU control method is governed by the CPU operating cycle, complexity of processing and control algorithms, and control method efficiency. Until recently, the microprogram method of control was not widespread in high-throughput machines. This was because this method had a number of shortcomings, the most important of which were reduction in machine throughput and the high cost of microprogram development. Therefore, as a rule, the hardware method of control that permitted achieving a given level of speed was used in high-speed machines, including the large models of the YeS EVM-1.

The situation has changed sharply in recent years. Microprogram control began to be used extensively in designing high-throughput computers. This interest in microprogramming was caused by the development of the technology of the micro-element base, the expansion of the computer instruction set and the considerably greater capabilities offered by this method compared to the hardware.

The emergence of large-scale integrated semiconductor circuits with a low level of delay made it possible to develop control storage, the parameters of which made it possible to substantially reduce the effect of microprogramming on computer throughput.

A study of CPU control methods revealed the dependence of their efficiency on the size of the instruction set used in operations. An evaluation of the composition of the instructions in the YeS EVM shows that the microprogram method is more efficient. It is evident that CPU effectiveness will grow with extensions of the composition of instructions and functions of the CPU. However, in building control circuits for the individual units in the CPU, a large role begins to be played by the factor of speed of the control circuit and complexity of algorithms. Indeed, performed for high-throughput computers in one CPU cycle simultaneously are resolving the priority of the many requests from CPU units and channels, checking the key for storage protection and accessing independent blocks of main storage, translating logical addresses, fetching from buffer storage and other operations.

Complex algorithms, on the one hand, can be more economically implemented with microprogram control, but on the other hand, these algorithms have a large number of branching conditions, which complicates the microinstruction addressing scheme and in the final analysis both the cycle itself and the number of cycles in the algorithm are increased. In this case, a compromise is required, which leads to a mixed hardware-microprogram method of control.

It is most efficient to use microprogram control in CPU executive units, in which are implemented multicycle algorithms having a large number of linear sections and a limited number of branching conditions. Based on this, small models have primarily microprogram control in all CPU units, while CPU's in the large models use it mainly in the executive units.

The advantages of the microprogram method of control compared to the hardware from the point of view of raising CPU efficiency and organization of maintenance are expressed in the following:

with microprogram implementation, the structure of the control circuit is more regular, hardware facilities are used more economically, and the level of unification of individual assemblies is raised, which leads to simplification and reduction in cost of equipment;

microprogram control permits developing a system of microdiagnostic tests to automatically search for malfunctions; and

the control algorithms are documented more simply and clearly, which supports simplicity in training service personnel and facility in CPU operation.

The structure of the microprogram control unit (MPU) is governed by three factors: the principle of construction of control storage, the structure of the microinstruction and the method of generating the address of the following microinstruction.

Control storage for a microprogram control unit comes in two types: read-only and writable. Read-only storage is implemented on the basis of modules or integrated circuits of semiconductor programmable storage (IS PPZU). Special programming devices are used to program the storage when the CPU is manufactured. The information in the programmable read-only memory is preserved under all conditions of CPU operation, even when power is disconnected.

Writable storage is implemented on the basis of integrated circuits of high-speed storage (OZU). The user loads information into this storage unit from external media each time power is switched on to the CPU. In the YeS EVM, cassette recorders are used to hold files of microprograms. Microprograms are replaced or changed by replacing the cassettes which are prepared at the computer manufacturing plant.

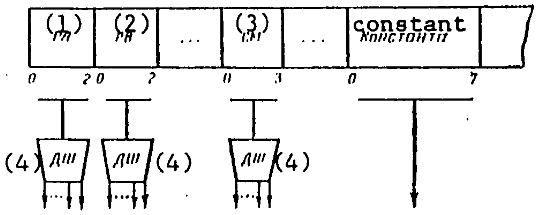
All YeS EVM-2 models are oriented to using writable storage of microprograms, since it has invaluable advantages in optimizing algorithms for operations and developing a dynamic system of microprogramming.

Microprogramming with the use of writable storage of microprograms has important advantages in debugging prototypes and makes possible raising the efficiency of machines in series production and operating at a using installation thanks to the introduction of new software that makes use of the extended set of instructions and new CPU functions. The latter is very important in selecting the method for implementation of the control circuits. This is in connection with the observed trend of reducing time spent on the work of the operating system by "integrating" some frequently used subroutines and operating system modules into the hardware. Such integration can be performed only when the CPU functional capabilities are extended, i.e. when the control circuit algorithms are updated, which is easily done by changing the contents of control storage.

The main characteristics of control storage are fetch time, word width and number of words. Memories with a capacity to 4K words, width to 144 bits and fetch time to 60 ns are used in the Unified System of Electronic Computers.

The width of a word of control storage is governed by the type of coding of the sets of microoperations and by the method of generating the address of the next microinstruction.

Functional as well as instruction-oriented coding is used in coding the microoperations. Functional or field coding presupposes the presence in the operation part of individual fields that contain sets of compatible microorders that control the various functional parts of the processor that permit parallel operation of them. Usually in the composition of the fields there is a field that includes the immediate operand or instruction that can be used for setting certain registers. Fig. 6 shows an example of such a microinstruction. Control signals are generated by decoding of the fields. Field size is from 1 to 4-5 bits which permits coding from 1 to 32 microorders.



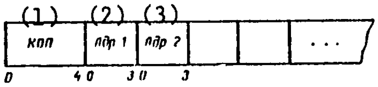


Fig. 6. Microinstruction with functional coding

Key:
1. RA -- field that controls reception into register A
2. RB -- field that controls reception into register B
3. SM -- adder function control field
4. DSh -- decoder

Fig. 7. Microinstruction with instruction-oriented coding

Key:
1. KOP -- operation code
2. Adr 1 -- address 1
3. Adr 2 -- address 2

With instruction-oriented coding, the microinstruction, just as a nominal instruction, contains the operation code field, fields in which addresses of registers or processor functional assemblies are specified, and fields containing additional control information (fig. 7).

A comparison of these two methods shows that outlays for equipment and time for generating control signals with instruction-oriented coding are greater than that with functional coding, although the width of the microinstruction is smaller.

Instruction-oriented microprogramming lends itself more easily to automation of design of microprograms. With this method, the microprogram can be written by a programmer who has a formal description of the microinstructions. Although functional coding is considerably more complex and requires detailed knowledge of all processor assemblies for writing the microprograms, in this case, more efficient microprograms are obtained in terms of time of execution and number of microinstructions. Functional coding is intended primarily for the large models in the Unified System.

Two methods of addressing, natural and compulsory, are used to generate the address of the next microinstruction.

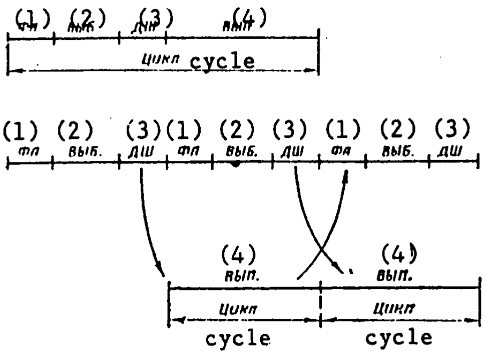


Fig. 8. Time chart of operation of control storage

Key:

1. FA -- microinstruction address generation
2. VYB - fetch from control storage
3. DSh - decoding of microorders
4. VYP - execution of operations

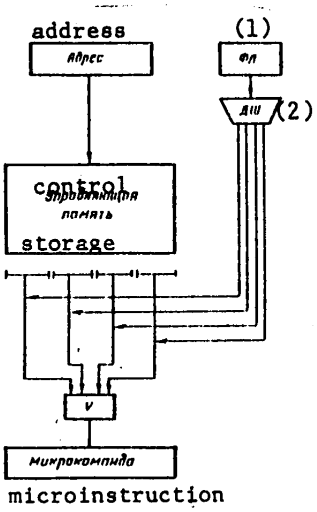


Fig. 9. Diagram of microprogram control of processor

Key:

1. FA - microinstruction address generation
2. DSh - decoding of microorders

The compulsory addressing method presupposes that each microinstruction contains the base address of the next microinstruction to be executed and the fields that define the conditions that affect a change of the base address. Thus, all conditions for microprogram transfer are specified in the microinstruction.

The natural addressing method presupposes that after execution of the microinstruction with address A, the microinstruction with address A + 1 will be executed, which eliminates the need for microinstruction addressing fields within the microinstruction. But used in the process in the microprogram in addition to operation microinstructions are those of the control type that contain only fields for effecting microprogram transfer. This causes complication of the microinstruction decoding circuit and extension of the microprogram, although the length of a microinstruction is shortened. Since this method of microprogramming is similar to designing ordinary programs, natural addressing is often used with instruction-oriented coding.

Various methods are employed to raise the speed of operation of a microprogram control unit. The unit operating cycle consists of four phases: microinstruction address generation (FA), fetch from control storage (VYB), decoding of microorders (DSh) and execution of operations (VYP) (fig. 8). Usually, the operation execution phase coincides with the address generation, fetch and decoding phases.

Thus, with proper selection of relations of times of operation of executive units and microprogram control, the processor cycle can be shortened and is governed usually by the operation of the control circuits. The cycle can be shortened further by reducing address generation time or eliminating it from the operating cycle.

For example, if the branching in the microprogram can be in no more than four directions, then a method is used in which the four words are fetched at once from control storage, starting at the word specified by the base address, and the final fetch of one of the four words is effected according to the transfer conditions generated simultaneously with the fetch of the microinstruction (fig. 9).

The time for the microorder decoding phase can be reduced by using for critical purposes control fields with direct coding, when each bit of a field is directly a control signal.

2.5. Principles of Organization of Array Processor

An array processor (MP) is a supplemental, specialized porocessor connected to a main computer instead of one of the IO channels or directly as an operating resource. Standard YeS OS channel programs are used in the first case to organize communication with an array processor; the second case requires a special supplement to the YeS EVM software.

The main operations executable in an array processor include correlation, convolution, vector, scalar and matrix multiplication, translation of fixed-point to floating-point format and the functions of indexing, counting, fetching and storing of input and output data. Operations are performed in the arithmetic unit of the array processor. Two buffers with a capacity of 32 words each are used to match the rates of operation of computer main storage and the arithmetic unit in the array processor during iterative operations. These buffers are used to hold both input data and the result. The basic arithmetic function executable in the array processor is the function UX+Y with data having the short floating-point format. Fixed-point numbers the length of a halfword may also be used as input data in the array processor; prior to processing, they are translated to floating-point format in the arithmetic unit.

As mentioned earlier, the array processor is considered as one of the selector channels, connectable to the main processor and main storage in the computer. Such organization permits simultaneous data processing in the main and array processors.

Four IO instructions, SIO, TIO, HIO and TCH, can initialize operation of the array processor. Each instruction contains a channel address and an external device address for identification of the array processor. In the process, the channel address (bits 16-23 of the instruction) must contain the code 3 and the external device address (bits 24-31) is arbitrary. The external device address code 01 (hexadecimal) is intended for initializing special diagnostic operations. Upon completion of execution of an IO instruction, one of four values of a condition code (00, 01, 10, 11) is issued to the processor. The array processor priority for access to main storage is higher than that of the processor and lower than that of a channel. Table 4 gives the condition code values and state of the array processor in the various modes when IO instructions are executed.

Just as the IO channels, the array processor fetches from main storage the channel address word (ASK), then the channel command word (USK) and generates the channel status word. Control information for data (USO--operand control word) is fetched for executing matrix operations in the array processor. Operand control word format is shown in fig. 10.

Table 4. Condition codes and array processor states

array processor Состояние МП state	Команды Instructions							
	SIO		TIO		HIO		TCH	
	(1) действие	(2) код условия	(1) действие	(2) код условия	(1) действие	(2) код условия	(1) действие	(2) код условия
available Доступное	Нет(3) ошибок Инициализация операции	00	Нет(4) действий	00	(5) Запоминание	01	(4) Нет действий	00
	(6) Ошибка запоминания	01						
executing Выполнение операции operation	(4) Нет действий	10	(4) Нет действий	10	(7) Останов	10	(4) Нет действий	10
discon- Отключен nected	(4) Нет действий	11	(4) Нет действий	11	(4) Нет действий	11	(4) Нет действий	11

Key:

1. operation
2. condition code
3. no errors; initialization of operation
4. no operations
5. store
6. store error
7. halt

Формат байта format of byte	Адрес первого элемента address of first element
0 7	8 31
Индекс index	count Счет
32 47	48 63

Fig. 10. Format of operand control word

Specification of bits 0-7 of the byte format field is shown in table 5.

Table 5

		State		Used for control word of operand		
Bits	Function	0	1	Y	X	U
0	data format	floating-point	fixed-point	+	+	+
1	used only for fixed-point data		data in complement code	+	+	+
2	data quantity			+	+	+
3	operation	algebraic addition	absolute subtraction	+	+	+
4	stack control	without stack	with stack	+	-	-
5	not used					
6	not used					
7	not used					

Bits 8-31 contain the address of the first element in the matrix. This address must correspond to the data type, i.e. integral boundaries of storage data (word or halfword).

Bits 32-47 include a halfword with fixed point and are used as the index for addressing the current byte in main storage. Both positive and negative index values are possible. Bits 48-63 contain the quantity defining the number of operands in the matrix, considered as a 16-bit positive number.

Three types of operations may be executed in the array processor: vector, scalar and matrix. Examples of executable operations are given below.

Vector move operation with translation into floating-point format (VMC). This operation may be shown by the following expression:

$Y_i \longleftarrow X_i$ for i = 1, 2, ..., n, where n = min (CTY, CTX) and $Y_i = 0$ for i = n + 1, ..., CTY, if CTY $>$ CTX; here and from now on the symbol $\longleftarrow$ denotes putting one variable into the place of another; CTY, CTX and CTU is the value of the count field in the operand control words for Y, X and U, respectively; min (a, b) and max (a, b) is the minimum and maximum values of quantities a and b; and $\{Y\}$ is the supplementary operand.

Thus, vector operand X is put into the place of vector operand Y in main storage. If the dimension of vector X is less than the dimension of vector Y, then the components of vector Y, for which i $>$ n, are replaced by zeros.

In the process of executing the operation, operand X is translated into floating-point format.

Vector move operation with translation into fixed-point format (VFX). This operation is described by the expression:

$$S_1 = X_1 \cdot X_2 - X_1$$
$$S_2 = X_1 \cdot X_2 + S_i$$
$$Y_i \leftarrow |S_2| + U_i X_3 \text{ for } i = 1, \ldots, n, \text{ where}$$
$$n = \min(\text{CTY, CTU})$$
$$\text{and } Y_i \leftarrow 0 \text{ for } i (n + 1) \text{ to CTY, if CTY} > \text{CTU}$$

In the process of executing the operation, floating-point numbers are translated into fixed-point format. For this, bits 8-22 of the floating-point number are put into bits 1-15 of the fixed-point field, and the zero bit is reserved for the sign if bit 8 equals 1, otherwise bits 9-23 are put into bits 0-15 if bit 8 of the original number equals 0. The factors X_1, X_2 and X_3 are used to extend the precision during translation of the formats.

Element-by-element multiplication of vectors (VEM). This operation may be described by the expression:

$$Y_i \leftarrow Y_i + U_i \cdot X_i \text{ for } i = 1, 2, \ldots, n,$$
$$\text{where } n = \min(\text{CTY, CTX, CTU})$$
$$\text{and } Y_i \leftarrow |Y_i| \text{ for } i = n + 1 \text{ to CTY, if}$$
$$\text{CTY} > \min(\text{CTX, CTU})$$

In this operation, the components of vector U are multiplied by the corresponding components of vector X, and the elements of the product are added to the corresponding components of vector Y. The result is moved into the location of vector Y.

Element-by-element addition of vectors (VES). The operation is described by the expression:

$$Y_i \leftarrow U_i + X_i \text{ for } i = 1, \ldots, n, \text{ where}$$
$$n = \min(\text{CTY, CTX, CTU})$$
$$\text{and } Y_i \leftarrow U_i + 0 \text{ for } i \text{ from } (n + 1) \text{ to } \min(\text{CTY, CTU}), \text{ if}$$
$$\min(\text{CTY, CTU}) > \text{CTX}$$
$$\text{and } Y_i \leftarrow X_i \text{ for } i \text{ from } (n + 1) \text{ to } \min(\text{CTY, CTX}),$$
$$\text{if } \min(\text{CTY, CTX}) > \text{CTU}$$
$$\text{and } Y_i \leftarrow 0 \text{ for } i \text{ from } \max(\text{CTX, CTU}) + 1 \text{ to CTY, if}$$
$$\text{CTY} > \max(\text{CTX, CTU})$$

Scalar Multiplication (SMY). The operation is described by the expression:

$$Y_i \leftarrow \{Y_i\} + U \cdot X_i \text{ for } i = 1, 2, \ldots, n, \text{ where}$$
$$n = \min(\text{CTY}, \text{CTX})$$
$$\text{and } Y_i \leftarrow Y_i \text{ for } i \text{ from } (n+1) \text{ to CTY, if}$$
$$\text{CTY} > \text{CTX}$$

In this operation, the components of vector X are multiplied by the scalar U. The result is put into the location of vector Y.

Sum of Squares (SSQ). This operation is represented by the expression:

$$Y \leftarrow \{Y\} + \sum_{i=1}^{n} X_i(X_i),$$
$$\text{where } n = \text{CTX}.$$

Sum of Vector Elements (SVE). For this operation,

$$Y \leftarrow \{Y\} + \sum_{i+1}^{n} X_i,$$
$$\text{where } n = \text{CTX}.$$

Partial Matrix Multiplication (PMM). This operation may be represented by the expression:

$$Y_i \leftarrow \{Y_i\} + \sum_{j+1}^{n} X_j U_{(i-1)p+j} \text{ for } i = 1, 2, \ldots, m, \text{ where}$$
$$m = \text{CTY}, \; n = \text{CTX and}$$
$$p = \frac{\text{index of U operand control word}}{\text{number of bytes in operand U}}$$

The Y operand control word specifies a row of matrix Y. CTY defines the number of elements both in a row of matrix U and in a row of matrix Y.

The X operand control word specifies a series of matrix Y.

The U operand control word specifies the matrix U. The value of CTU is ignored since CTY defines the number of elements in a row of the matrix, and CTX the number of elements in a column of the matrix. The value of the index in the U operand control word defines the number of bytes between the sequential elements in a row of the matrix. For clarity, the structure of the data array is given below:

$$[X_1 X_2 \ldots X_n] \begin{bmatrix} U_1 U_{p+1} \ldots & U_{mp-p+1} \\ U_2 U_{p+2} \ldots & U_{mp-p+2} \\ \vdots & \vdots \\ U_n U_{p+n} & U_{mp-p+n} \end{bmatrix} - [Y_1 Y_2 \ldots Y_m]$$

The first component of the result is $Y_1 = \{Y_1\} + X_1U_1 + X_2U_2 + \ldots + X_nU_n$. The remaining components are computed similarly.

Multiple Convolution (CVM). This operation is described by the expression:

$$Y_i \leftarrow |Y_i| + \sum_{j=1}^{n} U_j - X_{i+j-1} \text{ for } i=1, 2, \ldots, m, \text{ where}$$
$$n = \mathrm{CTU} \text{ and } m \leftarrow \min(\mathrm{CTY}, \mathrm{CTX})$$
$$\text{and } Y_i \leftarrow |Y_i| \text{ for } i \text{ from } m+1 \text{ to } \mathrm{CTY}, \text{ if } \mathrm{CTY} > \mathrm{CTX}.$$

Functional Structure of the Array Processor. In one of the possible methods for functional organization, the array processor is divided into two units: the control (fig. 11) and the arithmetic (fig. 12). Each is functionally independent of the other and they are controlled by two synchronous microprograms.

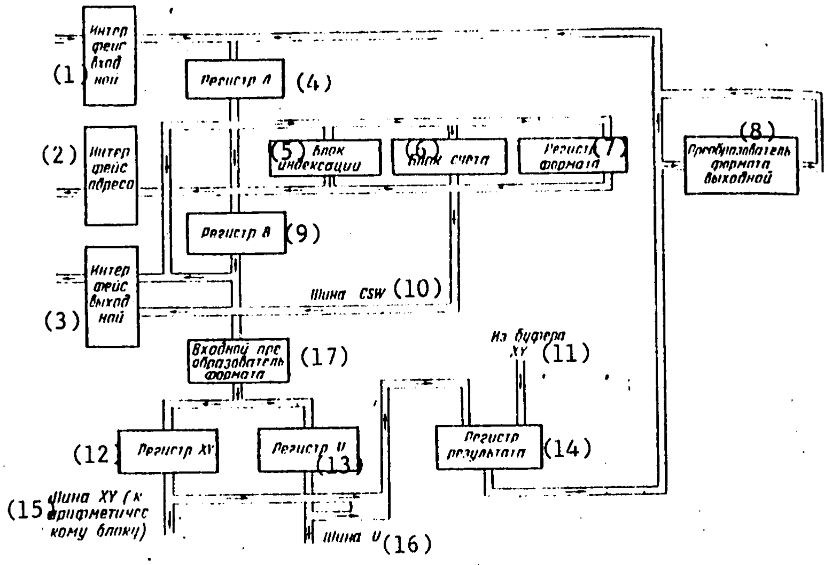


Fig. 11. Diagram of array processor control unit

Key:

1.	input interface	9.	register B
2.	address interface	10.	channel status word bus
3.	output interface	11.	from XY buffer
4.	register A	12.	register XY
5.	indexing unit	13.	register U
6.	counting unit	14.	result register
7.	format register	15.	XY bus (to arithmetic unit)
8.	output format translator	16.	U bus
		17.	input format translator

The control unit is intended for:

receiving and processing channel instructions;

reading control words from OP [main storage];

reading source data from main storage and storing computation results in main storage;

performing translation of data formats; and

writing the SSK [channel status word] and logout information during normal and abnormal end of execution of current operation, respectively.

The control unit consists of input and intermediate registers A and B. input and output data format translators, unit for signals, unit for generation of and output registers for U and XY addresses, and format register.

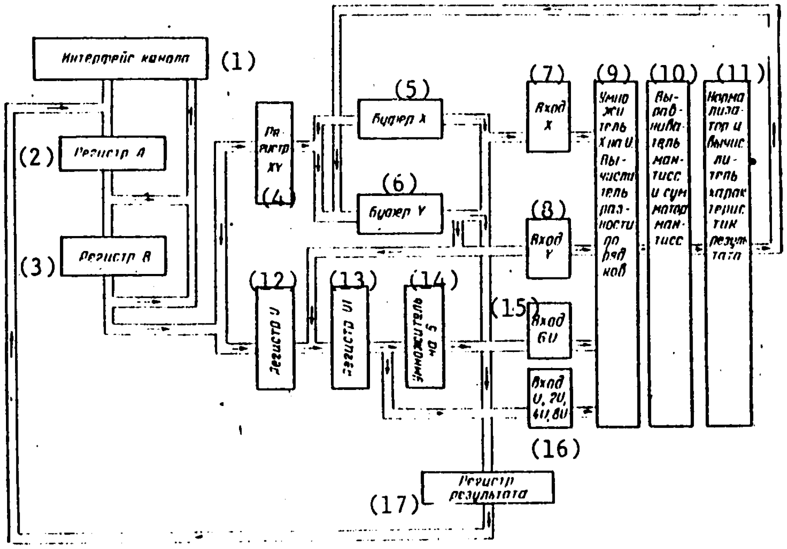


Fig. 1Z. Diagram ot array processor arithmetic unit

Key:

1. channel interface
2. Aregister
3. B register
4. XY register
5. X buffer
6. Y buffer
7. X input
8. Y input
9. X by U multiplier. Calculator of difference of exponents
10. mantissa aligner and mantissa adder
11. normalizer and calculator of characteristics of result
12. U register
13. Ul register
14. by 6 multiplier
15. 6U input
16. U, 2U, 4U, 8U input
17. result register

The A **register** is intended for receiving data and control information from main storage. Input data is sent to the arithmetic unit and results are written to main storage through the B register. The format register holds information on the format of data used in calculating. The format translators translate input and output information from fixed-point to floating-point format and vice versa. The U register is intended for receiving input data from the format translation unit and then moving it into the arithmetic unit. The XY register performs the functions of receiving and buffering the input operand. The indexing and counting units generate operand addresses and count the vector elements based on control information specified in the channel command word.

The arithmetic unit is used to perform arithmetic operations on data and consists of two buffer memories (BX and BY), buffer register U, multiplication and addition units and an output result register.

Information is entered into registers XY and U for processing through the A and B registers.

The X and Y operands are entered into the corresponding buffers with a capacity of 64 words of 82 bits each. The U operand is not buffered, but used as the multiplier. The arithmetic part of the unit consists of three stages. In the first stage, the mantissas of operands X and Y are multiplied and the difference of the exponents of the numbers XU and Y computed; in the second, the mantissas are aligned and added; and in the third, the result is normalized and the final characteristic computed. The result is stored in buffer Y and is sent through the result register to computer main storage. Using the array processor makes it possible to raise computer throughput 5-fold to 30-fold when suitable specialized problems are being solved.

Chapter 3. Organization of Data Storage

The hierarchical structure of computer storage systems has two basic levels--external and main storage, whose main characteristics are information access time and capacity. External storage, as a rule, is implemented with magnetic tapes and disks. Depending on the unit, its capacity ranges from 10^2 to 10^4 megabytes, while access time varies approximately from

10^5 to 10^2 microseconds. Main storage in computers in the Unified System has a capacity from 64K bytes to 16M bytes with an access time of about 1 microsecond.

This chapter covers organization of main storage which has a substantial effect on the efficiency of the entire computer system. This is determined not only by the increase in internal computer throughput, but also to a considerable extent by the fact that for organization of its operation, an ever growing importance is being assumed by the software system, in the development of which are required an ever larger size of main storage and considerable time on realization of control functions.

In turn, main storage parameters are determined by the physical medium used as the storage medium and by the structure of organization of data storage. Main storage parameters are also affected by the design and technological solutions adopted.

3.1. Development of Main Storage Units

Ferrite cores and semiconductor circuits with a high degree of integration are used as storage media (or memory elements) in main storage units in the Unified System of Electronic Computers. In the YeS EVM-2, storage units have varied functions and are oriented to using integrated circuits. But for economic and throughput reasons, this does not exclude the use of ferrite core storage.

Ferrite Core Storage. In developing main storage for the YeS EVM-2, ferrite cores were used that had been developed for the YeS EVM-1 with practically no major change in characteristics. In the process, thanks to the use of a more modern element base in the electronic framework circuits and more progressive design and technological solutions, a considerable decrease was achieved in design dimensions of storage calculating on information capacity. At the same time, the basic operating characteristics of storage were improved.

Serving as an example is the YeS-3206 main storage unit with ferrite cores that was developed for the YeS-1060 computer and which uses the same cores employed in the development of the YeS-3203 and YeS-3205 units for the YeS-1030 and YeS-1050 computers, respectively. Used in the YeS-3206 unit are structurally complete, interchangeable modules of storage with a size of 64K bytes, that contain ferrite matrices and electronic framework circuits. This has made possible a fourfold increase in information density compared to the YeS-3203 and YeS-3205 units. In addition, equipment bulk was cut in half as a result of using newly developed power supplies. Thus, equipment bulk of the YeS-3206 is one-eighth that of the YeS-3203 and YeS3205 units while main storage capacity is identical.

Ferrite cores 3VT with an external core diameter of 0.8 mm and 5VT with an external core diameter of 0.6 mm have received the most use. For these cores, real cycle time of the storage units (ZU) is 1.2-2.0 microseconds. With that, access time ranges from 0.5 to 1.0 microsecond. Used in practically all ferrite core main storage is access by circuit 2.5D, which is more economical in equipment than cicuit 2D and has higher speed and noise immunity than circuit 3D.

Integrated Circuit Storage. Using TTL (triode-transistor logic) semiconductor memory microciruits was begun in the YeS EVM-1. These circuits have an information capacity of 16, 32 and 64 bits per package, access time of 100-200 ns, and a cycle of 1 microsecond or more.

In developing the YeS EVM-2, integrated circuit storage units have received further development and extensive application as memories for various purposes, which is due primarily to the improvement in characteristics of memory microciruits.

For the most part, integrated circuits (IC) with a storage capacity of 4K and 16K bits have found application in Unified System main storage units. Storage IC's with information capacity of 4K bits have an access time of 200 ns and a cycle time of 400 ns. The 16K-bit storage IC's have an access time of 250 ns and cycle time of 400 ns. These storage IC's are the dynamic type, the essential shortcoming of which is the necessity of periodic refresh of the stored information to prevent loss of it. This leads to either the asynchronous mode of main storage operation with the CPU or to a relative increase (true, insignificant) in main storage cycle time. The latter occurs when the standard refresh interval has been provided for in the main storage cycle. However, the use of dynamic IC's for storage is warranted by their information capacity, which is several fold greater than that of static circuits.

In internal storage for YeS EVM-2 processors (microprogram storage, buffer storage, address matrix, general-purpose registers, etc.), three static, high-speed, ECL (emitter-coupled logic) type IC's are used: with an information capacity of 64 bits, access time of 15 ns and write time of 10 ns; with an information capacity of 256 bits, access time of 40 ns and write time of 30 ns; and with an information capacity of 1024 bits, access time of 50-60 ns and write time of 40 ns.

Storage IC characteristics determine their role in implementing computer storage. Thus, high-speed IC's with small information capacity are used in developing internal processor storage. Circuits with high information capacity and slow speed are, as a rule, used in developing main storage units.

A characteristic feature of all types of IC storage units, in contrast to ferrite core storage, is their dependence on power, i.e. information in them is lost when the power supply is interrupted. But the capability of keeping information in main storage still does not solve all the problems in recovering the computational process after a power interruption. This is connected with the fact that control information kept in the internal processor registers is lost when power is interrupted and the operating system cannot continue computations without it. In connection with this, additional measures have be to provided for to preserve information in any type of storage. The main ones are:

generation of a signal in advance of a full drop in voltage from power mains to halt the processor and lockout main storage, as well as to switch to reserve power;

maintenance of electric voltage for the time needed to copy information required for recovery of the computing process onto external media; and

recovery of control information in a computer after power is switched on that permits continuing execution of a program from the point of interruption or from a checkpoint.

With the emergence of storage IC's, the possibility emerged for solving the problem of designing main storage units in a new way. In this case, the storage units are designed at the level of development of logic assemblies, since the design is built by employing standard solutions: storage TEZ [standard exchange card], panels and racks. This also makes it possible to make use of the entire apparatus for design automation that had been developed earlier to design logic units.

Using semiconductor circuits in main storage units has made it possible to raise further the information capacity of storage per unit of design volume. The main storage units made with 16K-IC's developed in a package for the YeS EVM-2 have a capacity of 8M bytes in the standard rack together with the power supply. With further integration of semiconductor storage circuits, the possibility is emerging of placing 8M-16M-byte main storage in one-two standard processor panels.

The storage IC's listed above are not the final composition of circuits. The effort underway now on developing new microciruits and the results obtained from it are making it possible to plan on using in the Unified System of Electronic Computers in the near future storage IC's with information capacity increased several fold. The development of IC technology practically has no limitations, in connection with which this direction is very promising for both the YeS EVM-2 and in further development of the Unified System. This does not exclude development of storage elements and units with other physical principles. In the near future, one should expect development of different types of storage with storage and speed characteristics better than those available now.

It is necessary to note that the same storage elements, modules and units as a wholė, developed within the framework of the Unified System, are finding application in various models, which substantially raises the effectiveness of developing them.

3.2. Organization of Main Storage

Various means of organizing operation with main storage are used in the various models of the YeS EVM-2 as a function of their parameters. In the general case, to reduce the dependence of operation of the CPU and IO channels on the time parameters of main storage, and to reduce their mutual effect on each other, the following facilities are provided for in the structure of the YeS EVM-2:

high-speed main storage buffer;

channel buffer; and

interleaving of main storage.

Efficiency of Buffer Storage. The main storage buffer has a relatively small capacity (8K-64K bytes as a function of model throughput) with a short cycle time (40-100 ns) that permits reference to data at the CPU operating rate. Buffer storage is inaccessible to the programmer in the sense that it is not taken into account in programming and the programmer need not be aware of its existence. It is usually placed in the CPU and its operation is supported by hardware control facilities.

The expediency of introducing buffer storage lies in reducing the effective time of access to information kept in storage.

Effective access time is defined by the following expression:

$$T_{ef} = T_b \cdot P + T_s (1 - P)$$

where T_{ef} is effective access time; T_b is buffer storage cycle time; T_s is main storage cycle time; and P is the probability of reference to buffer storage (probability of finding the requested information in buffer storage).

The probability of finding requested information in buffer storage is a function of its capacity and the size of the data block exchanged between main and buffer storage. The latter parameter, although not so eveident, also affects the average frequency of processing requests from the CPU. For example, if the buffer and main storage exchange a unit of information (one instruction or one operand), probability of reference to main storage declines after the buffer is full just through repeated use of the same information in the section of the program located in the buffer. If the buffer and main storage exchange blocks (several instructions or operands), probability of reference to main storage declines even more through the capability of organizing prefetching in this case. The size of the block exchanged depends on the specific time parameters of the CPU, buffer and main storage and is usually selected as equal to 16, 32, 64 or 128 bytes.

Thus, the basic parameters of buffer storage that determine effective time of access to main storage are capacity, cycle time and size of data block exchanged between main storage and the buffer.

Since cycle time is a constant quantity determined by storage element speed, the effectiveness of using buffer storage in many respects depends on the proper selection of its capacity and size of the block exchanged.

In the ideal case, buffer storage capacity would be such that all information for which storage references are made is located in buffer storage. In this case, probability of reference to buffer storage is one and effetive time of access to main storage equals buffer storage cycle time. But in an actual system, probability of reference to buffer storage does not equal one and varies as a function of buffer storage parameters, while adhering to a law close to the exponential. With any exchange block size, probability of reference to buffer storage increases as its capacity increases. With a given buffer storage capacity, as exchange block size increases, probability of reference to buffer storage increases at first, but then declines since the number of independent data blocks that can be placed in buffer storage becomes too small. Buffer storage capacity has the strongest effect on this relationship. Large-capacity buffer storage provides the highest frequency of processing of CPU requests even with large sizes of the exchange block, while small capacity buffer storage completely loses efficiency. To achieve maximum efficiency in using buffer storage, its parameters must be mutually coordinated. In this case, probability of reference to buffer storage is 0.89-0.95, i.e. effective time of access to main storage is about equal to buffer storage cycle time.

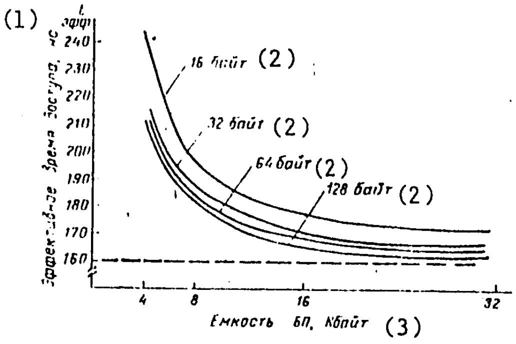


Fig. 13. Effective time of access to main storage as a function of buffer storage capacity

Key:

1. effective access time, ns
2. bytes
3. buffer storage capacity, kilobytes

Shown in fig. 13 is the effective time of access to main storage as a function of buffer storage capacity for the different sizes of data exchaneg blocks. In this case, buffer storage cycle time is 160 ns, and main storage cycle time is 2 microseconds.

Structure of Buffer Storage. This storage is intended for storing individual, most frequently used sections of a program and for organizing rapid access to it by the CPU.

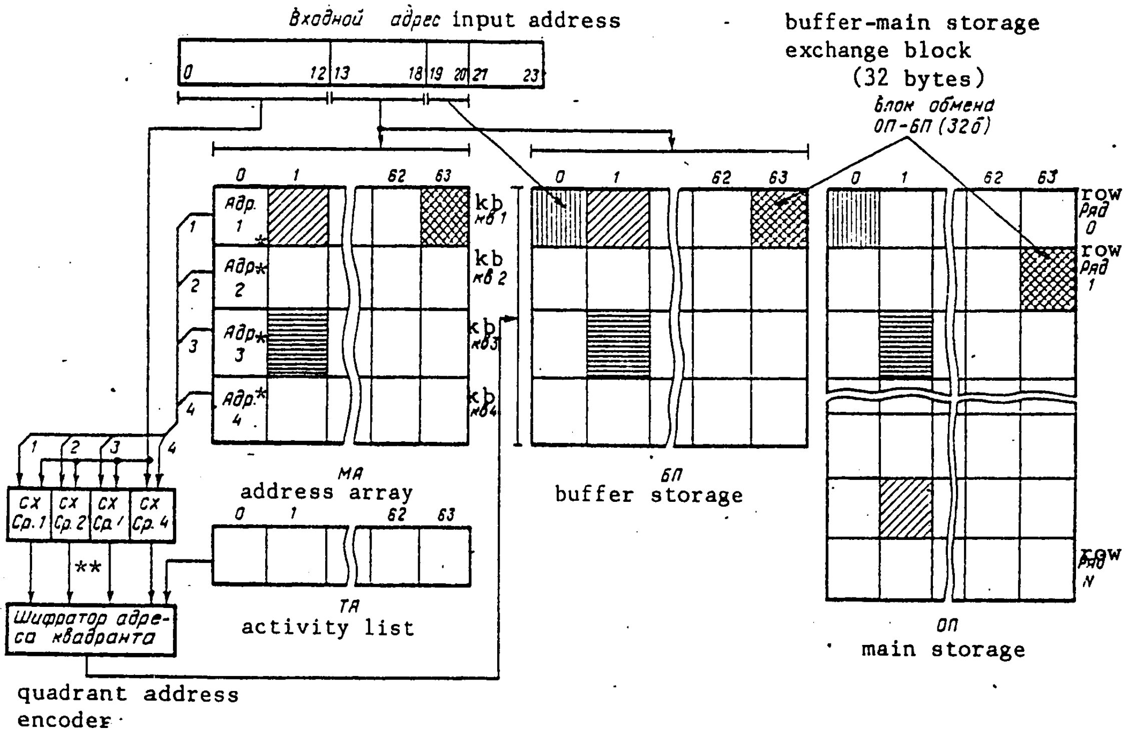


Fig. 14. Diagram of location of information in buffer storage

Key:

* address 1, 2, 3 and 4 ** comparator circuits 1, 2, 3 and 4

Buffer storage structure and a diagram of the location of information in it are shown in fig. 14. Buffer storage capacity is 8K bytes, exchange block size is 32 bytes, and the main and buffer storage access width is 8 bytes.

The entire extent of main storage is conventionally divided into individual data blocks along the horizontal and vertical. Data located along the horizontal form a row of information blocks with a total size of 2048 bytes, while that along the vertical form columns of information blocks.

The number of rows depends on main storage size. Thus, with 8M bytes, main storage has 4096 rows. The number of columns is fixed at 64. As a result of this subdivision, each row has 64 information blocks of main storage. Each block has 32 bytes or four doublewords of data located in main storage by sequential addresses. Data is exchanged between main and buffer storage by these information blocks.

Just as main storage, the buffer is divided into information blocks along the horizontal and vertical. The number of rows in buffer storage is fixed at four and from now on we shall call them quadrants. Just as main storage, the buffer has 64 columns. Thus, buffer storage has 256 information blocks of 32 bytes each.

Any information block can be placed from main storage into one of the four quadrants of the corresponding buffer column. If the buffer has a different capacity and exchange block size, the number of rows and columns of data blocks may vary, but the principle of mapping main storage information into the buffer remains the same.

The buffer storage is managed with an address array and a replacement array. The address array is divided into columns and quadrants in precisely the same way as buffer storage. The address array holds the addresses of the main storage information blocks that are contained in the buffer storage. Thus, the array holds 256 addresses, one for each buffer storage information block.

The address structure is as follows: the address of the corresponding main storage row and four presence bits. The address of the column of the address array and the address of the main storage row recorded in this column uniquely specify the address of the information placed in the buffer from main storage. The four presence bits specify the storage in the buffer of the corresponding doublewords within the bounds of a specific information block.

Each column in the address array is assigned a six-bit string from the replacement array called the activity list. The state of the bits in this list specifies the information block in the buffer with minimum activity, i.e. the block at the bottom of the list. This information block is subject to replacement when new data has to be brought in from main storage.

In accordance with the division of main and buffer storage into rows, quadrants and columns, the 24-bit address used to reference main storage is conventionally divided into four fields (table 6).

Table 6

Field Number	Address Bits	Function
1	0-12	specify main storage row address
2	13-18	specify address of column of the address array, buffer and main storage
3	19-20	specify the number of the doubleword within the bounds of the 32-byte information block
4	21-23	specify the number of the byte within the doubleword

The process of information block exchange between main and buffer storage occurs the following way.

When the CPU requests a unit of data from main storage, the address of the column (bits 13-18) of the address array is used in a simultaneous reading of the corresponding four main storage row addresses (one from each quadrant). These addresses are compared to the main storage row address specified by the reference address (bits 0-12). If there is no match, this means the requested information block is not in the buffer and must be read from main storage. In this case, four logical blocks of storage are sent sequentially at the rate of the machine cycle (the principle of interleaving of storage is used) and a doubleword of data is selected from each block. The entire group (information block) is put into the buffer, and the doubleword specified by bits 19-20 of the reference address is also sent to the CPU. The information block read from main storage is put into the buffer storage column specified by bits 13-18 of the reference address.

If the requested information is in the buffer (a match occurred), the doubleword of data is fetched from the buffer and sent to the CPU. In the process, the address of the buffer storage cell is specified the following way: the quadrant address is specified by the signal from the corresponding comparator circuit, the column address by bits 13-18 and the doubleword address by bits 19-20 of the reference address.

If the CPU is referenced to store data in main storage in an information block that is in the buffer, both main and buffer storage are updated accordingly.

Buffer storage contains an exact copy of the data put into the corresponding main storage information blocks. Consequently, if an IO channel stores data in a main storage location that contains data held in the buffer, then the buffer information block is either updated too or deleted. In this case, the following interaction between the IO channels and buffer storage is provided for. The channels fetch information from main storage, and in the process the status of the address array and activity list is not changed.

When data is stored in main storage through the IO channels, the information block referenced by the channel may be in main storage or may be located in the buffer.

In the first case, no change is made to the address array, activity list or buffer contents. In the second, the presence bits for the corresponding information block in the address array are set to zero (reset), and the activity list is adjusted to reflect minimum activity for this block.

A second algorithm for operation with buffer storage is possible. It differs from the first in that information is stored in main storage somwhat differently.

If the CPU stores data in main storage and in the process the information block required is in the buffer, only the contents of buffer storage are changed. The information is stored in main storage by blocks either when the blocks with the least activity are replaced, or by a special algorithm at times when main storage is free of servicing channel and CPU requests.

Since main storage cycle time is several-fold longer than buffer storage cycle time and according to statistics, store instructions make up a considerable portion of programs (15 percent according to Gibson 1), organization of operation with buffer storage under the second algorithm yields a substantial increase in CPU throughput. This is its chief advantage.

Of course, under the second algorithm, IO channels must use the entire control mechanism to store data and store/fetch to/from the buffer storage is the required information block is there. The same occurs in a multiprocessor system with a common extent of main storage, when one of the processors operates with data placed in buffer storage by another processor.

Thus, requests from processors and channels coincide at the buffer storage level, which leads to an increase in the number of conflicts between them and the overall throughput of the storage system is reduced. This is the main shortcoming in organizing operation with buffer storage by the second algorithm.

Which of the two algorithms is more efficient is resolved in a specific design based on the parameters of CPU throughput, IO system throughput, main and buffer storage cycle times, and equipment cost parameters. Both algorithms are used in designing the Unified System computers.

Storage Interleaving. Main storage is subdivided into individual logical blocks that can be addressed and managed independently of each other. The principle of storage interleaving, also called the address interlacing method, consists of addressing logical blocks of storage in such a way that contiguous addressed cells are located in different logical blocks.

Prior to the introduction of buffer storage into the CPU structure, storage interleaving was perhaps the only organizational means permitting a reduction in effective time of access to main storage. This decrease is especially noticeable for linear sections of programs with little logical dependency of instructions (connectivity of instructions). With the introduction of buffer storage, the effectiveness of the method was preserved with the grouped exchange (data blocks) between main and buffer storage.

The order of location of addresses in logical storage blocks with the degree of interleaving is shown in table 7.

Table 7

Logical Block Number	Address of Storage Cells			
1	0	n	$2n$	$3n$
2	1	$n+1$	$2n+1$	$3n+1$
3	2	$n+2$	$2n+2$	$3n+2$
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
n	$n-1$	$2n-1$	$3n-1$	$4n-1$

The effective time of access to main storage during operation of the interleaving mechanism is reduced because of the overlapping in time of operation of several logical blocks of storage. In fact, if each consecutive block of storage is sent after the end of the operation of the preceding one (operation without interleaving), then time I for reading a block of information of n words is defined by the expression:

$$T = (n - 1)\ T_c + T_a,$$

where T_c is main storage cycle time;

and T_a is main storage access time.

If each block of storage is sent at the rate of one machine clock pulse (operation in the interleaving mode), then this time is:

$$T = T_a + (n - 1)\ T_m,$$

where T_m is the CPU clock pulse time.

It is evident that $T_m \ll T_c$ and time of access to main storage in the second case is less than in the first.

The selection of the number of logical blocks of storage is governed by the time relationships between the CPU and main storage with regard to that spent on control. As a rule, used in Unified System computers are two-way, four-way and eight-way interleaving of storage with an access width of 4 and 8 bytes.

Conflicts in Referencing Main Storage. Conflicts in referencing main storage are a factor that reduces the effective time of access to data. The problem of reducing conflicts in referencing main storage became especially severe in third-generation machines that use a CPU with several levels of overlap and a large number of high-speed IO channels. In the process, each CPU overlap level may have its own independent access to main storage. Thus, at the level of storage control, the conflict between both CPU and channel requests and within groups of requests from these units is resolved. The mutual effect of these requests leads to delay in servicing individual requests in accessing storage. Simulation results show that under certain conditions, this delay is very significant and is 20-23 percent of the effective time of access.

Conflicts can be reduced both through improving main storage time parameters and development of methods discussed above and selecting control strategy. By control strategy is emant determination of priorities for processing of requests in the storage control unit and organization of servicing them.

A structural diagram of a storage control unit and its interaction with the CPU, channels and main storage is shown in fig. 15. Using this drawing, let us consider the structural solutions employed in the YeS EVM-2 that make it possible to reduce conflicts in accessing main storage by selecting a certain control strategy.

In the Unified System of Computers, based on system operating efficiency in accessing main storage, IO channels have a higher priority than the CPU. Use of main buffer storage in itself reduces the probability of hardware interaction in processing streams of information from channels and the CPU as a result of separation of these streams. Also, buffer storage permits reducing the effective time of access to main storage by the CPU and thereby reduces the average time main storage is engaged in processing a request. A channel buffer (BK) is used for the very same purposes in channel accesses to main storage.

The channel buffer unit is designed for rapid communication between IO channels and main storage. For each channel, this unit contains a register group that forms the channel storage buffer (BPK) for holding data, addresses and control characters. The number of registers in a group is equal to the degree of main storage interleaving or a multiple of it. Stored in them is information from channels prior to transmission of it to main storage in the write mode, or information read from main storage prior to sending it to the channels in the read mode. In the first case, the possibility emerges of rapid transfer of data from channels with main storage,

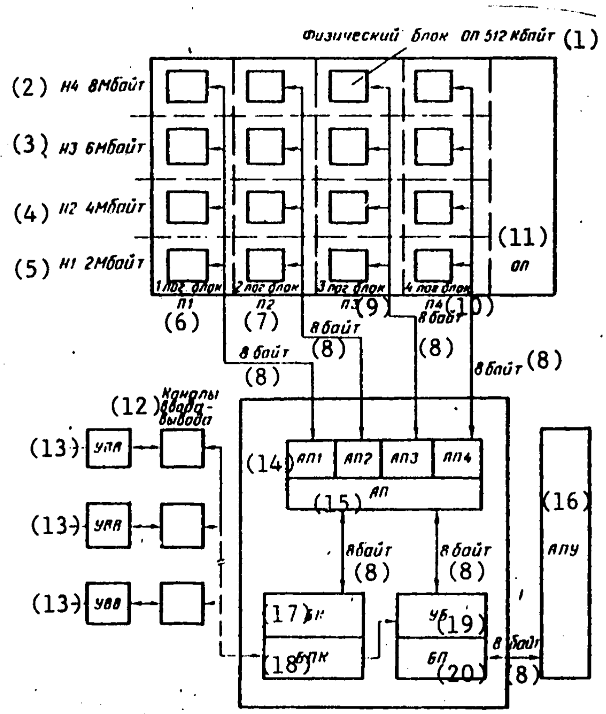


Fig. 15. Diagram of Storage Control Unit

Key:

1. 512K-byte physical block of main storage
2. N4 8M bytes
3. N3 6M bytes
4. N2 4M bytes
5. N1 2M bytes
6. 1 logical block P1
7. 2 logical block P2
8. 8 bytes
9. 3 logical block P3
10. 4 logical block P4
11. OP [main storage]
12. IO channels
13. IO device
14. AP 1, 2, 3 and 4
15. AP [storage adapter]
16. APU
17. BK [channel buffer]
18. BPK [channel buffer storage]
19. UB [buffer storage control]
20. BP [buffer storage]

irrespective of its engagement, and rapid release of the channel for execution of other operations. In the second case, main storage is rapidly released during a data exchange with a channel. In the channel write mode, data stored in the registers are sent to main storage, operating in the address interleaving mode, sequentially in each machine clock pulse. In the read mode, data are also read from main storage sequentially in each machine clock pulse and stored in the register group for the corresponding channel. Thus, in the write mode, prior to all registers being filled, the IO channels operate only with their own buffer storage, without referencing main storage. In the read mode, the first channel reference is made to main storage, but the following ones are made only to the register group of the channel buffer storage. The use of channel buffer storage is efficient for the burst mode since in this mode information is transferred in large blocks and in the process, reference to storage, as a rule, is made by sequential addresses. Thus, the channel buffer enables grouped exchange of data between main storage and the IO channels.

Since channel and CPU references to storage are independent, a conflict between them is possible only when both the channel buffer and CPU require at the same time (in one storage cycle) the same apparatus, in the case discussed, the main storage or address array. To efficiently resolve such situations, the buffer storage control unit provides for two address processing paths and two priority schemes: one for accessing the address array, and one for accessing main storage.

In the first priority scheme, conflict situations are resolved between accesses to the address array by the CPU (in determining whether the data required are in buffer storage) and by the channel buffer (in a write to delete an information block being changed in buffer storage). In the process, the CPU reference has the highest priority.

In the second priority scheme, data situations are resolved between channel buffer and CPU requests for access to main storage. The CPU always accesses main storage during a write, but during a read, only in those cases when the required information block has to fetched from main storage. In this case, channel buffer requests have the highest priority.

In reading information, a request comes from the channel buffer only for the priority schemes for referencing main storage. During a write, the buffer storage control unit gets two requests from the channel buffer: one for the prioirty scheme for main storage, and the other for the priority scheme for the address array. Both these requests are processed independently of each other. Since requests from the channel buffer are grouped, the entire information block is analyzed at once instead of individual data words, and this requires only one machine clock pulse.

3.3. Raising Reliability of Data Storage

The increase in size of main storage and complication of its organization has required taking anumber of measures ro raise the validity of operating reliability data obtained from storage. In the YeS EVM-2, these measures include:

use of checking codes to eliminate single errors and detect double ones; and

the application of special structural solutions aimed at maintaining computer system serviceability when a storage module malfunctions by reducing the overall capacity of main storage.

Checking and correcting of storage errors is based on using information redundancy, i.e. when a certain number of check bits is added to the checked word. Used for this purpose in the YeS EVM-2 is a modified Hamming code, which is a systematic code, i.e. a code where the positions of the check bits are fixed.

Check bits are formed by calculating the parity of sum of ones for certain groups of information bits. The required number of check bits is determined from the condition:

$$2^k \geqslant n+k+1 \text{ or } 2^k-k-1 \geqslant n,$$

where n is the number of information bits; and
k is the number of check bits.

It follows from this relation that seven check bits are needed for a doubleword (8 bytes). To correct single errors in a checked block and to detect double errors, an eighth bit is added to enable checking the parity of the entire information block, including the check bits. Thus, each storage cell holds 64 information bits and 8 check bits, formed by the Hamming code in the mode of storing information bits in the corresponding block of main storage. Coding and decoding matrices are shown in figs. 16 and 17, where C0-C32 are check bits and CH0-CH32 are error syndromes. C0H is the bit for overall parity of all eight bytes that permits detection of double errors and is formed either by the modulo-two sum of the information bits marked in the coding matrix or by the sum of all information and check bits. Such construction of the coding matrix permits using the same equipment to calculate C0-C32 and the byte signals of parity for transmission to the CPU. The error syndrome is derived by comparing the check bits fetched from main storage with the check bits computed by the coding matrix for the information read.

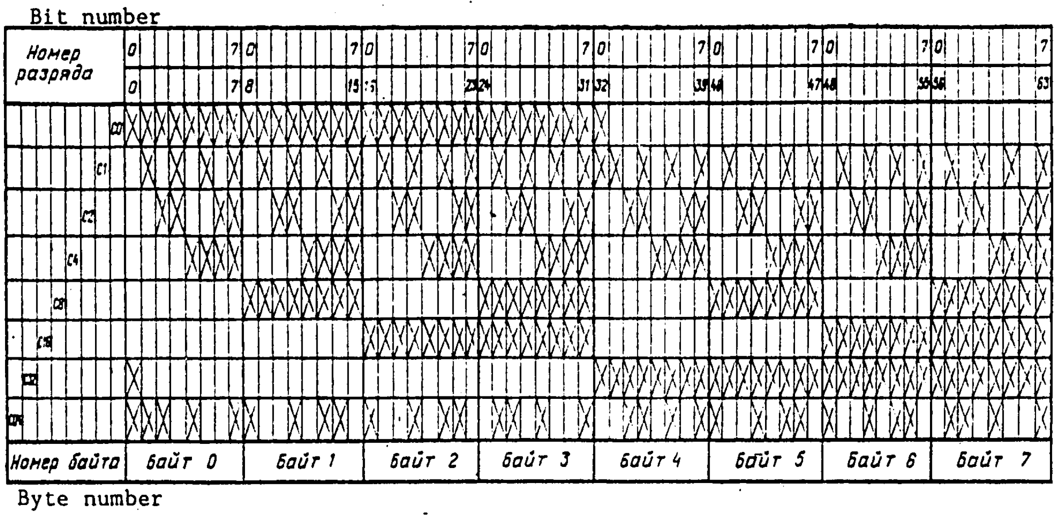


Fig. 16. Coding of information by the Hamming code

Key:
1. byte number
2. number of bits in byte

Syndrome Разряды синдромов Bits				СН1	0	1	0	1	0	1	0	1
				СН2	0	0	1	1	0	0	1	1
				СН4	0	0	0	0	1	1	1	1
СН 32	СН 16	СН 8	СН 0	Номер байти	Номер разрядов в байте (2)							
					0	1	2	3	4	5	6	7
0	0	0	0	(1)	СОЧ	С1	С2		С4			
0	0	0	1	0	С0	1	2	3	4	5	6	7
0	0	1	0		С8							
0	0	1	1	1	8	9	10	11	12	13	14	15
0	1	0	0		С16							
0	1	0	1	2	16	17	18	19	20	21	22	23
0	1	1	0									
0	1	1	1	3	24	25	26	27	28	29	30	31
1	0	0	0	4	С32	33	34	35	36	37	38	39
1	0	0	1		0	32						
1	0	1	0	5	40	41	42	43	44	45	46	47
1	0	1	1									
1	1	0	0	6	48	49	50	51	52	53	54	55
1	1	0	1									
1	1	1	0	7	56	57	58	59	60	61	62	63
1	1	1	1									

Fig. 17. Decoding matrix

If the syndrome points to the empty squares in the decoding matrix, there is a multiple error. In the check, computation of this relation is made:

$$\Lambda = \Sigma\,(C_i + \text{СН}_i) \bmod 2 + \text{СОЧ}$$
$$i = \{0,\ 1,\ 2,\ 4,\ 8,\ 16,\ 32\}$$

Analysis is performed in accordance with table 8.

Table 8

Syndrome Status	A	Conclusion
zeros	0	no error
≠ 0	1	single error
≠ 0	0	double error
zeros	1	error in COЧ bit

Irregularity in bits 0 and 32 of the coding matrix permits distinguishing errors in bits 0, 32, C0 and C32.

With a multiple error, data in main storage is regenerated for further program processing.

Facilities for checking data of main storage are implemented in the storage adapter unit (AP) (see fig. 15). A storage adapter is incorporated in the CPU structure because the blocks of main storage are essentially a group of storage modules without general address registers and registers for information being written or read and also do not have facilities for generating and checking the correctness of check bits. These functions are imposed on the storage adapters.

A storage adapter unit has its own group of registers for the address of data being written or read for each logical block of main storage. Address and data buses are common for all courses of a logical block of main storage. Checking facilities located in the storage adapter are common for all logical blocks of main storage.

This solution has enabled simplifying the structure of the main storage unit and the CPU-storage communication interface. This made it possible to standardize it which enabled using the same storage units in various models. In addition, the specific volume of equipment was reduced while the capacity of main storage was increased.

In the Unified System computers, main storage capacity for each model is established in accordance with its throughput and ranges from the minimum, defined as a rule by the standard complete set of the basic version of the computer, to the maximum, determined by the physical connection capability. The capability of increasing main storage capacity is supported by the modularity of design at the level of blocks of storage and units. In main storage made with integrated circuits, main storage capacity can be increased at the level of storage TEZ's [standard exchange cards].

For an example, let us discuss in more detail the YeS-3206 main storage unit for the YeS-1060 computer. The basic scheme for dividing the YeS-3206 main storage unit into individual modules and the principle of control of them coincide with the diagram shown in fig. 15.

The minimum main storage capacity for the YeS-1060 computer is 2M bytes and consists of two YeS-3206 units. Design of the YeS-3206 unit is based on the modular principle, expressed in that the total capacity of 1024K bytes is divided into two identical storage blocks, each with a capacity of 512K bytes. Each storage block is independent within the unit and has autonomous interface and control. In turn, each storage block uses storage modules as magnetic storage. The information capacity of each storage module is 64K bytes. A storage block is made up of eight storage modules, and a unit has 16. Storage modules are structurally complete, replaceable and interchangeable units

Maximum main storage capacity for the YeS-1060 is 8M bytes and is obtained by connecting additional YeS-3206 units. For this purpose, each unit has circuits for relaying input and output signals from the CPU. Capacity is increased in 2M-byte increments by serial connection of storage blocks. Thus, storage blocks connected by common address, informational and control buses make up a logical storage block. The number of storage blocks within a logical block is defined by the complete set of storage in the computer and may vary from one to four. Selection of one of the four storage blocks within the bounds of a logical block occurs by direct feed of control signals to it from the CPU through radial lines. The address and information are fed through main lines. At each moment of time, one of the four storage blocks is operating in a logical block. Storage blocks are addressed by the horizontal course number (N1-N4) and the logical storage block number (P1-P4).

Facilities for reconfiguring main storage in event a block malfunctions are provided on the computer control console. Reconfiguration operations depend on storage capacity. If total main storage capacity is more than 2M bytes, when one of the blocks malfunctions, an entire course is disconnected and reconfigured into the area of high-order addresses. Disconnection of an entire course is required to maintain the mode of storage interleaving. If storage capacity is the minimum, then when one block malfunctions, the interleaving mode is removed automatically and the faulty block is reconfigured into the area of high-order addresses.

The control console also provides facilities for disconnecting buffer storage if it malfunctions, and in this event, operation continues with main storage.

Modularity of design of main storage units combined with special structural solutions permits maintaining computer serviceability when modules fail at various levels. In the general case, these facilities permit achieving an increase in reliability, improvement in interchangeability and repairability, and enhancement of adaptability to manufacture and debugging of main storage.

Chapter 4. Development of Input/Output System

4.1. Organization and Logic Structure of Input/Output Channels

In third-generation machines, as a result of optimization of IO functions, the need arose for developing specialized processors that enable data exchange between IO devices, and also external storage devices, and main storage in parallel with processing of data in the processor.

The hardware and software that facilitate exchanges between main storage and one external device make up a subchannel. The aggregate of subchannels form an IO channel.

The different modes for transferring information and ways of organizing interaction between channels and the CPU on the one hand, and peripherals on the other, determine the IO processor channel.

Three types of channels can be connected in the YeS EVM-2: selector, byte-multiplexer and block-multiplexer.

A selector channel is intended for data exchange with high-speed external storgae units in the burst mode. In connection with this, the logical basis of a selector channel is formed by a separate subchannel that executes one channel program at a time, by strictly regulating the end of the preceding IO procedure before starting the next.

The byte-multiplexer channel effects simultaneous execution of several channel programs for several medium or low-speed peripherals. The main mode of operation is multiplex, although in many cases, the burst mode of operation of the multiplexer channel is provided for. The logical basis of a multiplexer channel is formed by the set of multiple subchannels used for data exchange with various peripherals in the time-sharing mode. For this purpose, there are individual equipment and means of interface for each subchannel or common facilities for a group of subchannels. A new type of channel, the block-multiplexer, has been introduced in the YeS EVM-2.

It is intended for organization of parallel operation of several high-speed pheıals over one information line. This channel has the properties of both the selector and the multiplexer channels of the YeS EVM-1.

The processor initiates operation of a channel program using IO operations by specifying the operation code, the channel number and the address of the peripheral with which data is to be exchanged. In addition, the address of the first command in the channel program in main storage is specified. The channel program establishes the type of operation assigned and determines the order of its execution. Fetched from main storage for this purpose is a series of channel command words [CCW] containing the operation code of the channel command, the beginning address of the main storage block allocated for the exchange, the amount of data to be exchanged and the channel modes of operation.

The function of the CCW bits is given in table 9.

Table 9

Bit Number	Bit Function
0-7	channel command code. There are six IO operations: READ, WRITE, READ BACKWARD, SENSE, CONTROL and TRANSFER IN CHANNEL
8-31	address of the first byte of information in main storage
32-36	modification flags that permit changing the sequence of execution of a channel program
32	chain-data flag. It causes fetching of the next CCW if in the preceding CCW, the information byte counter becomes equal to zero. The operation code in the new CCW is ignored and execution of the operation begun earlier continues
33	chain-command flag. It causes the next CCW to be fetched and executed after completion of execution of the operation specified in the current CCW.
34	suppress-length-indication flag. It causes suppression of indication of incorrect length that indicates that the data format required by the peripheral does not match the number of data bytes specified by the CCW
35	skip flag. Specifies suppression of transfer of information to storage during a READ, READ BACKWARD or SENSE operation
36	program-controlled-interruption flag. Causes the channel to generate an interruption condition
37	indirect data address flag
38-39	must be zero, otherwise the system specifies an error in the channel program
40-47	not used
48-63	specifies number of bytes to be exchanged

In addition to the instructions START I/O, TEST I/O and TEST CHANNEL, the following instructions are also used in the YeS EVM-2: START I/O FAST RELEASE, CLEAR I/O, HALT DEVICE and STORE CHANNEL ID.

These IO instructions cannot be used in a problem program since they are privileged and executed only when the processor is in the supervisor state. For communication between the IO supervisor and a specific channel program, there are two fixed locations of main storage with addresses 72 and 64, in the directly addressable area of processor storage, that contain, respectively, the channel address word [CAW] and the channel status word [CSW]. The function of the CAW bits is given in table 10.

Table 10

Bits	Function
0-3	storage protection key. Intended to protect the area allocated for the IO operation
4-7	zeros, otherwise the system specifies an error in the channel program
8-28	address of first CCW in the channel program

By the CAW, the channel obtains additional information on the location of the channel program in main storage and the keys for protection of the exchange blocks, while the processor is informed by the CSW on the status of the channel and peripheral as well as on the results of an IO instruction execution.

The states of the channel, subchannel and peripheral are reflected by a combination of the following signals: available, interruption pending, working or not operational. These states change continuously in the process of executing a channel program and when necessary are stored by command in location 64 of main storage in the form of the CSW just as are special situations that occur in execution of IO operations. As a rule, the CSW is stored in main storage at the end of execution of an IO operation, but in excpetional cases, the store may be effected by an interruption in the process of executing an IO operation. The procedure for processing interruptions that may occur simultaneously in different peripherals is defined by the channel priority schemes that provide for sequential processing of all interrupt requests stored to this moment in the peripherals and subchannels. To avoid loss of information on the IO state, the operation of all channels must be masked before the end of analysis of the contents of the CSW.

The states of a channel and peripheral that occur during execution of IO operations become clear when the CSW bit functions given in table 11 are examined.

The designation of the CSW bits given in table 11 is modified by the equipment used and the specific causes that occurred during execution of the IO operation and which caused storage of the CSW.

The principles of IO operations selected in the Unified System have determined the logical structure of the channels, each of which must perform the following functions:

accept and decode IO instructions from the processor;

establish communication with the external device (VU) specified by an IO instruction;

Table 11. CSW Fields

Bits	Function
0-3	storage protection key, from CAW
4-5	zeros. Flag for unexecuted logging
6-7	deferred condition code
8-31	address of following CCW, differing from current CCW by a byte
32-39	external device status byte
32	attention
33	status modifer
34	UVU [control unit] end
35	busy
36	channel end
37	device end
38	unit check
39	unit exception
40-47	channel status byte
40	program-controlled interruption
41	incorrect length
42	program check
43	protection check
44	data check
45	control check
46	interface check
47	data chaining check
48-63	residual count for the last CCW used

fetch the channel program from main storage;

decode channel commands, test them for validity and execute the operations specified by them;

accept/send control signals from/to an external device over the IO interface;

provide for receiving, sending, checking, counting and storing data during an exchange between main storage and an external device;

generate signals on channel status, receive and store signals on peripheral status, form the CSW and store it in location 64 of main storage; and

accept interrupt requests from an external device, organize them and send them to the processor.

4.2. Development of Principles of Input/Output Operations

To enable operation of the block-multiplexer channel and to raise the effective data transmission rate, the YeS EVM-2 offers new capabilities:

the mode of multiplexing blocks of data;

additional IO instructions;

channel command retry;

expanded IO interface; and

indirect data addressing.

Data Blo.k Multiplexing Mode. From a programmer's viewpoint, the block-multiplexer channel is an independent device having several subchannels and operating in parallel with the processor program. The mode of operation of this channel depends on the state of the multiplexing control bit (bit 0 of control register 0 in the CPU). If the value fo this bit equals zero when a START I/O instruction is received, the block-multiplexer channel will operate in the selector mode, and in this case, it is functionally equivalent to the selector channel (multiplexing inhibited). If the value of this bit is one when a START I/O instruction is received, the block-multiplexer channel will execute the operation requested by a peripheral on the basis of block multiplexing simultaneously with operations requested by other peripherals (multiplexing allowed).

The selector mode or block-multiplexing mode is maintained by the channel for the entire time of execution of the channel program. In the block-multiplexing mode, during execution of a channel operation by a program not associated with data transfer, the channel may be switched to execution of the data transfer operation for another channel program. Such switching may occur between data blocks, if command chaining is specified in the channel program or the channel command retry procedure is being executed, since data transfer within the bounds of a block is effected by the channel in the burst mode.

Fig. 18 shows the timing diagram for the sequence of operation of the block-multiplexer channel in the data block multiplexing mode.

If a channel receives a channel-end signal from a peripheral while executing a channel program in the command chaining mode, it stops execution of this program without waiting for the arrival of the device-end signal, specifying thereby readiness for execution of an IO operation with another peripheral. The channel keeps the status of the discontinued operation in subchannel storage in the form of a unit control word (UCW).

When the disconnected external device is ready to operate with the channel, it tries to communicate with the channel and again make use of its resources. If the channel is free, the value of the UCW is read from subchannel storage and the channel resumes execution of the interrupted channel program. But if the channel is busy, the external device must wait until the channel is free.

The efficiency of operation of the block-multiplexer channel in the block mode is achieved only with the support of an external device in this mode.

Additional IO Instructions. To raise the efficiency of use of the IO system in the YeS EVM-2, additional IO instructions have been provided (table 12).

Table 12

Name	Mnemonic	Operation Code
START IO FAST RELEASE	SIOF	9C01
CLEAR IO	CLRIO	9D01
HALT DEVICE	HDV	9E01
STORE CHANNEL ID	STIDC	B203

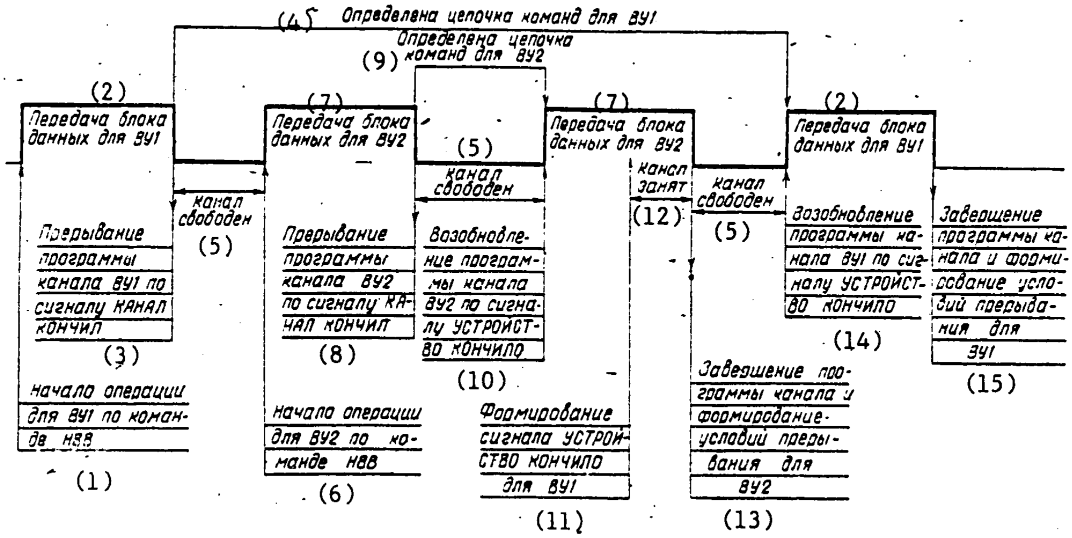

(4) Определена цепочка команд для ВУ1
(9) Определена цепочка команд для ВУ2
(2) Передача блока данных для ВУ1
(7) Передача блока данных для ВУ2
(7) Передача блока данных для ВУ2
(2) Передача блока данных для ВУ1
(5) канал свободен
(5) канал свободен
(5) канал свободен
(12) канал занят
(3) Прерывание программы канала ВУ1 по сигналу канал кончил
(8) Прерывание программы канала ВУ2 по сигналу канал кончил
(10) Возобновление программы канала ВУ2 по сигналу устройство кончило
(14) Возобновление программы канала ВУ1 по сигналу устройство кончило
(15) Завершение программы канала и формирование условий прерывания для ВУ1
(1) Начало операции для ВУ1 по команде НВВ
(6) Начало операции для ВУ2 по команде НВВ
(11) Формирование сигнала устройство кончило для ВУ1
(13) Завершение программы канала и формирование условий прерывания для ВУ2

Fig. 18. Timing diagram of sequence of operation of block-multiplexer channel in the mode of block-multiplexing of data

Key:

1. start of operation for external device [ED] 1 by SIO instruction
2. transfer of data block for ED1
3. interruption of channel program for ED1 by channel-end signal
4. command chaining specified for ED1
5. channel free
6. start of operation for ED2 by SIO [START I/O] instruction
7. transfer of data block for ED2
8. interruption of channel program for ED2 by channel-end signal
9. command chaining specified for ED2
10. resumption of channel program for ED2 upon device-end signal
11. generation of device-end signal for ED1
12. channel busy
13. end of channel program and generation of interrupt condition for ED2
14. resumption of channel program for ED1 upon device-end signal
15. end of channel program and generation of interrupt condition for ED1

The instruction START I/O FAST RELEASE starts an IO operation in the external device addressed. If the value of the multiplexing control bit is 0, the channel executes this instruction just like a START I/O instruction.

If the value of the multiplexing control bit is 1, then the channel fetches the CAW from main storage location 72 and immediately releases the processor, rather than after completion of the device-selection procedure as is done when the instruction START I/O is executed. At the same time, the data block-multiplexing mode is started in the block-multiplexer channel. In parallel with processor operation, the channel performs the device-selection procedure and starts the IO operation.

The IO operation is not initiated when the channel or device detects errors in executing the START I/O FAST RELEASE instruction. In this case, the status of the channel and device, as well as the deferred condition code (bits 6 and 7), are indicated in the CSW which is stored during the next IO interruption. Bits 6 and 7 in the CSW indicate whether conditions have been encountered subsequent to the setting of a condition code 0 for START I/O FAST RELEASE that would have caused a different condition code setting for START I/O.

Table 13 shows the possible setting of these bits and their meanings.

Table 13

Bit 6	Bit 7	Meaning
0	0	normal IO interruption
0	1	deferred condition code is 1
1	0	not used
1	1	deferred condition code is 3

The instruction START I/O FAST RELEASE has the advantage over START I/O that the processor can be released sooner for other work. A disadvantage, however, is that if a deferred condition code is presented, the resultant CPU execution time may be greater than that required in executing START I/O.

When operating in the mode of block multiplexing of data, the block-multiplexer channel enables simultaneous execution of several channel programs and IO operations.

Each channel program requires its own subchannel. If several channel programs using all available subchannels are being executed in the block-multiplexer channel, then the next START I/O FAST RELEASE instruction will start an IO operation and switch the channel to the selector mode. The channel selector mode is maintained until the end of execution of the channel program. After the end of execution of the channel program, the channel is switched to the mode of block multiplexing of data and continues the IO operations begun earlier with other devices.

Execution of the CLEAR I/O instruction depends on the value of the multiplexing control bit.

When the multiplexing control bit is 0, this instruction is executed just like the TEST I/O instruction. If this bit equals 1, then the CLEAR I/O function causes the current operation with the addressed device to be discontinued, the state of the operation at the time of the discontinuation to be indicated in the stored CSW, and the subchannel to be switched to the available state. After completion of these operations, the processor is released and the condition code is set to 1.

The CLEAR I/O instruction will switch a subchannel to the available state considerably faster than the HALT I/O instruction. To free a subchannel after completion of a HALT I/O instruction, the processor has to process the IO interruption from the given subchannel.

The HALT DEVICE instruction causes the current operation at the addressed IO device to be terminated. With this instruction, a channel does not halt burst operation with a device other than the one addressed (in contrast to the HALT I/O instruction).

After termination of the current operation in the addressed device, the channel is switched to the interruption pending state, and to free the subchannel, the processor must process the IO interruption.

The instruction STORE CHANNEL ID causes information describing the addressed channel (channel type, channel model, maximum length of the extended logout) to be stored in locations 168-171. The channel stores this information in main storage. This information is used later by the program to allocate storage for the extended logout as well as during channel error-recovery software operation.

Channel Command Retry. The YeS EVM-2 has the capability of channel command retry. Command retry is a function of the channel and device that permits retrying the command without requiring an IO interruption. Command retry is initiated by the device presenting either of two device status-bit combinations by means of a special IO interface sequence.

Fig. 19 shows the algorithm for using the command retry procedure in the channel and device.

If a retry can be performed immediately, the device issues these indicators in the status byte: channel end, unit check, status modifier and device end. After receiving the status byte with these flags, the channel starts executing the command retry procedure. During the command retry, operations in the channel are similar to those performed for command chaining.

If the device end flag was not put into the device status byte, then the channel is just adjusted for command retry. After a status byte with the device-end flag is received, the channel will start execution of the command retry procedure. If the VU [external device] presents a status byte with the device-end and status-modifier flags, then command retry is suppressed and when necessary, either command chaining is continued or an IO interruption condition is generated.

Expanded IO Interface. Compared to the YeS EVM-1, IO interface capabilities have been considerably expanded in the YeS EVM-2. To raise IO interface throughput, parallel two-byte transfer of data has been introduced. Two sets of data buses have been provided for this: channel buses (18 lines) and subscriber buses (18 lines). Two additional channel lines (channel marker) and two additional subscriber lines (subscriber marker) specify the presence of data on these buses.

For compatibility with the YeS EVM-1 interface, the channel and subscriber buses are divided into two sets: the first sets of buses (with 9 lines each) are used in both the first and second phases of the Unified System, while the second sets of buses (with 9 lines each) and the marker lines can be used only in the YeS EVM-2.

Fig. 20 shows the capabilities of operation of the expanded two-byte IO interface in the data transfer mode during reading (a) and writing (b).

Accelerated data transfer has been provided for in the YeS EVM-2 IO interface. For this, two lines (DAN-A and DAN-K) have been added that accompany the corresponding data on the buses and are similar to lines INF-A and INF-K. The DAN-A line signal may be present in the IO interface while the INF-A line signal is cut-off and vice versa.

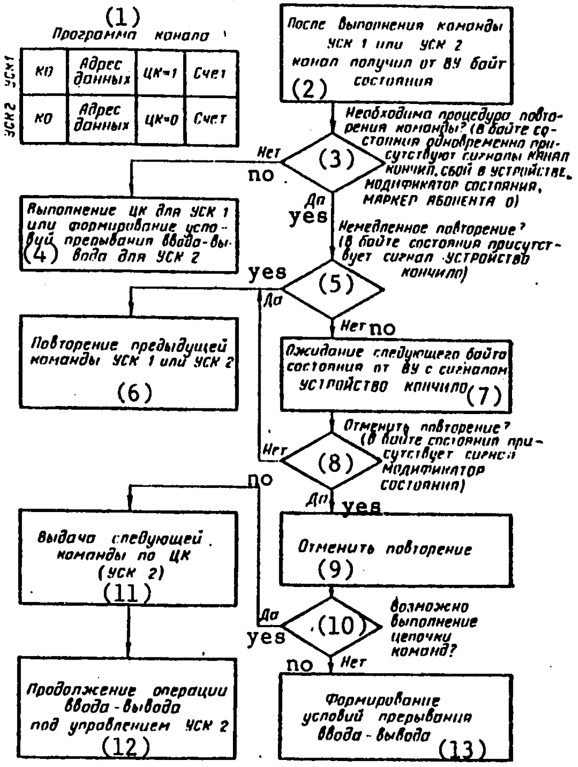

(1)
Программа канала
УСК1
УСК2
КО
Адрес данных
ЦК=1
Счет
КО
Адрес данных
ЦК=0
Счет
После выполнения команды УСК 1 или УСК 2 канал получил от ВУ байт состояния
(2)
Необходима процедура повторения команды? (В байте состояния одновременно присутствуют сигналы КАНАЛ КОНЧИЛ, СБОЙ В УСТРОЙСТВЕ, МОДИФИКАТОР СОСТОЯНИЯ, МАРКЕР АБОНЕНТА 0)
(3)
Нет
no
Да
yes
Выполнение ЦК для УСК 1 или формирование условий прерывания ввода-вывода для УСК 2
(4)
Немедленное повторение? (В байте состояния присутствует сигнал УСТРОЙСТВО КОНЧИЛО)
(5)
yes
Да
Нет
no
Повторение предыдущей команды УСК 1 или УСК 2
(6)
Ожидание следующего байта состояния от ВУ с сигналом УСТРОЙСТВО КОНЧИЛО
(7)
Отменить повторение? (В байте состояния присутствует сигнал МОДИФИКАТОР СОСТОЯНИЯ)
(8)
Нет
no
Да
yes
Выдача следующей команды по ЦК (УСК 2)
(11)
Отменить повторение
(9)
Возможно выполнение цепочки команд?
(10)
Да
yes
no
Нет
Продолжение операции ввода-вывода под управлением УСК 2
(12)
Формирование условий прерывания ввода-вывода
(13)

Fig. 19. Algorithm for using the command retry procedure in a channel and device

Key:

1. channel program
   CCW1: operation code/data address/ TsK [chain-command flag] = 1/ count
   CCW2: operation code/data address/ chain-command flag = 0/ count
2. status byte received from device by channel after execution of CCW1 and CCW2 commands
3. is command retry procedure required? (channel-end, unit-check, status-modifer and subscriber marker 0 signals are present simultaneously in the status byte)
4. execution of command chaining for CCW1 or generation of IO interrupt condition for CCW2
5. immediate retry? (device-end signal is presented in status byte)
6. retry of preceding CCW1 or CCW2 commands
7. wait for next status byte from device with device-end signal
8. cancel retry? (status-modifer signal is present in status byte)
9. cancel retry
10. execution of command chain possible?
11. issue next command under command chaining (CCW2)
12. continue IO operation under control of CCW2
13. generate IO interruption condition

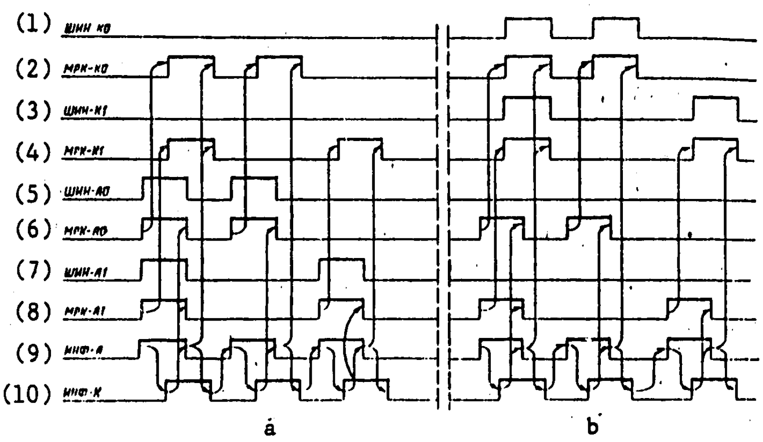


Fig. 20. Capabilities of expanded two-byte IO interface in the mode of data transfer during reading (a) and writing (b)

Key:

1. bus-K0 -- first set of channel buses
2. marker-K0 -- channel marker line
3. bus-K1 -- second set of channel buses
4. marker-K1 -- channel marker line
5. bus-A0 -- first set of subscriber buses
6. marker-A0 -- subscriber marker line
7. bus-A1 -- second set of subscriber buses
8. marker-A1 -- subscriber marker line
9. INF-A -- signals accompanying data on subscriber buses
10. INF-K -- signals accompanying data on channel buses

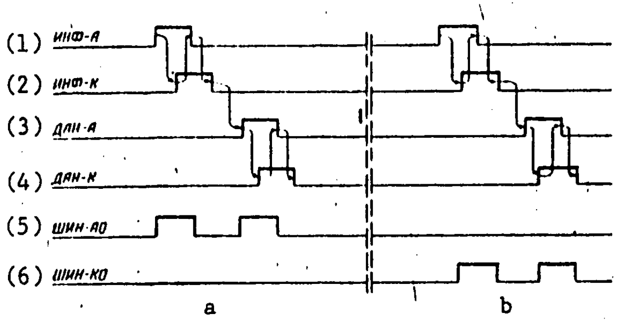


Fig. 21. Interaction of signals INF-A, DAN-A, INF-K and DAN-K in mode of data transfer during reading (a) and writing (b)

Key:

1. INF-A
2. INF-K
3. DAN-A -- signals accompanying data on subscriber buses
4. DAN-K -- signals accompanying data on channel buses
5. bus-A0
6. bus-K0

The introduction of the additional lines, DAN-A and DAN-K, permits the IO interface to provide throughput to 1.5M bytes/sec. for one-byte data transfer, and to 3M bytes/sec. for two-byte.

Fig. 21 shows the interaction of the signals INF-A, DAN-A, INF-K and DAN-K in the mode of data transfer during reading (a) and writing (b).

In the YeS EVM-2, all communication lines in the IO interface are structurally combined in four cables, of which two are fully compatible with YeS EVM-1 IO interface cables, which permits connecting YeS EVM-1 external devices to YeS EVM-2 IO channels and vice versa (fig. 22).

Connectors A1-A4 are used both for connecting YeS EVM-1 external devices and for connecting YeS EVM-2 external devices (one-byte interface). Connectors A5-A3 are used for connecting only YeS EVM-2 devices (two-byte interface). Blocks of matching resistors are shown at the connectors marked with an asterisk.

Table 14 gives the IO interface connector designations and their function.

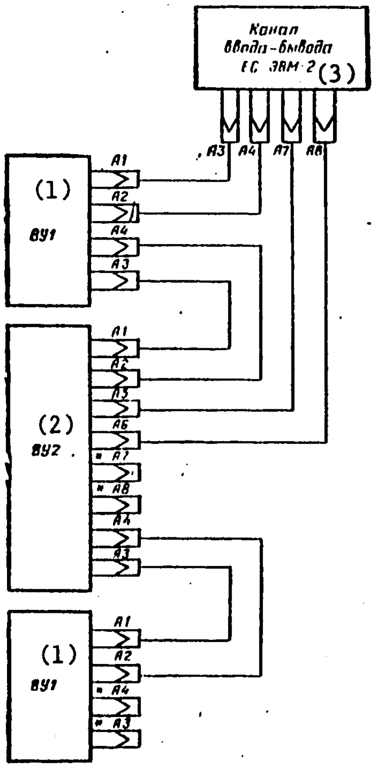


Fig. 22. Diagram of connection of external devices to IO channel

Key:

1. VU1 -- YeS EVM-1 external device
2. VU2 -- YeS EVM-2 external device
3. YeS EVM-2 IO channel

* A3, A4, A7, A8 -- blocks of matching resistors

Table 14

Conventional Connector Designation	Function
A1	Intended for connection to IO channel or preceding device and for transfer of data signals (first sets of buses)
A2	Intended for connection to IO channel or preceding device and for transfer of control signals (one-byte and accelerated modes)
A3	Used for connection of next device and for transfer of data signals (first sets of buses)
A4	Used for connection of next device and for transfer of control signals (one-byte and accelerated modes)
A5	Intended for connection to IO channel or preceding device and for transfer of data signals (second sets of buses)
A6	Intended for connection to IO channel or preceding device and for transfer of control signals (two-byte mode)

[Continuation of Table 14]

Conventional Connection Designation	Function
A7	Used for connecting the next device and for transfer of data signals (second sets of buses)
A8	Intended for connection of next device and for transfer of control signals (two-byte mode)

Channel Indirect Data Addressing. Channel indirect data addressing [CIDA], a companion facility to dynamic address translation in the processor, translates data addresses for IO operations. It permits a single channel command word [CCW] to control the transmission of data that spans noncontiguous pages in real main storage.

CIDA is specified by flag bit 37 in the CCW which, when one, indicates that the data address in the CCW is not used to directly address data. Instead, the address points to a list of words, called indirect-data-address words (IDAW's), each of which contains an absolute address designating a data area within a 2,048-byte block of main storage.

When the indirect data addressing bit in the CCW is one, bits 8-31 of the CCW specify the location of the first indirect data address (KAD) word to be used for data transfer for the command. Additional IDAW's, if needed for completing the data transfer for the CCW, are in successive locations in main storage.

The number of IDAW's required for a CCW is determined by the count field of the CCW and by the data address in the initial IDAW.

Each IDAW is used for the transfer of up to 2,048 bytes and can designate any location in main storage. Data is then transferred, for read, write, control and sense commands, to or from successively higher storage locations or, for a read backward command, to successively lower storage locations, until a 2,048-byte block boundary is reached. The control of data transfer is then passed to the next IDAW. The second and any subsequent IDAW's must specify, depending on the command, the first or last byte of a 2,048-byte block. Thus, for read and write commands, these IDAW's will have zeros in bit positions 21-31. For a read backward command, these IDAW's will have ones in bit positions 21-31.

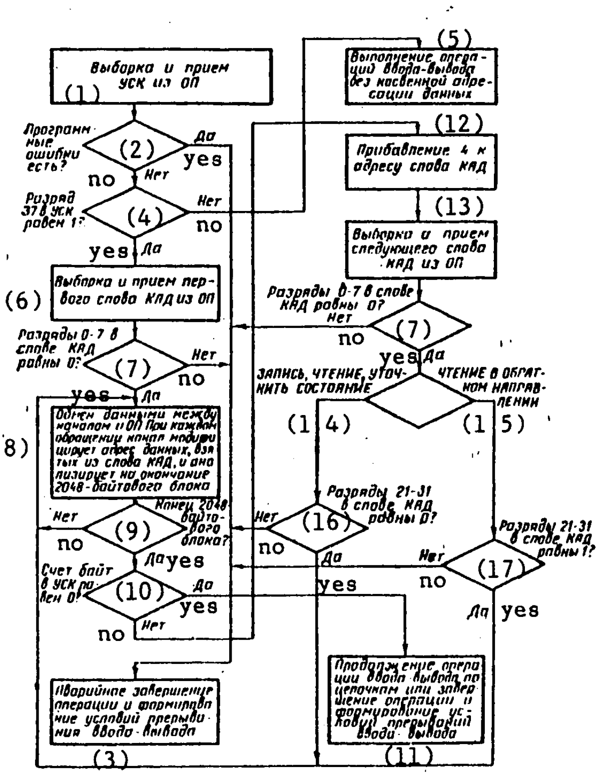
Except for the unique restrictions on the specification of the data address by the IDAW, all other rules for the data address, such as for protected storage and invalid addresses, and the rules for data prefetching, remain the same as when indirect data addressing is not used.

The format of the IDAW and the significance of its fields are as follows:

0 7	8 31
zeros	data address

Bits 0-7 must contain zeros. Bits 8-31 specify the location of the first byte to be used in executing the IO operation. For the first IDAW, the data address does not have to be a multiple of a 2,048-byte block. For subsequent IDAW's, the data address must be located on a boundary of a 2,048-byte block.

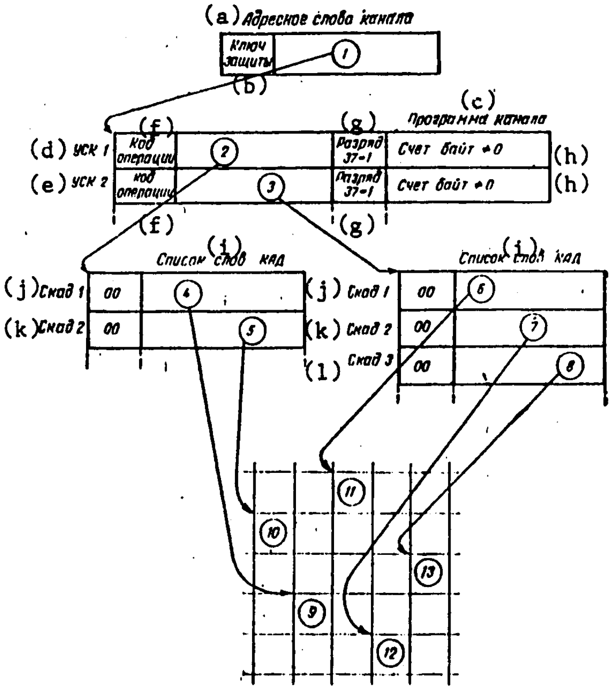
Fig. 23 shows the algorithm for executing an IO operation with indirect data addressing. Fig. 24 shows the replacement of data addresses in using indirect data addressing.


(1)
Выборка и прием УСК из ОП
(2)
Программные ошибки есть?
Да
yes
Нет
no
(4)
Разряд 37 в УСК равен 1?
Нет
no
yes
Да
(5)
Выполнение операции ввода-вывода без косвенной адресации данных
(6)
Выборка и прием первого слова КАД из ОП
(7)
Разряды 0-7 в слове КАД равны 0?
Нет
no
yes
Да
(8)
Обмен данными между каналом и ОП. При каждом обращении канал модифицирует адрес данных, взятых из слова КАД, и анализирует на окончание 2048-байтового блока
(9)
Нет
no
Конец 2048-байтового блока?
Да
yes
(10)
Счет байт в УСК равен 0?
Да
yes
Нет
no
(3)
Аварийное завершение операции и формирование условий прерывания ввода-вывода
(12)
Прибавление 4 к адресу слова КАД
(13)
Выборка и прием следующего слова КАД из ОП
Разряды 0-7 в слове КАД равны 0?
Нет
no
yes
Да
Запись, чтение, уточнить состояние
Чтение в обратном направлении
(14)
(15)
(16)
Разряды 21-31 в слове КАД равны 0?
Нет
no
Да
yes
(17)
Разряды 21-31 в слове КАД равны 1?
Нет
no
Да
yes
(11)
Продолжение операции ввода-вывода по цепочкам или завершение операции и формирование условий прерывания ввода-вывода

Key:

1. fetch and reception of CCW from main storage
2. program errors?
3. abnormal end of operation and generation of IO interrupt condition
4. bit 37 in CCW = 1?
5. execute IO operation without indirect data addressing
6. fetch and reception of first IDAW from main storage
7. bits 0-7 in IDAW = 0?
8. exchange data between channel and main storage. With each access, the channel modifies the address of the data taken from the IDAW and analyzes for end of 2,048-byte block
9. end of 2,048-byte block?
10. CCW byte count = 0?
11. continue IO operation by chaining or end operation and generate IO interrupt condition
12. add 4 to IDAW address
13. fetch and reception of next IDAW from Main storage
14. write, read, sense
15. read backward
16. bits 21-31 in IDAW = 0?
17. bits 21-31 in IDAW = 1?

Fig. 23. Algorithm for execution of IO operation with indirect data addressing



Key:
- a. channel address word [CAW]
- b. protection key
- c. channel program
- d. CCW 1
- e. CCW 2
- f. operation code
- g. bit 37 = 1
- h. byte count ≠ 0
- i. list of IDAW's
- j. IDAW 1
- k. IDAW 2
- l. IDAW 3

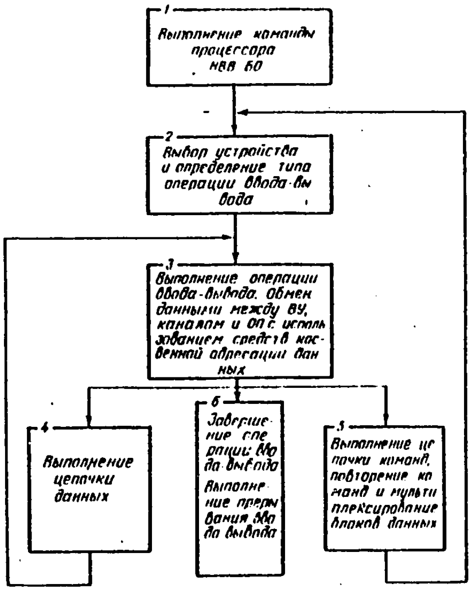
Fig. 24. Replacement of data addresses when indirect data addressing is used in a channel:

1 - address of start of channel program;
2-3 -- address of start of IDAW list;
4-8 -- address of start of data block in IDAW;
9-13 -- data blocks (no more than 2,048 bytes) in main storage

4.3. Operation of Input/Output System

IO operations include data transfers between main storage and external devices.

In accordance with the channel program stored in main storage, an IO channel organizes and executes an IO operation, and controls data exchange between an external device and main storage. The processor establishes communication with a channel at the start of the execution of the operation to issue the initial data for its organization, at the end of the operation to obtain information on the nature of its execution, in certain cases and during execution of the operation to obtain current information or to halt execution of the operation. Fig. 25 shows a block diagram of the IO system operation.


1
Выполнение команды процессора НВВ БО
2
Выбор устройства и определение типа операции ввода-вывода
3
Выполнение операции ввода-вывода. Обмен данными между ВУ, каналом и ОП с использованием средств косвенной адресации данных
4
Выполнение цепочки данных
6
Завершение операции ввода-вывода
Выполнение прерывания ввода-вывода
5
Выполнение цепочки команд, повторение команд и мультиплексирование блоков данных

Key:
1. Execution of processor instruction START I/O FAST RELEASE
2. Selection of device and determination of type of IO operation
3. Execution of IO operation. Data exchange between device, channel and main storage using facility of indirect data addressing
4. Execution of data chain
5. Execution of command chain, command retry and multiplexing of data blocks

Fig. 25. Flowchart of IO system operation

The CPU program initiates IO operations with the instruction START I/O FAST RELEASE (NVVBO) (1).* When the addressed channel is available, not performing an operation previously begun and has no interruption pending, it fetches from location 72 of main storage a CAW that contains the protection key and the starting address of the channel program; otherwise, execution of the SIOF instruction is completed with the condition code set to 3 or 2. When program errors are detected in the CAW received, the channel generates and stores in location 64 of main storage the CSW and ends the SIOF operation with the condition code set to 1.

When there are no errors, SIOF is completed with the condition code set to 0 and the CPU proceeds to execution of the operating commands. The channel, operating in parallel with the CPU, fetches from main storage the first CCW and in the indirect addressing mode the first IDAW, and also generates the real address of the data in the CCW.

When program errors are detected in the CCW or IDAW, in executing the SIOF instruction, the channel generates a CSW with the deferred condition code set to 1 and requests an IO interruption.

*The number in parentheses denotes the block number in the algorithm diagram.

After the CCW and IDAW are fetched successfully from main storage, the channel organizes selection of the addressed external device by sending the address of the device to all devices connected to the interface for this channel (2). If the device with the given address is not in the IO interface, the channel generates a CSW with the deferred condition code set to 3 and requests an IO interruption. A device that recognizes the address connects itself logically to the channel and receives from it the command code specified by the CCW. In response to the command code, the device sends a status byte to the channel. The channel analyzes the status byte received and when necessary may generate a CSW with deferred condition code set to 1 and request an IO interruption.

If the device can execute the IO operation specified by the CCW, the channel and the device start this IO operation and in parallel with the CPU operation organize the data transfer (3). In executing the operation, the write bus of the channel receives the data from amin storage and transfers it through the IO interface to the external device (one or two bytes at a time depending on the mode).

In executing a read operation, the device transfers one or two bytes at a time through the IO interface, and the channel receives and places them in main storage. With each reference to main storage, the channel changes in the CCW the data address field and byte count field and analyzes their value.

If the data address in the CCW is a multiple to a 2,048-byte block boundary and the indirect addressing mode is specified, the channel fetches from main storage the next IDAW, generates the new real data address in the CCW and in the absence of errors continues the IO operation (3). IO operations are possible that consist in transfer of data in several areas of storage. In this case, a CCW chain is used, each of which indicates an area of storage for the operation specified at the start of the chain.

If the byte count in the CCW equals zero and the data chaining mode is specified, the channel fetches the next CCW from main storage. When operation with indirect addressing is required, the channel also fetches the first IDAW from main storage, generates the real data address in the CCW and continues the same IO operation under control of the new CCW (4). If program errors occur in receiving the CCW's or IDAW's, the channel ends the IO operation abnormally and receives from the device the status byte describing the end of the operation.

If the byte count in the CCW is zero and the data chaining mode is not specified, the channel completes the IO operation, receives the status byte from the device and analyzes it. If the values of the device status byte specify the command retry mode, the channel fetches from main storage the preceding CCW and organizes the command retry mode (5), similar to command chaining. In the process, the channel again selects the device (2) without CPU participation.

Command chaining is used for transferring several data blocks through a channel or when sequential execution in one device of a series of different operations is required.

If command chaining is specified in the CCW and the value of the status byte permits execution of this chain of commands, the channel and device determine the necessity of organizing the data block multiplexing mode. In organizing the block-multiplex mode, the channel fetches from main storage the next CCW and activates the chain of commands for another, higher priority device (5). In the process, work with the previous device is halted temporarily and the necessary control information is stored in the subchannel.

If the block-multiplexing mode is not specified, the channel fetches from main storage the next CCW and continues operation on the chain of commands with the same device (5). If command chaining is not specified in the CCW or the status byte does not permit execution of it, the channel generates the CSW and requests an IO interruption (6).

During execution of the procedure for following up an IO interruption, the channel stores the CSW in location 64 in main storage, and the address of the channel and device in the PSW register. After completing the procedure, the CPU replaces the PSW and starts execution of the program for processing the IO interruption (6). At this time, the channel is available for receiving a new CPU instruction or for receiving requests for service from a device.

By the instructions CLEAR I/O, HALT I/O and HALT DEVICE, the CPU can stop execution of a previously started operation, and by the instructions TEST CHANNEL and TEST I/O, can test the status of the entire IO system, including the addressed channel and device.

At the end of channel program execution, the signals describing the halt of the IO operation are sent to the main program either through an IO interruption (6) or during execution of the CPU instruction TEST I/O.

In any case, these signals cause storage of the CSW which contains information pertaining to the IO operation being performed.

4.4. External Storage Units

External Storage in the Unified System of Computers. Computer throughput and computing capabilities to a considerable extent are determined by the composition and characteristics of its storage. The main function of external storage consists in substantially extending the amount of computer storage for storing the main bulk of data, intermediate computing results and the necessary software. In the largest computing systems, external storage is measured in tens and hundreds of gigabytes compared to the megabytes of main storage.

In contrast to main storage which can be represented by a specific block (or blocks) made with particular physical elements (ferrite cores, films, integrated circuits, etc.), it is rather difficult to designate external storage with the same definiteness. External storage is more properly thought of as the aggregate of media with information, as a library of these media in which by using particular devices (external storage units, etc.) any independent amount of information is removed and sent to computer main storage or received from it and recorded for storage on a suitable medium.

From this point of view, external storage units (VZU) are functionally similar to IO devices and organization of operating with them is similar to the organization of operation of UVV [IO devices].

The external storage for modern computers and complexes is built on the principles of magnetic recording on movable media in the form of tapes, disks and drums covered with magnetic material.

Additional requirements for the development and operation of external storage units emerged with the creation of the Unified System of Computers:

to provide for interchangeability of data media. While this requirement was met earlier within the limits of a single computer, for the Unified System interchangeability is required between different computers located not only in diffe:·.nt computer centers, but even in different countries. This led to standardization of the main parameters of data media: electrical (signals, resolution), physical (length, width, thickness, diameters), recording methods, recording densities, recording formats and organization of a service for standards;

to standardize communications between computers, control units and storage units, i.e. to introduce interfaces at various levels. This made it possible to make use of the entire pool of external storage units and the different computers in the Unified System and to put into the composition of computers the new devices as they were developed and assimilated in production;

to increase the capacity of external storage which, as a rule, in modern computers is several hundreds of megabytes and more; and

to improve technical characteristics: data exchange rate, information access time, reliability and others.

The efforts on external storage units made within the framework of the YeS EVM-1 and -2 made it possible to solve these problems to a large extent.

Types and Characteristics of External Storage Units. It its composition, the existing pool of external storage units is far from uniform and includes devices with various technical characteristics. Because of mutual contradictions, the main parameters--capacity of an individual medium, data exchange rate, specified data access time--cannot assume the desired values at the same time. Therefore, in each external storage unit these contradictions are resolved in accordance with particular needs.

The classification of units used in various combinations in computer external storage is shown in fig. 26.

A structural feature is the availability of two types of units: direct access units (disks or drums) and sequential access units (tape). The former are considerably more complex than the latter in design and technology of manufacture, and storing data in them is more expensive. However, their relatively fast access time ensures their mandatory use in external storage of modern computers in "online modes," as well as further improvement and development.

In the YeS EVM-2, this group is represented by these subsystems: YeS-5066/YeS-5566, YeS-5050/YeS-5551, YeS-5052/YeS-5551, YeS-5061/YeS-5561 and the YeS-5061/YeS-5568.

Using the disk subsystem based on 100M-byte units (the YeS-5066/YeS-5566) has made it possible to substantially increase computer external storage capacity and expand computer functional capabilities. A number of technical problems was solved in

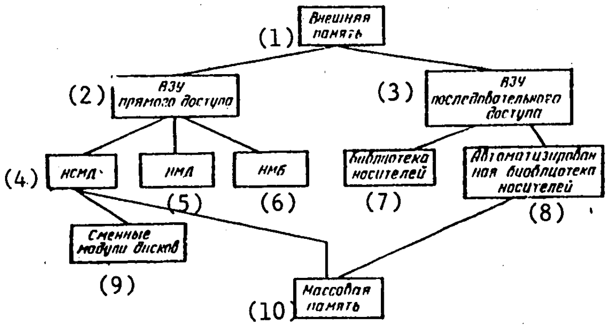


Fig. 26. Classification of external storage units

Key:

1. external storage
2. direct access units
3. sequential access units
4. NSMD [removable disk storage unit]
5. NMD [disk storage unit]
6. NMB [magnetic drum storage unit]
7. library of media
8. automated library of media
9. removable modules of disks
10. mass storage

in developing this storage unit: development of thinner disk magnetic coating, obtaining stable floating of the magnetic head at a small height above the disk surface, development of a read/write path with high resolution and development of a sensitive servo system. The YeS-5566 control unit also has important advantages over earlier used devices of the same type.

External sequential access storage units are now represented by magnetic tape storage units (NML). Despite their relatively slow average information access time, they are firmly secure in the composition of computer external storage because of the low cost and convenience of storing information recorded on magnetic tape. Large computer centers have libraries of tens of thousands of reels of magnetic tape with 10^{12}-10^{13} bits of stored information. All source data and results of computations, irrespective of which external storage units they appear on in processing, are recopied for storage and "dumped" onto magnetic tape.

The most common magnetic tape storage units operate with a data recording density of 32 and 63 bits/mm and the data transfer rate ranges from 32 to 315 kilobytes/second. In the Unified System of Computers, this group consists of the YeS-5012, YeS-5017, YeS-5025, YeS-5002, YeS-5003, modifications of them and subsystems based on them.

Characteristics of the main external storage units in the Unified System of Computers are given in tables 15 and 16.

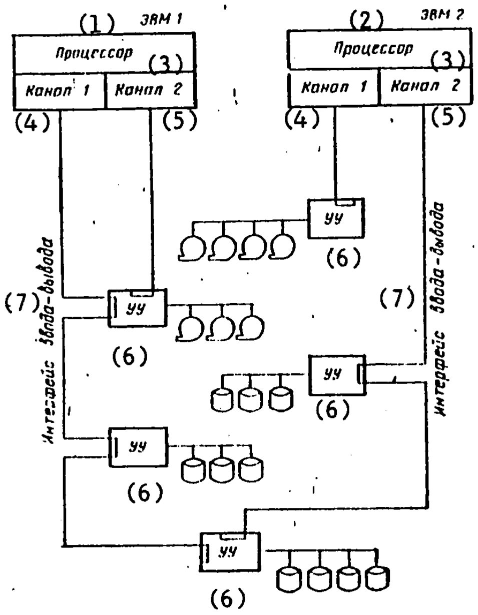
Table 15

		Basic Technical Characteristics		
Model number of removable disk storage units	Developed by	Information capacity per spindle, megabytes	Data transfer rate, kilobytes/second	Mean access time, ms
YeS-5050	USSR	7.25	156	80
YeS-5052	Bulgaria	7.25	156	60
YeS-5056M	USSR	7.25	156	90
YeS-5061	Bulgaria	29	312	75
YeS-5066	USSR	100	806	45
YeS-5067	Bulgaria	200	806	30
YeS-5080	USSR	200	806	30

Table 16

		Basic Technical Characteristics					
Model number of magnetic tape storage units	Developed by	Reel capacity, megabytes	Recording density, characters/mm	Tape speed, m/sec	Recording Method	Start/stop time, ms	Maximum data transfer rate, kilobytes/second
YeS-5017	USSR	20	8/32	2	NRZ-1	5	64
YeS-5019	Poland	20	8/32	3	NRZ-1	4	96
YeS-5022	CSSR	20	8/32	4	NRZ-1	3	128
YeS-5025	USSR	20/40	32/63	2	NRZ-1/PE	5	64/126
YeS-5612	Bulgaria	20/40	32/63	3	NRZ-1/PE	3	189
YeS-5002	GDR	20/40	32/63	3	NRZ-1/PE	3	96/189
YeS-5003	Bulgaria	20/40	32/63	5	NRZ-1/PE	2	160/315
YeS-5004	CSSR Bulgaria	20/40	32/63	2	NRZ-1/PE	4	64/126
YeS-5027	Bulgaria	120	63/246	3	PE/GC	1	750

Organization of Operation of External Storage. The decisive factor in organizing the operation of external storage units is the attempt to maximally decrease the effect of the parameters of these devices on the rate and reliability of operation of the individual computers and the systems as a whole. In the process, great importance is attached to questions associated with ensuring the capability of changing the set of equipment in complexes already in operation, modernization and connection of new, more improved devices, convenience of operation and others.



Key:
1. computer 1
2. computer 2
3. processor
4. channel 1
5. channel 2
6. UU [control unit]
7. IO interface

Fig. 27. Diagram of connection of external storage units

In the Unifies System of Computers, the following scheme has been implemented for connecting external storage units to a machine: processor--channel--control unit (UU)--external storage unit. In this chain, several channels can be connected to the processor, several control units to a channel, and each control unit can serve a certain number of external storage units. Thus, dozens of external storage units can operate with a machine in accordance with a user's desire.

This scheme, supplemented by unified principles of software and one type of connection between channels and control units and the latter with external storage units, allows organizing the operation of external storage units, while meeting the requirements for their functioning within the computer structureand for changing the external storage equipment set upon user's requests.

Interaction of external storage units with a machine is similar in its organization to the interaction of IO devices. Therefore, from the point of view of program processing, external storage units are part of the external device group in common with them.

Having relatively high speeds, external storage units are connected to selector or block-multiplexer channels and while monopolizing all facilities of the channel do not permit other external devices to be connected through it to the machine. The channel and control unit communicate through a standard IO interface.

In addition to the main, standard IO interface, there is a so-called minor interface between the control unit and an external storage unit which is a set of lines and signals. The set and function of the minor interface communication lines are largely similar to that of the standard IO interface. But the time relations of signals, since they reflect the external storage unit design features, vary for different groups of equipment. Therefore, the minor interface is standardized within the bounds of the same type of device (for example, magnetic tape storage units of a particular class).

The channel--control unit--external storage unit communication system permits several versions of mutual connection.

1. Channel--control unit--external storage unit. This is the simplest connection scheme.

2. An external storage unit may be connected through a special control switch to two or more control units, and the latter themeselves to one or more channels. With such a connection, the possibility of access to a specific external storage unit is ensured while bypassing a busy or malfunctioning control unit. The flexibility and reliability of a system are enhanced, but a new unit, the control switch, appears in the circuit in the process.

3. A control unit has independent circuits for connection to two channels or a switch is used for the same purpose. This method allows two channels of one machine to operate with a specific external storage unit, and also allows several machines access to one external storage unit, i.e. allows providing a common extent of external storage for several computers joined into a system.

4. The most complex method of connecting external storage units, both in the hardware and the software sense. It includes all features of the methods discussed above.

Various schemes may be used for various types of external storage units with regard to their functions and purpose in creating intricate computer complexes and systems. In typical computer configurations in the Unified System, the third version is now used mainly for connecting external storage units.

The scheme for organization of interaction between external storage units and the machine considered and adopted in the Unified System has the folowing advantages:

convenience and ease of programming with the presence of a large variety of external storage units, since the basic principles of organization are the same and device-independent;

flexibility of changing the mix of external storage units in accordance with the user's desire (variety of configurations of external storage), as well as the capability of increasing within broad limits the composition of external storage units without any changes in the processor and channels;

the capability of maintaining system operating capability when units malfunction with several paths for access to external storage units;

the capability of operation of several computers joined in a system with a common extent of external storage; and standardization of control unit interface to a channel which permits modernization and development of new external storage units and connection of them as part of a computer system as they become available.

Prospects for Development of External Storage Units. An active quest is now in progress to find ways of applying new physical principles in the storage of large amounts of data by using laser technology, holography, optical and magnetic-optical effects.

On the other hand, considerable progress in the development of microelectronics, especially in the field of integrated storage circuits, charge-coupled devices, cylindrical magnetic-bubbles and others, already makes it possible today to develop storage units competitive with equipment of lower level of external storage units in terms of amount of data storable, throughput and reliability.

Despite this, the capabilities of traditional magnetic tape units are far from exhausted.

Tape units are being improved in two directions: development of large-capacity units and improvement in reliability and operating characteristics.

Efforts in the first direction should result in units with a higher recording density, including through new methods of coding, that will permit a several-fold increase in data capacity of a reel of magnetic tape and data transfer rate.

The second direction should result in a considerable extension of the service life of the most critical unit elements, the magnetic heads, as well as elimination of direct operator contact with magnetic tape thanks to the introduction of an automatic tape loading mechanism.

A natural development of removable magnetic disk units is the development of units with an increased data capacity of a pack which in the near future will be 200 and 300 megabytes. However, development of such devices even on a large scale will exacerbate the problem of ensuring interchangeability of removable packs.

In connection with this, a promising direction in the development of units with removable disk packs is the development of devices with removable modules in which sets of disks are combined with magnetic heads. This design solution eliminates the complex and laborious operation of alignment of magnetic heads, and substantially increases the capacity of the devices through a lower level of floating of magnetic heads on the surface of a disk. The one-time amount of data storable in such a subsystem of external storage may reach 20 gigabytes. These devices substantially increase reliability and facilitate system operation.

The next step in development of computer external storage is the so-called "mass" storage that combines the basic positive qualities of both tape and disk units.

The capacity of mass storage units reaches 10^{12}-10^{13} bits, data transfer rate is 800-900 kilobytes/second and maximum access time is no more than 15 s.

With virtual organization, for the user, this storage in parctice becomes online direct access storage with very high capacity and corresponding time characteristics.

Structural mass storage is represented in the form of a two-level hierarchy: automated storage and an executive buffer unit that is linked directly to computer main storage. Information in the library is kept on sections of magnetic tape and by requests is sent to or removed from the executive level through the write/read apparatus. The executive level, the parameters of which are determined by throughput, number of logical addresses, capability of virtual organization and other system indicators, consists of disk units with a capacity of 100-200 megabytes or of semiconductor storage units.

Optomechanical external storage units are arousing great interest. Their development has become possible thanks to the evolution of laser technology. Optomechanical external storage units have a number of advantages over electromechanical; the most important are: high surface density of data recording (on the order of 10^4-10^5 binary-coded characters/mm^2), high data transfer rate and high data capacity of the unit.

By method of data recording, optomechanical external storage units are divided into two main groups: discrete storage units with bit-by-bit recording and holographic storage units. The former are realized with media made in the form of disks or bounded segments of tape, and the latter by the use of tape media.

A substantial improvement in computer characteristics can be expected in the near future when these directions of development of external storage and other solutions on its improvement are realized.

4.5. Input/Output and Data Preparation Units

Expansion of the nomenclature and improvement of the parameters of data IO devices are major aspects in the development of the IO system that largely determine the efficient and versatile use of computers. Operation of these devices is supported by a complex of data preparation devices and development of them has also received serious attention.

Despite the fact that a rather broad nomenclature of IO and data preparation units was developed for the YeS EVM-1 and that these devices are being successfully applied in the YeS EVM-2 too, efforts were continued both in the direction of improving the parameters of the devices developed earlier and in the direction of developing new units.

IO and data preparation units were evolved in the following directions:

enhancement of rate of operation and reliability;

development of combined and grouped devices;

development of problem-oriented IO complexes; and

development of data preparation units oriented to magnetic media.

Given below are the major characteristics of the basic IO devices that, on the one hand, confirm the distinguishing features of the YeS EVM-2 devices, and on the other, describe the new capabilities of their application.

The rate of operation and reliability of the IO devices (the YeS-6019 card reader (USSR), alphanumeric printers YeS-7037 (USSR) and YeS-7039 (CSSR), typewriters and printers YeS-7076 (Poland), YeS-7181 (CSSR), YeS-7183 (GDR), YeS-7186 (Poland) and YeS-7188 (Poland)), as a rule, were improved thanks to application of the new principles of their operation.

Applied in the YeS-6019 card reader were feeding of cards by the narrow side in a horizontal plane and a vacuum feed system; thanks to this, the card input rate was 1200 cards/minute. Data is read from cards having 45 and 80 columns. Also, the card usage factor was increased several fold with the use of the vacuum feed system.

The principle of tape or chain printing which makes use of removable type began to be used for line impact printers.

The use of the continuously moving print chain has allowed increasing the printing rate, providing for high repairability through convenient and easy replacement of both the individual sets of characters without additional mechanical adjustment and the entire print carrier, and also using perforated paper with various formats.

The YeS-7037 prints 132 characters per line at 800-1000 lines per minute. The YeS-7039 prints 160 characters per line at 1200 lines per minute.

The dot matrix principle is the basis for the serial printers and their mechanisms (the YeS-7076, YeS-7181, YeS-7183, YeS-7186 and the YeS-7188) which provides a printing rate of 180, 150, 100, 180 characters/second and 200 lines/minute, respectively.

The YeS-7902 (GDR and CSSR) and YeS-7903 (USSR) perforated tape IO stations with a read rate of 1000 and 1500 characters/second and a punch rate of 100 and 150 characters/second respectively were developed in place of individual perforated tape devices. The introduction of these data stations allows reducing equipment cost and power consumption and enhancing operating convenience.

Efficient use of computers in time-sharing modes is possible only with man-machine interactive facilities. In connection with this, much attention was paid in developing the YeS EVM-2 to developing cluster units for data input from a screen and output of it to a CRT screen that allow online communication with a computer.

Added to the YeS-7061 and YeS-7063 (Hungary), the YeS-7063 (CSSR), the YeS-7064 and YeS-7096 (USSR) alphameric displays in use in the YeS EVM-2 were the YeS-7920 alphameric cluster display station, which was developed with the participation of specialists from the VNR [Hungarian People's Republic], the GDR, the PNR [Polish People's Republic], the USSR and the CSSR, and the YeS-7905 (USSR) cluster display terminal for input/output of graphic information.

The YeS-7920 is designed for operation as an operator's console and as an information display unit in various operating modes, for which several modifications have been provided:

local cluster: YeS-7920-00, YeS-7920-01;

remote cluster: YeS-7920-10, YeS-7920-11; and

remote single: YeS-7920-20, YeS-7920-21.

Table 17. Basic Technical Characteristics of Individual Units in the YeS-7920 Display System

Unit Устройство	Model Шифр (1)	Data Transfer Скорость обмена информацией Rate	(2) Код обмена	(3) Емкость буфера, знаков	(4) Формат экрана или формат печати	(5) Потребляемая мощность, ВА	(6) Масса, кг	(7) Интерфейс связи
Cluster Control Групповое устройство управления Unit	ЕС-7921-00	600, 1200, 2400, 4800, бит/с (8)	КОИ-7 KOI-7	480	—	350	200	S2 joint Стык С2
	ЕС-7921-01	600, 1200, 2400, 4800, бит/с (8)	КОИ-7 KOI-7	1920	—	350	200	
	ЕС-7922-00	Не менее (9) 250 Кбайт/с	DKOI ДКОИ	480	—	350	200	YeS EVM Ввода-вывода ЕС ЭВМ IO
	ЕС-7922-01	Не менее (9) 250 Кбайт/с	DKOI ДКОИ	1920	—	350	200	
Display Дисплей	ЕС-7925-00	600, 1200, 2400, 4800, 9600, бит/с (8)	КОИ-7 KOI-7	480	(10) 12 строк по 40 знаков	450	70	S2 joint Стык С2
	ЕС-7925-01	600, 1200, 2400, 4800, 9600, бит/с (8)	КОИ-7 KOI-7	1920	(11) 24 строки по 80 знаков	450	70	
	ЕС-7927-00	760—864, Кбит/с (12)	КОИ-8 KOI-8	480	(10) 12 строк по 40 знаков	300	45	Последовательный 14-битовыми словами (13)
	ЕС-7927-01	760—864, Кбит/с (12)	КОИ-8 KOI-8	1920	(11) 24 строки по 80 знаков	300	45	
Printer Печатающее устройство	ЕС-7934-00	Не менее (14) 66 знаков/с	—	480	80 знаков (15)	650	100	
	ЕС-7934-01	(14) Не менее 66 знаков/с	—	1920	64 знака (15)	650	100	
	ЕС-7934-02	(14) Не менее 66 знаков/с	—	—	40 знаков (15)	400	40	(16) Параллельный для механизмов печати

[Key on following page]

Table 17. [Continuation]

Key:

1. [Russian "EC" = English "YeS"]
2. interchange code
KOI [information interchange code]
DKOI [decimal information interchange code]
3. buffer capacity, characters
4. display or printer format
5. power consumption, VA
6. weight, kg
7. communications interface
8. bits per second
9. no less than 250 kilobytes/second
10. 12 lines, 40 characters each
11. 24 lines, 80 characters each
12. kilobits per second
13. serial by 14-bit words
14. no less than 66 characters/second
15. characters
16. parallel for print mechanisms

The YeS-7920 cluster display stations enable a large number of operators to interact with computers both in direct proximity to the machine and at a large distance away from it (remote). Cluster local units are connected to the computer directly through the selector, byte-multiplexer and block-multiplexer channels by the YeS-7922-00(01) cluster control unit. In doing so, the YeS-7920 peripherals, the YeS-7927 display and YeS-7934 printer (no more than 32 units), can be connected to the cluster control unit at a distance up to 1200 meters.

The YeS-7920-10(11) cluster remote units, which contain the same peripherals and have the same characteristics, are connected to selector, byte-multiplexer and block-multiplexer computer channels through a data transmission multiplexer, modems and dedicated telephone lines by using the YeS-7921-00(01) cluster control unit.

The YeS-7920-20(21) single remote units, which include the YeS-7925-00(01) display and the YeS-7934-02 printer, are connected to the same channels both directly and through a data transmission mutliplexer in the case of remote use.

The main technical characteristics of the individual units in the YeS-7920 display system are given in table 17.

The YeS-7905 cluster unit is designed for input/output of alphanumeric and graphic information in the real-time mode with automated design of documentation.

The YeS-7905 unit includes:

the YeS-7565 cluster control unit; and

the YeS-7065 display console which consists of a display and plotting unit for input of graphic information with a 400 X 400 mm working field.

Each YeS-7565 cluster control unit can be connected to two to four YeS-7065 display consoles at a distance of no more than 500 meters. The main technical data for the display are given in table 18.

Problem-oriented complexes which have begun to be developed in place of individual two-coordinate graphic units allow broad functional capabilities of the Unified System. Problem-oriented complexes are built with the application of minicomputers and function both autonomously and as part of the models of the Unified System. Four such complexes have now been developed: the YeS-7907 (CSSR), the YeS-7908 (USSR), the YeS-7941 (CSSR) and the YeS-7942 (CSSR). The first two complexes are designed for IO of graphic information, and the latter two for performing graphic design operations and technological production operations, respectively.

Table 18. Basic Technical Data for the YeS-7065 Display Console

Characteristics	Parameters
working field, mm	250 X 250
number of addressable raster units	1024 X 1024
character code	DKOI [decimal information interchange code]
number of brightness levels	2
number of characters in a line:	
basic size	74
extended size	49
number of lines for characters	
basic size	52
extended size	35
maximum number of characters on the screen	2100

Data preparation units were developed further in the YeS EVM-2. These include:

the YeS-9004 (Bulgaria) unit for preparing data on magnetic tape with a recording density of 32 characters/mm;

the YeS-9111 and YeS-9112 (CSSR) individual units for preparing data on floppy disks for one or two work places;

the YeS-9113 (Bulgaria) unit for copying data from floppy disk to magnetic tape with a width of 12.7 mm;

the YeS-9003 (Bulgaria) multiconsole system for preparing data on magentic tape with 16 operator consoles;

the YeS-9006 (Bulgaria) cassette system for preparing data on magnetic tape with eight operator consoles; and

the YeS-9150 (Poland) cluster system for preparing data on magnetic tape with up to 32 operator consoles.

The new units in this class are oriented mainly to magnetic media: standard and cassette magnetic tapes and magnetic disks.

Chapter 5. Means of Organization of Computer Systems

A distinguishing feature of phase-two computers in the Unified System is the building on their basis of different computer systems that offer the user new capabilities and are characterized by high efficiency.

These systems include first of all those with virtual organization of main storage.

In shared-use systems, the Unified System is supported by software and hardware for data teleprocessing facilities (facilities for organizing the mode of time sharing of the processor, local and remote terminals with the capability of independent access to a computer, data transmission equipment, data transmission multiplexers, etc.).

Complexing facilities included in the models for building computer systems meet the increased requirements for throughput and reliability and join computers at the levels of common main and common external storage.

There are two types of complexing of computer systems, multiprocessor and multimachine, which differ in the level of computer coupling, software and hardware.

5.1. Multiprocessor Systems

A multiprocessor system presupposes two or more processors, each of which is controlled by a common operating system. Each processor has access to a common main storage and to all or some of the IO devices.

Complexing of several processors at the level of common main storage is supported by facilities of multiprocessor operation or multiprocessing facilities.

Multiprocessing facilities include a permanently allocated area assigned to each processor in a common main storage, prefixing, interprocessor signaling and time-of-day-clock synchronization.

The structure and purpose of the fields of the permanently allocated area of storage are discussed in chapter 2.

Prefixing. Since each processor in a multiprocessor system has its own permanently allocated storage area, these areas are allocated within the bounds of the common extent of main storage by using the hardware mechanism of prefixing, i.e. the storage area with addresses 0-4096 is displaced by the value of the prefix assigned to each processor.

The mechanism of translating addresses by using a prefix allows achieving the minimum mutual effect of the processors operating with a common extent of main storage.

All addresses subject to translation by the prefixing mechanism are referred to as real. Storage addresses not subject to this translation and all translated addresses, whether or not they are changed, are referred to as absolute. As a result of the computation of the absolute address, real addresses 0-4095 are increased by the value indicated in a 12-bit prefix register. Two new instructions have been provided for operating with the prefix in the YeS EVM-2: SET PREFIX and STORE PREFIX. During execution of the first instruction, a 12-bit value located in bits 8-19 of the second operand is stored in the prefix register. During execution of the second instruction, the contents of the the prefix register is stored at the address of the second operand which specifies a 32-bit storage location. In this case, zeros are put into bits 0-7 and 2-31, and the prefix value is placed in bits 8-19.

Prefixing is effected in the multiprocessing mode, does not depend on the control mode and does not alter the address where it is generated. The address is changed by using the prefix in the storage control unit immediately before referencing. For this purpose, the storage control unit contains circuits for analysis of the incoming main-storage reference addresses, as a result of the operation of which the addresses are either translated (table 19) or remain unchanged.

Table 19.

Storage Address Bits	Value of Bits	Actions in Translation
8-19	equal to zero	replaced with the prefix
8-19	equal to the prefix	replaced with zeros
8-19	not all zeros and not equal to the prefix	remain unchanged
20-31	in all cases	remain unchanged

The essence of translating storage addresses can be understood proceeding from the purpose of prefixing: allocation of a permanent directly addressable area of storage of each processor within the bounds of a common extent of main storage.

Interprocessor Signaling. In addition to exchange of large amounts of data or their joint use in a multiprocessor system, there must be direct communication between the processors to exchange control information. This necessity may occur in many cases, for example when a processor has to start or stop another processor, and also stop or interrupt its operation. A special instruction, SIGNAL PROCESSOR, has been provided in the YeS EVM-2 to organize communication on control information between processors, in addition to the instructions READ DIRECT and WRITE DIRECT. This instruction, format RS, specifies the address of the processor with which communication has to be established, the code of the order being called and the result of execution of the order.

The address of the processor with which communication is being established is placed in a general register. The number of this register is specified by the third operand in the field R_3 of the instruction (see fig. 2). The address of each processor is fixed and assigned during assembly of the special switch of the control circuits intended for this.

The order codes are specified in bits 24-31 (see table 20) in place of the address of the second operand specified by fields B_2 and D_2 of the instruction.

Table 20. Orders and Codes for Communications among Processors

Codes	Orders	Order Function
0000 0000	unassigned	
0000 0001	sense	obtain information on status of called processor
0000 0010	external call	generate an external interruption with the external call order at the addressed processor. The code and address of the processor sending the signal are stored when the interruption occurs.
0000 0011	emergency signal	generate an external interruption with the emergency signal order at the addressed processor
0000 0100	start	addressed processor is placed in operating state
0000 0101	stop	addressed processor is placed in stop state
0000 0110	restart	addressed processor performs restart function
0000 0110	initial program reset	effect initial program reset at the addressed processor, i.e. reset of processor and all channels
0001 1000	program reset	program reset at addressed processor, i.e. reset of processor and channels. Differs from preceding in that other control information must be reset in this case
0001 0000	stop and store status	place addressed processor in stop state and store status after that
0010 0000	initial microprogram load [IMPL]	perform program reset and after that start initial microprogram load if provided for in model
0011 0000	initial CPU reset	perform initial reset of addressed processor
0100 0000	CPU reset	perform reset of addressed processor. Differs from preceding in information that must be reset
0101 0000- 1111 1111	unassigned	

In response to the sense order, the addressed processor issues information on its status, stored in the general register designat~d by the R_1 field of the SIGNAL PROCESSOR instruction. The status conditions and their bit positions in the register in which they are stored are given in table 21.

Accepted or rejected orders at the addressed processor by the SIGNAL PROCESSOR instruction are confirmed by storing of the corresponding condition code and status condition by the issuing processor.

Clock Synchronization. Each processor in a multiprocessor system may have its own time-of-day clock or all processors may share a clock. In any case, a processor accesses only one clock. When more than one clock exists in a system, they must be synchronized so that all programs obtain the same time readout.

The clock synchronization facility together with the operating system supervisor ensure that time-of-day clock readouts appear as if one clock were used, irrespective

Table 21. Status Conditions for Indication of Response to an Order

Register Bits	Status Condition	Definition
0	equipment check	malfunction of addrsed processor that affects only execution of this instruction
1-23	unassigned (zeros)	
24	external-call pending	addressed processor has not yet completed processing of interrupt with external call because of processing of previously issued SIGNAL PROCESSOR instruction
25	stopped	addressed processor is in stopped state
26	operator intervening	operations from console in progress at addressed processor
27	check stop	addressed processor is in check-stop state
28	not ready	required microprogram not loaded at addressed processor
29	unassigned (zero)	
30	invalid order	addressed processor received invalid order or order has not been implemented
31	receiver check	malfunction in addressed processor that may have affected generation of other status conditions

of whether a clock is used in one or more processors. The operation of them consists in the fact that they simultaneously change the contents of low-order bits of all clocks. In addition, they ensure that a clock is started by another. Lack of synchronization is considered a disruption to the operation of a multiprocessor system and causes an external interruption.

These multiprocessing facilities are governed by the principles of operation of the YeS EVM-2 and presuppose the capability of organizing multiprocessor operation. To implement this capability in each specific model, it is necessary to provide for equipment that provides access by a processor to the main storage of another, and vice versa, i.e. access by each processor to a common extent of main storage. Organization of this access varies and specific implementation of it largely determines the efficiency of operation of a multiprocessor system. By topology of communications between the main resources, there are now several types of organization of multiprocessor systems. The main ones are:

systems with a common bus;

systems with a cross switch; and

multiple bus systems.

Organization of multiprocessor systems with a common bus is based on the principle of connecting all system resources through a common bus. Information is transferred between processors, IO channels and independent blocks of main storage through the common bus with time sharing.

Such organization is noted for simplicity of implementation, flexibility, convenience of adding on additional resources, standardization, simplicity of interfaces of the individual devices and low cost of switching equipment. Also, because of time sharing of transfers, the problem of conflicts and protection of information is essentially solved since each unit monopolizes the bus from start to end of the transfer.

At the same time, such systems have a number of shortcomings, among which the main ones are long time delay in data transfer and relatively low system reliability.

The data transfer delay increases as the load increases, i.e. as additional units are connected. This is due to time-sequential transfers and the need of additional time for identification of the unit establisging communication and for organization of communication. The low system reliability compared to that of other types of systems stems from the operation of the common bus being provided for by one control unit.

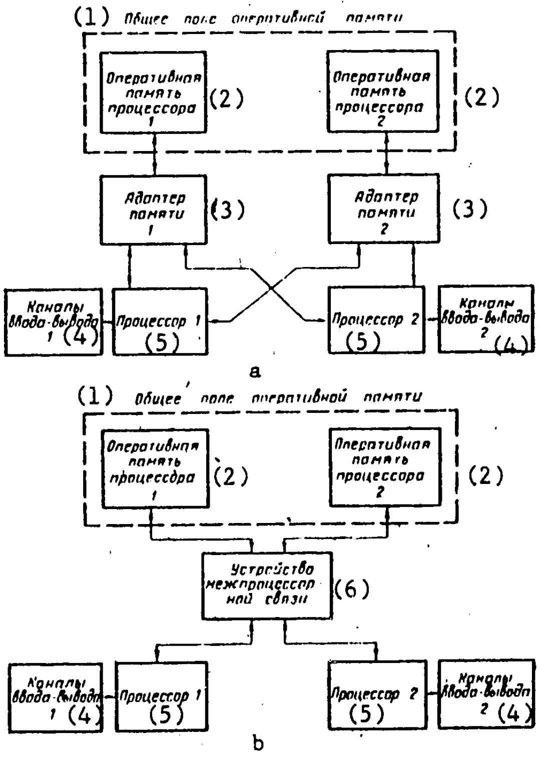
It should be noted that these parameters may be improved by dividing the common bus into two or more and such systems have already emerged in design practice.

Organization of systems with a cross switch is based on a switch matrix with spatial division that allows connecting uniport devices with each other in any combination. In this system, the transfer rate is higher than in the common bus system since the processors can access storage over several paths. The system with a cross switch is somewhat less flexible than the common bus system, yet this system can grow to a sufficiently large extent and this is governed by the amount of equipment of the switching matrix. The switching matrix device is separate functionally from the main equipment and may be used to interface IO channels with a processor that has to communicate with a peripheral at a given time. Conflicts arising between processors during reference to main storage are resolved by special apparatus placed in the swicthing matrix. The relatively high functional complexity and cost of the switching equipment for this system should also be noted. To organize operation on the order of 10 processors with a common extent of main storage, the amount of apparatus of the switching matrix is commensurate with the amount of equipment of one processor.

For their organization, multiport multiprocessor systems make use of main storage units with multiple ports or units for interprocessor communications that allow expanding the number of ports to storage. Just as the preceding system, multiport multiprocessor systems have a high data transfer rate and high reliability through the capability of organizing several paths of access to main storage. Also, there is less switching equipment and consequently lower cost. At the same time, the add-on capability in this system is limited to the number of main storage ports.

Selection of a particular organization of multiprocessor systems is governed both by their purpose and the structure of the components, especially the processor.

The architecture of the Unified System of Computers is designed both for building both general-purpose large-scale and economic single-processor models and for building multiprocessor systems that maintain these qualities. The logic structure of the processor, channels and main storage of the Unified System makes it possible to organize, as a rule, multiport multiprocessor systems and systems with cross


(1) Общее поле оперативной памяти
Оперативная память процессора 1
(2)
Оперативная память процессора 2
(2)
Адаптер памяти 1
(3)
Адаптер памяти 2
(3)
Каналы ввода-вывода 1 (4)
Процессор 1 (5)
Процессор 2 (5)
Каналы ввода-вывода 2 (4)
a
(1) Общее поле оперативной памяти
Оперативная память процессора 1
(2)
Оперативная память процессора 2
(2)
Устройство межпроцессорной связи
(6)
Каналы ввода-вывода 1 (4)
Процессор 1 (5)
Процессор 2 (5)
Каналы ввода-вывода 2 (4)
b

Key:
1. common extent of main storage
2. main storage for processor
3. storage adapter
4. IO channels
5. processor
6. device for interprocessor communication

Fig. 28. Dual-Processor System with Common Extent of Main Storage

switching. These systems may be built, for example, on the base of models YeS-1035, YeS-1045, YeS-1060 and YeS-1065. A common bus is used in the YeS-1015 computer; a multiprocessor system can be built that enables data transfer through this bus with time sharing.

Two possible configurations for a dual-processor system with a common extent of main storage based on Unified System models are shown in fig. 28.

Shown in (a) is a multiport multiprocessor system in which multiport storage adapters are used for switching of data buses. The storage adapters can be physically placed in both the main storage unit and in the processor. In both cases they perform the same role: they expand the number of main storage ports and resolve conflicts between the processors during reference to main storage.

Shown in (b) is a multiprocessor system in which a device for interprocessor communication (UMS) is used for processor communication with main storage. This functionally independent unit may be implemented in the form of both an adapter that expands the number of main storage ports and a switching matrix. An interprocessor communication unit affords great system flexibility and at the same time allows releasing the computer from superfluous equipment when the single-processor version is used.

The logic structure of the processor imposes specific requirements on the organization of data transfer control in a multiprocessor system. In the YeS EVM-2 processors, especially rigorous requirements are imposed by buffer storage, protection key storage and the apparatus of dynamic address translation in organizing virtual storage. The presence of these facilities, located in the storage control unit, requires an exchange of control signals between the two processors, by which specific actions are performed in the storage control unit for one processor intiated by the other. These actions include:

establishing of invalidity of data in buffer storage;

executing operations with storage keys; and

establishing invalidity of information in the translation-lookaside buffer.

Since in a multiprocessor system, main storage is common and buffer storage of each processor contains data that are copies of data of some areas of main storage, when the data in these areas are altered by a processor, the data must be deleted in the buffer storage of the other processor. These actions are performed in a manner similar tot he actions for establishing the invalidity of data in buffer storage in writing data to main storage by IO channels in a single-processor system (see section 3.2).

Main storage protection keys may be placed in both the storage unit itself and in the processor. In the first case, no special difficulty occurs in organizing operation with protection keys in a multiprocessor system. But quite often, computer structure organization requires placing storage protection keys in the processor. In this case, in organizing a multiprocessor system, the operations with storage keys, SET STORAGE KEY, INSERT STORAGE KEY and RESET REFERENCE BIT, must be executed in all processors. In executing these modes, real addresses, storage protection keys and operation flags must be exchanged between all processors.

To speed up the mode of translation of a virtual (logical) address into a real address, a translation-lookaside buffer is used in the processor. Each location of the translation buffer contains the logical address of a page, the real address of the page corresponding to it and the storage protection key for this page. Full translation (using tables) is performed only when the first reference is made to some page and the result of the translation is placed in the translation buffer. Subsequent references to this same page do not need full translation since the real address for the corresponding logical address and the storage protection key can be fetched from the translation buffer.

In a multiprocessor system, if a page-table entry is changed or storage protection key of some page is changed in a processor, the information in the corresponding location of the translation buffer in which the real address of this page is found must be deleted in all processors. For this purpose, exchange of control signals between all processors must also be organized.

The capacity of common main storage in multiprocessor systems based on Unified System models varies but is no more than 16M bytes. Also, it must have the capability of being changed in event some logical block of storage malfunctions. The necessary control facilities have been provided to specify the required capacity of common storage and to reconfigure storage in event a block malfunctions. These controls are on the computer control consoles or on special panels for the multiproeessor system.

Thus, these multiprocessing facilities and hardware facilities for organizing the common extent of main storage in conjunction with the operating system allow building multiprocessor systems with efficient organization on the basis of Unified System computer models.

5.2. Multimachine Systems

Multimachine systems are basically two-machine computer complexes (VK) based on series computers that are united for joint operation by one or more complexing facilities. In contrast to multiprocessor systems, multimachine systems have no common extent of main storage and each computer that is part of the complex is controlled by its own operating system.

Computer complexes are intended for solving a broad range of information, computing and administrative problems in the modes of multiprogram batch processing, real time and shared use through communication channels. They are used as the central link in data processing at the top levels of automated management systems as well as in systems for acquiring, storing and processing large files of information.

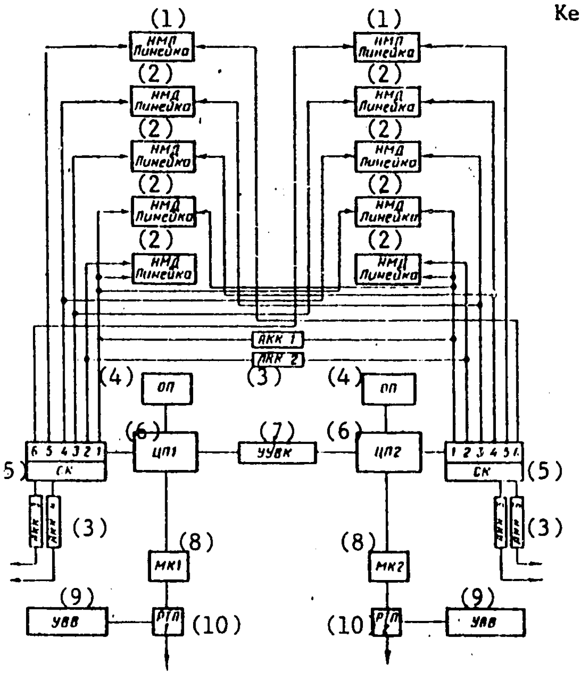
Within the Unified System of Computers, there are now the VK-1033, the VK-2R-35, the VK-2R-45 and the VK-2R-60 computer complexes built, respectively, on the base of the YeS-1033, YeS-1035, YeS-1045 and YeS-1060 computers.

Configuration of Computer Complex. Each of these computer complexes has its own complement of equipment in accordance with its purpose and makes use of various complexing facilities; nevertheless, they are all built on the same principle that allows forming a generalized configuration of a computer complex based on Unified System models (fig. 29).

Each computer in the complex has a central processor (TsP), main storage (OP), one or more byte-multiplexer channels (MK), a group of selector or block-multiplexer channels, a set of IO devices linked to a multiplexer channel through a logic repeater (RTL), a set of external storage units and a set of channel-to-channel adapters (AKK). Each line of external storage units consists of a control unit and the storage devices served by it.

The processors communicate through the computer complex control unit (UUVK), and the main storage units for the first and second computer through the chain of the first computer's IO channel, the channel-to-channel adapter and the second computer's IO channel and vice versa.

In the configuration of a computer complex, one can allot external storage units to be accessed by just one computer and units to be accessed by both computers in the complex forming a common extent of external storage. In fig. 29, the common extent of external storage is formed by the group of lines of magnetic disk storage units and the line of magnetic tape storage units in each computer. In doing so, two



Key:
1. line of NML [magnetic tape storage units]
2. line of NMD [magentic disk storage units]
3. AKK [channel-to-channel adapter]
4. OP [main storage]
5. SK [selector channel]
6. TsP [central processor]
7. UUVK [computer complex control unit]
8. MK [multiplexer channel]
9. UVV [IO device]
10. RTL [logic repeater]

Fig. 29. Generalized configuration of computer complex based on models of Unified System of Computers

lines of the group of common magnetic disk storage units of each computer are used as alternative access for the unlike selector channels of both computers, and one line for like channels.

Each computer usually uses its own external storage units to store the operating system and the programs used often and most important for operation. The common units are used to store the input batch of jobs and data files containing both information for common use and information needed to organize the interaction of both computers in the complex and the backup to replace a computer that has gone down.

One group of channel-to-channel adapters is used for intermachine communication within the complex effected through an IO interface of like channels of each computer, and two other groups afford communication for the first and second computer, respectively, with other similar computer complexes.

This structure of multimachine systems stemmed from the need to meet the main requirements for such systems: ensure continuous functioning of the computer complex with high reliability. Computer complexing raises the reliability of a computer complex not only through the backup to the main equipment that allows switching computers during failures, but also through backup storage of vital data in computer storage and the magnetic disk and tape storage units used jointly by both computers. Computer complexing also allows achieving higher throughput than the throughput of one computer when the complex functions in the mode of distributing tasks.

The system throughput of each individual computer in the complex is a function of the processor load which is governed by the size of main storage and the number of IO channels. The structure of the computer complex makes it possible to have for each computer the largest size of main storage and the maximum number of IO channels defined for a specific model. This enables achieving a high level of multiprogramming, a high degree of reactivity of the operating system and an extension of the degree of overlap of IO operations for various programs in various IO channels. This results in a considerable increase in the number of programs ready for processing by the processor which leads to an increase in its load and the computer throughput.

Large amounts of data to be processed and the requirements for problem solving time in ASU [automated management systems] have determined the availability in the computer complex of external storage with considerable capacity and rather high speed. The Unified System architecture allows solving this problem both by connecting large-capacity units and by increasing the total number of units.

Computer Complexing Facilities. Meeting the main requirement of a computer complex, ensuring high reliability of operation, imposes a number of additional requirements on the organization of the structure and the composition of the complex equipment.

Thus, for continuous functioning of a computer complex when the main computer solving the main problem in the complex goes down, there must be provided the capability of replacing it by a backup that while switched on does not assume the load, but only "keeps track" of the operation of the main computer.

During operation of the complex, loss of input messages, intermediate results and statuses is impermissible. To prevent this, the dependent computer has to be able to assume part of the load of the host computer, i.e. both computers must have the capability of solving some common problem, exchanging results of computations and control information.

The requirement of preventing a malfunctioning computer from affecting the operation of the complex, restoring it without interrupting complex operation or performing scheduled preventive maintenance, as well as the presence of problems that must be solved simultaneously and independently in each computer in the complex dictate the necessity of dividing the complex in these cases into two independent computers.

To meet these requirements, the computer complex must implement the mode of hot standby, distribution of tasks and independent functioning; the assignment of modes and the shifting from one to the other must be effected both by software to achieve the highest efficiency and by the manual method. The latter is needed for an emergency change of modes at the option of the complex operator, for example, in case a complex computer fails when it not only cannot change the complex operating mode, but cannot even message its status to the other computer.

To implement manual control of computer complex modes, there must be provided within the computer complex a computer complex control unit that is built into the line for exchauge of codes of control information and equipped with mode controls and indicators that record the status of the complex as a whole and each computer separately.

When computers are united into a complex, facilities for intermachine exchange must be provided. Change of modes must be effected by the transfer through some line of a control code from the computer acting as the host to the other machine in the complex and the corresponding response from the latter.

Cases of exchange between complex computers not only of codes, but also of files of control information are frequent. This necessity arises, for example, when changing from the mode of distribution of problems or independent functioning to the mode of hot standby. For this, the capability of rapid exchange of considerable portions of information must be provided in the computer complex.

The modes of hot standby and distribution of problems require both complex computers to access the same programs and data files located on some common external storage unit. Therefore, there must be facilities in the computer complex to provide each complex computer free access to some of the external storage units.

The majority of these requirements are met by facilities that allow joining two Unified System computers at the processor level by direct control, at the external storage unit level by using a two-channel switch, and at the IO channel level by using a channel-to-channel adapter; each complexing level operates under the control of corresponding software.

Direct control facilities ensure rapid communication through the standard direct control interface between central processors by using the special instructions, WRITE DIRECT and READ DIRECT, and the mechanism of external interruptions. Direct control facilities are used for rapid exchange of small amounts of data which are usually control or synchronizing information.

The channel-to-channel adapter is connected to two IO channels, using the standard IO interface for the Unified System. It is intended for exchange of control information, but its main function is exchange of data files at a rate close to the rate of operation of the channels.

The channel-to-channel adapter can physically be integrated into the IO channels (the YeS-4060 in the YeS EVM-1) or be a structurally standalone unit, which is the case for the YeS-4061 used in the YeS EVM-2 that contains two functionally independent adapters.

The YeS-4061 channel-to-channel adapter operates in the burst mode with maximum throughput on the order of one megabyte/second. For each IO channel, the adapter connected to it responds to channel requests, and accepts and decodes channel commands just like any peripheral control unit, but it differs from it in that these commands are used to provide communication between channels and synchronization of their operation. Thus, during communication of two computers using the channel-to-channel adapter, any one of the computers is considered an external unit with respect to the other. The adapter functionally consists of two exchange control units connected to each other by signal lines.

In contrast to the YeS-4060, the YeS-4061 channel-to-channel adapter can operate in two modes: in the compatibility mode, i.e. in the YeS EVM-1 mode, and in the expanded mode, in the YeS EVM-2 mode. In the latter mode, the YeS-4061 has certain features compared to the YeS-4060:

new commands are introduced: MODIFIED IDLE, SENSE STATUS BYTE and WRITE END OF FILE;

the sensed status byte is introduced which allows defining in detail the nature of a malfunction situation that has occurred in the adapter;

the additional indicators of ADAPTER CHECK and SPECIAL CASE are introduced in the status byte;

mandatory processing of the interruption condition is effected for the indicator ATTENTION which prevents exchange of information with random matching of commands;

the status NOT READY is introduced and used to inform a channel that the adapter is not ready for operation.

The two-channel switches enable connecting an external storage control unit to two channels of different computers, forming thereby a common extent of external storage on the units connected to them. The mutual effect of computers in accessing an external storage unit belonging to the common extent of external storage is eliminated by reserving the device for the time of its operation with a given channel and releasing it after completion of the operation with the channel.

The levels of device reservation differ for the units that control disk storage and tape storage. The command RESERVE DEVICE for disk storage control units reserves one storage unit connected to it, but for tape storage control units, the control unit itself is reserved, i.e. all devices connected to the given control unit are reserved. The instruction RELEASE DEVICE is intended to release the reserved units.

When computers are united into a complex, the problem arises of creating facilities for centralized control needed to organize expeditious control of the complex to make the fullest use of its resources and increase its readiness. In developing centralized control facilities, a compromise between software and hardware solutions is expedient. The computer complex control unit is a hardware facility for centralized control that supplements and expands the corresponding software control facilities.

The computer complex control unit is intended for assigning modes of operation of the computer complex, indication of its status and performing the functions of operator consoles for both computers that are part of the complex. It enables performing the following functions:

manual switching of the computer complex to the required mode of operation;

specification of the functional states of each computer included in the complex;

receiving from a computer and sending to a computer the functional states of a computer;

indication of the modes of operation of the complex;

indication of the functional state of each computer in the complex;

automatic audible and light signalling in event of malfunction of at least one computer in the complex or during emergency disconnection of power when power supply malfunctions for any computer and the control unit itself;

computer power on/off;

power status indication;

processor status indication;

initial program loading; and

external interruption of program.

The computer complex control unit is connected to the two processors through a standard interface for direct control and is structurally a standalone unit.

Computer Complex Operating Modes. It is evident that computer complex operating modes are determined as a function of a large number of factors: restrictions on time for solving problems, required validity of results of their solution, current status of complex hardware and other operating conditions. In the process, the meaning incorporated in the computer complex operating modes and computer functional states is determined by the specifics of the ASU [automated management system] in which the computer complex is used. In exactly the same way, the specifics of the ASU is taken into account in the specific interpretation of the control signals and response signals to them that are generated by software or by using the controls on the computer complex control unit and transmitted between computers or to both computers when a change in the functional state of the computer complex is required. The possible computer complex operating modes are listed in table 22 and a most general description of these modes is given.

Table 22

	Functional States	
Modes	Computer 1	Computer 2
1	working	backup
	backup	working
2	working	parallel
	parallel	working
3	working	auxiliary
	auxiliary	working
4	independent	independent
5	working	standby
	standby	working
6	independent	working
	working	independent

Mode 1. Used when the dynamics of the control problems require expeditious connection of the standby computer and maintenance of past history of control. In this case, both computers in the complex execute the same job, but the results of operation of the backup machine are not output (blocked), and are used only for a check comparison. Connecting the standby machine consists in blocking the working computer and unlocking the output of the backup machine.

Mode 2. Used to get over overloads on the working computer that may occur, for example, when control algorithms are changed so that the time of processing certain jobs on the working computer will be extended.

The working computer can keep track of its own load; for example, it counts the number of jobs in the queue for processing. When a certain critical load factor is reached, the standby computer is readied for the mode of parallel operation. With the onset of an overload, the standby computer assumes part of the load on the working computer, operating in parallel with it, and after the overload is worked off, it may switch, for example, to the backup mode.

Mode 3. In this mode, both computers handle control problems, but the working computer solves the main, basic problems of the ASU, while the standby computer processes control problems of an auxiliary nature, for example, some preprocessing of jobs.

Mode 4. In this mode, the computer complex is essentially split into two separate, independent computers operating under completely different programs and processing individual problems.

Mode 5. In this mode, the working computer handles all ASU problems, and the standby computer may be either in the waiting mode or the test check mode.

Mode 6. In this mode, one complex computer (the working one) processes all ASU problems, and the second solves problems not associated with control.

In all complex operating modes, the capability of switching one complex computer to the state for preventive maintenance or repair has been provided for.

The functional capability of the computer complex operating modes is governed to a considerable extent by both the capabilities of the operating system for each computer and by the capabilities of the software developed for a specific ASU.

In the YeS EVM-2, to support the computer complex operating modes, the operating system includes complexing facilities that provide for:

program control of the direct control facilities;

program control of the operation of the channel-to-channel adapter;

program control of the common extent of external storage on magnetic disks; and

program control of the common extent of external storage on magentic tapes.

These software facilities are discussed in detail in chapter 7.

5.3. Data Teleprocessing Hardware

One of the most effective ways of meeting user system requirements now is the development and application of facilitie; for remote processing of data. Modern development of information and computing systems and control systems is characterized by extensive use of data teleprocessing. The number of automated data processing systems in which teleprocessing is the main component of operation with data is increasing.

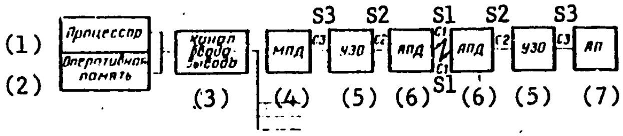


Fig. 30. Diagram of teleprocessing of data in the Unified System of Computers

Key:

1. processor
2. main storage
3. IO channel
4. MPD [data transmission multiplexer]
5. UZO [error protection device]
6. APD [data transmission device]
7. AP [terminal]

S1, S2, S3 -- junction interfaces

In the program for the Unified System of Computers, a central problem is that of developing the hardware and software for data teleprocessing systems and networks that provide the capability of distributed processing. It should be noted that the modular structure of the Unified System of Computers and the availability of interfaces at various functional and structural levels provide the necessary prerequisites for establishing systems for various purposes, primarily shared-use systems with a developed network of remote terminals and systems for intermachine exchange of data.

In the broad sense, data teleprocessing facilities (STD) using a computer include:

the teleprocessing hardware proper (data transmission multiplexer, data transmission equipment, terminals);

communication systems (telephone and telegraph channels, radio relay lines and others); and

the software.

This section covers hardware; the software is described in chapter 7.

Fig. 30 shows the basic diagram of the data teleprocessing system adopted in the Unified System of Computers. The data transmission multiplexer (MPD), connected to an IO channel, a multiplexer as a rule, through a standard interface controls the transmission and partial processing of information from a computer to terminals and back. If the level of processing of transmitted information increases, the MPD moves up to the rank of a data teleprocessing processor (PTD). A PTD allows reducing the load on the central processor. The YeS-8371 communications processor (Poland, Bulgaria) is an example of a PTD in the Unified System.

Data transmission equipment (APD) is designed to join logical discrete devices, data transmission multiplexers and terminals, to communication channels.

Terminals (AP) transmit information to a computer and receive data from it. If a terminal preprocesses data received and transmitted, it is called an intelligent terminal.

Logical channels for communication are classed as switched and dedicated. The latter include the physical lines enabling regional transmission of data.

For dedicated communication channels, permanent electrical communication is maintained between a computer and the terminals. Switched communications implies that both the computer and the terminals are subscribers to some switched communication system (the telephone, for example). To establish communications, either the computer or the terminal must give the communication system the required address (number) of the subscriber being called.

By a physical communication line is meant wire, cable communication between the computer and terminal without use of state-wide (and departmental) telephone-telegraph communication channels. As a rule, physical lines operate within the bounds of some region.

If one channel connects a computer with one terminal, such communication is called single-point. One channel may connect a computer with several terminals too, so-called multipoint communication, under which terminal-terminal exchange is possible without use of a communication channel.

There are three modes for transmitting data over communication channels:

simplex (transmission in one direction only);

half-duplex (alternate transmission in two directions); and

duplex (simultaneous transmission in two directions).

There is also a special transmission mode, when service information is transmitted (at a rate of 75 bits/sec) over the return channel at the same time in the half-duplex data transmission mode.

The validity of data transmission over communication channels is on the order of 10^{-3}. Therefore, in a data teleprocessing system, effective checking and recovery of transmitted information is performed at both the hardware and software levels. The main method of obtaining correct data is the method of retransmission when errors are detected. Various types of hardware checking are used to detect errors (parity check, cyclic codes, check sums and others).

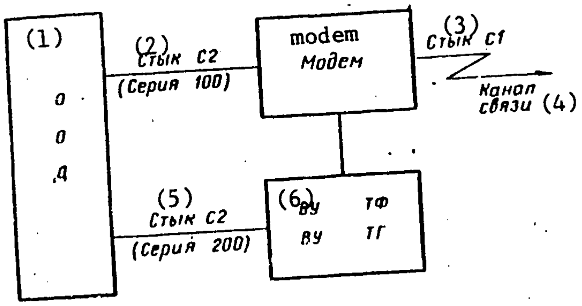
Two-wire and four-wire channels are used in communications. The latter are used in duplex transmission. There is also a version of duplex transmission over a two-wire channel that incorporates additional equipment for separating the forward and reverse channels.

Data transmission equipment mainly includes the following:

modems and signal converters (UPS);

calling units (VU) for switched communication lines; and

error protection devices (UZO).



Key:
1. OOD [data processing equipment]
2. junction S2 (series 100)
3. junction S1
4. communications channel
5. junction S2 (series 200)
6. VU TF, VU TG [telephone and telegraph call units]

Fig. 31. Diagram of connection of call units

Data transmission equipment is classed as low-speed to 200 bits/s (over standard telegraph channels), medium-speed to 4800 bits/s (over voice-grade channels) and high-speed of more than 4800 bits/s (over wide-band channels).

Modems convert the binary signals from a terminal (MPD or AP) into the modulated signals sent on a communications line. Reverse conversion occurs for receiving.

A typical representative of the family of modems is the modem 200 (YeS-8001, Bulgaria, USSR, Romania) used for synchron^us or asynchronous duplex transmission over voice-grade channels at 200 bits/s. Frequency modulation is used. Logical 1 and 0 are transmitted on the forward channel at 980 and 1180 Hz, and on the reverse channel at 1650 and 1850 Hz. During transmission, the frequencies are mixed, and separated by filters upon reception. The frequency of 2100 Hz is used for automatic call.

Differential phase shift modulation (single, doube and triple) is used for modems 2400 and 4800. In single modulation, the information bits 1 and 0 are coded by the two phases 0° and 180°; in double, a pair of bits (a dibit) is coded by four values of the phase; and in triple, a group of three bits is coded by eight values of the phase. There are experimental modems with 16-phase coding. Modems allow service telephone conversations when data is not being transmitted.

Communication is possible up to 13,900 km over telephone lines with retransmission. The number of retransmitters depends on the transmission rate. For the YeS-8010 at 600 or 1200 bits/s, the number of retransmitters is 12, and at 2400 bits/s, 6.

Physical lines are used in a limited region. To match terminal equipment to the physical lines, the YeS-8027, YeS-8028 and YeS-8029 low-level signal converters (UPS NU) are used. Line length depends on the transmission rate. Thus, for the YeS-8028:

two-wire line, duplex mode, 2400 bits/s--10 km;

four-wire line, duplex mode, 2400 bits/s--14 km;

and a four-wire line, duplex mode, 4800 bits/s--7 km.

To establish communication over switched lines, one uses telephone and telegraph call units (VU TF and VU TG) connected to data processing equipment according to the diagram shown in fig. 31.

Unified System data transmission equipment includes the YeS-8122, YeS-8135 and YeS-8140 error protection units designed to raise the validity of transmitted data. An UZO [error protection unit] is connected between the OOD [data processing equipment] (junction S3) and the modem (junction S1). This unit organizes transmission of data in blocks with cyclic checking. Different generating polynomials are used for checking:

for the YeS-8135: $X^{16} + X^{12} + X^5 + 1$; and

for the YeS-8140: $X^{24} + X^{23} + X^6(X^7) + X^4(X^5) + X^2 + 1$.

The polynomial for the YeS-8140 is reorganized for two versions.

When an error is detected, a request is made to repeat the data transmission. With a validity of data in a communication channel at 10^{-3}, the error protection unit provides for overall validity of 10^{-7}. The size of the buffer storage for the YeS-8135 enables data transmission to 13,900 km (at 2400 bits/s) and to 6,000 km (at 4800 bits/s).

Data transmission multiplexers are connected to Unified System computers through a standard IO interface. With a two-channel switch, a data transmission multiplexer can be connected to two multiplexer channels of one or two computers. The data transmission multiplexer operates according to the channel programs generated in computer main storage. The data transmission multiplexer establishes and disconnects communication with a remote terminal, transmits data with checking from and to a computer, provides a response to control symbols and generates corresponding control information to implement algorithms for data exchange.

If a data transmission multiplexer has sufficiently developed facilities (software) to preprocess data and control the data transmission network, then as mentioned earlier, it belongs to the category of communications processors or data teleprocessing processors.

Another variety of these units are the so-called remote data transmission multiplexers that allow economizing on communication channels. The YeS-8421 (Hungary) remote data transmission multiplexer can combine low-speed communication channels into one high-speed and vice versa.

Let us briefly discuss the principle of operation of a data transmission multiplexer using the YeS-8402 (MPD-2, USSR) as an example. This multiplexer enables operation in the half-duplex mode with the maximum number of communication channels (176) of which 64 can be switched. The number of connectable channels depends on the transmission rate and on the exchange algorithm (synchronous--asynchronous, duplex--half-duplex). Two communication channels with a remote terminal are used to organize the duplex mode. Operation of the MPD-2 over switched and dedicated telephone and telegraph lines, as well as over physical lines, is provided for as a function of the data transmission equipment used.

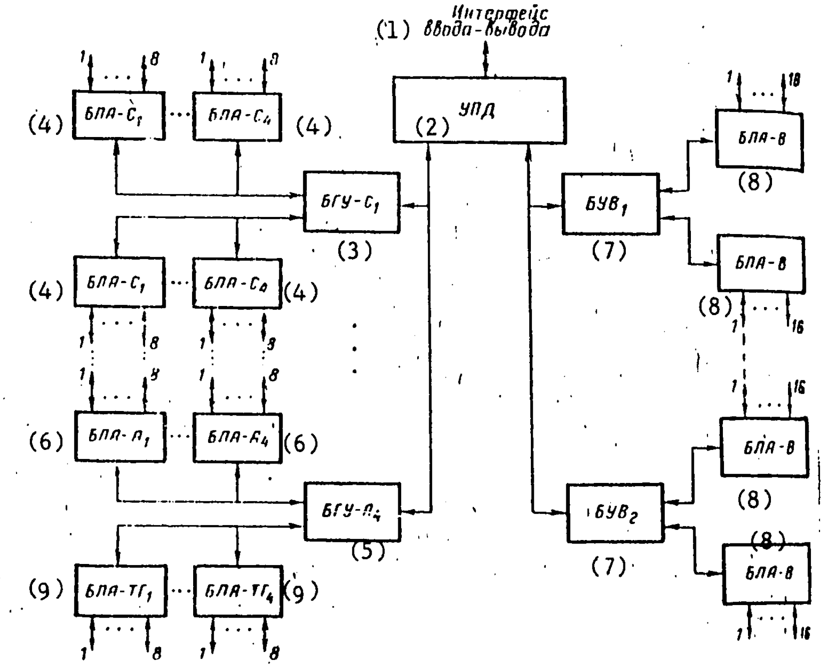


Fig. 32. Diagram of MPD-2 data transmission multiplexer

Key:

1. IO interface
2. UPD [data transmission control unit]
3. BGU-S_1 [group control unit, synchronous]
4. BLA-S [line adapter unit, synchronous]
5. BGU-A_4 [group control unit, asynchronous]
6. BLA-A [line adapter unit, asynchronous]
7. BUV [call control unit]
8. BLA-V [line adapter unit, call]
9. BLA-TG [line adapter unit, telegraph]

The YeS-8001, YeS-8002, YeS-8010, YeS-8011 and YeS-8015 modems, YeS-8030 telegraph line interface unit, and the YeS-8027 physical line interface unit can be connected to the MPD-2.

Transmission rate is up to 4,800 bits/s. Maximum device throughput is 40K bytes/s.

It provides for operation of the AP-1, AP-2, AP-4, AP-61, AP-63 and AP-70 terminals as well as for computer-computer exchange.

The validity of the transmitted information is no lower than 10^{-6} with a probability of error in the channel of no more than 10^{-3}. A general diagram of the MPD-2 is shown in fig. 32.

The data transmission control unit is intended for:

interface to the channel;

storage of control information and data coming from the channel and terminal;

control of all units in the MPD [data transmission multiplexer];

implementation of exchange algorithms; and

checking and diagnostics of all MPD units.

The basic logic for operation of the UPD [data transmission control unit] is implemented by microprogram read-only memory with 4096 words of 72 bits each.

The BGU-S and BGU-A plug-in units perform group control of the BLA-S and BLA-A line adapter units respectively in the synchronous and asynchronous (start-stop) modes of data transmission. The BLA units send and receive the next bit of information in turn for the communication line. The BLA-TG operates either through a UPS-TG [signal converter, telegraph] or directly for a telegraph line. The call control units (BUV) and the line adapter units, call, (BLA-V) control the automatic call units on switched communication lines (YeS-8061).

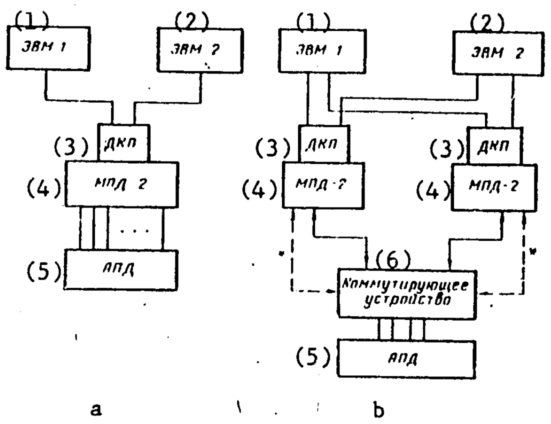
In sending data to a terminal, the MPD-2 executes the WRITE command. UPD storage (1024 x 36 bits) accumulates up to 8 bytes of data from the channel; then the data 1 byte at a time for the synchronous mode or 1 bit at a time for the asynchronous mode are sent to local storage in the group control unit and then 1 bit at a time to the line adapter unit. In dialing the call number, the UPD sends one digit at a time to the call control unit and the line adapter unit, call.

The MPD-2 command system includes both the standard channel commands (such as RESERVE, SENSE, TEST I/O and others) and the commands needed to implement the algorithmsfor exchange with the various terminals. A brief description of the second group of commands is given in table 23.

Table 23

Command Code	Command	AP-2, AP-4, AP-61, AP-63 (synchronous transmission)	AP-70 telegraph	AP-61, AP-63 (asynchronous transmission)
01	WRITE	+	+	+
02	READ	+	+	+
05	LOOP WRITE	+	+	+
06	PREPARE	-	+	-
06	PATH READY	+	-	-
09	POLL	+	-	-
0A	READ-M	-	+	-
23	SET STATE	+	-	-
27	ON	+	+	-
29	DIAL NUMBER	-	+	-
2D	OFF	+	+	-

The ON command connects the MPD-2 to the APD [data transmission equipment] and establishes character synchronization in the data transmission channel.



Key:
1. computer 1
2. computer 2
3. DKP [two-channel switch]
4. MPD-2 [data transmission multiplexer]
5. APD [data transmission equipment]
6. switch unit

a. without backup
b. with backup

* control signals (MPD READY, MPD NOT READY, MPD CONNECTED)

Fig. 33. Diagram of backup during operation with the MPD-2

The SET STATE command is used only in operating with synchronous terminals and is the first command, with which exchange of information with terminals begins. This command is intended for setting in the UPD [data transmission control unit] the state of operation with a check of the intermediate unit.

The READ command takes data from the communication channel and sends it to computer main storage.

The WRITE command sends data from the computer to the communications channel and then to the terminal.

The PATH READY command looks after character synchronization in the communications channel and the start of operation of a terminal. This is an expectation, characteristic for the MPD, of a response from a terminal.

The POLL command polls the terminal equipment (AP [subscriber station] and its components) in the communication channel.

The LOOP WRITE command checks the MPD-2. Information sent through some IO subchannel is always sent back to the computer through the zero subchannel.

The OFF command is intended for disconnecting the MPD-2 from the APD.

The READ-M command differs from READ in that there is no measurement of time-out (for the start-stop mode of transmission, when time between two transmitted characters is not defined).

The PREPARE command defines the start of operation of a start-stop terminal.

Table 24. MPD [Data Transmission Multiplexer] Characteristics

MPD Model Country	Максимальное число каналов (1) при режимах		rate, Скорость, бит/с bits/s	(2) Метод защиты от ошибок	Equipment connectable at remote end																				
	half- полудуплексном duplex	дуплексном duplex			АП-1	АП-2	АП-3	АП-4	АП-5	АП-6	АП-14	АП-15	АП-31	АП-50	АП-61/1	АП-61/2	АП-62	АП-63	АП-64	АП-70	АП-74	ТА-5	ТА-7	ЕС-7925	ЕС-7921
MPD-1A (YeS-8400) USSR	16	8	50-2400	M C	X	X		X							X	X	X	X	X	X		X	X		
MPD-1A (YeS-8401) Bulgaria	64	4	50-2400	M	X			X	X	X			X	X		X	X	X	X	X		X	X		
MPD-2 (YeS-8402) USSR	176	88	50-4800	M C	X	X		X	X	X	X		X	X	X	X	X	X	X	X		X	X	X	X
MPD-3 (YeS-8403) USSR	4	2	50-4800	M C	X	X	X	X												X		X	X	X	X
MPD-4 (YeS-8404) GDR	12		200-1200	M	X				X	X			X				X	X	X	X					
MPD-10 (YeS-8410) Hungary	32		50-2400	M C	X	X	X	X	X	X				X		X	X	X	X	X	X	X		X	X
UMPD (YeS-8421) Hungary	23/1		50-200		X												X		X	X		X	X		

Key:

1. maximum number of channel with modes
2. error protection method: M = matrix check (longitudinal-transverse parity); C = cyclic check

TA-5 and TA-7 are telegraph sets, 5- and 7-element, respectively

Russian "АП" and "EC" = English "AP" and "YeS", respectively

Operation of the MPD-2 with a backup is provided for systems with high reliability (fig. 33). Characteristics of other Unified System MPD's are given in table 24.

Terminals. The nomenclature of Unified System terminals is rather extensive and it is continually being expanded and modernized.

A terminal is one or more peripherals with a special control unit that performs various functions. The terminal control unit implements the terminal operation algorithm, performing actions to establish and disconnect communication between the MPD and terminal and to control the data exchange itself between them. For this, the control unit generates and recognizes the necessary control characters, counts and compares check sums of data blocks when the exchange takes place with division of information into blocks, and messages the MPD on its readiness for operation, desire to change direction of transmission or to end the communication session. Upon receiving information from the computer, the terminal control unit translates the data from the transmission code to that of the peripheral at the station to which this data is directed. Reverse translation occurs when sending information from a terminal to a computer. The control unit enables standalone operation of the peripherals making up the terminal station. As mentioned before, the mode of multipoint operation of a group of terminals for one communications channel is possible for some terminals. In this case, terminal-terminal communication is possible. Multipoint connection can be implemented both through an S1 junction (using matching units, fig. 34) and an S2 junction (fig. 35).

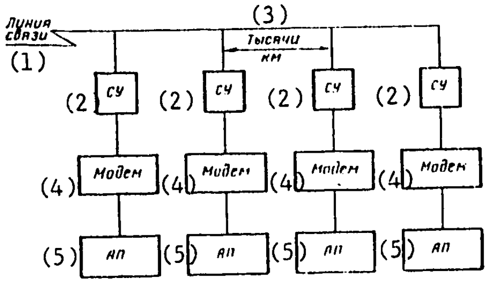


Fig. 34. Multipath connection of terminals by an S1 junction

Key:
1. communication line
2. SU [matching unit]
3. thousands of kilometers
4. modem
5. AP [terminal]

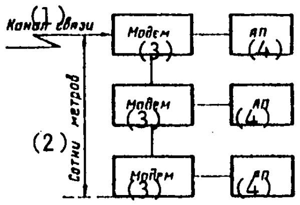


Fig. 35. Multipath connection of terminals by an S2 junction

Key:
1. communication channel
2. hundreds of meters
3. modem
4. AP [terminal]

Terminals differ in composition of peripherals, exchange algorithms (synchronous, asynchronous) and connectable data transmission equipment (table 25).

Table 25. Terminal Characteristics

	Input							Output								Data Transmission Device						Modes		Line Type			Data Transmission	
Terminal Model Country	keyboard	perf. cards	perf. tape	edge cards	mag. tape	jetons	passport	perf. cards	perf. tape	edge cards	ser. printer	A/N printer	mag. tape	CRT	error control	modem 200	modem 1200	modem 2400	modem 4800	sig. conv. LL	sig. conv. TG	duplex	half-duplex	telegraph	telephone	low-level	synchronous	asynchronous
AP1 YeS8501 BU	x	x	x	x					x	x	x				M	x	x			x	x		x	x	x	x		x
AP2 YeS8502 HU	x		x	x					x	x	x				C	x				x	x		x	x	x		x	
USSR	x		x	x					x	x	x				C	x					x	x	x	x	x		x	
AP3 YeS8503 BU HU	x		x						x		x				C		x			x			x		x		x	
AP4 YeS8504 UR	x	x	x		x				x		x	x	x		M			x					x		x		x	
AP5 YeS8505 GE	x	x	x			x	x		x			x			M	x	x			x								
AP6 YeS8506 GE	x	x	x			x	x		x		x				M		x			x								
AP14 YeS8514 PO	x	x	x		x				x		x	x	x	x	M,;C	x	x	x										
AP31 YeS8531 BU	x	x	x	x					x	x	x				M,C	x	x			x	x							
AP50 YeS8550 HU	x	x	x		x	x			x		x	x	x	x	M,C		x	x	x	x								
AP61 YeS8561	x										x			x	M	x		x				x					x (Mod 1)	
USSR	x										x			x	M			x				x						x (Mod 2)
AP62 YeS8562 HU	x										x			x	M	x	x	x	x	x	x							
AP63 YeS8563 UR	x										x			x	M			x					x					
AP64 YeS8564 HU	x										x			x	M	x	x	x	x	x	x							
AP70 YeS8570 BU USSR	x										x				M	x												
TA5 YeS8591 CZ	x	x							x		x																	
TA5 YeS8592 GE	x	x							x		x																	
TA7 YeS8593 GE CZ	x	x							x		x																	

Key:

BU = Bulgaria	M = matrix
CZ = Czechoslovakia	C = cyclic
GE = German Dem. Rep.	LL = low-level
HU = Hungary	TG = telegraph
PO = Poland	
UR = USSR	
A/N = alphameric	

Prospects for Development of Data Teleprocessing Facilities. There is now a definite trend in shifting from data teleprocessing systems based on a single computer to data teleprocessing network systems where the end user has the capability of accessing a computer network.

With the network organization of information and control systems, there arise many intricate and laborious problems both in compatible software and in realization of hardware. The network organization implies a multilevel structure with mandatory availability of standard protocols for interface between levels. Development of these interface protocols on the union and international levels is a complex problem.

Among hardware for the network organization, there will be needed devices such as data teleprocessing processors, remote data transmission multiplexers and various types of communication line concentrators. Data teleprocessing processors, on the one hand, must reduce the program load on the network central computers, and on the other, provide optimal control of information transmission to the networks (message and packet switching). In connection with the great progress in electronics technology, the intelligent level of terminals is growing and a family of intelligent terminals, which in a certain sense are minicomputers, is emerging.

For our country, developing data teleprocessing network systems is especially important and at the same time this is very difficult because of the great geographic expanses.

To solve these problems, it is necessary to develop a unified principle of the systems approach to organizing data processing systems, including shared-use systems, computer networks, and data transmission systems or networks. This principle must reflect the unity of the various parties: the user (list of services offered the user) and the developers of the communication lines and computer hardware (ensure coordination at all levels and compatibility).

Resolving these questions has led to the problem of the architecture of open systems of network teleprocessing (OSST), which can be formulated as the capability of interaction by a user or program of one computer system with a user or program of another computer system. It is evident that this problem requires providing for system-wide compatibility in computer and data transmission networks.

Development of the OSST includes:

developing a logical model of the system architecture with a formalized description of facility functioning logic;

distribution of functions at functional levels between end users and development of protocols for inetraction of the distributed components of a level; and

introduction of a language for describing the functions and interaction in an open system.

In the system plan for an OSST, the network is logically built from sequentially positioned functional levels: the lower levels correspond to the telecommunications facilities in the data transmission subsystem, and the upper to the standard application of data processing and to the users (computers, terminals and user application programs.

Open systems of network teleprocessing are a major step in the evolution of systems use of computer technology in the socialist community countries; it is expected that they will promote considerable expansion of computer applications.

Chapter 6. Development of Checking and Diagnostic Facilities

Unified System computer use in various sectors of the national economy, the broad range of users, complication of equipment and the need to ensure high technical and economic indicators required taking special steps to raise computer operating reliability. Achievements in structural solutions or technology may not yield an end high effect for a user if the problems of checking the operation of computers and recovery of the computing process are not solved.

The main directions in implementing requirements for raising reliability of computer operations are hardware and software checking facilities, diagnostics and recovery. At today's level of computer technology development, these facilities are intertwined and separating them is very arbitrary; because of this, in further discussion we will use the concept of hardware-software facilities, which provide:

machine-check handling;

correction of errors in main storage;

CPU retry;
IO operation retry;

microdiagnostic procedures;

recovery within operating systems; and

maintenance programs.

6.1. Machine-Check Facilities

In connection with the byte organization of data, checking each byte for parity is the main method of hardware checking in the YeS EVM-2, just as in the YeS EVM-1. The eight information bits of a byte plus one check bit always have an odd number of ones. This method allows detecting single bit errors and malfunctions of the type ≡ 0 (constant lack of a signal) in the control circuits. In the general case, checking modulo two allows detcting errors present at the same time in an odd number of bits. Assuming the appearance of double errors in any two bits of an information word has equal probability, the percentage of such undetected errors when both bits are part of one nine-bit byte is:

$$\frac{4 \cdot C_9^2}{C_{36}^2} \cdot 100\% = 23\%$$ for a four-byte word and

$$\frac{8 \cdot C_9^2}{C_{72}^2} \cdot 100\% = 11\%$$ for an eight-byte word.

Parity check circuits easily and economically allow checking ripple through transfers of information and storage of information in registers. On the other hand, parity checking of functional assemblies such as decoders, counters and adders requires considerable equipment (50-80 percent of the main hardware). Considering

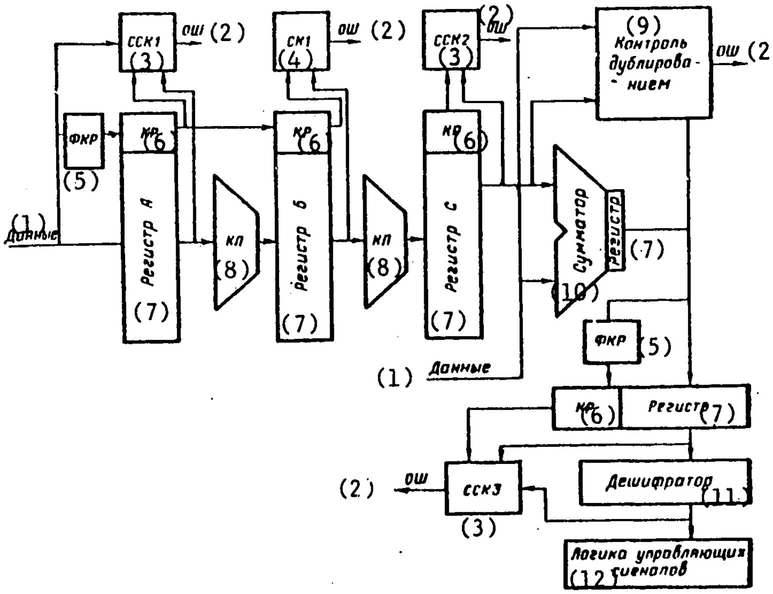


Fig. 36. Self-Checking Check Circuits

Key:

1. data
2. OSh--error signal
3. SSK--self-checking check circuit
4. SK--check circuit
5. FKR--check bit generation
6. KR--check bit
7. register
8. KL--combinational logic
9. duplication check
10. adder
11. decoder
12. logic of control signals

that modulo three and higher checking and duplication checking are more efficient in the degree of error detection, the application of mixed methods of hardware checking in YeS EVM-2 CPU's is apparent. Thus, the CPU's in the YeS-1035 and YeS-1060 computers are fully checked for parity, inc.'uding the adders. In the YeS-1045 computer CPU, parity check is combined with duplication of the main adder. The adders in the YeS-1065 computer CPU are modulo three checked.

Mixed checking is critical from the point of view of validity in assemblies for shifting from one form of checking to another. Therefore, self-checking check circuits are used at such points, for example, in the YeS-1045 computer, in which these circuits are employed wherever loss of the check signal itself is possible due to malfunction. As a rule, these are the beginning and ending points of the check path (fig. 36).

The use of self-checking check circuits seems to be promising with the incorporation of LSI and VLSI circuitry.

C. Shannon posed the problem to computer developers: Who will check the checkers? Prior to the introduction (let us note--widespread introduction) of self-checking check circuits (and here additional research is still needed), Shannon's problem was solved by introducing through the special CPU instruction DIAGNOSE procedures for simulating incorrect parity and testing of check circuits in the diagnose mode with test programs. Periodic execution of these test programs practically guaranteed operative readiness of the check circuits to perform their finctions. This method has been implemented in the CPU's for the YeS-1052 and YeS-1060 computers.

Selecting a specific check circuit for units that transform information is governed by:

depth of checking;

bulk of check equipment;

effect of check circuits on operating time diagram;

convenience (inconvenience) of packaging operating and check circuitry in plug-in units; and

the diagnostic capability of the check circuits themeselves and the operating circuits.

In modern computers, especially great attention is paid to the problems of validity of information kept in main storage. This is because memory sizes have increased sharply as a result of the increased packaging density of storage elements. It may be said that the weight of storage bit reliability malfunctions is declining. Correcting codes have long been accepted and applied in external storage units and they are now being used extensively in the YeS EVM-2 for main storage. The popular and efficient modified Hamming code (see chapter 3) that allows correcting single and detecting double and some multiple errors is being used to check information in main storage.

The Hamming code apparatus also requires solving the problem of checking the checkers. Check circuit test modes must be specified by the DIAGNOSE instruction. Progress in diagnosing main storage malfunctions is due to the generally accepted method of structural-bit organization.

6.2. Machine-Check Handling

A major factor in maintenance efficiency is the mechanism adopted to handle signals from the circuits that detect errors in the machine-check system. YeS EVM-1 computers perform the following operations upon signals from check circuits:

recording of status of storage elements in the CPU and channels in the fixed area of main storage;

machine-check interruption; and

software recovery of the computing process.

It should be noted that in the process the main storage recording area was inadequate, and at the time of the interruption, except for the "old" PSW, no additional information was recorded, i.e. the interruption code was zero. Software was responsible for recovering the computing process, and it had three levels:

I. recording and acquisition of statistics on malfunctions in the system log for YeS OS (or DOS). There is actually no recovery and as a consequence of this, the operating system is reloaded after a malfunction;

II. recording, acquisition of statistics and analysis of in what kind of program, system or user, the malfunction was observed. If it is a system program, it is reloaded; if a user program, the job is removed and the operator has to restart it.

III. recording, acquisition of statistics and analysis of the status recording area and an attempt at software retry of the program instruction or fragment that caused the malfunction.

Programs at level III are the most effective. But at the same time, these programs are considerably model-dependent, complicated and laborious in development. On the other hand, level III programs in a number of cases yield false solutions in the area of recovery: the program requires reloading when actually retry of the computing process is possible; the program notes the fact of recovery when in fact there is no recovery (or it is partial), and in this case, the job is ended with incorrect results. The latter situation is most dangerous.

The experience of operating with level III programs led to the necessity in the YeS EVM-2, on the one hand, of imposing some of the recovery functions on hardware, making use of microprogram control, and on the other, of standardizing procedures for handling error signals and expanding the composition of information on the malfunction situation and in turn standardizing this information. In the operating system enabling operation of the YeS EVM-2 models, level III programs are clearly divided into model-dependent and model-independent modules. The laboriousness of development and maintenance has been reduced. The instruction that has failed is retried mainly by hardware (microprogram) facilities which raises the validity of recovery considerably.

Depending on CPU structural complexity, hardware retry is effected either from the beginning of the instruction that has been incorrectly executed or from a checkpoint within the instruction. The instruction that failed is retried up to eight times, after which an interruption signal is generated if recovery has not occurred. In low and medium throughput CPU's with nonconcurrent structure and microprogram control, retry, as a rule, proceeds at the level of a microinstruction. In CPU's with concurrent levels of instruction processing and with units having hardware control, retry is performed from the beginning of the instruction, except for instructions with variable-length operands; these are retried starting from the current doubleword of the operand. During retry, the CPU is switched to the nonconcurrent mode which makes it possible to "push through" more easily the instruction that failed.

The retry facilities do not affect CPU speed because additional equipment was introduced for expeditious buffering of certain information during program operation. This equipment takes up one-two percent of the CPU apparatus and allows buffering of instruction addresses, operand addresses, operands changed during instruction execution, and other information needed to recover the computing process. Anti-hang-up facilities are also introduced into the processors.

A hang-up occurs when no hardware malfunction is recorded, yet the system cannot continue operation since some procedure begun in the system has not been completed. This may happen, for example, in event of CPU looping in an algorithm of some instruction without disturbing parity, or when signals are dropped in the CPU-storage or channel-storage interface.

The anti-hang-up facilities in the processors contain an apparatus that upon expiration of the maximum permitted interval of time for execution of any desired long instructions generates a signal causing amchine-check interruption and informing the program of the hang-up with recording of the CPU state. In case of an incompleted instruction with an IO channel, the correspodning channel is reset to obtain access to the system external devices. After receiving the signal on the CPU hang-up, the software, just as when a check malfunction occurs, performs the procedures to recover system operating capability.

Let us discuss and analyze the standard machine-check handling facilities that have been incorporated in the YeS EVM-2. The effectiveness of these facilities has been primarily facilitated by the clear classification of machine-check interruption conditions into two types: repressible and exigent.

Repressible conditions are those in which the instruction processing capability of the CPU has not been affected. Therefore, these interruptions can be delayed until the completion of the current instruction or even longer if necessary. Repressible conditions are of three types: recovery, alert and repressible damage.

Recovery conditions can be either circumvented or discarded. Malfunctions causing them are either not reported or if reported are grouped in one subclass, system recovery.

A machine-check interruption condition not directly related to a hardware malfunction is called an alert condition. The alert conditions contain two subclasses:

degradation (for example, size of buffer storage has been reduced, fast multiplier has been disconnected, counter main storage errors to be corrected has overflowed and others): and

warning (for example, signal for disconnection of air conditioning or fans, exceeding climatic conditions and others).

Repressible damage conditions are divided into three subsclasses:

timer damage;

timing-facility damage; and

external damage (IO malfunction, channel buffer malfunction in the storage control unit, for example, and others).

Repressible condition interruptions enable execution of the CPU instruction, then the waiting program and supervisor interruptions are processed, and only after that is the machine-check interruption taken if the corresponding masks are open.

Conditions that govern exigent interruptions do not allow the CPU to continue execution of instructions since direct damage has occurred to its operation.

Exigent conditions are divided into two classes:

instruction-processing damage or incapability of continuing an interruption; and

system damage (all severe situations in the system that cannot be isolated to the preceding class).

Exigent conditions terminate execution of the instruction and cancel processing of program and supervisor interruptions.

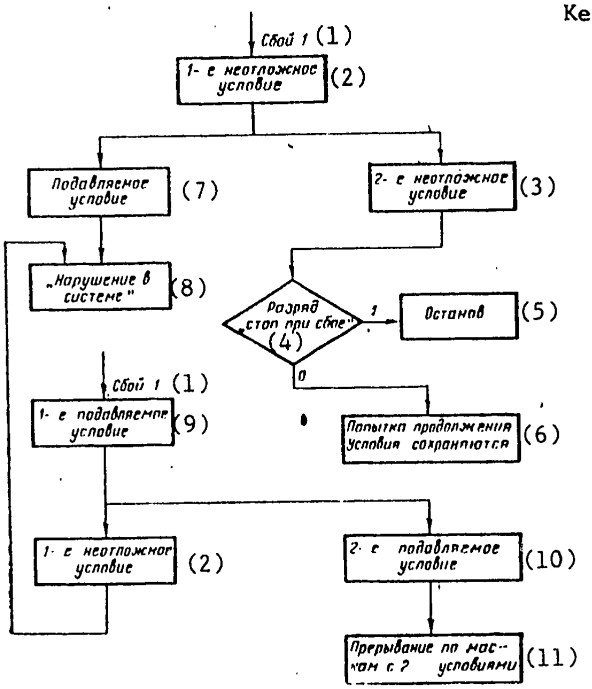
In accordance with the classes and subclasses of the interruptions, a system of masks is built that is defined by bits 4-7 of control register 14 (table 26). Just as in the YeS EVM-1, PSW bit 13 controls masking of the entire check as a whole. If this bit is zero, the check in the machine is inhibited; if one, it is permitted.

Table 26

Mnemonic Designation	Mask	Control Register Bit Number
RM	Recovery	4
DM	Degradation	5
EM	External Damage	6
WM	Warning	7

The EM mask is propagated for all three subclasses of the repressible damage class. In contrast to the YeS EVM-1, another signal has been introduced: CHECK-STOP which in certain malfunction situations leads to a system halt. This signal is governed by the status of the zero bit in control register 14. Special attention should be paid to this. The computer halt mode was not provided in the YeS EVM-1; it could hang up, loop and stay in the wait mode. Experience of YeS EVM-1 operation showed that it was necessary to isolate a set of malfunction situations in which it is advisable to provide for halts to preclude destruction of the results of the system operation prior to the malfunction, to analyze situations from the engineer's console and make a decision. These are malfunctions after which recovery cannot be transferred to level III programs. The logic of operation of the check-stop signal, check masks in the PSW and exigent interruption conditions are model dependent to a considerable extent. The main principle is that a system halt (hard machine-check) occurs with an exigent interruption condition when the PSW bit 13 is 0 and there is a check-stop signal.

The algorithm for system actions when a second interruption condition occurs is shown in fig. 37. The flowchart describes the system response when two machine-check interruption conditions occur in succession. With two exigent conditions, the response depends on the state of the check-stop bit. It should be mentioned that a halt occurs in certain YeS EVM-2 models even when this bit is zero. Three combinations of two interruption conditions are classified by the status of SYSTEM DAMAGE. Two or even more repressible conditions lead to a single interruption, but all repressible conditions that have occurred are reflected in the interruption code.


Сбой 1 (1)
1-е неотложное условие (2)
Подавляемое условие (7)
2-е неотложное условие (3)
„Нарушение в системе" (8)
Разряд „стоп при сбое" (4)
1
Останов (5)
0
Сбой 1 (1)
1-е подавляемое условие (9)
Попытка продолжения Условия сохраняются (6)
1-е неотложное условие (2)
2-е подавляемое условие (10)
Прерывание по маскам с 2 условиями (11)

Key:

1. malfunction 1
2. 1st exigent condition
3. 2nd exigent condition
4. check-stop bit?
5. halt
6. attempt to continue; conditions maintained
7. repressible condition
8. system damage
9. 1st repressible condition
10. 2nd repressible condition
11. interruption by masks with two conditions

Fig. 37. Flowchart of processing of secondary interruption condition

The standard procedure for processing an error signal from the check circuits is shown in fig. 38. The real algorithms for processing machine errors are more complex and consider the structural features of the processors. With a machine check, upon a signal from the check circuits, in certain cases information from the storage elements is stored in main storage. This procedure is called a machine-check logout.

The status logout is divided into two types:

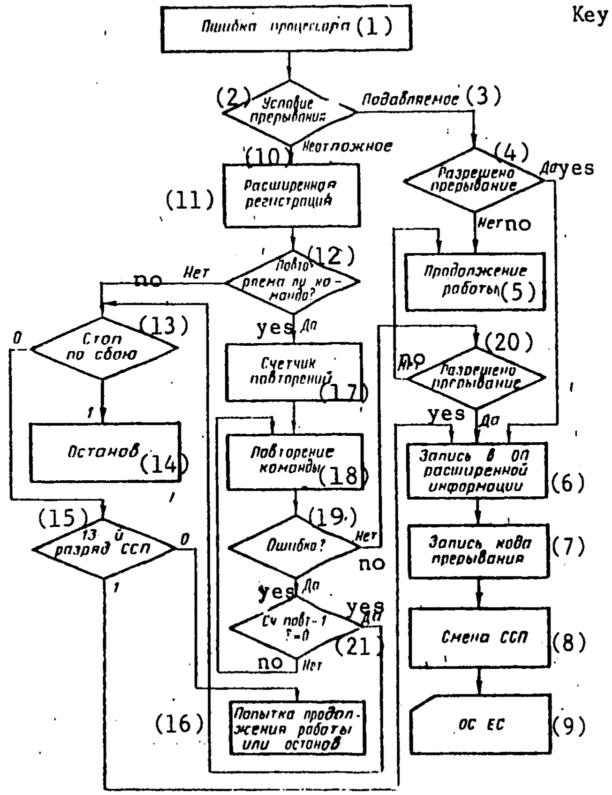
synchronous--during the interruption procedure; and

asynchronous--not at the same time as an interrupt, if there is no interrupt or in the case of a delayed interruption.

In turn, each logout type is divided into two forms:

logout in a fixed storage area; and

extended logout in a storage area, the starting location of which is specified by the contents of control register 15.



Key:
1. CPU error
2. interruption condition
3. repressible
4. interruption enabled?
5. continue operation
6. write extended information in main storage
7. store interruption code
8. replace PSW
9. YeS OS
10. exigent
11. extended logout
12. instruction being retried?
13. check stop?
14. halt
15. PSW bit 13?
16. attempt to continue operation or halt
17. retry counter
18. instruction retry
19. error?
20. interruption enabled?
21. retry counter - 1 ? = 0

Fig. 38. Flowchart of CPU error signal handling

The fixed logout area is used primarily for storing additional information in a hardware retry of instructions (the YeS-1045 computer CPU). Because of this, fixed logout can only be asynchronous. To control logout, three bits of the mask in control register 14 are used: bit 1 -- synchronous extended; bit 8 -- asynchronous extended; and bit 9 -- asynchronous fixed.

Extended logout is used for model-dependent level III recovery programs and for statistical diagnostics on malfunctions. The size of the area in main storage for extended logout for the different YeS EVM-2 models is given in table 27.

Table 27

Models	Size of Extended Logout, Bytes
YeS-1025	16
YeS-1035	128
YeS-1045	644 (about 20 percent of the logout area is reserved for future development)
YeS-1055	128
YeS-1060	512

The machine-check extended interruption information consists of seven fields, which are stored at machine-check interruption time. Two fields are allocated for logout of timing facilities: the comparator and the CPU timer.

The contents of the clock comparator are stored in the doubleword starting at location 224. The contents of the CPU timer are placed in the doubleword starting at location 216.

The address of the main storage location in which an uncorrected error in information or protection key occurred is stored in bits 8-31 of the word at location 248.

The region code is stored at location 252 and contains model-dependent information which more specifically defines the location of the error. This may be, for example, the address of the system unit that caused an external interruption or recovery report.

The register save area uses three fields starting at loactions 352, 384 and 448. The contents of the floating-point, general and control registers, respectively, are stored in them.

The main integrated information is an eight-byte interruption code (KP). Eight bits in the interruption code identify the interruption subclasses, and 14 bits in this code indicate the validity of information stored in the interruption process.

The validity bit of some field is set to one if the stored information is valid with respect to the point of interruption, fully stored and no additional errors occurred in storing. Otherwise it is set to zero. It should be stressed that the validity bits are important to reliable operation of the YeS OS recovery programs.

The interruption code also contains bits specifying the location of the interruption point relative to the error location and more specifically defining the nature of an error in main storage and protection key storage; it also contains the size of the area for extended logout of information.

6.3. Microdiagnostic Procedures

YeS EVM-2 processors and channels make use of the mode of microdiagnostics (the terms microtesting and mcirotest are also used) that permits a substantial reduction in the time needed to find a malfunction and the skill-level of maintenance personnel.

Compared to software checking and diagnostics, the main advantage of microtesting on principle is the capability of logging hardware status at each step of synchronization and the capability of accessing additional checkpoints within the makeup of the extended logout area.

The microtesting mode requires additional hardware. Thus, it makes up three-four percent of that for the YeS-1060 computer processor. It is hard to name a precise figure for part of this hardware is used for other purposes: extended logout during a malfunction, loading of microprograms, etc. The microtesting mode also requires rather laborious development of an information base:

microtest files proper with a size up to one megabyte;

diagnostic information including glossaries and manuals for operation for when the glossaries do not give a unique diagnosis; and

diagnostic monitors.

As an illustration, given below are a brief description of the microdiagnostic system for the YeS-1060 computer processor and some features of its realization in the YeS-1035 and YeS-1045 computers.

The YeS-1060 computer includes the YeS-5009 console cassette storage unit that effects bit-by-bit storage on two tracks with duplication. Size of the cassette is about 70K bits. The YeS-5009 storage unit is used both to load working microprograms and to input microtests. In the processor, there is a special unit, the console storage unit adapter (APN), with 512 bytes of internal high-speed storage that controls input of information with checking either into microprogram control storage or into circuits for decoding APN diagnostic instructions.

To understand the capabilities of the microtesting mode, it is expedient to examine the system of APN diagnostic instructions. Let us stress right away that APN instructions are in no way related to CPU instructions. In this sense, the APN together with the console storage unit and the storage proper is a kind of minimachine, a service processor.

Instructions have a fixed format of 1 to 506 bytes. Instructions with a length of 1 to 3 bytes are generally used. Long instructions are intended for loading storage of microprograms. One byte is allocated for the operation code.

The APN instruction system contains a set of instructions for setting two microinstruction registers. A three-byte instruction sets two register bytes. The setting instructions are divided into two groups: with and without execution of microorders. There is an instruction to reset the microinstruction registers.

An elementary microtests consists of a string of instructions for setting without execution and one instruction for setting with execution. Prepared microorders are executed within one machine step.

Two three-byte instructions are used to poll the status of the CPU storage elements: DIAGNOSTIC BIT COMPARE and DIAGNOSTIC BYTE COMPARE.

The polled bits and bytes have conventional addressing used in the mode of extended logout of status and during display on the engineer's console. The bit address, containing 12 bits, or the nine-bit byte address is specified in the instruction. The instruction also contains a reference for comparison consisting of one or eight bits. If the polled result conincides with the reference, the next APN instruction is executed; if not, a halt occurs and the instruction address and instruction itself are displayed on the engineer's console. After this, the operator refers to the diagnostic information on the microtest printouts or in the glossary.

In the testing mode in question, execution of microorders is blocked and blocking is removed only for one step. Operating synchronization is not blocked. To perform multistep microtests, several instructions for setting with execution have to be used in succession. In the process, fetching of microinstructions from control storage is blocked.

The APN instruction system includes the one-byte instruction LOAD PROCESSOR which allows loading with initial information the internal processor registers (instruction buffer, general registers and others), by making use of the LOAD service microprograms located in microprogarm control storage. Service microprograms are loaded at the same time as working ones. Before the LOAD PROCESSOR instruction, instructions are used to set the microinstruction register without execution to set its address part. That is how entry into the service microprogram is defined. The LOAD PROCESSOR instruction transfers control to the service microprogram. The loaded data comes directly after the instruction into the APN. There may be individual service routines for loading the instruction buffer, general registers, etc. It is possible, but not advisable, to compile general-purpose loading routines. Service microprograms must be as simple as possible and require minimum operating capability of the hardware "nucleus," which should be tested using elementary microtests. A service microprogram is terminated by a synchronization halt, blocking of microorder execution and transfer of APN control. Then, by using strings of setting instructions, execution of more complex microtests is possible.

The START SYNCHRONIZATION instruction allows turning on synchronization for a given number of steps (up to 256).

The one-byte instruction, EXECUTE OPERATING MICROPROGRAMS, allows transferring control of some operating microprogram, the address of entry of which is specified by setting instructions, and stepping the operating algorithms for execution of CPU instructions, starting with the START SYNCHRONIZATION instruction with the number of steps as one. The latter mode allows reducing the laboriousness of development of microtests and partially automating this process. There has been developed for this an experimental generation system which on a computer in good working order upon the DIAGNOSE instruction logs by steps the individual instructions or combinations of instructions of the CPU. The information obtained is selected and used to automatically derive sets of microtests.

The APN instruction system contains a number of auxiliary instructions:

RESET MICROINSTRUCTION REGISTER;

SIMULATE KEYS--for microtests that test the circuits for manual operations for the engineer's console;

END MICROTEST--for organization of looping in case of manual diagnostics;

SET TESTING FROM BUFFER--specifies the mode of loading a microtest into the 512-byte buffer and cyclic execution of it. The alternative mode--the microtest read from the console storage unit is executed at once one time; and

SIMULATE INCORRECT PARITY--for checking check circuits.

The YeS-1045 computer APN executes the system of instructions for its structure quite similarly to that described above for the YeS-1060 computer. It is also used for checking individual TEZ's [standard exchange cards], and the check tests are entered from the console storage unit. This APN operating mode is called the auto-tester mode.

Finally, the YeS-1045 computer APN also controls automatic switching of the modes for routine checking of the power supply, providing for the running of microtests together with the unit for the automatic system for power supply checking and diagnostics when there is deviation in the ratings of the secondary power sources.

In the YeS-1035 computer, there is no special system of instructions for the APN. Microtests are fragments of microprograms and are loaded from the console storage unit directly into the micorprogram control storage.

Development and debugging of the microtests themselves involve considerable difficulties. In connection with the large size of the microtests, special automation facilities are required for these purposes. The MD60 symbolic language for CPU microtests has been developed for the YeS-1060 computer. The translator from MD60 into the codes of the APN instructions generates a listing of the microtests and the data medium on cassette tape. The translator incorporates facilities for making changes from a display.

The MD60 language is modeled on YeS Assembler and the macro apparatus has been provided for. The latter substantially facilitates writing of microtests. Variables in the MD60 language are mnemonic designations of the registers and flipflops adopted in the technical documentation. A list of the main MD60 statements and their functions are given in table 28.

Table 28

Statement	Statement Functions
SBR	is translated into the instruction SIMULATE KEYS with the CPU reset feature
SRMK	is translated into the instruction RESET MICROINSTRUCTION REGISTER
UST	is translated into a series of instructions for setting without execution. Used as operands are the designations of the fields of the microinstruction register and the values to which they are to be set
USTV	similar to the UST statement, but is translated into instructions for setting with execution
KT	is translated into the instruction START SYNCHRONIZATION. Operand specifies number of steps in decimal form
OPR	is translated into a series of instructions for byte and bit polling. Operand structure is similar to that of the UST statement. For example, RAV (0-7) = FF means that in polling the register RAV its reference value must equal FF

The macro apparatus for the MD60 language was designed on the basis of Assembler macro facilities. The symbols SET and internal macro instructions are used. The instructions of conventional compilation SETA, LCLA, AIF, AGO and ANOP have been defined.

6.4. DIAGNOSE Instruction

In the instruction set for YeS EVM-2 CPU's, just as in the YeS EVM-1, there is the special model-dependent instruction, DIAGNOSE, intended to extend the capabilities of program checking and diagnostics. The functions of the DIAGNOSE instruction vary with the model but at the same time have general properties.

These are primarily simulation of hardware errors for checking check circuits and retry. The DIAGNOSE instruction uses a control word, the individual bits of which specify the mode of parity inversion at particular points in the CPU. The inversion mode may be specified for a prolonged time before the next DIAGNOSE instruction or for two-three machine steps. The latter is needed in checking retry circuits for simulation of malfunction. Using the control word also enables storing incorrect information in main storage to check the Hamming code circuits. There is the capability of turning on and off correction by the Hamming code, for example, in detecting malfunctions in the check circuits. In general, turning some CPU equipment on and off is a characteristic function of the DIAGNOSE instruction. It can be used to partially or completely disconnect buffer storage, the translator buffer, units for speeding up operations and other equipment.

The DIAGNOSE instruction also specifies modes for channel checking or diagnostics. In doing so, it provides for checking check circuits, disconnection of the interface from peripherals and connection of it through the simulation register located in the channel. Disconnection of the channel-main storage interface is also possible. Channel diagnostics, if built into the CPU, are performed in the same way as for the CPU. If the channel is made as a standalone device, diagnostics are performed by synchronization steps with polling of internal checkpoints.

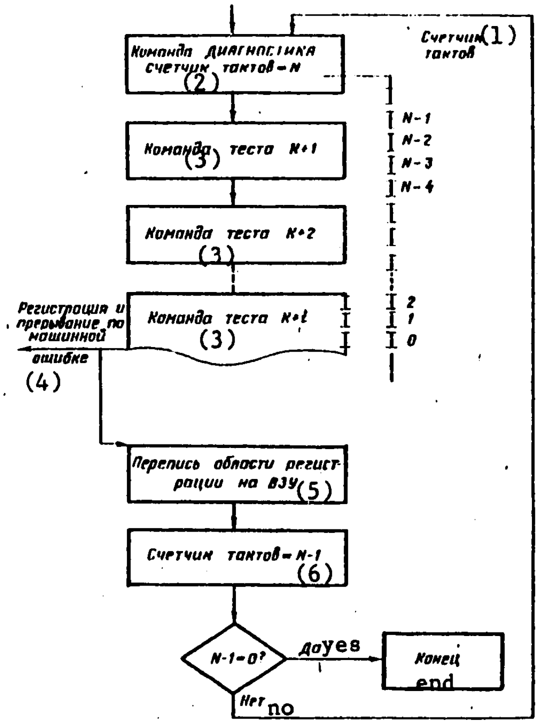
In the YeS-1045 computer, the DIAGNOSE instruction allows pecifying the microprogram storage address and number of microinstructions required for execution; this provides for the capability of realization (for testing purposes) of the microprogram fragments that are loaded from the console storage unit.

Using the DIAGNOSE instruction, logging of the CPU status at a given time can be effected. To this end, for example, a two-byte field for the synchro-pulse counter has been defined in the instruction control word in the YeS-1060 computer. In operating in this mode, the instruction DIAGNOSE loads the two-byte counter that subtracts one with each pulse of the main synchronization. When the counter is filled with zeros, a ficticious error signal is generated, CPU status is logged and a machine-check interruption occurs. Thus, one can produce a "frame-by-frame" picture of the execution of an individual operation or group of operations. The algorithm for "frame-by-frame" logging of CPU status in some sequence of instructions is shown in fig. 39. The log "images" are serially accumulated on external storage.

Based on the information obtained, one can construct a program for diagnosing equipment that goes beyond the "nucleus" (in this case, required as the "nucleus" is equipment that permits execution of the algorithm shown in fig. 40, i.e. statements marked with an asterisk). This program, using the DIAGNOSE instruction, effects step-by-step "photography" of the sequence of test instructions ("instruction 1," "instruction 2," "instruction i") and comparison of the frames obtained with the reference information derived earlier on a computer in good working order.

6.5. Software Checking and Diagnostic Facilities

Software checking and diagnostic facilities are divided into off-line and on-line.



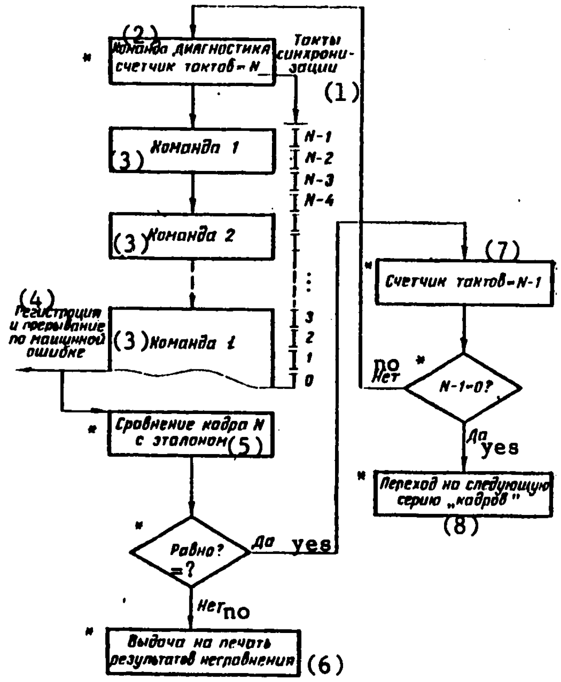

Команда диагностика счетчик тактов = N (2)
Такты синхронизации (1)
Команда 1 (3)
Команда 2 (3)
Команда i (3)
N-1
N-2
N-3
N-4
3
2
1
0
Регистрация и прерывание по машинной ошибке (4)
Сравнение кадра N с эталоном (5)
Равно? =?
Да yes
Нет no
Выдача на печать результатов несравнения (6)
Счетчик тактов = N-1 (7)
N-1=0?
Нет no
Да yes
Переход на следующую серию „кадров" (8)

Fig. 39. Flowchart of routine for obtaining reference results

Key:
1. step counter
2. DIAGNOSE instruction step counter - N
3. test instruction (K+1, K+2, K=i)
4. logging and machine-check interruption
5. copy logging area to external storage unit
6. step counter = N-1

Fig. 40. Flowchart of diagnostic routine

Key:
1. synchronization steps
2. DIAGNOSE instruction step counter - N
3. Instruction (1, 2, i)
4. logging and machine-check interruption
5. compare frame N to standard
6. print out results of no match
7. step counter = N - 1
8. go to next series of "frames"

On-line routines operate under control of operating systems in parallel with running of user jobs in the multiprogram mode. The peripheral to be tested is switched to the off-line mode for user jobs and teh job is started for execution of particular test sections of the given device. On-line tests allow performing preventive maintenance and repair of computers without reducing considerably computer throughput as a whole. This pertains primarily to computers with a large set of peripherals among which these is redundancy. Using on-line tests, it is also convenient to perform preventive maintenance of peripherals operating occasionally in the system (plotters, optical readers and others). The main aim of on-line tests is to substantially reduce the time for preventive maintenance of computers and thereby raise the factor of technical use of the equipment.

Off-line test routines in both the YeS EVM-1 and EVM-2 are part of the set of YeS TEST-MONITOR (TMYeS) programs. The set includes the TMYeS test control program, utilities for generating and maintaining media of the set (tapes, disks, cards) and a set of test-sections of all devices, including central.

The control routine interprets a simple and convenient job language, enables standard output of messages to the operator and has a set of utility subroutines facilitating programming of test-sections. The control routine is compiled so that only 40 of the simplest instructions (of 183) are used which require minimum serviceable hardware "nucleus" of the CPU and the input path.

To ensure serviceability of the control routine, the CPU and minimum path of input from magnetic tape are checked with the so-called base test. The latter checks the above mentioned 40 instructions and path of input from magnetic tape only in the read mode.

In the base test, messages are not printed out. Incorrect execution of test samples is indicated by looping of an unconditional branch instruction to itself or by a halt upon error signals from the check circuits. Then maintenance personnel perform analysis by the display on the engineer console and when necessary the microtest system is used (see section 6.3).

The set of off-line routines loadable directly from the medium of an external storage unit amy include any sort of measuring routine for one-time use, for example, routines to determine throughput of the CPU and channels, checking the accuracy of timers, checking disk rotation rate and the like.

Development of the complex automatic testing system (SKAT) for the YeS EVM-2 is planned as a prospect for development of test software. A characteristics feature of SKAT is the parallel testing of peripherals by the method of elimination: the devices are serially connected for parallel operation. If n devices are already operating and an error appears in the system when the n+1-th device is started, by a halt and start retry among the n+1 devices SKAT defines the minimum configuration in which the error remains. In the minimum configuration, it is easier to localize the source of the error. As a rule, these are complex interface errors occurring in parallel operation of the devices. During operation of SKAT, the CPU handles background test jobs. By operating modes, SKAT approximates the work of operating systems and at the same time has testing facilities ("elimination," tracing of events prior to the error, program halts upon specified conditions and others). Use of SKAT, an off-line system, permits checking computers with a large set of IO peripherals within 5-10 minutes.

A second promising direction in development of maintenance test software is the trend to having one (and not two) set of test-sections for a peripheral both in the off-line and in the on-line testing systems. This will lead to introduction of new control routines in the TEST-MONITOR and operating systems.

The third direction involves increasing the diagnsotic properties of software tests. This will require incorporating additional hardware diagnostic modes in the device.

6.6. Computer Maintenance

In maintaining computers during preventive maintenance and prescribed operations and when malfunctions occur, all hardware-software facilities for checking and diagnostics typical for the YeS EVM-2 are used:

hardware checking and improvement of the machine-check handling system;

hardware retry of instructions upon failure;

system of microdiagnostics;

DIAGNOSE instruction;

software testing;

hardware facilities for recovery of channels and IO units; and

software facilities for recovery of operating systems.

Daily prescribed operations include checking the computer with the base test and a limited set of the test-sections of the TMYeS. The configuration sufficient for starting YeS OS and the initial batch of user jobs is checked. A complete check of the configuration needed for the current day is performed with on-line tests under control of YeS OS.

A malfunction may be detected either during performance of the prescribed operations or in the working mode. When a peripheral fails, the malfunction may be located by using the on-line tests. For this, the peripheral is switched to the off-line mode, tested by off-line facilities through the unit console and again connected to the system for checking. If there is no capability of repairing the peripheral by using the on-line tests, repair is either put off till the next scheduled prescribed operations (if possible) or a search for the malfunction is made with the TMYeS test-sections, as a result of which running user jobs on the computer is terminated.

Failures of the central part (CPU, main storage, channels), as a rule, lead to computer failure. It is possible to continue operating in some cases when one can get by with reduced size of storage or without some number of channels. When the CPU fails, the base test and the set of TMYeS test-sections are executed to find the location of the malfunction.

When maintenance personnel have some preliminary information on the malfunction, the single-purpose set of test-sections is executed in specific order. The test sections enable localization down to a block, instruction and mode. Then the microtest system is used selectively. And in the most necessary case, a switch is made to manual diagnostics using either microtests, the base test or test-sections (tracing, step-by-step mode, time logical analyzers and the like).

Information from operating system recovery facilities is used by the engineer for forecasting the source of failures and further planning of the next scheduled operational checks.

Chapter 7. Software

Software, which largely determines the capability of hardware, holds a major place in the development of computer technology. The role of software in extending the sphere of application and efficiency of the use of computers is decisive. This is due to the relation of cost of development of hardware and software steadily and continually changing in favor of the latter.

A considerable stock of user software has now been accumulated and continues to increase rapidly; this must be taken into account not only in the development of new computer systems, but also in choosing the directions of development of software. Therefore, in developing the YeS EVM-2, considerable attention was paid to improving Unified System software.

The Unified System software system consists of operating systems, application program packagaes (PPP) and maintenance programs.

The main function of operating systems is to increase the efficiency of use of the resources of a computer installation and to raise the convenience of operator-computer interaction. Extensively used in the Unified System are the YeS OS and YeS DOS operating systems that are discussed in detail in this chapter.

The makeup and function of PPP are also considered in this chapter; they extend the capabilities of operating systems and are oriented to solving specific user problems.

Maintenance routines are intended for determining the technical condition of computers both in the mode of preventive maintenance and while jobs are being executed (see chapter 6).

7.1. Evolution of the YeS OS Operating System

YeS OS is designed for use in medium and large Unified System models. Its evolution in developing the YeS EVM-2 was determined by:

expanding the sphere of services offerred the computer user;

raising the efficiency of software proper; and

raising the efficiency and reliability of use of hardware capabilities.

Within the YeS EVM-2, expanding the sphere of services is expressed in the following:

development of multiprogramming;

expanding the makeup of basic data structures and standardized methods of operating with these structures;

development of teleprocessing support;

improvement in the facilities for program debugging and those for checking the operation of a computer system.

Fundamentally important to the YeS EVM-2 is the incorporation of virtual main storage, one of the main resources in a computer system. In the YeS EVM-2, up to 16M bytes of virtual storage are made available to all job steps to be executed and are allocated among them dynamically. The use of virtual storage for mutual protection of jobs from each other as well as protection of the operating system control program from the jobs has made it possible to increase the number of simultaneously executable jobs and thereby raise the degree of multiprogrammability.

The development of multiprogramming in conjunction with the improvement in teleprocessing facilities and methods has allowed creating the time-sharing mode that enables shared access to the computing resources of the Unified System of Computers. The user is also offerred the capability of remote job entry based on the developed capabilities of teleprocessing.

From a software point of view, the transition from the YeS EVM-1 to the EVM-2 was effected by an evolutionary development of the YeS OS and DOS operating systems as well as by accumulating a stock of application programs.

YeS OS was developed in the following directions:

A new version of the control program was introduced: SVS (virtual storage mode), available with MFT and MVT versions from the YeS EVM-1;

A new common telecommunications access method, OTMD, is used in all versions of the YeS OS control program;

The time-sharing mode (SRV) and interactive remote job entry (DUVZ) is supported in all versions of the control program;

YeS EVM-2 disk storage facilities with 100M and 200M bytes with sector search are used in all versions of the control program;

In the SVS version, the user is offerred a new type of data organization on direct access devices and the corresponding access method, the VSAM virtual access method;

A new component, the dynamic debugging monitor (DDM), has been developed and included in OS to enable debugging of user programs and extend the operating system capabilities; and

A new component, the generalized trace facility (UST), has been developed and included in OS to trace the operation of individual programs and obtain statistics.

To make use of the advantages of the new functional capabilities of the YeS EVM to the full extent, measures to raise reliability and immunity to failures were also taken at the software level. In the YeS OS, existing software facilities for recovery and logging of malfunctions were improved and new facilities included, and the capabilities of dynamic reconfiguration of peripherals and selecting alternative paths for peripheral access were extended. Programs for handling specific errors of YeS EVM-2 peripherals were developed and included in the corresponding components of OS.

The inclusion in OS of facilities for building multimachine configurations also helps solve the problem of reliability and meet growing requirements for throughput of computer systems.

7.2. General Structure and Functions of YeS OS

Closely associated with the development of information processing hardware is the development of operating systems oriented to specific structures and functional capabilities of CPU's and allowing efficient resolution of the problems facing the user.

The YeS OS operating system is an integral part of the computer system and is a software extension of Unified System computer hardware. It is based on the modular principle that allows selecting facilities for a specific computer system in accordance with user requirements. The process of adjusting an operating system to a specific application is called generation.

Functions of the YeS OS Operating System. In working with YeS OS, the user formulates his jobs in one of the high-level programming languages: PL/1, FORTRAN, COBOL, ALGOL or RPG or in the Assembler machine-oriented language. He then sends his work to the computer center in the form of a job consisting of a sequence of individual operations associated with the execution of the YeS OS defined programs. The operating system executes the job in three stages:

preparation for job processing;

execution of the individual steps of job processing; and

output of results.

Based on the job, the operating system forms a sequence of tasks making up the individual units of operations for the YeS OS. The operating system optimizes use of the resources in the computer installation by overlapping in time processing of several tasks belonging to various jobs. The operating system itself functions on the basis of tasks that are system tasks. The used principle of overlapping of execution of several different operations making up the basis for the mode of processing of several jobs is one of the versions of the multiprogramming mode of operation.

Needed to execute user jobs are certain hardware resources (such as main storage, IO devices and others) and logical resources stored in the form of programs or files. YeS OS special components coordinate the use of all resources in the computer system, allocating them among the different jobs in accordance with requests and priority (job control). Requests for computer system resources are a job component and are formulated in job control language.

The resources needed are allocated after a job is fetched from the job queue. The job is entered into main storage and put on the list of active jobs. Execution of the basic tasks located in main storage at the same time is controlled by special operating system programs (task management).

Results of the operation that emerge in executing jobs in the form of data are intended for slowly operating devices (printer, puncher, microfilm output) and may be preliminarily accumulated in storage with direct access. The accumulated information is output to the required devices after the job is completed under control of system output programs.

YeS OS Structure. The YeS OS operating system consists of a control program and a set of processing programs (fig. 41). The control program, intended to control the CPU for data processing in a computer system, consists of the following components:

job management;

task management;

data management; and

diagnostic facilities.

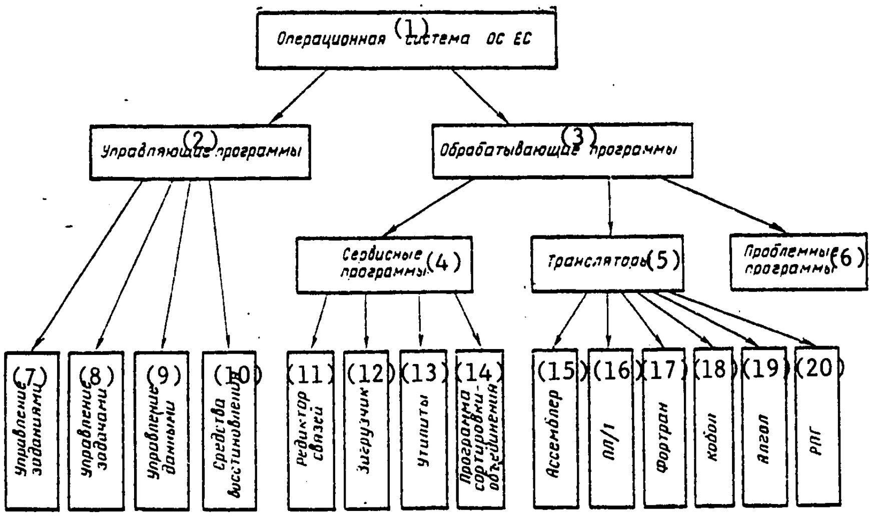


Fig. 41. YeS OS Operating System Structure

Key:

1. YeS OS operating system
2. control programs
3. processing programs
4. service programs
5. translators
6. problem programs
7. job management
8. task management
9. data management
10. recovery facilities
11. linkage editor
12. loader
13. utilities
14. sort/merge program
15. Assembler
16. PL/1
17. FORTRAN
18. COBOL
19. ALGOL
20. RPG

Processing programs are the translators and service programs supplied by the software developer as well as the programs written by computer users.

There are several different configurations of the control program that can be formed at generation time:

the MFT mode--control program with a fixed number of tasks;

the MVT mode--control program with a variable number of tasks; and

the SVS mode--system with virtual storage.

Under MFT, several jobs can be executed simultaneously. Except for the area for the nucleus of the operating system, all main storage is divided into fixed partitions. The number of partitions, equal to the number of simultaneously executable jobs, and the sizes of the partitions are defined at system generation time. The computer operator has the capability of reducing the number of partitions and changing their size.

Under MVT, the number of simultaneously executable jobs may vary dynamically. A storage partition is allocated for each job step and its size may also vary from job to job.

The maximum number of partitions used for job processing under MFT and MVT is 15 and all partitions have different values of protection keys.

In contrast to MVT and MFT, the number of partitions is not restricted under SVS, because not only protection keys but also the structure of dynamic addressing are used for protection of main storage. The number of partitions in which jobs are executed depends on the number of operating initiators. Other limitations on the number of simultaneously executable jobs under SVS are discussed in section 7.3.

The principles of functioning under SVS are considered in detail in section 7.3.

Operating System Modes of Operation. The mode of operation governs the basic organization of the process of processing by compuetr. The YeS OS may be used in two modes: batch and shared use.

Operating in the batch mode ensures continuous and efficient use of a computer. However, the time between receipt of the user job and return of the results to him may be substantial.

The user operating in the shared-use mode is linked directly to the computer and controls the processing from a terminal, which substantially reduces the time for obtaining results from execution of jobs. YeS OS offers the following facilities for operating in the shared-use mode:

conversational remote job entry (DUVZ); and

time sharing system (SRV).

Job Management. Job management facilities effect input of user jobs into the system and perform all actions associated with allocation of resources, execution of jobs and output of results to IO devices. In accordance with functions performed, the software for job management is divided into the job scheduler, that receives and processes jobs prepared by the user in the job control language, and the master scheduler, the facility for interaction between the computer operator and YeS OS. Jobs coming in for processing from remote terminals are controlled not by the job scheduler, but by the correspodning programs that are part of the shared-use facilities.

A programmer uses the job control language (JCL) to plan the processing of the job and request the resources needed for this. The language includes the JOB statement, the EXEC statement for a job step and the DD data definition statement. The JOB statement assigns the job name, class and priority. The job name is used to identify the job in operator communication with the operating system. The job class affects the process of forming the input queue, and the priority determines the order of job entry from the queue of the given class for processing. Also indicated in the job statement are the size of the storage area needed to execute the job and a number of other parameters specifying modes for job execution. The EXEC statement is used to indicate the program or procedure to be executed. A procedure consists of previously prepared job control statements that can be modified when referenced. Each job may contain several EXEC statements. For each data set to be processed or formed during execution of a job step, a DD statement must be provided to code the data set description, type and number of IO devices requested.

In the operating system, jobs are processed in several stages. In the first stage, the read-interpret program, a component of the job scheduler, reads the stream of jobs from the input device and forms the input queue in accordance with the job class and priority. If there is input data in the input stream among the JCL statements, it is placed for storage on external storage with direct access. Input of jobs into the system can be effected simultaneously by several input streams from different IO devices. Jobs in which JCL errors are detected are excluded from further processing.

Another componentof the job scheduler, the initiator, selects jobs from the input queue in accordance with class and priority, allocates the resources needed and sends them for processing, effecting subsequently the planning of each successive job step. Parallel execution of several operations results from the start of several initiators, each of which serves the jobs of well defined classes. Results of the operation of each job are sent to defined output classes. For each output class, the statement assigns a system output program and the corresponding external device. Operation of the system input and output programs and job processing are effected in parallel, which results in a high degree of automation of the process of job processing and high throughput of the operating system.

In job processing using the job entry subsystem (KROS), raising the rate of job processing is provided for by preliminary reading of the input stream and placing it in direct access storage. The statements are interpreted when the jobs are selected for processing. As a result, the job input unit operates at the maximum rate (fig. 42). After allocation of the needed resources, the problem program is laoded and receives control. Output data sets are stored on direct access storage. KROS ensures output of output files and system messages to the appropriate output device.

The user can control output by using DD statement parameters or through computer operator commands.

In loading the operating system and in the process of its operation, the computer operator communicates with the operating system. The operating system requests data on operator actions and decisions pertaining to removal and mounting of data media and reports the status of IO devices. In turn, the operator has the capability of controlling system operation, requesting information on its status, and changing parameters of the system and individual jobs to raise system operation

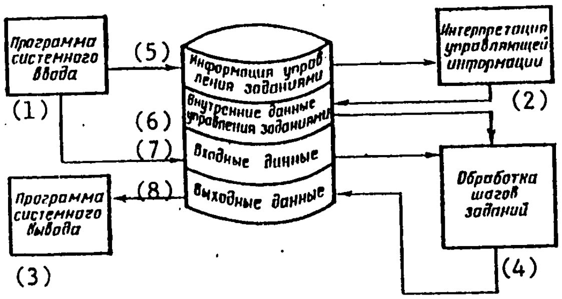

Программа системного ввода
(1)
(5)
Информация управления заданиями
(6)
Внутренние данные управления заданиями
(7)
Входные данные
(8)
Выходные данные
Программа системного вывода
(3)
Интерпретация управляющей информации
(2)
Обработка шагов заданий
(4)

Fig. 42. Structure of the KROS job entry subsystem

Key:

1. system input program
2. interpretation of control information
3. system output program
4. job step processing
5. job control information
6. job control internal data
7. input data
8. output data

efficiency. Using the appropriate commands, the operator starts system input and output programs, monitor programs and the shared-use mode.

The master scheduler communicates with the operator for the system. In generating an operating system, both a single-console and a multiconsole mode for its operation may be selected. In the latter case, the capability is available for using up to 32 devices operating as consoles. In doing so, one device must be selected as the basic console since certain commands can be sent only through this device.

Task Management. YeS OS ensures efficient passage of user tasks thanks to operation in the multiprogramming mode. Operating system multiprogramming facilities that allow simultaneous handling of several tasks on data processing permit these tasks to make joint use of system computer resources. Steps belonging to the same job are executed in strict sequence. Overlapping of execution is possible only for steps of different jobs. However, within a step, several tasks or subtasks may be formed that will be executed at the same time with each other and with other tasks. Processing efficiency is raised substantially through use of programs used in parallel that can serve several different tasks at the same time, which saves main storage (it is sufficient to have one copy of the program in storage) and reduces program loading time.

The supervisor is the main task control program. Frequently used programs of the YeS OS supervisor are part of the nucleus kept permanently in main storage. The other programs are loaded to main storage as needed. The main function of the supervisor consists in servicing other operating system programs as well as computer user programs. The supervisor prevents an uncalled-for effect of programs on each other and on the operation of a control program, making use of storage protection facilities and effecting complete control over IO operations.

In operating in the multiprogramming mode implemented in a single-CPU computer installation, a maximum of one task can be processed by the CPu at any given time. This task is called the active task. The active task is selected from those ready based on priority in accordance with the dispatcher algorithm. When an active task can not make use of the CPU, for example while awaiting the result of an IO operation, or when a higher priority task becomes ready, a new task is selected and made active.

The task control program receives control of the CPU after interruptions. The cause of interruptions may be the emergence of an event that a task is waiting for. In processing interruptions, information ne 'ed to restart operation of the interrupted program is stored. Further processing essentially depends on the cause of the interruption. For example, in the virtual storage mode, in event of an interruption due to the absence of a needed page in main storage, the mechanism for retrieving and loading the required page to main storage begins operating. An interruption may occur as a result of a special request for supervisor facilities from another operating system program or a computer user program.

Data Management. YeS OS standard data management programs support user program in data processing. Considering that the entire operating system is kept in data sets on volumes of external storage, data management facilities, in organizing exchange of information betweeen external and main storage, play a key role in the organization of control and processing processes in a computer system. Data management implements control of I/O operations, initial processing of data, overlap of IO operations with processing and protection of data sets from unauthorized access. The operating system has a catalog used by data management facilities to identify and find any data set. An increase in the number of programs and data during system operation leads to an increase in the size of the system catalog.

A data set is a collection of logically associated records with a specific structure. The operating system offers the user the capability of using fixed, variable and undefined-length records (fig. 43). A record is considered the smallest logical unit of information transferred between a program and data management facilities. To raise the efficiency of information transfer and make fuller use of the surface of the magnetic media, several logical records are combined into a block that is written/read to/from the medium in one reference to IO facilities. Blocking and unblocking are performed by data management routines. The blocking method depends on the type of data used. In using fixed-length records, the block size, as a rule, is a multiple of the record size; in the case of variable-length records, an additional field describing the total block length is included in the block. The length of a variable-length record is contained in the initial bytes of each record.

Data set characteristics are usually stored on external storage with the data set itself and reflect data set organization, location and size.

The operating system provides an extensive set of IO routines (access method routines) that free the programmer from writing routines to acess data sets. These routines automatically perform such functions as overlap of data processing with IO operations, and preparation and checking of labels for volumes and data sets. Instead of using YeS OS data management routines, a programmer has the capability of writing his own channel routines and IO are management routines. A programmer may request data by a similar method irrespective of the features of a specific IO device.

Key:
1. fixed-length records
2. blocks
3. records
4. blocked
5. interblock gaps
6. unblocked
7. variable-length records
8. block length
9. data record
10. data
11. record length
12. undefined-length records

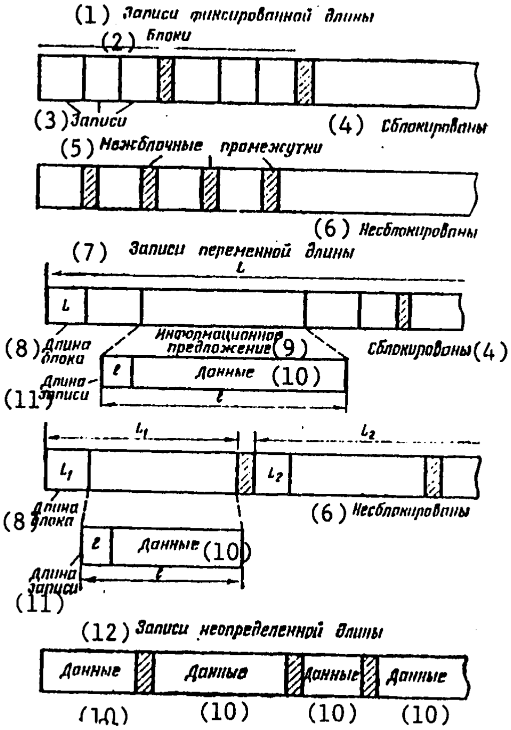


Fig. 43. Fixed, variable and undefined-length records

The YeS OS permits the following types of data set organization: sequential, direct, library, indexed sequential and telecommunications. Let us note also the organizations associated with creating microfilms and the generalized form of organization of files on direct access devices that is characteristic for devices with a capacity of 100M and 200M bytes.

Sequential organization of data sets is characteristic for the majority of existing devices and is oriented to sequential processing of records without skipping.

In indexed sequential data sets, records are ordered by the value of a key that precedes each record. Records in this data set may be processed both sequentially and randomly. In the latter case, the key of the required record has to be specified. Similar capabilities are offerred by the direct organization of data sets that permits an arbitrary value of the record key.

A data set with partitioned organization consists of a collection of sequentially organized partitions. A directory is located at the beginning of the data set. A record in the directory is selected by keys, and within the bounds of a partition, sequentially. This form of organization is used to store program libraries and groups of logically interrelated sequential files of data.

Data sets with partitioned, direct and indexed sequential organization may exist only on direct access devices.

Messages sent over communication channels and kept in an IO buffer are no different from ordinary data obtained from local devices, and the telecommunications organization is provided for message queues.

Data management facilities provide for two data access methods: the method of access with queues that automates execution of IO operation to the maximum, and the basic method that permits a programmer to directly participate in managing IO operations. The queued access method effects automatic synchronization of processing programs and execution of IO operations.

An access method may be defined as the aggregate of the data set organization and the access method used used to process the data set. Basic access methods are intended for all types of data set organization, but queued access methods are used only for sequential, indexed sequential and telecommunications data sets. The list of access methods supported by YeS OS in all modes of the control program is given in table 29.

Table 29

Data set organization (accesses)	Access Method	
	Basic	Queued
sequential	BSAM sequential method	QSAM sequential
indexed sequential	BISAM indexed sequential	QISAM indexed sequential
partitioned	BPAM for data sets organized by partitions	
direct	BDAM direct access graphic access method (GMD)	
remote processing of data	method of access for remote processing of data (BTMD) [BTAM]	extended method of access for remote processing of data with program control of messages (OTMD) [TCAM]

It is important to note that the graphic access method (GMD) [GAM] is organized in a way that a request for transfer of data may originate not only from a program, but also from a display operator. The capability of identification of messages sent in random sequence between terminals and main storage is provided in the GAM in data teleprocessing.

BTAM and TCAM differ in the number of services offerred the user. BTAM, the simpler and more economical method, is a series of macro instructions for organizing IO and translating messages. Using these macros, the problem program participates directly in controlling the teleprocessing hardware. TCAM presumes use of a special message control program oriented to a specific configuration of a teleprocessing network. As a result, the problem program becomes independent of the specifics of the teleprocessing network.

BTAM and TCAM are considered in more detail in sections 7.5 and 7.6.

VSAM has been developed for virtual storage systems (see section 7.9).

Recovery Facilities. Software recovery facilities are part of the OS control program and process interruptions from the machine check circuits.

Recovery facilities enable acquisition of information on system errors and the time of their appearance, recovery of the serviceability of a task or the system after an error occurs or emergency termination of processing if system recovery is unsuccessful.

Recovery facilities are discussed in more detail in section 7.13.

Programming Languages. The YeS OS user is offerred various programming languages to compile programs: In addition to the Assembler machine-oriented language, he may use the problem-oriented languages of PL/1, FORTRAN, COBOL, ALGOL and RPG. Problem-oriented languages are closer to conversational language and mathematical notation than to machine instructions. Using problem-oriented languages to write an algorithm makes it independent of the computer instruction list. The program is written in brief form and outlays for developing it are far less than when programming in Assembler.

Problem-oriented languages have certain limitations with respect to external devices used and functions of the operating system, but at the same time do not require precise knowledge of the operating system and specifics of specific devices on the part of the programmer.

Programs written by the user are teh source data for the corresponding translators that translate the source data into machine codes.

The Assembler machine-oriented language is maximally adapted to computer features. Programming outlays are rather high compared to using the problem-oriented languages. Assembler is divided into the basic language and a macro language. Mnemonics are used to code machine instructions.

PL/1 is a multipurpose language suitable for both commercial and scientific and technical problems. The broad range of application of this language is characterized by the wealth of different types of data. PL/1 offers the user a large set of standard functions. There are several different versions of the compiler for this language. The standard compiler translates source statements into machine code and produces a listing of the translation and messages on errors detected. The optimizing compiler generates object code that results in reducing the time for program execution and substantially reducing storage taken up.

The debugging version of the PL/1 compiler is intended for operation in the interactive mode with the capability of checking individually input statements of the source language. The optimizing and debugging versions of the PL/1 compilers may be efficiently used in the time sharing mode.

FORTRAN is a problem-oriented language for representation of formulas and calculations in science and engineering. It has a simple structure and is easy to learn.

Computational operations are in the form of expressions in which arithmetic and logic operations as well as operations with binary strings and compare operations may be performed.

When the source text is compiled, a listing is produced of the translation and messages on syntactic errors detected.

There are different versions of the compilers. The ST (standard) FORTRAN compiler has additional debugging facilities. The OP (optimizing) FORTRAN compiler is used to generate optimized machine code.

The SS interactive version of the FORTRAN compiler is intended for use in the time-sharing mode, in which the new expanded versions of the standard and optimizing compilers, FORTRAN SE and FORTRAN OE, may also be used.

RPG is a problem-oriented language developed for solving simple commercial problems which requires processing of extensive data sets using simple operations and output of them on a printer. Only alphanumeric strings and decimal numbers with fixed point are used as data. The RPG compiler prints out the translation, detects syntactic errors and corrects some of them. The new expanded version of the RPG-2 compiler has autoreport facilities.

ALGOL is one of the early problem-oriented languages. Programs written in this language have a blcok organization. The compiler supports all capabilities provided in ALGOL-60.

COBOL is a problem-oriented programming language oriented to commercial applications. Programs in this language process numeric and alphanumeric data.

COBOL compiler capabilities under YeS OS are the same as those under YeS DOS. The only difference is in system-oriented elements.

The new expanded version of the COBOL compiler includes additional facilities enabling its efficient use in the time-sharing mode. There are English and Russian language versions of the COBOL compilers.

Service Programs. These YeS OS programs include the linkage editor, the loader, the sort/merge program and YeS OS utilities.

The linkage editor and loader combine load modules of separately compiled parts (object modules) and other load modules. The input data for these components consist of object and load modules and control instructions. In the linkage process, links are formed between the different parts of the load module in a way as if all parts of the program had been translated together. The linkage editor and loader form a load module in relocatable formIn the loading process, the module is adjusted for a specific location in main storage.

The load module may have an overlay structure (structure with planned overlap), for which it is divided into individual segments loadable into main storage in accordance with a scheme specified by the user. This allows saving on main storage needed for execution of the module. In operating with a virtual storage system, the overlay structure loses its value.

The sort/merge program sorts or merges fixed or variable-length records in ascending or descending order. This program may be used with Assembler, COBOL and PL/1.

YeS OS utility programs include standalone service programs, system service programs, data set service programs and system programmer service programs.

7.3. Virtual Storage Support

The operating system supporting virtual storage in the YeS EVM-2 is an evolution of those versions of YeS OS used with YeS EVM-1 models. Based on a number of hardware and software facilities, the user is offerred virtual storage with a capacity of 16M bytes, which substantially exceeds the size of computer real main storage. Let us note a number of advantages stemming from the use of virtual storage.

1. Using virtual storage, the user has little need for the overlay structure and subdividing tasks into a sequence of smaller tasks. Programs are automatically overlapped based on the page swap mechanism.

2. Real storage size ceases to be a critical factor in locating components of the operating system and user tasks, since they are located in virtual storage. In extending the real storage of a computer installation, operating efficiency automatically increases based on the concept of paging.

3. Real storage fragmentation is reduced. Using the page organization for storage management eliminates fragments of real storage between and within partitions. Fragments may exist only in virtual storage. In real storage, fragmentation is substantially reduced.

4. The rather efficient apparatus of dynamic management of a basic resource of a computer installation, real storage, is realized. If a task is not very active, because of paging it will use an insignificant amount of real storage.

5. The degree of multiprogramming rises significantly, since the most active pages of a substantially greater number of tasks than in a conventional system may be present in the same real storage.

6. The system has a greater area for resident functions that may be used simultaneously by many tasks. Virtual storage can hold a great number of supervisor programs and access method programs. When they are not being used, they do not take up real storage.

The virtual storage system is supported by the control program SVS which implements multiprogramming with a variable number of tasks that use virtual storage at the same time.

Virtual Storage. The main aspects of the function and use of virtual storage may be described by using the concept of address space. There is address space for a program, understood as the tange of variation of addresses of its instructions and data, and address space of real storage with reflects the capacity of computer main storage. There is also the address space of a computer installation that describes the maximum possibel size of real storage that can be connected to the computer installation. A computer system with a 24-bit address may have an address space of

16M bytes. This concept of address space that may be many times larger than real storage is called virtual storage.

The traditional principles of multiprogramming that became widespread in the computer systems on the base of the YeS EVM-1 were based on the total size of address spaces of programs being processed at the same time not exceeding the address space of real storage. The placement of programs in real storage in these systems was based on the concept of static relocation that consisted in translating program addresses at loading time in accordance with its location in real storage. Increasing the degree of multiprogramming was curbed by teh sizes of real storage and the presence of unused fragments between the partitions occupied by tasks.

The use of virtual storage is based on using the strategy of dynamic relocation implemented by the facilities of dynamic address translation (DPA) [DAT] and on using the segment-page organization of the address space and external page storage.

Dynamic translation of virtual addresses into real is based on the segment-page organization of virtual storage. A virtual address has the structure (S, P, i) (see fig. 4), where S is the segment index, P is the page index and i points to the specific address (displacement) within a page. In the YeS EVM-2, 4 or 8 bits are allocated for coding the segment index; therefore, there may be either 16 1M-byte segments or 256 64K-byte segments in virtual storage. For coding the displacement of i, 11 or 12 bits may be allocated, and page size is 2K or 4K, respectively.

One-megabyte segments may contain, as a function of the structure of the virtual address, 512 or 256 pages, while 64K-byte segments may have 16 or 32 pages. Translation tables, a segment table and several page tables, are used in the operation of the hardware facilities for DAT. A segment table consists of elements that, as shown in fig. 44, define the availability of segments and the length and addresses of the corresponding page tables. Page tables consist of elements that indicate the availability of a page in real storage and the high-order bits of its real address. There is an entry in the page table for each page of virtual storage. The structure of a page table entry is shown in fig. 45.

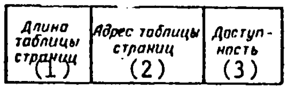


Fig. 44. Structure of segment table entry

Key:
1. page table length
2. page table address
3. availability

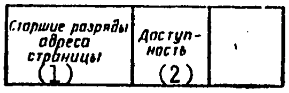
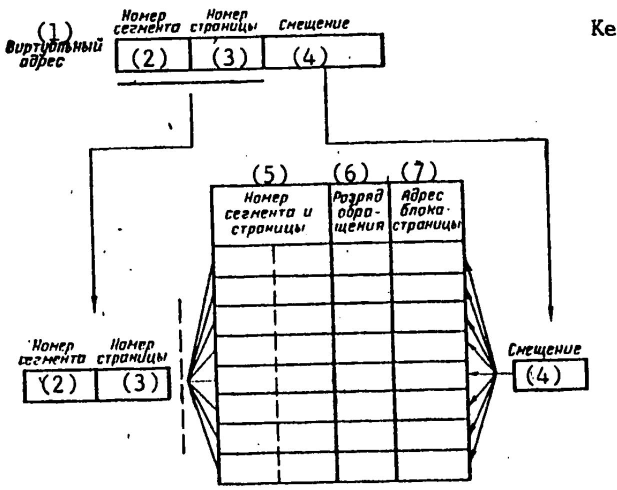


Fig. 45. Structure of page table entry

Key:
1. high-order bits of page address
2. availability

The beginning address of the segment table is stored in control register 1. Segment and page tables must be fixed in real storage.



Key:
1. virtual address
2. segment index
3. page index
4. displacement
5. segment and page index
6. reference bit
7. page block address

Fig. 46. Associative Registers

Program interruptions with codes X'10', X'11' or X'12' may occur in the process of dynamic address translation that indicate that reference was made to an unavailable segment of virtual storage or an unavailable page or an invalid address was detected for the segment or page table.

Special hardware facilities have been provided to increase the translation rate. Associative registers are used in small models, and a rapid translation buffer in the large ones. A system of eight associative registers is shown in fig. 46. In referencing storage, the part of the virtual address containing the page and segment index is compared with the first part of the registers. The registers hold information on the eight last used pages, identified by their segment and page indexes. If there is a match in one of the registers, the real address is generated based on the displacement and the contents of the right portion of the register. If there is no match, the tables are used for translation. The information obtained on the location of pages in real storage and the correspodning segment and page indexes are stored in one of the registers in accordance with the replacement algorithm.

External Page Storage. Since virtual storage is many times larger than real storage, the latter holds only the most active pages. Pages to be used are kept in external storage in the system data set, SYS1.PAGE, that is called the page data set.

In the process of translating virtual addresses, when the system detects that the required page is not in real storage, a copy of this page is moved into real storage. The location of pages in external storage are reflected in a special table, called the external page table.

Paging organization is a function of the operating system component called the page supervisor.

Page Supervisor. From the viewpoint of paging organization, real storage is divided into fixed and page-replaceable (dynamic) areas. The special table reflects the list of real storage pages available for exchange and their status.

The page supervisor performs the following basic functions:

management of the segment table, page tables and real page tables;

start of operations associated with page swapping;

processing of interruptions associated with the absence of needed pages in real storage;

management of page replacement in real storage; and

management of the intensiveness of page exchange.

The algorithm for replacement of pages in real storage is the basis for page exchange.

Two replacement algorithms affording common interface with YeS OS components have been implemented in the page supervisor. The YeS OS user-programmer has the capability of including the required algorithm in the specific version of YeS OS when it is generated.

Translation of Channel Programs. In contrast to the CPU, YeS EVM IO channels can not translate virtual addresses into real ones. Therefore, virtual addresses used in channel programs have to be translated into real ones by software. Channel programs with virtual addresses are translated into logically equivalent programs with real addresses. Because of this, access methdo rpograms that exist in previous YeS OS editions may be used with virtual storage with no change. The channel program is translated right before the start of the IO operations. Created in the system queue area is the logically equivalent duplicate of the channel program, but now with real addresses, which is used in starting the IO operations. In such translation, it may happen that all addresses required are in non-adjacent real pages. In the YeS EVM IO channels, there is the hardware facility of indirect addressing that allows sending data to nonadjacent pages of real storage.

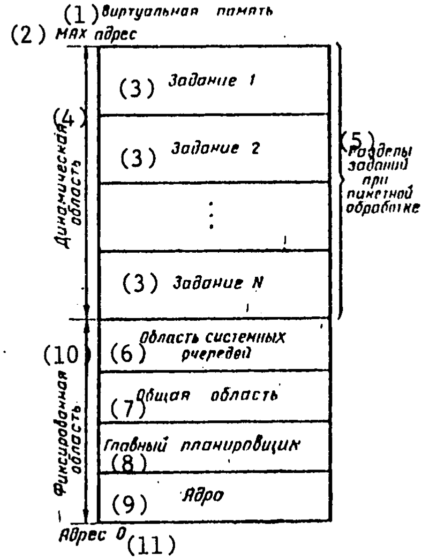
During execution of the IO operation, all needed pages need not undergo page exchange. Before starting the IO operation, these pages are fixed in real storage, and unfixed after successful completion of it.

Jobs in which dynamic modification of channel programs is effected during execution of IO operations are executed in a special mode, when the the virtual addresses of the instructions and data match their real addresses. In this case, channel program translation is not required.

Channel program translation facilities are part of the IO supervisor. Channel program translation does not require fixations on the part of the programmer and is an internal function of the operating system.

7.4. Virtual Storage Mode SVS

The mode of multiprogramming with a variable number of tasks, that make joint use of virtual storage (SVS), is oriented to YeS EVM-2 hardware facilities and is a practical implementation of the concept of virtual storage in the YeS EVM. It was developed on the basis of the MVT mode and affords multiprogram execution on a compuer of programs, the total size of which substantially exceeds the capacity of real storage.



Key:
1. virtual storage
2. MAX address
3. job (1, 2, ... N)
4. dynamic area
5. job partitions in batch processing
6. system queue area
7. general area
8. master scheduler
9. nucleus
10. fixed area
11. address 0

Fig. 47. Structure of virtual storage in SVS mode

Virtual storage is used jointly as one address space by both the control program and all computer users in accordance with the principles of allocation of storage in the MVT mode. Virtual storage is divided into two areas: the fixed (containing the nucleus of the control program, the system queue area, the general area, and the master scheduler partition) and the dynamic, in which partitions are organized for execution of jobs (fig. 47). The maximum size of the dynamic area is

$$V = 16\text{M bytes} - V_{fix}$$

where V_{fix} is the size of the fixed area.

The virtual storage size provided with a specific loading of YeS OS is governed by the size of the page data set used, since the more space for page storage, the greater the size of virtual storage.

Job execution in the dynamic area is affected by page size: program interruptions occur when pages are not in real storage and there are delays in program execution due to page transfers. This effect can be avoided, if desired, by fixing in real storage both individual apges and entire partitions. Therefore, under SVS, there are two types of partitions: with virtual ddresses (V=V) and those in which virtual addresses equal the real (V=R).

Virtual storage for V=V partitions is allocated by segments. The size of a V=V partition is set to a multiple of segment size, even when less storage is requested in a job for the execution of which a partition is created. The same protection key, equal to one, is set for all these partitions, because storage protection is organized not only by using protection and storage keys as in the MVT mode, but also by using the translation tables. According to the principles of operation of the YeS EVM-2, in the segment table used in dynamic address translation, in the entry for each segment there is a bit for the "availability-unavailability" of access to this segment. Any reference to the segment (either a write or read) for which this bit is set as "unavailable" causes a program interruption, perceived by YeS OS as an interruption due to a storage protection exception.

In a transfer of control to a task being executed with a nonzero protection key, availability in the segment table is indicated only for partition segments and segments occupied by YeS OS. In a transfer of control to a task being executed with a zero protection key, i.e. having the right of access to any partition, availability for all segments is indicated in the segment table.

To reduce the time required to readjust segment tables when tasks are switched, two segment tables are created that differ only in the availability bit setting: a user table, in which availability is indicated for partition segments and those taken up by the computer YeS OS, and a system table in which availability is indicated for all segments. Because of this, readjustment of segment tables is required only when tasks are switched that are being executed in different partitions. In transferring control between system programs of the user, readjustment of tables is not required; loading the address of the corresponding table into the control register suffices.

In transferring control to tasks being executed with a zero protection key, the address of the system segment table is loaded for translation; in transferring control to tasks being executed with a nonzero protection key, the user table is used.

For V=R partitions, virtual storage is allocated by pages. The allocated pages are fixed in real storage, and the page and segment tables are adjusted so that the virtual addresses equal the real in dynamic address translation. The size of this partition may be less than the segment size and a multiple to the page size. For storage protection in this case segment tables are not used: various protection keys not equal to 0 or 1 are set for the partitions. Consequently, there may be no more than 14 V=R partitions. They are created in the real storage area directly following the nucleus of the control program. The size of this area is set during generation of the specific version of YeS EVM OS. During loading of it, the computer operator has the capability of changing the size of the area for creating V=R partitions.

In the partitions of both types, local system queue areas (LSQA) are created in which control blocks related only to the given partition are placed (under MVT, they are placed in the SQA [system queue area]). This substantially localizes references of tasks to pages of virtual storage, for example in processing storage control block queues, and reduces the number of references to pages not present.

One segment of virtual storage is allocated for the LSQA. It is allocated by the control program automatically, above the size of virtual storage specified in the job for the execution of which the partition is created.

This organization of partitions and storage protection under SVS simplifies the restriction on the number of partitions under MVT: The number of partitions created is limited not by the number of protection keys, but by the size of virtual storage provided. Page exchange for the fixed and dynamic parts is organized as follows:

in the fixed part, pages containing the nucleus and SQA storage sectors, allocated by the virtual storage control program, are fixed in real storage and not subject to page exchange; pages containing the general area and the master scheduler partition are subject to page exchange;

in the dynamic part, pages containing V=R partitions and LSQA storage sectors, allocated by the virtual storage control program, are fixed in real storage and not subject to page exchange; pages containing V=V partitions are subject to page exchange.

For the page data set, it is necessary to allocate a size that enables holding the pages of the fixed part subject to page exchange and the needed number of partitions from the dynamic part. The size of pages in the fixed part subject to page exchange varies with the different YeS OS versions and averages about 3M bytes. The size and location of the page data set are set only when YeS OS is loaded.

Paging means that a page is kept in real storage only when it is referenced; it is withdrawn to the page data set by the paging algorithm as soon as the need arises for placing another page in real storage. Placing a virtual page in real storage or removing it may also be performed upon program requests.

Program requests for placing, fixing, unfixing and removing pages change the normal conditions for functioning of the paging algorithm, since they reduce the size of real storage accesible for organization of the page exchange. Therefore, the number of simultaneously fixed pages is a parameter that depends on the amount of real storage, the intensity of requests for replacement of pages and on the intensity of servicing them.

SVS components subject to page exchange make up a large part of the system and use external page storage in the process of operation. However, unallocated segments of the dynamic area of partitions does not use external storage. Similarly, all pages not used in the segments do not require external page storage. Unallocated segments and pages unused within segments are "potential" address space.

The distribution of pages in the page data set is reflected in the external page table.

If an interrupt occurs due to unavailability of a page in the process of program operation, the system makes use of the external page table to determine the location of the needed page.

A page selected for replacement is moved to the page data set only if it is altered. Pages are stored in an order that ensures a balanced load on the devices holding the page data set and minimum movement of each device's reading mechanism. The control blocks describing the location of the external pages are used to determine the position of the reading mechanism and the free space closest to it to which the page is moved from real storage. This achieves minimum movement of the reading mechanism which reduces the time for page exchange.

To prevent unproductive, extremely intense exchange of pages that is possible when the number of problems executed at the same time increases, management of page exchange intensity has been implemented. The criterion for management is the page exchange intensity, i.e. the number of pages transferred between real storage and the page set within a unit of time. When paging increases to the value

$$I_{\max}=I_{\text{opt}}+\Delta_1 I,$$

where I_{opt} is the optimal intensity and $\Delta_1 I$ is the permissible increment of intensity, execution of one or more low-priority tasks is halted. When intensity declines to the value

$$I_{\min}=I_{\text{opt}}-\Delta_2 I,$$

where $\Delta_2 I$ is the permissible intensity decrement, execution of some halted tasks is resumed. The optimal intensity of paging I_{opt}, the permissible interval of change in paging intensity $\Delta I=\Delta_1 I+\Delta_2 I$, and the time interval used in measuring the intensity are the major parameters affecting the throughput of computer systems and depend on its structure, i.e. on the number of IO channels, types of IO devices intended for placement of the page set.

The main difference in the process of loading programs under SVS from that under MVT is that the programs are loaded not into real, but into virtual storage. Before loading, the program is in the system or personal library. The loading program reads the program to be loaded and translates the addresses of the program being loaded in accordance with the location of the partition in virtual storage. In doing so, the program being loaded autoamtically assumes the segment-page format and is withdrawn to external page storage in the paging process.

After completion of the program loading, control apsses according to the virtual address to the entry point of the loaded program. The required page will be placed in real storage and the loaded program starts its execution.

7.5. Basic Telecommunications Access Method

The basic telecommunications access method (BTMD) [BTAM] is used for creating programs that implement exchange of data between a computer and remote terminals by using communication channels. BTAM affors use of only half-duplex channels. Communication is established through switched or unswitched communication channels.

This access method provides the programmer with facilities for creating a subscriber list that contains the information needed to establish communication with a remote station. BTAM effects addressing and interrogation of terminals, sending and receiving of messages, insertion of changes to the the subscriber list, buffering and code translation.

In developing programs by using BTAM, execution of the following procedures that enable proper functioning of the teleprocessing hardware must be provided for:

definition of the teleprocessing system;

use of the code translation table;

opening and closing of data sets; and

management of the data transmission channel.

The teleprocessing system is defined by creating descriptions of channel groups, data transmission multiplexer parameters and terminal parameters.

The main aim of uniting channels into groups is the capability of making use of the common resources of the access method for all channels in the group. Therefore, channels with similar characteristics are united into a group.

Using the Code Translation Table. The YeS EVM internal code is the DKOI [decimal information interchange code], while the teleprocessing equipment sends information in other codes. The macro definitions of BTAM contain code translation tables; indication of the name of the needed table in the macro instructions ASMTRTAB and TRNSLATE by the programmer suffices. The programmer may also specify his own translation table if required for specific applications.

Opening and Closing of Data Sets. The procedures for opening data sets of communication channels are performed by using the macro instruction OPEN. As a result, BTAM programs are loaded into main storage and auxiliary tables are generated. If it turns out that some channels were not opened due to their unavailability for transmission, one can later open a single channel by using the maco instruction LOPEN. After completion of data transmission, data sets are closed by using the macro instruction CLOSE.

Data transmission channel management consists in establishing communication between the computer and the remote station and adhering to the algorithm for transmission of messages over the communications channel. This management is effected by using the access method programs taking into account the specifics of the operation of the MPD--APD-AP [data transmission multiplexer--data transmission equipment-terminal] link. In the process of receiving and transmitting information, a block by block check is made by using the check sum. When information is received with an incorrect check sum, the block in error is retransmitted.

The standard READ and WRITE macro instructions are used for message transmission. To establish communication with a remote station and send the first block of data, these maco instructions are used in the modes of intial read and initial write. There is the capability of transmitting data in the transparent mode, when any binary code combination can be transmitted as data, even if it matches the control symbol code.

7.6. General Telecommunications Access Method

Compared to BTAM, the general telecommunications access method (OTMD) [TCAM] is at a higher logical level. Using TCAM frees the applications programmer from having to take into account in programming the features of the algorithms for data transmission through the communication channel, to program terminal polling and sampling and code translation, and to define the network configuration. All this is performed in a separate task by using the message control program (PUS) [MCP].

The MCP is generated for a specific teleprocessing system by using macro instructions and TCAM program modules; it operates in the partition with the highest priority. The MCP has several sections:

activation and completion of the teleprocessing system operation;

definition of data sets needed for MCP operation;

definition of the teleprocessing network configuration; and

message handlers.

The first sectiop of the MCP must be the section for activation and completion of the teleprocessing system operation. The first macro in the section is INTRO, which defines the basic parameters of the MCP. OPEN opens the MCP data sets. READY transfers control to the MCP dispatcher which plans all MCP operations; if there are none, it is placed in the wait state. After completion of the MCP operation, control is passed to the macro CLOSE, following the macro READY, which close the MCP data sets; then, control is returned to the operating system by using the RETURN macro.

The data set definition section contains macros:

for the groups of communication channels, for a data set, for the message queue, for the MCP system log, for check-point data sets, as well as PCB macros for user application programs.

The section for defining the teleprocessing network configuration contains macros that describe communication channels, terminals and application programs. In this section, polling lists are built and other parameters that define system configuration are specified.

The section for message handlers contains macros used to specify unique handling of messages circulating in the system for each group of terminals or application programs. In particular, message handler macros can be used to control putting messages into a queue in accordance with priorities, edit messages, control routing by sending messages to the needed destination queue, translate messages from the transmission code to computer code and vice versa, etc.

The MCP keeps message queues on NMD [magnetic disk storage] and in main storage; it has facilities allowing the computer operator to control network operation by using commands issued from the console. The facility for creating checkpoints allows resumption of MCP operation at the point previously planned after computer malfunctions.

The MCP has facilities affording interface between the MCP and user application programs.

User application programs operate usually in individual storage partitions; they interact with terminals through the MCP and by using the usual sequential access methods in YeS OS. This allows writing application programs in high-level languages and debugging them in advance without using the data teleprocessing devices.

7.7. Remote Job Entry Mode

Conversational remote job entry [CRJE] facilities allow job entry from remote terminals, connected to a central computer by communication channels, and from local displays. They offer the CRJE user, working at a terminal, the same capabilities the ordinary YeS OS user has at a central computer installation. The CRJE ensures efficient use of central computer equipment through joint use of computer resources by several users at the same time. There is also the capability of joint use of information media by several users executing jobs similar in content.

The CRJE system operates within a system task similar to system IO tasks. A terminal keyboard and printer (or display) are available in the CRJE system for input of source information and output of final results. Jobs sent by a user come into the operating system for subsequent planning and execution. When a job input from a terminal completes its operation, all its output is placed in a special class for the CRJE system. Then the contents of this class are sent to the appropriate user terminal where the job was input. The CRJE system enables creation of new data sets, every possible operation on them and modification of any accessible information, including jobs immediately prior to their input into the system stream of jobs at the central computer installation.

There are three modes of user operation in the CRJE system: command, edit and input.

When a user terminal is connected to the system, operation occurs in the command mode. The user switches to the edit or input mode by using the EDIT command. In the edit mode, commands are entered that are used to perform different operations on the data sets. Operation in this mode is terminated by entering the END command. The input mode is used to create a new data set or add records to an existing file.

In the edit mode, the terminal user may request syntactic checking of PL/1 or FORTRAN statements, if the appropriate syntactic checking programs have been included in the system. Checking is done on a single statement; therefore, errors will not be detected when a check of the relations between statements is needed to find them. For statements entered in the input moae, there is the capability of automatic checking. When errors are found, an appropriate message is sent to the terminal user.

In the command mode, the CRJE system user has the capability of inputting jobs from data sets located in the terminal user's library and checking the process of its execution at the central computer installation by requesting information on job status. The terminal user may cancel a job input by him at any time. By using commands, the terminal user may obtain information on the status of data sets prior to job execution. When job execution ends, the terminal user receives a message indicating normal or abnormal end. This means that job output is available for the user. By special command, all the output from the given job, placed in the output class for CRJE, is sent to the terminal. If the terminal does not have special interruption facilities, the user may use the capability of receiving the output in groups containing a specific number of lines. Information output may be interrupted after any group. If the terminal is equipped with interruption facilities, the user of CRJE may interrupt output at any time. The data set in the output class for CRJE is erased after it is completely output to the terminal. If output is interrupted during data set transmission, this set is not erased. When necessary, the user may resume information transmission, as well as sent it to the printer at the central installation.

7.8. Time Sharing System

The YeS EVM software includes the time sharing system (SRV) [TSS], implemented in the form of an independent program functioning within YeS OS. TSS assumes shared, conversational and simultaneous use of a computer system by several users. This system offers users the most advanced form of multiprogramming and functions under control of YeS OS/MVT versions.

TSS offers a terminal user a sufficiently universal method of control of the computing process, implemented in the form of a command language and suitable for many practical applications. Terminal software is based on YeS OS TCAM. Including TSS in YeS OS has no effect on the amount and nature of functions offered computer users during operation in other processing modes. The capability is provided for combining TSS and batch processing modes.

In a functional sense, TSS capabilities are considerably broader than those offered users by the CRJE system. The fundamental differences between CRJE and TSS are as follows. TSS jobs are not put into the usual job queue, but are executed directly under user control. In contrast to CRJE, which operates at the system task level, TSS affects the operation of the YeS OS control programs. The number of TSS users whose jobs execute in parallel exceeds considerably the number of CRJE system users because of the efficient use of external storage used to hold programs when they are withdrawn from main storage and awaiting their turn for execution.

A command language similar to that in CRJE is used in TSS. The syntax of problem-oriented language statements is checked during input. Error messages are output to the terminal.

TSS Operating Algorithm. The set of jobs to be executed in TSS form a queue that is placed in external storage. Jobs are successively entered from the queue (fig. 48) into real storage and processed within a specific time slice Z, which depends on queue service time T and the number of jobs K.

$$Z = \frac{T}{K}$$

Selection of the tiem slice depends on a number of contradictory factors. The time slice must be sufficient for the system to be able to perform a certain quantity of useful work and at the same time ensure a sufficiently quick response to user requests, equal to the queue servicing period. At the same time, a decrease in the slice leads to a decrease in the frequency of replacement of jobs in real storage and increases system overhead for transferring them.

The selection of these parameters is affected by the number of partitions in real storage allocated for execution of jobs in the TSS mode, the specific composition of the job mix and a number of other factors.

Jobs in real storage at the same time make use of the resources of the computer installation during smaller time slices just as is done in the usual batch processing mode (fig. 49).

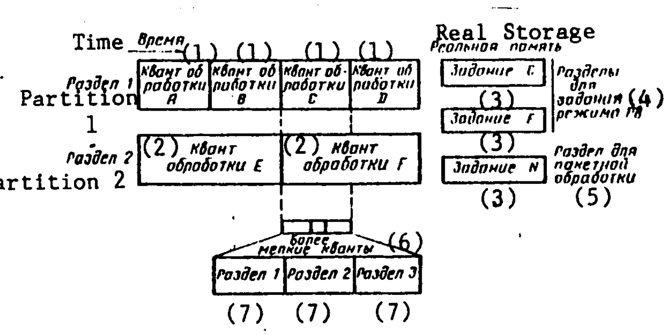


Fig. 49. TSS job processing together with batch processing jobs

Key:
1. processing slice (A, B, C, D)
2. processing slice (E, F)
3. job (C, F, N)
4. TSS job partitions
5. batch processing partition
6. smaller slices
7. partition (1, 2, 3)

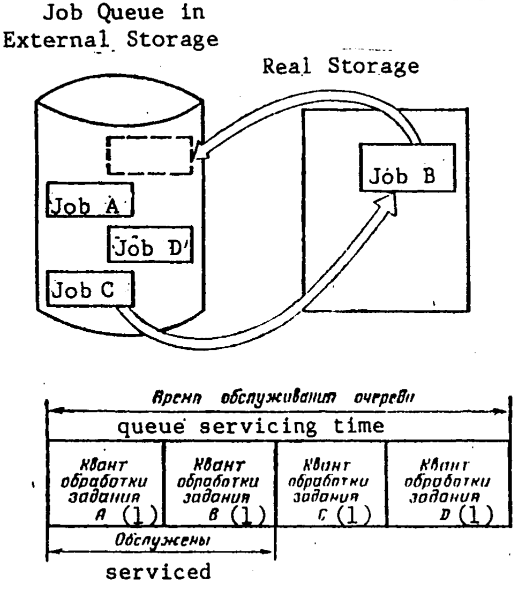

Job Queue in External Storage
Real Storage
Job A
Job D'
Job C
Job B
Время обслуживания очереди
queue servicing time
Квант обработки задания A (1)
Квант обработки задания B (1)
Квант обработки задания C (1)
Квант обработки задания D (1)
Обслужены
serviced

Fig. 48. TSS job processing

Key:
1. job processing slice (A, B, C, D)

The TSS control task functions under OS/MVT and controls TSS job exchange between external and main storage. When the operator issues the command TSS START, the TSS module receives control and forms the storage sector needed for the task in which buffer areas are built, partitions for job execution are formed and a partition control task is built for each partition.

In the process of functioning, the TSS mode control task effects replacement of jobs in real storage, while optimizing the use of resources of the computer installation. It builds channel programs for each request for job input or output. By using the apparatus of program-controlled interruptions, high efficiency of the job replacement process is achieved: the interrupt processing program completes the next channel program after the current channel program. Considering the high intensity of exchange, the process flows as one unending IO operation. Information on the location and sizes of the programs being transferred is formed in partition control task tables. If the operator issues the STOP command, each active user is informed of the termination of the operating session; the storage area occupied by TSS is returned to the operating system and operation is terminated.

The partition control task operates in each partition allocated for job execution in the TSS mode. It switches to active status those jobs given a partition for the current time period and monitors the conditions associated with the end of the servicing time. In particular, if IO operations are not complete at the end of a processing slice, the job is retained in the partition until the end of these operations.

LOGON-LOGOFF Handler. The functions of this task are similar to the functions of the initiator and read (interpretation) program that operate in the conventional batch processing mode. The LOGON-LOGOFF handler is loaded into a partition for planning of the new job when the immediate user is connected to the TSS.

The terminal control program is a monitor and the specific user operates under its control. As a function of the request, it retrieves the module needed for processing and passes control to it. After completion of processing of the request, the terminal is sent the message READY and the program is ready to receive the next command.

The message control program (PUS) [MCP] is a TCAM component. It contains the description of the configuration of terminals connected to the system and the IO areas for storing information transferred between main storage and the terminals; it effects data exchange between the IO areas and buffer areas of TSS. The MCP functions in YeS OS as an ordinary problem program in a real storage partition. It must have the highest priority among the problem tasks operating in the operating system.

Command and User Program Handlers. At this level of control are effected checking of the correctness of commands, correctness of operand coding, calls of modules that implement the standard servicing algorithms, and programs prepared by the TSS user.

7.9. Virtual Storage Access Method

The virtual storage access method (VSAM) is based on the concept of virtual storage and is used in systems with virtual storage; therefore, data sets with virtual organization are stored as pages. This access method is most efficient when disk storage with 100M and 200M-byte capacity is used.

This access method combines all the functional capabilities of all the other access methods used in the MVT and MFT operating systems, except operation with partitioned data sets. VSAM substantially reduces data access time, especially because with the indexed structure, the process of adding new records is organized without use of overflow areas. VSAM offers the capability of indexing data sets by several keys at the same time, which is very important for ASU [automated control systems].

7.10. Facilities for Building Multimachine Complexes

Hardware facilities have been provided in the YeS EVM-2 to implement multimachine systems consisting of several CPU's and the corresponding IO devices and channels. Multimachine systems allow increasing the overall throughput of a computer installation and have increased reliability when errors and malfunctions occur, since a computer going down in a multimachine system leads only to a reduction in throughput.

Computer complexing facilities (direct control facilities, channel-to-channel adapter, common extent of external storage) enable organization of communication and exchange of information between several computers in a multimachine computer system. Direct control facilities are used for rapid exchange of control and synchronizing information between two computers in a multimachine computer installation. A channel-to-channel adapter (AKK) may be designated for exchange between several computers in a system of both control information and data files at the rate of operation of IO channels. A common extent of external disk (NMD) or tape (NML) storage units allows two computers to make use of common information files stored in this extent.

Multimachine system software is implemented at two levels. On the first level, facilities are provided to control information transfer between CPU's, effected by problem programs. Data exchange through an AKK occurs similar to the operation with other IO devices with the use of the operating system access method. On the second level, multimachine system control is implemented, which optimizes distribution of the programs, subject to processing, between the various CPU's making up the complex. Central control of the complex is performed by the host machine which organizes distribution of operations.

A modified YeS OS control program, which monitors the entire system and controls the common job file, is used to control the multimachine system.

Appropriate software is provided within the YeS OS for each level of organization of the multimachine system.

Direct control software facilities presuppose data exchange between two computers connected by direct control interfaces.

The DIRECTWR macro is provided for access to another CPU through the direct control interface. Reception of information sent through the direct control interface by using the DIRECTWR macro is effected by YeS OS. Depending on the operating mode chosen, information feedback for monitoring may be initiated. Then control is passed to a procedure for analysis of the information received, which is a program interface between the programs for processing the received information.

In YeS OS, there are standard interface programs for direct control facilities and an asynchronous exit to a program prepared by a user is provided.

In asynchronous operation of two computers, a conflict situation may arise when requests for information exchange over the direct control lines come into the two computers at the same time. To overcome this, in the system loading process there is an exit to the console for the purpose of specifying the priority when simultaneous requests from two computers come in. Operator responses on the various computers in the multimachine system must be coordinated.

Program Control of Channel-to-Channel Adapter. In connection with the sequential structure of data sets sent through an AKK, two access methods, QSAM and BSAM, have been provided. Each provides its own set of macros to describe a data set and request IO operations.

Program checking of the correctness of transmission of information through a channel-to-channel adapter is provided in the YeS OS. The program generating a block of data must include in it the check sum

modulo 2^{32}. A secondary check of the check sum and comparison of it with the original sum are effected by the YeS OS programs. After reception and checking, the information is sent to the user program.

Program Control of the Common Extent of External Disk and Tape Storage. The common extent of external storage may contain data sets with any data organization valid in YeS OS and all valid access methods may be used for operation with them.

Appropriate macros have been provided to reserve the devices of any of the computers and to subsequently release them. After completion of a task, if the operating system detects a resource not released by the task, the resource is automatically released.

It should be stressed that the level of reservation for the common extent of external tape storage differs considerably from that for disk because of the sequential nature of magnetic tapes. Reserving a tape storage unit from the common storage extent actually results in reservation of all the tape units connected to the same control unit as the reserved unit. The specifics of operating with data sets on tape also imposes a restriction on the place of issuance of the macros for device reserving and releasing. Reservation is required to create and modify data sets on disk. It is not needed for reading data sets and in the intervals between data modifications.

When problem-oriented languages are used, the functions of reserving and releasing peripherals may be performed by a special program written in Assembler.

Tne software for the common extent of external storage is included in the YeS OS when it is generated. There is a restriction on system data sets, which cannot be placed in the common extent of disk storage. In operation with the common extent of external storage, special attention is paid to placement of a data set on certain extents, since the stipulated arrangement may result in conflict situations.

7.11. Dynamic Debugging Monitor

The dynamic debugging monitor (DDM) is designed to debug both YeS OS programs and user programs operating under operating system control. The DDM can be used to perform:

halts of the system being monitored when specified program events occur, which include: transfer of control according to a specified address, and occurrence of program events recorded by using the hardware facilities of program event recording (PER); and

recording of certain program events (various types of interruptions, times of execution of privileged instructions) pertaining to the operation of the system being debugged.

After a halt of the system being debugged at the specified point, one can perform:

a change in the status of the system of the contents of the general, control and floating-point registers, a specified area of main storage, and a PSW change;

a printout of the information reflecting the status of the system being debugged at the the halt point: printout of the general, control and floating-point registers, the PSW, conetnts of certain areas of the operating system, etc.

The DDM is included in the system being debugged when the YeS OS is generated. DDM is initiated during operation of the program to intiate the nucleus of the operating system.

By method of organization, DDM programs are divided into DDM supervisor programs, which operate in the supervisor state and receive control as a result of the occurrence of an interruption, and DDM processes, which operate in the "task" state. The system being debugged operates in the "task" mode; therefore, DDM programs receive control when interruptions occur during execution of privileged instructions in the system being debugged, having the capability of tracking the flow of control in the system being debugged without interfering in its operation.

DDM interrupt handlers distinguish interruptions pertaining to monitor operation which are sent to DDM programs, and interruptions pertaining to operation of the system being debugged which are sent for processing to the programs of the system being debugged.

The control processors of the monitor receive control by means of the DDM dispatcher. A program operating in the "task" mode is provided for each process. These programs include a program to issue messages to the console, a program to control the processes of the printout and change of data of DDM, and progarms to control the mode of debugging, processing of printout instructions and data change.

7.12. Generalized Trace Facility

The generalized trace facility (UST) [GTF], pertaining to a class of monitor programs, is designed to acquire information on a specified class of events occurring during the functioning of the computer under operating system control. The GTF is included in any specific OS version when it is generated. It exists in the form of modules of the control program nucleus and in system libraries. To use the GTF, a monitor program is generated which implements its functions.

An event occurring in the computer functioning process is considered recognizable if with each onset of this event, the monitor, in which the processing program for this event is provided, receives control. After processing of the recognizable event, the monitor returns control to the program, during operation of which this event occurred.

The monitor places records on events recognized in the event trace and stores it on an external storage unit. The event trace is a sequence of records on recognized events, ordered by the time of their onset and which reflects the history of functioning of a progarm or the computer as a whole during some time interval.

Events recognized by the monitor form three groups: interruptions, privileged instructions and the GTRACE macro instruction.

Of the interrupt group, the following events are distinguished:

external interruption;

supervisor-call interruption;

program interruption; and

IO interruption.

The group of privileged instructions includes the following events:

LPSW instruction;

SSM instruction;

SIO, TIO, HIO, TCM, WRP, ROD instructions; and

SSK, ISK instructions.

The GTRACE macro instruction is designed for inserting into the event trace data especially provided for this purpose.

The monitor can eb started only by computer operator command and functions as a system task. Certain requirements on main storage and the IO system must be met for successful functioning of the GTF. To store the evnt trace, the GTF monitor may use a data set on either tape or disk. The computer must have a peripheral for storing the event trace and ensure its serviceability.

An appropriate operator command is issued to terminate monitor operation and cancel the GTF task.

7.13. YeS OS Recovery Facilities

In the computer functioning process, various hardware malfunctions occur periodically which result in errors. The complexity of modern computers requires a rapid response to an error; otherwise, delays in localizing errors and eliminating their consequences will result in irregularities of the information in main storage and extreme losses of time.

To raise the efficiency of using modern computers, the operating system includes special programs to recover the serviceability of the YeS OS after errors occur in the CPU, main storage, IO channels and peripherals, and to record information on the status of hardware and software at the time the error occurs.

These facilities include the following programs that have become traditional:

SERO, SER1 for recording the status of the hardware and the YeS OS;

CCH for handling IO channel errors;

ERP for handling peripheral errors;

OBP for recording the status of peripherals;

EREP for editing and printing records on errors; and

the standalone program SEREP for editing and printing information on errors.

With further development of the YeS OS recovery facilities based on generalization of the experience of operating with the preceding versions of the YeS OS, the need arose to create additional program components that substantially expand the capabilities of the recovery facilities. The new components reduce many situations that result in overloading YeS OS and strive maximally to overcome the effect of errors on system serviceability.

The additional recovery facilities include:

the POSK program which enables recovery of YeS OS serviceability on the functional and system levels;

the APR program for retry of IO operations over an alternate access path;

the DDR, dynamic device reconfiguration, program; and

the MIC [missing interruption checker] program for processing missing IO interruptions.

Software recovery facilities are classified as facilities for recovery after IO or machine errors.

Facilities for recovery after IO errors are components of the IO supervisor or operate under its control. Let us touch in more detail on an analysis of each of the programs.

The CCH program receives control when an error occurs in an IO channel. It analyzes the error, generates information on the status of the YeS OS and hardware when the error was detected and informs the operator that the error occurred. If it is established that the error is transient, control is passed to the ERP program for retry of the IO operation. In event of a permanent error, control is passed to the APR program (if it has been included in the specific OS configuration during system generation) to find an alternate path for access to the peripheral. But if CCH determined the need to terminate YeS OS operation, it generates a record on the error and passes control to the SER1 or POSK programs for recording these data in the system log SYS1.LOGREC and switches the CPU to the wait state. When termination of YeS OS operation is not required, control is passed to the OBR program for recording information on the error in SYS1.LOGREC and sending a warning message to the computer operator.

The APR program finds an alternative path for access to the peripheral specified in the IO operation, but only when the channel error occurs on the main path. When an alternate path through another channel is available, APR passes control to the ERP program for retry of the IO operation. When an alternate path is not found, control is passed to the SER0, SER1 and POSK programs for recording the status of the YeS OS and hardware in the data set SYS1.LOGREC. APR gives the operator the capability of adding or removing one of the paths for access to the peripheral.

The ERP program, a mandatory component of YeS OS, is a set of standard procedures for processing peripheral errors. There is an individual procedure for each peripheral. The ERP program analyzes the error. If the error is permanent, retry of the operation is not possible. The computer operator issues the appropriate message and control is passed to the OBR program. In the other cases, ERP attempts a retry of the IO operation. The number of retries depends on the peripheral type. If the error is not corrected after the specified number of retries, it is classified as permanent.

The OBR program, a component of the IO supervisor, gathers statistics on the status of the YeS OS and the peripherals in which the error occurred. The information is logged in the system log SYS1.LOGREC. OBR also monitors how full the SYS1.LOGREC data set is and informs the system operator about this.

The DDR program reconfigures devices. After a permanent error is detected, the IO supervisor passes control to the DDR program which determines the possibility of moving the IO operation to another IO device. If this is possible, the DDR performs the appropriate rearrangement of the systems tables, issues instructions to the operator for moving the volume needed and returns control to the ERP program for a retry of the IO operation. DDR offers the capability of reconfiguring devices of the same type, including card devices, disk and tape storage units and printers.

The MIC program monitors completion of IO operations. It identifies situations associated with losses of IO interruptions that are a result of both hardware malfunctions and errors in YeS OS operation. If an expected interruption has not occurred within a specified time, MIC messages the computer operator, logs the information on the error that occurred in the system log and simulates the missing interruption, completing the request for IO.

Facilities for recovery after machine checks consist of the SER0, SER1 and POSK programs that log the status of the hardware and YeS OS, depending on the type of computer used. The programs receive control when machine checks occur through a new PSW for interruptions from the check circuits and in the case of IO errors, from the CCH program. The SER0, SER1 and POSK programs are mutually exclusive.

SER0 determines the type of error detected, generates a record of the error, completes execution of all IO operations, records the information on the error in the system log SYS1.LOGREC, sends the computer operator a message on the error, recommending reloading of YeS OS, and puts the CPU in the wait state. If a second machine check occurs during operation of the SER0 program, SER0 logs special information in the SYS1.LOGREC data set and puts the CPU in the wait state.

In addition to the functions implemented by the SER0 program, the SER1 program analyzes the nature of the error based on information supplied by hardware facilities, checks the nucleus area for parity errors, and attempts to recover system serviceability, after abending the task affected by the error. If SER1 has established the fact of distortion of the control programs or has not detected a damaged task, the computer operator is sent a message on the error with the recommendation of reloading YeS OS, and the CPU is switched to the wait state.

The POSK program begins operation with analysis of the interrupt code that contains information on the nature of the machine check detected. If the error was overcome by hardware facilities, POSK logs the appropriate information in the system log and performs no recovery operations. But if the interruption code is for an unsuccessful hardware recovery, POSK tries maximally to keep YeS OS in a serviceable state. An attempt is made at functional recovery, as a result of which the damaged task remains fully serviceable. If functional recovery is not possible, recovery is made at the system level, which consists in abending the damaged task. When the control program is damaged, recovery is considered impossible and the computer operator is issued a message with the recommendation to reload YeS OS.

7.14. YeS DOS-3 Operating System

The YeS DOS-3 multiprogram operating systen is intended for organization of efficient functioning of small and medium YeS EVM-2 computers with virtual storage. It supports batch job processing, data teleprocessing facilities and data base management facilities. The concept of the DOS-3 control program differs considerably from that of previous versions of the DOS. This new concept has the following main advantages:

more flexible allocation of system resources;

more efficient IO control system in a virtual environment;

program independence of type and model of peripheral; and

simplified systems maintenance.

The DOS-3 modular structure permits expansion and improvement of software, which allows relying on its viability over the entire period of operation of the YeS EVM-2 in traditional and new applications. DOS-3 is based on using the concept of virtual storage, dynamic allocation of resources and centralization of all major system functions.

YeS DOS-3 Composition. The first phase of the YeS DOS-3 includes the following program components:

control program: supervisor, job management, initial loading, program for peripheral concurrency;

logical IO control system;

system utility routines: editor, librarian, special routines, sorting routines, debugging facilities, device test routine;

Assembler translator;

translators for programming languages: FORTRAN, PL/1, COBOL, RPG-2 with autoreport; and

BTAM.

Development plans call for the following program components in the second phase of the YeS DOS-3:

expansion of the control program: support of secondary consoles, catalog of files, expansion of remote job entry;

system utility routines: facilities for checking system generation, facilities for managing statistics;

translators for programming languages: optimizing PL/1, PL/C, PASCAL, SIMSCRIPT, SYSTRAN; and

the DL/1 system.

Compatibility with YeS DOS-2. YeS DOS-3 is compatible with YeS DOS-2 at the level of:

high-level programming languages;

Assembler language, if the program does not refer to internal control units and system tables (including DTF tables) by means of non-system functions; and

object modules, if the program does not refer to the system nucleus and the requirements for compatibility at the Assembler level are met.

Compatibility is not supported for programs dependent on specific characteristics of computer models or devices.

To facilitate programming and increase compatibility, YeS DOS-3 has declarative macro instructions that permit reference to control units and system tables in symbolic form.

System Resources. The main function of the operating system nucleus is control of utilization of all system resources. DOS-3 offers the following resources:

area of virtual addresses;

area of real storage;

area of storage on auxiliary storage direct-access devices for system use;

IO peripherals;

data files on external storage units; and

operating system programs.

The best possible distribution depends on the nature of these resources. Some of them, control program components for example, are used jointly by many users at the same time. Similarly, libraries and data files may be used jointly (only as input information).

Unshared resources (for example, a storage area, unit record peripherals, output and update files) are allocated dynamically, but largely to the entire process of operation to avoid system impasses.

Sequential and unit record peripherals are essentially not shared. DOS-3 supports definition of peripherals used jointly by several users and indirect IO (SPOOLING).

The system work area is primarily implemented in an area of virtual addresses, and in doing so, a permanently allocated work area on disks is not required. Because of this, the usual "critical" elements of a DOS system (SYSUT system utility files and SYSLNK files) are not needed; therefore, the multiprogram mode is enabled for even small hardware configurations.

Main Storage Management. The main system resource is virtual storage. All user operations executed at the same time have their own virtual area not exceeding 16M bytes. This virtual area is statically divided into four parts:

area of identical addresses;

area of jointly used programs;

a jointly used virtual work area; and

a personal virtual area.

The identical address area takes up the lower part of any virtual area, starting with the zero address (real ro virtual). This area is always implemented in real storage, not reflected on magnetic disks and is common for all virtual areas, but the addresses of all objects are identical and real. This area contains the DOS-3 control program and control tables. The boundary of this area is set when the system is generated.

The area of jointly used programs is also common for all virtual areas. Stored on magentic disks is a copy of the DOS-3 control program phases in the form in which these phases are stored in the absolute module library. This area contains supervisor transient programs, logical IO control system modules, logical IO control system transient programs, the job management program, etc.

From the area of jointly used programs, parts of the DOS-3 are moved into real storage by the paging mechanism. Since these system programs are reentrant, they do not have to be copied into the page set. A copy of the jointly used virtual work area is kept in the page data set. The upper boundary of the jointly used virtual work area is set when the system is generated and defines the lower boundary of the personal virtual area. This address is the address, starting from which the user usually edits his programs. Each user job is given its own (personal) virtual area (i.e. appropriate page tables are built). At thsi same time (or during editing) the SYSLNK area is built. After a program is loaded for execution, a copy of it remains unaltered in the absolute module library; alterable copies of a program and data are kept in the page data set and used in the paging between real storage and the page set. The size of a personal virtual area may be dynamically extended to a maximum of 16M bytes.

Real storage or the area of real pages is divided into two parts: the area of identical addresses and the area of replaceable pages. The contents of the area of identical addresses is the permanent resident part of the DOS-3 control program, i.e. the real part of all virtual areas. The series of pages of the jointly used and personal virtual storage is transferred to real pages when necessary. The control program periodically executes a review procedure to determine which pages are no longer needed in real storage to keep the contents of changed pages in the page set in the end.

The page set is placed on one or more disk volumes; a unique adjacent sector must be allocated on each volume. The set forms the required storage area on disks to keep the form of changed or constructed contents of any virtual storage area. The entire page set area is formatted in advance into 1K-byte blocks and a segment is the unit of allocation. Dynamic allocation of segments is determined by using the segment set scheme in real storage. The maximum size of this set is a parameter of DOS-3 system generation.

Job Input and Planning. Jobs are input into the system by means of the device(s) assigned by the operator as the system input device. When several system input devices are available or if the system input device is a virtual device, the system operates in the multiprogram mode. The system input unit may be a card input unit, a tape or disk storage unit, a virtual system input unit that can be connected to any unit record device, a floppy disk input unit or a telecommunications unit.

The unit of assignment and planning of resources is the order, consisting of one or more jobs that can be processed sequentially and make use of the same system resources.

The first statement in any order [zakaz] must be a *$$JOB statement that can be conventionally described as the creation and description of a partition. The *$$JOB statement parameters are used by the system scheduler to determine whether all resources requested are accessible and, consequently, whether the partition needed can be created and started. The user first idnicates the required size of virtual storage in kilobytes and the actual size of real storage (size of a working set of pages). The required configuration of other resources (peripherals, disk and tape volumes, files) is indicated by a reference to a certain phase of the description of the partition in the absolute module library. Any number of different phases of a description of a partition may be written and cataloged by using the system macroinstructions. The partition description phase contains a list of standard assignments, in which all devices (by means of the physical address (CUU) or device class), tape and disk volumes needed for execution of the entire order are indicated.

This information is used by the system to determine the capability of starting the operation, reserving all required devices and building the appropriate table of logical devices for the partition.

The work scheduler builds a table of requests for operation and puts it in the system job queue. When all required resources are available, they are reserved, partition tables are built, and the first job in the order is started. Jointly used IO devices are not reserved for exclusive use. It is recommended that IO devices be allocated by class or volume registration number to make full use of system flexibility. Output and update files on disk are reserved when opened for individual use; input files may be shared.

Supervisor. DOS-3 consists of a relatively small nucleus that includes the interrupt handler, the queue handler, the SVS programs, programs for recovery of IO errors on disk, a program to find a missing page and others.

The nucleus stays permanently in the identical address area. In addition to the nucleus, the supervisor includes a number of programs kept permanently in the area for jointly used programs. The table of contents of all the programs in this area is built by the editor and stored in the identical address area during initial loading. The supervisor can control parallel processing of up to five independent operations (job streams), each of which is executed in its own partition; up to 99 subtasks may be created and executed concurrently in any partition.

Logical IOCS. Three basic aims were considered in developing the logical IO control system [IOCS]:

maintain the continuity of the logical IOCS;

improve the efficiency of the system with regard to its functioning in the virtual environment; and

increase the flexibility of the system.

The first aim was achieved by keeping all the data access methods that existed in preceding versions of YeS DOS in the YeS DOS-3 logical IOCS. System declarative and executive macro instructions retain at the symbolic level the main features from preceding systems; in the process, howver, a number of file description parameters are not considered. All system expansion methods were designed to maintain software compatibility. Organization of files of various formats was also retained. The exception is organization of index sequential files which underwent some changes because of implementation of more modern concepts of index sequential files. The system offers programs to translate index sequential files from previous versions of YeS DOS into those for YeS DOS-3.

The necessity of achieving the second aim occurred because the DOS-3 logical IOCS is intended for functioning in the virtual environment, while the hardware for IO operations in a virtual environment is insufficiently developed. The main task in executing any IO operation is the fixing of all storage areas needed in execution of the IO operation in real storage for the entire operation execution time. Work areas such as those for the channel program, list of addresses for indirect addressing in the channel, disk address for finding a specific block on disk and others are reserved in one area during the data file opening procedure. The user should not specify IO areas in his program. In this case, allocation of IO areas is performed automatically during the opening procedure. Automatic allocation of work areas during the opening procedure allows obtaining an adjacent area for all work areas associated with the IO operations; this saves on the number of apges needed for fixing in real storage and permits one fixing during the opening procedure instead of multiple fixing at the start of each IO operation.

Raising the flexibility of the logical IOCS is achieved by the following methods:

change in the principle for connecting executive modules for IO. Executive IO modules are connected during the data file opening procedure; since they are reentrant, they may be used in several tasks at the same time, which saves considerable storage in the multiprogram mode of operation;

offering the capability of dynamic allocation of IO areas during the file opening procedure;

the capability of dynamic change of parameters; for example, right before the start of a program, record size and record block size may be specified; and

the higher degree of independence of the peripheral makeup. In assigning devices, one may specify only the device type, and the operating system will allocate a free device of the given type.

Libraries. There are five types of libraries in YeS DOS-3: absolute module (phase), object module, source module, procedure and edited macro definition.

The absolute module library is intended for storing programs (phases) ready for execution. In addition to the usual phases, this library contains partition description phases that are used during start of partitions an list the resources needed for the partition; it also contains the table of contents of programs in the virtual area of jointly used progarms.

The object module library is intended for storing object modules that result from the programming language translators. Basically, this library is used to hold standard programs stored in object module form convenient for connecting to other programs.

The source module library is intended for storing modules in the source programming languages. A convenient apparatus is provided to make changes in source modules and connect other source modules as parts of a given source module.

The procedure library is designed to hold frequently used job sets. There is an apparatus for adjusting procedures for specific applications (substitution of assigned parameters).

A thorough syntactic check is made during cataloging of macro definitions. This speeds up debugging and translating of macro definitions from the usual format into the edited, which speeds up considerably the process of generating macro expansions from macro definitions. This information is contained in the edited macro definition library.

All libraries are disk files. At the beginning of each library, there is a table of contents of the library elements, which are sorted and blocked. To speed up searching, each block of a table of contents has a key. Library element text is stored in fixed-length blocks. All libraries may be system or personal.

The general library maintenance routine performs on library elements functions such as cataloging, deleting, renaming, correcting, perforating and printing individual elements or groups of library elements. The maintenance routine used to copy and reorganize libraries transfers entire libraries, sublibraries and groups of library elements from disk to tape, i.e. it copies libraries, sublibraries or individual elements from one or more libraries input to one output library. Ordering and compression of text occurs during this transfer.

Spooling. The real capabilities of multiprogramming with a small amount of hardware are limited by the number of unit record devices, especially by the number of line printers. Therefore, spooling is included in the DOS-3 control program. At system generation time, the user may specify any number of virtual unit record devices, each fo which is associated with a real device. Virtual devices may be used at the same time by several partitions.

If several additional virtual printers or other devices have been defined, any program in any partition may use all these devices at the same time. Data obtained as a result may then be printed sequentially on one real printer.

For input, the situation is somewhat different. In this case, the operator may start the process of buffering of data on any number of input units, including telecommunication devices. Direct input of other jobs into the system may occur at the same time, if a real peripheral has been assigned to the corresponding logical input unit.

Thus, one can combine as desired direct input and spooling, i.e. system IO of data through SYSRDR, SYSIPT, SYSLST and SYSPCH from a terminal or output to it is possible only in the spooling mode. In this case (indirect IO over teleprocessing lines), any program can read (write) its data from (and to) a terminal by using the conventional macro instructions in the logical IOCS just as if it came from a card reader or output to a printer, etc.

There is a job control language and operator command language that allows setting and changing priorities of individual operations and specifying the need for keeping or processing data, deleting after processing or moving from the output queue to some input queue. The majority of central operator control commands are available to a remote operator.

Command statements can be used in the data input stream to include data coming from other input units in the stream. This is especially useful in inputting data from floppy disks or perforated tape units prepared without the appropriate control cards. Data input from nonstandard units, from perforated tape input units for example, can be translated into the standard DKOI [decimal information interchange code] and then processed the usual way.

Spooling allows the operator to determine the actual status of IO queues, the status of current operations, use of resources, etc. This enables collection of basic accounting data, that is kept automatically in the form of a spearate data queue and output as required. The system is built to enable simplicity and efficiency in restarting after device errors occur.

7.15. Application Program Packages

Along with the operating systems and maintenance programs discussed above, application program packages (PPP) are an integral part of the YeS EVM software system. A PPP is a functionally complete complex of program facilities oriented to solving a specific logically complete class of problems.

The PPP developed within the YeS EVM differ in independence of the type of hardware and peripherals used, ease of adjustment, diversity of classes of problems solved (application areas) and by the various algorithms for solutions in each class. They are oriented to operating under the control of the YeS DOS and YeS OS operating systems and in many cases are a further expansion of them. There are now over a hundred packages embracing several millions of instructions in the Unified System.

PPP are classified by application sphere and class of problem solved as folows:

general-purpose;

for solving engineering, scientific and technical problems;

for solving economic problems and those in ASU's; and

expanding the capabilities of themain operating systems.

General-purpose PPP include programs that implement mathematical methods (for example, methods of queueing theory, probability theory, mathematical programming, etc.) for solving problems that pertain to many spheres of application. As a rule, these programs form the basis of the complex application problems of management and planning, operations research, simulation, etc., but may have an even greater value of their own.

Individual packages, as well as methodologically oriented (numerical mathematics, mathematical statistics, etc.) PPP for engineering, scientific and technical calculations, are designed to solve problems of computing special functions, interpolation, optimization, design, medical research and others. This class of PPP is characterized by a great diversity in problems solved and continual expansion of them. Resting on program creation as the basis and making use of the high-level programming languages designed specifically for scientific and technical applications, the user may develop packages oriented to solving his own problems.

The PPP for economic calculations and ASU are used to solve problems of production management and planning, operations and long-term planning, supply of materials and equipment and others. In connection with the widespread incorporation of Unified System computer facilities in various types of automated management systems, this class of PPP is assuming major importance; new programs have to be written and standardized especially in view of their widespread application.

Efforts by developers of Unified System software who specialize in the problems of applications programming are mainly focused on creating PPP that extend the capabilities of the main operating systems. These packages are designed to support certain modes of operation of computer systems, as well as the functioning of specialized hardware.

Since this class of packages by its functional value is of special interest to YeS EVM users, given below in detail are the characteristics of the programs that expand the functions of operating systems in the direction of supporting additional modes of operation and managing complex data structures.

A major quality of modern computer systems from the viewpoint of their use in building ASU's and IPS [information retrieval systems] is the availability within the standard software of facilities for organizing and managing large data files and facilities that afford shared access to these files in the mode of simultaneous servicing of user queries coming from remote and local terminals. There are now two PPP supporting these functions within the YeS EVM software system.

The "Oka" Data Base Management System (SUBD "Oka") [DBMS] is intended for building large-scale informational, reference and management systems with a large volume of processed information and complex logical ties between the data elements.

The system supports operation with data bases in the batch and teleprocessing modes. All types of remote and local YeS EVM terminals can operate in the system.

Data access is provided by application programming facilities in PL/1, COBOL and Assembler through the BETA data language. The DBMS data bases are implemented by hierarchical and inverted data structures.

The DBMS facilities support sequential and random access to the data bases and all types of processing: selection, insertion, replacement and deletion at the level of logical units of data.

The system has developed facilities for managing data bases that enable:

logging of all changes occurring in a data base and recording all messages;

data recovery that maintains the information stock of the system in the correct status in event of equipment failures and malfunctions;

resource protection from unauthorized access;

reorganization and loading of data bases; and

preparation of statistical reports.

In the teleprocessing mode, the "Oka" system performs all functions required for managing the reception, transmission, switching and translation of messages exchanged between the terminal network and a computer.

The "Kama" Data Base Teleprocessing System is intended for use as teh base software in building:

a wide class of conventional applications for teleprocessing systems, including message switching, reception of inquiries, data input in the interactive mode and data acquisition; and

information reference systems for shared use operating online and characterized by quick response.

The "Kama" system supports processing forms such as data input and accumulation online, message transmission from one terminal to another, online start of applications programs from terminals. This system also services user queries at the same time, coming from the terminal network. The number of simultaneously executable programs is limited only by the size of main storage available.

User requests are realized by application programs written in PL/1, COBOL or Assembler. Application programs are linked to a teleprocessing system by macro instruction facilities.

Data management in the data base teleprocessing system offers facilities for working with sequentially organized files and data bases built with YeS OS files with index sequential or direct organization. Information capabilities of the data base teleprocessing system enable management, selection, recording and updating of information in data bases.

Information system developers are fully responsible for the initial loading and integrity of the data bases.

The KROS PPP is designed to expand the capabilities of the YeS OS. Operation of the operating system together with the KROS PPP enables:

raising computer system throughput because of the change in the YeS OS job management algorithm and the use of special access methods for operation with job input streams and data output streams. How much throughput is increased depends on the computer configuration and the nature of jobs executed; and

automating computer operator functions. The KROS PPP performs automatically some of the actions usually performed by the computer operator.

For example, the KROS package plans and starts programs for system input, system output and initiators. Computer operators are also given a number of additional commands to manage job execution and peripheral operation:

dynamic ordering of task solving based on an account of the use of the CPU and peripherals; and

automatic inclusion in the job input stream of jobs formed by user programs.

Using the KROS package reduces the requirements for resources for a computer system. In particular, less direct access storage needed for holding the input and output data sets is required. Main storage size needed for KROS package operation is generally less than that needed by standard YeS OS facilities performing similar functions.

Since the KROS package supports remote job entry, the remote user can execute jobs under the control of YeS OS and KROS.

Unified System computers with the standard instruction set and the AP-4 (YeS-8504) terminals can be used as the remote stations.

Also note that using the KROS PPP does not require any change to YeS OS or user programs.

The KROS PPP is independent of the YeS OS edition and can be used as the basis for development of new facilities and capabilities that do not affect YeS OS and user programs.

The ROS PPP is intended for the functioning of a computer system that includes an arbitrary number of Unified System computers, each of which operates under control of the YeS OS.

The ROS PPP operates in one computer in a multimachine system. This computer is called the service computer and the others are called main computers. The service and main computers are linked by a channel-to-channel adapter.

Using YeS OS together with the ROS package in a multimachine computer system raises the throughput and reliability of the computer system compared to the computers used separately; it also permits reducing the number of computer operators maintaining the multimachine system. This advantage is achieved by dividing the functions for job execution between the service and main computers and making sue of the additional capabilities offered by the ROS PPP.

Job execution functions are divided as follows. The service computer takes care of:

input and building a common job queue for the whole multimachine computer system;

planning of job execution on the main computers with regard to user requirements;

sending a job for execution to a selected main computer;

receiving data from a main computer, obtained as a result of job execution;

output of data received on a printer and perforation; and

automatic issuance of required operator commands to the main computer.

Since all auxiliary functions for job processing are taken care of by the service computer, a main computer is concerned only with job execution. Also, instead of the slow devices for job input, data output and the operator console, a main computer uses high-speed channel-to-channel adapters for data transmission. These conditions for operation of a main computer within a multimachine computer system significantly increase the efficiency of its operation compared to the usual operating conditions.

Additional capabilities of the ROS package include:

user job processing management by using ROS command statements together with YeS OS JCL command statements. Use of ROS command statements is not mandatory;

capability of advance setting of data media needed for a job on a main computer before the start of job execution;

processing of internal jobs, which allows a user program executed on a main computer to generate new jobs and place them in the common ROS job queue;

management of dependent jobs, which allows a user to establish a relationship between jobs in some group of jobs and execute these jobs in a sequence described by the user;

remote processing of jobs, i.e. the capability of remote job entry from terminals and output of , processing results to them;

additional service routines in the form of a standard set of service functions operating under ROS control; and

the local execution mode, when one computer is used as the main and service computer.

The real-time supervisor (SRV) is a PPP that together with the YeS OS enables efficient use of all models in systems that control processes or objects in real time. The real-time supervisor offers facilities to extend the functional capabilities of the operating system in the following directions:

use of nonstandard IO devices (real-time devices) and lines for external interruptions (real-time lines) as sources of data to be processed in real time;

rapid system response in processing data coming into the computer from the real-time devices and lines;

a strict tie-in between times of execution of functional programs in a system and actual time transpiring; and

a number of additional optional capabilities (operation with resident and nonresident data queues, logging of events in a system, operator service and others) that may be used in process control systems.

To realize these capabilities, the real-time supervisor has its own facilities for organizing the data processing process, in the background of which occurs the operation of the operating system. As a result, the SRV offers an additional, with respect to those available in the operating system, mode for servicing programs, called the real-time mode; programs using SRV capabilities are executed under this mode.

Universality of SRV and the efficiency of the real-time mode supported by it are achieved because of the availability of the SRV generation facilities. The generation facilities allow obtaining specific versions of the SRV with regard to the configuration of available hardware, the configuration of the operating system version used and the features of the functional purpose of the process control system.

Chapter 8. Functional Characteristics of YeS EVM-2 Models

8.1. The YeS-1015 Computer

With a throughput of 19000-21000 instructions/second, the YeS-1015 computer is the smallest model in the YeS EVM-2 family. It is intended for use in batch processing and the interactive mode in traditional data processing systems. This computer is also used in rail transport management, enterprise management, in information systems for resource management, in power engineering, in executing commercial, banking and financial operations, in various data bank systems and in training.

In accordance with the YeS EVM-2 principles of operation, the YeS-1015 computer is compatible with YeS EVM-1 models and realizes:
extended precision floating-point operations;
facilities for organization of virtual storage with a 16M-byte capacity;
storage protection facilities;
extended facilities for processing of machine errors;
monitor program support facilities;
program event recording facilities; and
microdiagnostic facilities.

A structural feature of the YeS-1015 is the connection to a common bus of standalone processors that operate concurrently and implement the principle of decentralized data processing. Communication between the standalone processors that make up the CPU is effected by using simple and easily interpreted standard messages. The CPU also includes the main storage unit; display console (operator console), and three types of standalone processors oriented to execution of different functions: instruction processing (UEP), input/output (IOP) and service processor (RAP).

The operator console consists of the YeS-7168 display with 16 lines of 56 characters per line and the YeS-7186 matrix printer with a printing rate of 180 characers per second [cps] with 132 characters per line.

The instruction processing processor:
fetches data from main storage;
decodes instructions;
executes arithmetic and logic operations;
executes interruptions;
unpacks IO instructions and starts the operation of the IO processor.

The internal storage of the instruction processor is made up of general, floating-point and control registers. Processor cycle time is 550 ns.

Up to four IO processors that execute different functions may be used in the CPU. The first processor controls the directly connected YeS-7184 printer and IO peripherals through the multiplexer channel. The second processor controls the directly connected YeS-5061 disks, the maximum number of which is 8. The third processor

[1] Here and from now on, computer throughput is indicated for statistical mixes for solving scientific and technical problems.

is the selector channel; up to eight YeS-5017 units are connected to it through a YeS-5517 control unit. And finally, the fourth processor for IO effects a direct link to a data teleprocessing system.

Cycle time of the IO processor is the same as the instruction and service processors, 555 ns.

The service processor checks the operation of all CPU functions and supports: logging and analysis of errors, input of microprograms and computer-operator communication.

Main storage with a maximum capacity of 256K bytes is built with integrated circuits. Cycle time is 1 microsecond, access width is 2 bytes. An autonomous unit, the storage control unit (OTV), is provided to organize access to main storage within the CPU.

YeS-1015 computer software includes the YeS DOS-3 operating system that supports the YeS EVM-2 capabilities and a set of maintenance routines (KPTO) that includes not the traditional software, but a system of microdiagnostics that performs the following functions:
correction of errors occurring during operation;
logging of malfunctions in automatic mode;
offline testing of the machine during preventive maintenance; and
localization of defective units and assemblies

The standard peripheral composition for the YeS-1015 and the other YeS EVM-2 models is given in table 30.

8.2. The YeS-1025 Computer

With a throughput of 35000-50000 instructions/second, the YeS-1025 is the second member of the family of program compatible YeS EVM-2 machines and belongs to the class of small machines. It is oriented to solving a broad range of scientific and technical, economic and special problems both in the standalone mode and in information processing systems, including real-time and shared use systems. The YeS-1025 can also be used in large computer systems as a slave computer for preliminary processing of information. The sphere of application of the YeS-1025 is governed by the features of its structure:
higher technical and economic indicators and primarily of the throughput/cost ratio than the YeS-1021 computer because of a more flexible and efficient internal logical structure;
organization of virtual storage with a 16M-byte capacity;
connection of a broad complex of external devices;
direct connection of high-capacity external storage units; and
an easy and convenient method of connection to data teleprocessing systems and to other computers.

The YeS-1025 computer processor is similar in structure to that of the YeS-1015. It is designed in exactly the same way for decentralization of execution of functions that permit processing of arithmetic and logic instructions concurrently with IO of information from peripherals, with operations from the operator console and with diagnostic procedures. In accordance with this, the YeS-1025 processor is

Table 30. Standard Peripheral Composition for the YeS EVM-2 Computers (number in parentheses is number of units in the standard configuration; all model numbers are preceded by "YeS-")

	Computer						
Peripheral	1015	1025	1035	1045	1055	1060	1065
disk storage control unit	-	-	5561 (1)	5561 (1)	5561 (1)	5566 (2)	5566 (2)
removable disk storage unit	5061 (3)	5061 (5066) (2)	5061 (3)	5061 (4)	5061 (8)	5066 (6)	5066 (8)
tape storage control unit	5517 (1)		5517 (1)	5517 (1)	5517 (1)	5517 (2)	5525 (2)
tape storage unit	5017 (3)	5004 (6)	5017 (6)	5017 (6)	5017.02 (8)	5017.03 (8)	5025 (8)
typewriter with control unit				7077 (1)		7077 (2)	7077 (2)
alphanumeric printer	7184 (1)	7034 (7039) (2)	7032 (1)	7032 (2)	7033 (1)	7033 (2)	7032 (2)
card reader	6016 (1)	6016 (2)	6012 (1)	6012 (2)	6016 (1)	6019 (2)	6019 (2)
card puncher	7014 (1)	7014 (2)	7010 (1)	7010 (1)	7014 (1)	7010 (2)	7010 (2)
paper tape reader			6022 (1)	6022 (1)		6022 (2)	6022 (2)
paper tape puncher			7022 (1)	7022 (1)		7022 (2)	7022 (2)
paper tape IO station		7902 (2)					
alphanumeric and graphic information IO unit							7064 (2)
cluster control unit with branch consoles (four displays)						7906 (1)	7906 (1)
logic repeater							4080 (1)
channel-to-channel adapter							4061 (1)

made up of six autonomous functionally complete modules connected to a common bus. Communication between any two modules is effected by using special interface adapters built into each of them. Since the modules use a common bus with time sharing, the bus is physically divided in two to increase throughput. Also, one-third of the bus is used for asynchronous transmission of signals from the service module to the others and vice versa for convenience of diagnostics.

The YeS-2025 processor includes the following basic modules: operating, service, control, main storage, disk and multiplexer.

Additional facilities may be included in the processor as a function of the application:
a second main storage module;
a tape module; and
a switching module.

Each of these modules uses microprogram control and a set of internal registers to enable execution of the functions imposed on it.

The operating module realizes execution of the YeS EVM-2 universal instruction set, including the standard set of operations, operations with decimal numbers, operations with floating point, conventional and extended precision, operations for translation of addresses and conditional exchange, and operations enabling operation with time facilities and storage protection keys.

As noted earlier, a major feature of the YeS EVM-2 models is the availability of a developed diagnostic system that enables detection and rapid localization (and in some cases even correction of data) of faults due to malfunctions and failures occurring during computer operations. In this respect, the YeS-1025 is no exception and makes use of both software and special hardware for diagnostics. The software is intended mainly for diagnosing IO devices and disk and tape storage units. The hardware includes a system of links that enable direct writing and reading of information in each module or its major parts, and a special service module for initial recording of microprograms, control of procedures of microsignificance and processing of information on malfunctions. hardware facilities for the YeS-1025 computer diagnostic system also include the operator console with the YeS-4063 alphanumeric display, the YeS-7934 serial printer and the YeS-5074 or YeS-5075 floppy disk storage unit. In the process of computer functioning, information on malfunctions is accumulated on floppy disk and can be output on the display or printer for expeditious analysis of the reasons for the failures and malfunctions. Also stored on floppy disks is information on the causes of the most typical malfunctions that have ever occurred at a user site, which is essentially a set of diagnostic tests that permit rapid and efficient discovery and correction of damage to computer functioning.

The control unit in the YeS-2025 processor is essentially a storage control unit that performs all the encessary functions on organizing work with main storage and organizing access to data.

The main storage module is built with inetgrated circuits with a capacity of 1024 bits per package. One module holds 128K bytes. Using an additional storage module permits obtaining the maximum possible main storage, 256K bytes, for the

YeS-1025 computer. Main storage read cycle is 500 ns; write cycle is 750 ns, and width is 72 bits.

The disk module is an adapter for direct connection of up to four disk storage units, just as the tape module is used for direct connection of tape storage units.

The multiplexer module is designed to connect IO peripherals to a computer by means of the standard IO interface. Throughput of the multiplexer module in the burst mode is 29K bytes/second and in the multiplexer mode, 24K bytes/second. The number of multiplexer subchannels is 32 with maximum throughput for a one-byte interface of 29K bytes/second. Total throughput of the multiplexer module is 1.0M bytes/second.

A distinctive feature in the structure of the YeS-1025 computer processor is the presence of a switching module that has 16 lines for connecting both terminal equipment for a data teleprocessing system and small computers in the SM EVM series.

8.3. The YeS-1035 Computer

The YeS-1035 computer has a throughput rate of 140,000-160,000 instructions/second and is in the medium class. It is intended for solving a broad range of scientific and technical, economic and special problems in shared-use batch processing, teleprocessing and real-time systems as well as in areas that require a general-purpose computer with large computing capacity and a broad set of high-speed peripherals. This model can be used in small and medium-size computer centers and at the lower and mid levels of automated management systems. It implements the universal set of YeS EVM instructions and is program compatible with other Unified System models. Emulation facilities provide program compatibility with the Minsk-32 computer.

By connecting the YeS-8371 communications processor to the computer, a general-purpose computer complex can be created for shared use and teleprocessing of information. By using the special processor for array computations, a computer complex can be obtained for scientific studies oriented to array processing of large data files.

The processor (the YeS-2635) is the central unit in the YeS-1035 computer; it is intended for executing arithmetic and logic operations, organizing reference to main storage, organizing data exchnage between main storage and channels, controlling the sequence of instruction execution, actions during interruptions, operation of time readout facilities, initial program loading and operation of IO devices.

The processor processes numbers written in binary code, fixed-length floating-point numbers, variable-length decimal numbers, and fixed and variable-length logic information.

This processor consists of the central processor (the YeS-2435) with a byte-multiplexer and two selector channels, the YeS-3235 main storage unit and the YeS-0835 power supply.

This processor also has additional facilities: a channel-to-channel adapter enabling joint operation of two YeS-1035 computers and permitting transfer of data files from one computer to another through selector channels; two selector channels or an integrated file adapter.

Control of ALU operation is microprogrammed and width is two bytes. Control is effected by using reloadable storage of microprograms made with IC's with a cycle of 200 ns, width of 32 bits and capacity up to 64K bytes. Microprograms are laoded into storage from the console magnetic tape storage unit. If necessary, the user can replace the microprograms kept in storage. Automatic localization and indication of computer malfunctions is performed by using a special set of diagnostic microprograms. Checking computer serviceability by using the microdiagnostic system takes no more than 15 minutes.

When a reference is made to main storage or to microprogram storage, a correcting code is generated, stored and used to analyze the validity of the data transmission. If a single error emerges, the information is corrected.

If a malfunction occurs in the processor, control is passed to the microprogram for retry; it restores the situation that preceded the malfunction and passes control to the sector of the microprogram where the malfunction occurred. If the retry is successful, normal computer operation continues and the error is recorded in storage for analysis.

The instruction set includes arithmetic instructions to process floating-point 128-bit operands.

There are two versions of main storage for the YeS-1035 computer: the YeS-3235 ferrite core unit and the YeS-3238 integrated microcircuit unit. The YeS-3235 main storage unit has variable capacity: 256, 512 and 1024K bytes. Storage cycle time is 2 microseconds, access width is 72 bits and access time is 800 ns. The YeS-3238 is built with 4K-bit integrated circuits and has a capacity of 2M bytes. Storage cycle time is 850 ns and access time is 650 ns.

The YeS-1035 computer IO channels (selector and byte-multiplexer) make use of processor equipment for their operation, i.e. they are integrated. The selector chennel enables connecting high-speed peripherals (external storage devices) to the processor. The computer has two selector channels; up to 256 devices may be connected to each channel. The byte-multiplexer channel operates with low-speed IO devices. Up to 184 devices can be connected to a multiplexer channel.

The selector channels may be operated in the block-multiplexer mode which permits connecting up to 512 devices to the selector channels.

In a multiplexer channel, the number of subchannels (number of connectable devices) is governed by control storage size allocated to store channel control words, i.e. from 64 to 128.

Execution of IO instructions and data transmission is effected by multiplexer channel microprograms.

Maximum throughput of the multipelxer channel is 30K bytes/second.

The mixed principle of control is used in the selector channels. IO instructions are executed by selector channel microprograms, while data transmission between main storage and channels is effected by hardware with partial use of CPU equipment. Maximum rate of data transmission in the selector channel is 790K bytes/second and total throughput of the channels is 1.2M bytes/second.

All basic conecpts of the YeS EVM-2 have been implemented in the YeS-1035 computer:
extended instruction set;
virtual storage;
block-multiplexer mode of channel operation;
multisystem facilities and time readout facilities;
program event recording;
provision of monitor programs;
extended precision floating point operations;
correcting codes for main storage; and
advanced microdiagnostic procedures.

The YeS-1035 computer features Minsk-32 emulation facilities that permit use of the large stock of application programs developed for that machine. These facilities include:
facilities for emulating Minsk-32 programs;
facilities for translating programs in source languages; and
data transfer facilities.

Provision of Minsk-32 and YeS-1035 compatibility based on considerations of economic effectiveness is implemented on two levels: program and program-microprogram (emulation level).

The expediency in using the program level of compatibility stems from the fact that many Minsk-32 programs are written in the high-level programming languages and the process of their primary interpretation is not time critical. Primary interpretation provides for the sequence of conversion, translation and execution.

The second compatibility level, emulation, is used for programs written in the Minsk-32 machine language. Such programs make up a large part of the whole stock of programs since they are most economical. Therefore, the emulation level is the main and most efficient. Based on the great differences in data representation and formalization of computational processes and IO operations, in the second level of compatibility, 68 percent of the instruction set is covered by microprogram interpretation and 32 percent by prog am. Program interpretation of Minsk-32 instructions is largely accounted for by interpretation of general extra codes for exchange and extra codes for exchange with external devices. Microprogram interpretation is propagated for both computational and IO operations.

IO emulation has been implemented for external devices in the basic set for the Minsk-32 by using suitable analogs in the YeS-1035 set of external devices. In the process, units designed to operate with the Minsk-32 or converted in advance for use with the YeS-1035 may be used.

The YeS-1035 computer compatibility facilities provide emulation of the Minsk-32 computer with a 1.4-fold increase in throughput.

8.4. The YeS-1045 Computer

The YeS-1045 is a general-purpose computer with medium throughput of 880,000 instructions/second; it is designed to solve a broad range of scientific and technical, economic, informational and special problems both in the standalone mode and in information processing systems.

The universality of the YeS-1045 is provided by the balanced indicators of processor throughput, the universal instruction set, a unified method for connecting various external devices and advanced software. The logical structure of this computer meets the requirements imposed on the logical structure of the YeS EVM-2.

The YeS-1045 computer is program compatible with YeS EVM-1 models (in the basic control mode) and with other YeS EVM-2 models from below upwards.

The structure of the YeS-1045 computer permits organizing a dual processor system based on two computers and mutlimachine systems, ensuring in the process high throughput, reliability and viability.

Dual processor systems are organized by creating a common extent of main (up to 8M bytes) and external storage for both processors operating under the control of a single operating system.

Multimachine systems are created by complexing at the level of channels by using channel-to-channel adapters, direct control facilities and a common extent of external storage; each computer operates under control of its own operating system.

In the YeS-1045 computer, the capability has been provided through a special interface for connecting an array processor developed to substantially raise the efficiency of solving problems on pattern recognition, processing of geophysical data, etc.

A special access method under control of the OS 6.1 operating system allows the user to solve problems with the use of high-level languages.

The array processor together with the power supply and engineer panel is placed in a standard YeS EVM rack.

The YeS-1045 computer hardware has been developed on the modular principle, which allows a user to design specialized compuer systems corresponding to the purpose of the system being created to the greatest extent without making any changes to the structure and design of the machine.

Standard YeS-1045 computer facilities included in any system configuration created by a user include:
the processor;
facilities for processing in the basic and extended control modes;
the YeS EVM-1 universal instruction set and the majority of the new YeS EVM-2 instructions;
virtual storage;
instruction retry facilities;
monitor facilities;

program event recording;
high-speed 8K-byte buffer storage;
byte-multiplexer channel; and
five block-multiplexer channels.

Optional YeS-1045 computer facilities include:
the array processor;
direct control facilities;
facilities for organizing a dual processor system;
extension of main storage to 4M bytes by adding new 1M-byte blocks;
two channel-to-channel adapters;
logic repeater; and
configuration panel.

New configurations of the YeS-1045 computer may be derived by adding specific additional facilities to the basic machine composition or by connecting a broad range of external storage units, IO devices or teleprocessing devices to the IO channels through the standard interface.

The central part of the YeS-1045 computer is placed in three standard YeS EVM racks (processor and IO channels, main storage and computer power supply). Also placed in the power supply rack are the channel-to-channel adapters, logic repeater and their power supply.

When connecting main storage built with integrated circuits, the standard set for the YeS-1045 is palced in two standard YeS EVM racks.

The CPU for the YeS-1045 computer includes the following units: microprogram control, instruction fetching and interruption servicing, arithmetic and logic, storage control, checking and diagnostics, and the control panel.

Machine control storage consists of two parts:
permanent, designed to store processor and IO channel control microprograms, and loadable, for storing microprograms, console operations, diagnostic microprograms and the microprograms for array processor access to the YeS-1045 computer. The microprograms are loaded from the console magnetic tape storage unit, the ML-45 cassette type.

A special high-speed unit, an accelerator, is included in the processor to speed up execution of certain "long" arithmetic and logic operations. Executed in the accelerator are all multiplication operations, translations into the binary and decimal systems for packing and unpacking, all primary shift operations and some move and store operations, 25 operations in all.

The YeS-1045 computer processor has the following technical parameters:

duration of machine step, ns	120
control principle	hardware-microprogram
width of arithmetic unit, bits	32
number of overlap levels	2

The processor includes different types of integrated circuit storage: control, buffer, local protection key storage, etc. The YeS-3206 main storage unit, designed to receive, store and output information, is used as main storage in the YeS-1045.

Ferrite cores are used to hold the information. The storage unit is built on the principle of the 2.5D access. Structurally, the unit is made on the base of a standard rack, in which are placed the control unit, storage modules (with 64K bytes in each) and power supply system. There are facilities for protecting main storage for store and read.

The storage interface permits effecting a bus increment of capacity (up to 4M bytes) of main storage by adding new OP [storage] modules.

The technical characteristics of the main storage unit made with ferrite cores are:

capacity, megabytes	1.0
cycle time, microseconds	1.25
access time, microseconds	0.65
width, bytes	8

The YeS-3267 main storage unit made with integrated circuits has the following technical parameters (4K bits to a package):

capacity, megabytes	1.0
cycle time, ns	850
access time, ns	650
width, bytes	8

A rather advanced checking and diagnostic system is provided in the YeS-1045 computer to support high operating reliability and repairability.

A large share of the equipment (on the order of 95 percent) is hardware checked by self-checking circuits, which permits detecting malfunctions quite close to the time and place they appear.

The YeS-1045 computer recovery facilities provide the capability of continuing or recovering the computing process when a random failure occurs. This is done either by correction of single errors in main storage or by hardware-microprogram retry of the 179 CPU instructions, in which the error occurred. Upon successful retry, normal operation continues, but the error is logged for subsequent analysis. Otherwise, automatic retry of the situation that caused the malfunction is performed up to eight times. An unsuccessful execution of all retry attempts causes storing of the computer status and an interruption from the check circuits.

The high resolution of the diagnostics that enables localization of the malfunction with an average precision of down to two-three TEZ's [standard exchange cards] is achieved because of the use of the microdiagnostic procedures with a capacity to 1M byte, the programs of which are stored on the ML-45 tape storage unit. This provides the capability of checking the main assemblies of the central devices within 10-12 minutes.

The diagnostic system also includes the autotester apparatus designed to check TEZ's and permitting localization of a malfunction within individual cards with a precision down to one or more microcircuits. A malfunctioning element on a card is localized by using diagnostic tables and a special tester, and rapid recovery of computer serviceability is enabled after a failure. Provided in the YeS-1045 computer is the special unit for checking and diagnostics of the power supply (ASKDE) that permits performing an automatic change of voltages of power sources, as well as automatic checking of the status of power sources, fans, thermosensors, etc.

In the YeS-1045 computer, IO is performed through the byte-multiplexer and block-multiplexer channels. The combined hardware-micorprogram method is used to control the IO channels. Data is exchanged through the IO interface under control of the channel hardware facilities and is executed concurrently with processor operation. Data exchange between the IO channels and main storage and processing of control information are performed by processor i.cilities under microprogram control. The IO channels share the apparatus of control storage with the CPU.

Up to two byte-multiplexer and five block-multiplexer channels may be connected to the YeS-1045 computer. The byte-multiplexer channel can operate in the multiplexer or burst modes at a data transmission rate of 40K and 12K bytes/second, respectively, and has up to 256 subchannels.

Throughput of the block-multiplexer channels as a function of their number varies from 0.5M to 1.5M bytes/second, and total throughput of all IO channels is 5M bytes per second.

Up to 10 control units at a distance of 60 m may be physically connected to each channel. The special unit, the logic repeater, is included in the machine to increase the number (to 19) of devices connectable to the byte-multiplexer channel.

8.5. The YeS-1055 Computer

The YeS-1055 general-purpose computer has a medium throughput of 450,000 instructions/second and is designed to solve a broad range of scientific and technical, economic and special problems both in the standalone mode and in an information processing system, including real-time and shared-use systems.

The YeS-1055 computer has most of the properties of the YeS EVM-2 principles of operation, the main ones of which include:
the universal instruction set, except the four multiprocessing instructions (SPX, SIGP, STAP and STPX);
the basic and extended control modes;
organization of virtual storage by hardware facilities in the processor and channels in the aggregate with program support of the operating system;
the block-multiplexer mode of channel operation and the universal interface for channel communication with peripheral control units;
calls of special-purpose interruptions during execution of programs through use of the monitor facilities that permit acquisition, analysis and logging of information on the status of program processing at the time of interruption;
logging of events in a program on branches, changes in contents of general registers and main storage;
correction of single errors by using correcting codes during reference to main storage;
obtaining information on the time of processes and analysis of their status by using the time-of-day clock, comparator and CPU timer;
analysis of the status of the internal logic circuits in the CPU and channels by using microdiagnostic procedures;
instruction retry when an error occurs during instruction execution;
extended precision floating point operations; and
operand placement on an arbitrary byte boundary.

The standard facilities in the logic structure of the YeS-1055 computer are: the processor, control console, byte-multiplexer and block-multiplexer channels.

Optional facilities for the logic structure of the YeS-1055 computer are:
a second byte-multiplexer channel;
two block-multiplexer channels;
array processor;
direct control facilities;
channel-to-channel adapter; and
facilities for multiprocessor operation.

The YeS-2655 CPU has all the necessary function units for unpacking and executing instructions and controlling the computing process; it also has all the facilities for raising the efficiency of computations and extending system capabilities in accordance with the YeS EVM principles of operation. It meets the requirements for interactive operation with a large number of subscribers and common use of data files.

The principle of microprogram control is implemented in the processor. Control storage has a capacity of 8K instructions with 64 bits each, cycle time of 380 ns and access time of 140 ns. In the main, storage of microprograms is performed as a storage unit with permanent information. Only part of the microprogram storage is reloadable, the part used by the system for error correction.

The YeS-1055 computer CPU includes the processor proper, main storage and the IO channels.

The YeS-3204 main storage unit is made of MOS IC's with a 1024 x 1 organization. Used in the structure of main storage is the method of dividing the storage into four independent logic modules, access to which si effected by overlap according to the principle of address inetrleaving, which enables the specified parameters of model throughput. Main storage has two main versions: 1024K bytes and the maximum of 2028K bytes. Storage cycle time is 1140 ns; access width is 8 bytes. Facilities are provided to correct single and identify double errors in main storage, and to protect storage for reading and storing. Using the virtual principle allows increasing the efficiency of use of main storage.

The YeS-1055 computer IO system has byte- and block-multiplexer channels. The latter can also be used in the selector mode. Up to four block- and two byte-multiplexer IO channels can be connected. One byte- and two block-multiplexer channels are used in the main design.

The byte-multiplexer channel can operate in the multiplexer and burst modes at a data transmission rate of 40K and 1M bytes/second, respectively, and has up to 256 subchannels. The block-multiplexer channel has a data transmission rate of 1.5M or 3M bytes/second for the one-byte or two-b,te interfaces, respectively. The total throughput of all IO channels is 4-5M bytes per second.

The channel equipment contains a special unit to check the operation of the IO channels. This is a feature of the YeS-1055 computer. The channel check unit simulates operation of peripheral equipment and communication interfaces, which on the one hand, facilitates debugging and checking of the IO channels, and on the other, permits rapid and convenient localization of a malfunction that has occurred.

A supplementary unit for the YeS-2655 processor is the array module; it is not a functionally independent device and can be used only in the YeS-1055 computer. This is also a model feature. In contrast to the general method of connecting an array processor as an IO channel (see chapter 1), the array module is connected directly to the YeS-2655 processor as an executive resource. Structurally, it is made so that it can be connected to a machine already installed.

The array module is a specialized executive unit designed for rapid calculation of floating point operations used in array computations and Fourier transforms. In the process, because of parallel execution of different processes, a high rate of computation is achieved; depending on the size of fields, density of instruction stream and algorithm used, it may be 10-50 fold greater than that achieved in executing these operations in the arithmetic unit with floating point.

Connecting the array module directly to the processor opens the fundamental capability of organizing its operation concurrently with the operation of the executive units in the processor. This raises considerably the computation rate in the processor. However, specific realization of this operation involves modifying the operating system and considerable difficulties in control of the computer processor. Connecting the array processor through a standard IO interface does not provide to the full extent overlap of its operation with the operation of the arithmetic unit in the processor, but in return allows using standard control procedures. A quantitative evaluation of the effectiveness of the first and second methods of connection has not yet been exhaustively studied; therefore, the operating experience of the YeS-1055 computer will be useful in solving this problem and selecting the direction of development of specialized processors.

In the YeS-1055 computer, much attention has been paid to development of system control facilities and to the improvement of convenience in operation interaction with the computer. The YeS-7069 operator display console, a structurally independent unit, is used for these purposes. It is used both in controlling the system and in servicing it. It contains all the necessary controls and facilities for maintenance and display. The IO unit is a display with light pen and keyboard. The screen displays 25 lines of 80 characters each. The keyboard is used to input the necessary information and specify the control function; it has 26 alpha, 10 numeric and 27 special characters. One version of the keyboard also includes 20 Cyrillic characters. A high rate of communication is achieved through connecting a serial printer to copy the information output to the display. The serial printer operates at a rate of 45 cps.

The YeS-7069 operator console is connected to the processor by means of standard and special interfaces. The standard IO interface allows connecting the YeS-7069 unit to any YeS EVM-1 or EVM-2 model as an operator console. The special interface is designed to be used only with the YeS-1055 computer for display output and performing diagnostic procedures. The CPU allows connection of two YeS-7069 units. The second unit is supplied at user option. The YeS-7069 operator console has high esthetic and operating characteristics ensuring high comfort in computer servicing.

A peripheral feature for the YeS-1055 computer is the use of the YeS-7602 microfiche output unit. Output rate is 5 microfiche/minute, which is about 250,000 characters/minute. Lettering of each microfiche with visually readable microtype is possible. The operating system supports operation of this unit.

Another feature of the YeS-1055 computer is the availability of YeS DOS emulation facilities. These facilities allow operation of YeS DOS under control of the YeS OS control program. In other words, these facilities allow execution of programs, written by the user for YeS DOS, under control of the YeS OS operating system.

8.6. The YeS-1060 Computer

The YeS-1060 general-purpose computer has a high throughput of 1 million instructions per second. The concepts incorporated in the design of the YeS-1060 assume its use in major computer centers and automated management systems for solving a broad range of scientific and technical, plann-economic, informational and special problems in the modes of local and remote processing of information. Capabilities of versatile application of this computer are provided by the universal instruction set, large size of main and external storage, high-speed IO channels and broad set of peripherals. The availability of these facilities as well as the advanced system of interruptions, facilities for time readout and main storage protection allow efficient use of the YeS-1060 computer in the multiprogramming, time-sharing and interactive modes.

The YeS-1060 computer has all the capabilities of the YeS EVM-1 and realizes all the new concepts incorporated in the logic structure of the YeS EVM-2:
extended instruction set;
extended control mode in the processor;
dynamic address translation in the processor;
indirect data addressing in the channels;
block-multiplexer mode of channel operation;
new multiprocessor facilities;
extended precision floating point operations;
extended interruption system;
new facilities for time readout;
provision of monitor programs;
program event recording; and
increase in efficiency of checking and diagnostic facilities.

The architectural and structural features of the YeS-1060 computer are aimed at raising throughput, reliability and efficiency of use and creating simplicity and convenience in servicing.

The YeS-2060 processor provides for fetching of data from main storage, controlling the sequence of instruction execution, organizing interruptions, initiating operation of IO channels and implementing functions for checking and diagnostics. The processor includes: central control unit, arithemtic and logic unit, storage control unit, checking and diagnostics unit and the control console.

The central control unit is designed to fetch, unpack and buffer instructions, control the operation of the arithmetic unit and execute system functions. This device includes the units of instructions, data addresses, microprogram control, interruptions, timers and external communications unit.

The arithmetic and logic unit is used to execute arithmetic and logic operations and generate operand addresses. It includes: general and floating point registers and a parallel adder. An optional unit is a fast multiplier.

The storage control unit is designed for processor and IO channel communication with main storage and includes main buffer storage, dynamic address translation facilities, channel buffer unit and storage adapter.

The checking and diagnostics unit provides for control of the processor indication system, control of hardware checking circuits and check interruptions, recoraing processor status, hardware retry of instructions after detection of an error, execution of the functiors of the DIAGNOSE instruction, loading of microprograms into control storage, execution of microdiagnostic tests for localization of malfunctions and control of the synchronization system.

The control console is intended for manual control and display of computer status during maintenance and operation.

High speed of the processor is achieved through organization of rapid access to data, overlap of execution of operations and use of efficient computation algorithms.

The 8K-byte main buffer storage is built with fast integrated circuits. It is used to match processor operation time with main storage time parameters. Data is exchanged between main storage and the buffer in 32-byte blocks (pages). Four-way interleaving of addresses is used to speed up exchange. The buffer storage operating cycle, equal to the processor operating cycle, ensures rapid preparation of instructions and operands. Using an efficient algorithm for replacement of information in buffer storage combined with the use of a channel buffer permits reducing conflicts in the processor.

Efficient algorithms are used in the arithmetic unit. This makes it possible to combine a high rate of execution of operations with reasonable outlays for equipment. Using a special unit for fast multiplication has made it possible to raise the speed of execution of multiply operations 2-2.5 fold.

Instruction processing in the processor has three stages:
instruction fetching from the buffer, unpacking and generating of the operand address;
reference to storage and fetching of data;
execution of the operation and storage of the result.

In accordance with this, several instructions, at different levels of processing, are executed at the same time in the processor. Operation of the overlap levels is strictly synchronized in time, which simplifies organzation of control. Concurrently with processing of the instructions, the address of the next sector of the program is generated and instructions are read from storage to the buffer register.

Besides the purely logic capabilities of the computer, the simplicity and convenience of operation of the machine and rapid location of malfunctions are very important. During structure development, the designer almost always solves a compromise problem: achieving high throughput with reasonable computer complexity. In the YeS-1060 computer, the solution to these problems is shaped by:
the introduction of microprogram control not only in the arithmetic unit, but also in the central control unit; and
the introduction of an efficient checking and diagnostics system.

Microprograms control instruction processing in the processor, execution of operations and interruption processing. The writable control storage is built with integrated circuits and consists of two units: basic and control, which are accessed by independent addresses. The main unit of microprogram storage holds 4096 words of 144 bits each; the control unit holds 512 words of 24 bits each.

The microprograms stored in the control unit determine the sequence of processing stages and processor steps in instruction execution and also the degree of overlap of the processing stages. They also interrupt the sequence of the computing process when conflict situations, outside intervention or interrupt requests occur.

The microprograms in the basic unit of control storage determine the specific actions needed in the process of preparing instructions and executing operations and interruptions.

In the YeS-2060 processor, all units for central control and the arithmetic unit operate under microprogram control. The unit of system facilities for control and the checking and diagnostic unit have mixed hardware-microprogram control. This widespread introduction of microprogram control is a feature and advantage at the same time of the YeS-1060 computer, since until recently, this principle was not widely used for high-throughput machines.

The high requirements for operating reliability in the YeS-1060 are met by an advanced checking and diagnostic system. This system includes hardware and software facilities that interact both in the main operating mode and in the maintenance mode.

The hardware facilities effect online checking of processed information, elimination of the consequences of failures by retrying up to eight times instructions and sections of programs in the processor, retry of IO procedures in channels, and correction of single errors in main storage. To analyze the causes of failures, processor facilities allow fixing in main storage the status of equipment when an error occurs.

Under the conditions of preventive maintenance and servicing, it is possible to diagnose equipment by using special hardware facilities and a set of test programs and by executing microdiagnostic procedures to locate malfunctions. Online tests for checking external devices in the process of executing user tasks allow continual tracking of the status of external devices and when necessary deciding whether or not to use them in the system. These capabilities combined with the operating system recovery facilities create the prerequisites for efficient operation of the YeS-1060 computer at the user site. This is also facilitated by the software system that includes the latest version of YeS OS, application program packages that extend operating system capabilities, and application program packages for various purposes.

YeS-1060 system facilities permit combining the resources of several computers to execute a common task, ensuring in the process high throughput and reliability. Communication between the individual computers can be effected at the level of any system component by using specific multiprocessor facilities:
at the level of channels by using a channel-to-channel adapter;
at the level of peripheral control units by using a two-channel switch;

at the level of the basic main storage;
at the processor level by using direct control facilities.

The facilities that ensure efficiency in system application of the computer allow creating powerful systems and multimachine computer complexes based on the YeS-1060.

The YeS-1060 computer IO channels are physically independent standalone units that are connected to the processor and profit by simultaneous access to main storage. Information transfer rate depends substantially on the number and type of channels used. Up to seven IO channels, supporting the byte and block-multiplexer and selector modes of operation, may be conencted to the YeS-1060 computer processor. The block-multiplexer channels have the two-byte interface which allows connecting devices distinguished by a high data transfer rate.

To operate the computer in the virtual storage mode, indirect addressing facilities have been incorporated in the channel to complement the address translation facilities in the processor. Hardware retry of instructions in a channel has also been provided. The availability in the channel of facilities to log channel status makes it possible to recover system operation by program methods when errors occur in an IO procedure.

Structurally, the YeS-1060 computer channels are made in the form of YeS-4001 units that include four functional channels: one byte- and three block-multiplexer. The block-multiplexer channel can operate in the mode of a selector channel. The block-multiplexer channels are controlled by hardware facilities; the principle of microprogram control is used for the byte-multiplexer channel.

The byte-multiplexer channel consists of the main multiplexer channel and four selector subchannels. Throughput of a selector subchannel is 500K bytes/s. The integrated throughput of the byte-multiplexer channel is 1.5M bytes/s. The block-multiplexer channel has a throughput of 1.5M bytes/s with the one-byte interface and 3M bytes/s with the two-byte interface.

There are two versions of main storage for the YeS-1060 computer: ferrite cores and integrated circuits. The ferrite-core main storage unit has a capacity from 2M to 8M bytes, reference cycle of 1.25 microsecond and access time of 0.8 microsecond. To speed up access to main storage, it- entire extent is divided into four independent logical blocks. The YeS-3206 main storage unit is built on the modular principle and contains one megabyte per rack. The power supply is placed in the same rack. Structure, circuit and design solutions for the YeS-3206 unit have permitted reducing the bulk and raising the density of equipment packaging compared to the main storage unit in the YeS-1050 computer which uses the same ferrite cores.

The YeS-3266 semiconductor main storage unit for the YeS-1060 computer is made of dynamic inetgrated circuits with 16K bits per package. It has the same interface for communication with the processor and the same organization as the YeS-3206 unit. Capacity of the YeS-3266 unit is 8M bytes, cycle time is 680 ns and access time is 520 ns.

8.7. The YeS-1065 Computer

With a throughput of four to five million instructions per second, the YeS-1065 computer is the top model in the YeS EVM-2. It is intended for application in major computer centers and large data processing systems. This computer implements all data processing modes and uses all the architectural and logic capabilities provided in the YeS EVM-2. The main quality of the computer, high throughput, predetermines the features of its application and the features of the structural implementation.

First of all, the YeS-1065 model is a general-purpose computer. In these terms, it has to have the properties of universality and provide the capability of implementing the most varied classes of algorithms for user tasks. At the same time, it has to have the properties of a special system application. In these terms, the computer must have the characteristic features of an open system that allow new problem-oriented devices to be included in its makeup. Considering this, one can formulate the basic properties of the YeS-1065 computer architecture.

1. Structural solutions are aimed at achieving the maximum rate of execution of a random stream of instructions processing any data.

2. The computer structure provides the capability of incorporating in its makeup special processors to increase the rate of processing in systems used to solve a limited set of tasks. The purpose of these processors is to increase at least by an order the rate of execution of individual functions most often encountered in the computing process at a specific user site.

3. There are advanced complexing facilities in the computer to enable creating multiprocessor and multimachine complexes for a general increase in system computing capacity.

A processor structure that can be described as one with common resources was chosen for the YeS-1065 computer. This assumes the availability of several devices that prepare instructions for execution, but the execution of these instructions is performed in one operating device. Nevertheless, even with this structure, it is expected that special organizational steps are taken to reduce the processing time at all stages of instruction preparation and execution.

To reduce the data fetching and storing stage in the YeS-1065 computer processor, a 32K-byte main buffer storage unit is used; it operates at the rate of operation execution in the executive units. Information is exchanged between main storage and the buffer in 32-byte blocks. An additional gain in the operating rate at the stage of storage reference is obtained because the results of operations are stored only in buffer storage without immediate duplication of them in main storage. These results are stored in main storage when the least recently used information is destaged according to the standard discipline.

In the arithmetic and logic unit, a common resource in the YeS-1065 computer processor,operation execution time is reduced by the implementation of the fastest algorithms for oepration execution and by dividing this unit into several independently operating units (operating devices) for information processing. Each of these devices is oriented to executing a group of instructions close in type. This simplifies the structure of the operating device, reduces the outlays for implementation of each device and permits reducing the time for execution of operations compared to that for a general-purpose device, although total costs of equipment incr?ase.

Provided in the structure of the YeS-1065 computer processor are four independent units for execution of operations: fixed-point arithmetic, floating-point arithmetic, arithmetic for processing variable-length fields and decimal numbers and, finally, the fast multiplier (divider). Access to these units is independent and each may operate concurrently with the others. There is also the capability of increasing total capacity of the arithmetic unit by connecting both similar devices and problem-oriented devices for information processing.

The toatl throughput of all operating devices is several tens of millions of operations per second and is considerably higher that the throughput of the storage control unit and the central control unit (instruction processor). In this case, it is advisable to include in the processor structure several instruction processors and divide buffer storage into independent sections with their own control and access to main storage, i.e. to have several storage control units. The YeS-1065 computer processor has two storage control units, each with a 32K-byte buffer and two instruction processors. Each instruction processor manages its instruction stream to jointly maintain the required load on the arithmetic unit.

The structure with common resources, like the multiprocessor from the viewpoint of the operating system, operating with a common extent of main storage, has a number of advantages. First of all, there is the effective increase in throughput and provision for system viability. The computer continues functioning, though with less throughput, when the separate devices fail. When a second arithmetic unit is included in the computer, which is also a common resource, the reliability of such a computer system increases manifold. The inclusion of several instruction processors in the processor also simplifies organization of the structure of each. In doing so, organizing a large number of overlap levels is avoided and the microprogram principle of control is implemented.

In the computer with common resources, problem-oriented processors that raise the overall machine throughput may be connected to all the instruction processors with the rights of a common resource. For this, they have to have an interface to the instruction processor similar to the operating devices. For the instruction processor, this connection of new devices boils down to implementing additional instruction codes corresponding to the functions of the equipment being incorporated, which is rather simple with the availability of microprogram control. Far more complex is the software problem for these devices that has to be solved individually in each specific case.

The high rate qualities of the YeS-1065 computer processor are supported by a powerful IO system, large main storage and service facilities.

Along with the traditional structural solutions, when the channels are rigidly attached to the processor, the principle of floating channels (cross-call channels) is used in the YeS-1065. The principle of the universal channel (IO multiplexer), when execution of the functions of the byte- or block-multiplexer modes is enabled by the use of microprogram control, has also been implemented. The channels are operationally adjusted for the required mode by reloading of microprogram storage. All the new functions of the YeS EVM-2 are performed in the YeS-1065 computer channels: indirect data addressing, IO operation retry and the two-byte interface.

Up to 16 channels, enabling total throughput of about 30M bytes/s, are connected to the YeS-1065 computer. The main storage unit for this computer is made of 16K-bit microcircuits. The storage structure provides eight-way address interleaving. Storage size is 8M bytes, cycle time is 870 ns and access time is 650 ns.

A special processor for diagnostics and control, placed in the system control console, is used in the YeS-1065 computer. Its functions include performing diagnostic procedures and maintaining direct contact with all central units in the processor.

The YeS-1065 computer has an advanced error detection system, recovery facilities to eliminate the effect of random failures through automatic retry of operations, and error correction facilities using correcting codes. The microdiagnostic system permits detecting malfunctions during preventive maintenance and servicing. The computer structural features govern the specific functions of the console processor: tracking the functioning conditions of the individual devices, acquisition and analysis of information on failures, disconnection of malfunctioning devices and diagnosis of them, and decision-making on system configuration. To do this, it has large internal storage and the required set of its own external units.

Conclusion

Development of the Unified System of Computers has made it possible to a considerable extent to solve the problems of equipping the national economy of the countries in the socialist community with computer facilities. The Unified System is now a powerful, advanced system of computers in the socialist countries, has been assimilated into production and has found widespread application in various sectors of the economy. Work on the Unified System has not ended with the development of the first and second phases. The main directions of further development for the Unified System are:
increasing the efffectiveness (throughput/cost ratio) of computers with respect to the YeS EVM-2;
improving technical parameters and operating characteristics;
increasing the effectiveness of introduction into various spheres of the national economy through functional specialization of hardware, use of programmable hardware, built-in hardware control facilities and organizing data processing networks;
increasing further the effectiveness of stating and solving problems, including solving the problem of using a common information base and further introduction of problem-oriented language facilities; and
reducing the hardware and software maintenance costs by providing high reliability, and improving the methods of diagnostics and effective redundancy. The determining feature of this development is the shift to the new technological design base using LSI circuitry which is predetermining the creation of fourth-generation computers.

BIBLIOGRAPHY

1. "IBM System/360. Principles of Operation," ed. by V. S. Shtarkman, Moscow, Sovetskoye radio, 1969.

2. Gorbatsevich, S. L.; Makurochkin, V. G. and Cheremisinov, V. M., "Development of the Main Units for External Storage in the Unified System of Computers," in "Vychislitel'naya tekhnika sotsialisticheskikh stran" [Computer Technology of the Socialist Countries], No 6, 1979.

3. (Jermain, K.), "Programming for the IBM/360," Moscow, Mir, 1973.

4. Larionov, A. M., ed., "Yedinaya sistema EVM" [Unified System of Computers], Moscow, Statistika, 1975.

5. Zhimerin, D. G., "Application of Electronic Computer Technology in the Socialist Countries," in "Computer Technology of the Socialist Countries," No 5, Moscow, Statistika, 1979.

6. Katzan, H., "Operating Systems," Moscow, Mir, 1976.

7. Katzan, H., "System/370 Computers," ed. by V. K. Levin and L.D. Raykov, Moscow, Mir, 1974.

8. (Colin, A.), "Introduction to Operating Systems," ed. by V. V. Martynyuk, Moscow, Mir, 1975.

9. Lapin, V. S., "Development of Cooperation in Data Teleprocessing. New Tasks," in "Computer Technology of the Socialist Countries," No 4, Moscow, Statistika, 1978.

10. Lapin, V. S. and Semenikhin, V.S., "Concepts and Prospects for Development of Data Teleprocessing in the Unified System of Computers," in "Computer Technology of the Socialist Countries," No 1, Moscow, Statistika, 1977.

11. Larionov, A. M.; Levin, V. K. et al., "Basic Principles of Design and Technical and Economic Characteristics of the Unified System of Computers (YeS EVM)," UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 2(4), 1973.

12. Larionov, A. M.; Levin, V. K.; Przhiyalkovskiy, V. V. and Fateyev, A. Ye., "Technical and Operating Characteristics of Models in the Unified System of Computers," UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 2(4), 1973.

13. Enslow, P. H., ed., "Multiprocessors and Parallel Processing," Moscow, Mir, 1976.

14. Peledov, G. V. and Raykov, L. D., "Vvedeniye v OS YeS EVM" [Introduction to the OS for the Unified System of Computers], Moscow, Statistika, 1977.

15. "IBM System/370. Principles of Operation," ed. by L. D. Raykov, Moscow, Mir, 1975.

16. Przhiyalkovskiy, V. V. and Lapin, V. S., "Architecture of Open Systems for Data Teleprocessing Networks in the Unified System of Computers," AVTOMATIKA I VYCHISLITEL'NAYA TEKHNIKA, No 2, Riga, 1979.

17. Przhiyalkovskiy, V. V., "Some Results from the Development of the Unified System of Computers and Prospects for Further Development of It," in "Computer Technology of the Socialist Countries," No 6, Moscow, Statistika, 1979.

18. Pykhtin, A. Ya.; Gorelov, V. I. et al., "YeS-7920 Complexes: New Terminals with Broad Capabilities," in "Computer technology of the Socialist Countries," No 4, Moscow, Statistika, 1978.

19. Rakovskiy, M. Ye., "Decades of Cooperation," in "Computer Technology of the Socialist Countries," No 6, Moscow, Statistika, 1979.

20. Fateyev, A. Ye.; Roytman, A. I.; and Fateyeva, T. P., "Prikladnyye programmy v sisteme matematicheskogo obespecheniya YeS EVM" [Application Programs in the Software System for the Unified System of Computers], Moscow, Statistika, 1978.

21. Larionov, A. M., ed., "Elektronnaya vychislitel'naya mashina EVM YeS-1050" [The YeS-1050 Electronic Computer], Moscow, Statistika, 1976.

8545
CSO: 8144/1811

- END -