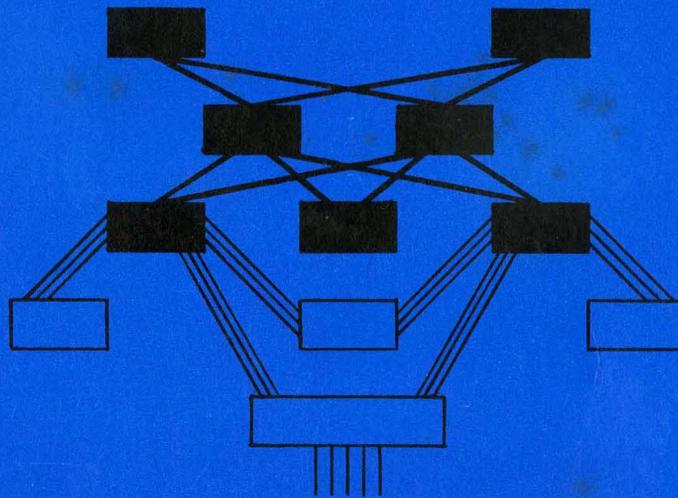


System Manual

GE-645

 Information
Systems

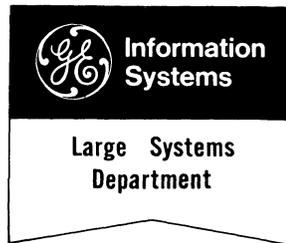
Large Systems
Department



GENERAL  ELECTRIC

GE-645

SYSTEM MANUAL



January 1968

In the construction of the equipment described, the General Electric Company reserves the right to modify the design for reasons of improved performance and operational flexibility.

GENERAL ELECTRIC

INFORMATION SYSTEMS EQUIPMENT DIVISION

COMPATIBLES / 600

PREFACE

Part I of this manual provides an introduction to the GE-645 Information Processing System. The first two chapters trace the thinking which brought about the development of an advanced information processing service concept. This concept assumes computer service available in the same way electricity is made available simultaneously to many users. The GE-645 is aimed at providing this type of service. Chapter 3 provides a summary which allows the readers to obtain a general

understanding of how the GE-645 approaches this problem.

Part II provides moderately detailed information about the major hardware components of the GE-645 System and contains a generalized description of the objectives and planned operational features of the Multics Operating System and related software as currently being developed for the GE-645 System.

ACKNOWLEDGEMENT

Multics (Multiplex Information and Computing Service) is an operating system concept developed from research by the Massachusetts Institute of Technology, Bell Telephone Laboratories and the General Electric Company. It draws upon the design and operating experience gained with CTSS (Compatible Time-Sharing System) at the Computation Center and Project MAC of the Massachusetts

Institute of Technology, and upon programming language research and system principles from Bell Telephone Laboratories.

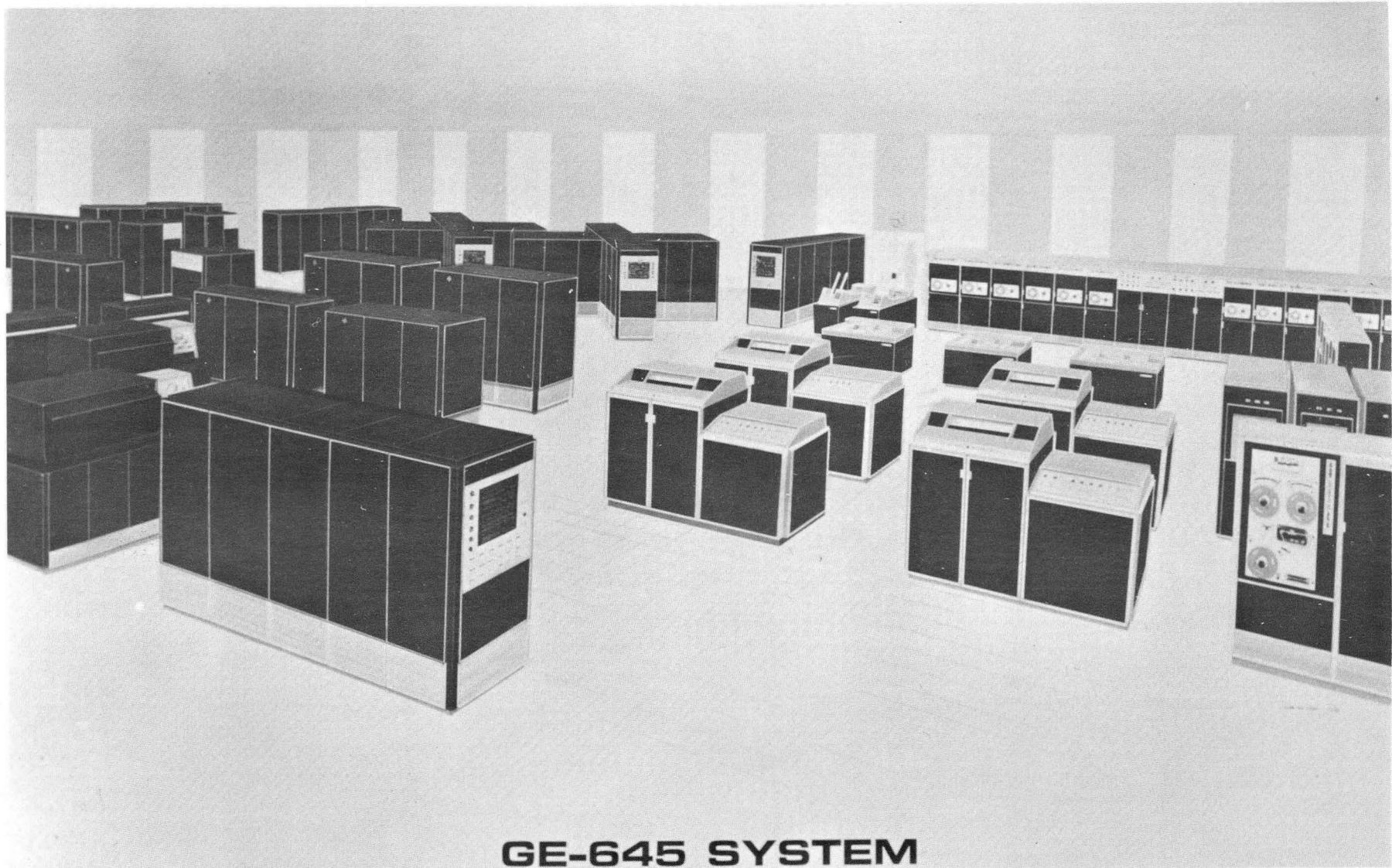
The descriptive material on Multics contained in this document is derived from the technical objectives established by the above organizations for the continuing implementation of that concept on the GE-645 System.

CONTENTS

		Page
PART I. Introduction to the GE-645 System		
1.	PROBLEM STATEMENT	1
2.	DESIGN CONCEPTS	3
3.	SYSTEM SUMMARY	5
	Total System	5
	Hardware	5
	Software	7
	Operating System	7
PART II. GE-645 System Description		
4.	HARDWARE SYSTEM CHARACTERISTICS	9
	Hardware System Organization	9
	System Configurations	10
	Connecting Input/Output Devices to the System	11
5.	SYSTEM CONTROLLER MODULE	15
	Storage Function	15
	Control Signals	15
	Connect Instruction	15
	Interrupt Cells	15
	System Clock	16
6.	PROCESSOR MODULE	17
	Modes of Operation	17
	Register Descriptions	17
	Instruction Repertoire	18
	Address Modification	19
	Segmentation	20
	Descriptor Segment	21
	Paging	27
	Associative Memory	29
	Faults and Interrupts	29
	Privileged Operation	34
	Access Control	35
7.	GENERALIZED INPUT/OUTPUT CONTROLLER	37
	Channels	37
	Adapters	37
	GIOC Controller	42
8.	EXTENDED MEMORY MODULE	45
	Organization	45

CONTENTS (Cont)

	Page
Performance Characteristics	45
Operation	45
Test Modes	46
9. SYSTEM CONFIGURATION CONSOLE	47
Reconfiguration	47
Reconfiguration Data	47
Master/Slave Reconfiguration	49
System Status Display	49
Operator Teletypewriter Station	49
System Initialization and Bootload	49
10. PERIPHERAL AND TERMINAL EQUIPMENT	51
Peripheral Equipment	51
Disc Storage Unit (DSU10F)	51
Disc Storage Unit (DSU204)	53
Magnetic Drum Storage Unit (MDU200)	55
Mass Storage Unit (MSU388)	56
Magnetic Tape Subsystem	58
Card Reader and Control (CRZ201)	59
Card Punch and Control (CPZ201)	61
ASCII Extended Character Set Printer (PRT202)	62
Perforated Tape Subsystem (PTS200)	64
Peripheral Switch Console (PSC200)	65
Terminal Equipment	65
GE-115 Information Processing System	66
Datanet-760 Keyboard/Display Subsystem	68
11. MULTICS	71
Concepts	71
User Interaction with Multics	72
Supervisor	73
Command System	74
File System	75
File Structure	75
Basic File System	76
Multilevel and Backup System	77
I/O System	77
APPENDICES	
A. INSTRUCTION REPERTOIRE	81
B. HARDWARE SUMMARY	89



GE-645 SYSTEM

PART 1.
INTRODUCTION TO THE GE-645 SYSTEM

1. PROBLEM STATEMENT

During the early growth period of computers, a computer was a unique and complex piece of equipment, but was too expensive to give to each person with a problem. In order to justify the cost of a computer and to take advantage of its speed, batch processing became the standard mode of operation. A staff of experts was required to manage, program, and operate the computer. As a result, the man with the problem gradually had barriers placed before him, and he became more removed from the computer.

The computer language was a primary barrier. The man with the problem didn't know how to state the problem in a language acceptable to the computer. Specialists were called in who understood the computer languages, but didn't understand the problem. As the problem passed from specialist to specialist, it went through several stages of translation. The elapsed time from problem statement to problem solution was lengthened, and often a clear statement of the original problem was lost in the several translations. As the layers of computer specialists became thicker, the computer gradually became less accessible to the man with a problem.

Computer technology developed at a rapid pace. However, much of the development was concerned with computer equipment and computer techniques, and very little was concerned with the techniques required to make the computer a convenient tool for solving individual problems. Many times, the person with a problem must have said, "I'd put more work on the computer if it weren't such a chore." In frustration, he used methods not well suited to the needs of the problem.

The computer schedule was another barrier placed between the computer and the man with a problem. A computer was scheduled in order to get maximum use of the equipment. However, the tight schedules set for maximum use of equipment were not realistic, were difficult to enforce, and caused almost everyone to endure a long turnaround time waiting for the results. Additionally, all jobs, regardless of priorities, were put aside for demonstrations, maintenance, and real-time problems.

Despite the best scheduling efforts; full equipment utilization could not be maintained. The entire computer system became unavailable when certain peripherals (e.g., typewriter or card reader) became unavailable. At other times the computer system was idle for many minutes waiting for tapes to be mounted or cards to be placed in the card reader. When performance characteristics of separate computer components were examined in conjunction with the requirements of the particular problems being solved, it was found that some parts were idle while other parts operated at full speed. During

calculations for a scientific problem, the input/output section worked only intermittently while the processor worked continuously; however, during calculations for a business problem, the processor worked part time while the input/output was kept busy.

The allocation of storage media was another problem closely associated with scheduling. Did management really have control of the costs in the computer system? A junior programmer could easily accumulate 50 reels of magnetic tape and collect drawer after drawer of cards. Who determined when it was more economical to store data in magnetic core, on magnetic tape, or on magnetic drum? Decisions were based on an individual's past experience rather than on statistics gathered from group experience over a longer period of time.

As middle and upper management took a broader view of the problems, the need to share programs and data files among several groups became apparent. In some instances, different groups worked on separate portions of the same large problem and each group developed its own set of programs and data files. In other situations, hardware and software limitations made it difficult for one group to utilize the files of another. A tremendous amount of this information should have been centralized and made available to several different groups at the same time. This lack of integration led to wasteful duplication of effort and fell short in supplying the total information overview sought by management.

Periodically, new and expanding computer applications required users to increase the size of their hardware systems. However, this was difficult and expensive when whole systems had to be added or replaced. Without hardware modularity, an entire hardware system consisting of all three capabilities (processing, storage, input/output) had to be added to provide more of any one capability. To take advantage of the new hardware configuration, applications had to be reprogrammed, often at considerable cost to the user.

The evolution and growth of the computer has not stopped. The technology associated with computers and their application continues to evolve at an ever increasing rate. In order for the computer to continue to serve the needs of business, industry, science, education, and government, the above problems must be solved, and the barriers between the computer and the man with a problem must be eliminated.

Specifically the next generation information processing service must provide:

- Dependable operation

- Remote terminals to communicate with the utility center
- Simple user-oriented languages for communication of problems and solutions
- Short turnaround time to permit direct interaction with the computer
- Full multiprogramming batch processing capability
- Concurrent operation of interactive jobs and batch processing jobs with high efficiency
- Memory management that frees the user from any concern about how to fit his program and data into the system
- Input/output management that frees the user from any concern about input/output operations, unless he wishes otherwise
- A storage system which permits easy sharing of data and programs
- File protection mechanisms to ensure that access to files is restricted as specified by the owners
- Modular hardware to meet the user's changing needs
- Modular software to permit its dynamic modification
- Mechanisms to permit management control over resource usage
- Bookkeeping procedures to record resource usage and charge users appropriately
- Clear, readable reference and training material to guide users, operators, maintenance engineers, and administrators in the use and control of the system.

Much of the necessary technology already exists. Many types of input/output terminals can be connected to the computer. Remote terminals (e.g. unit record devices, teletypewriters, graphic displays) can be placed at locations convenient for the user. Many users can have access to a computer and to its resources from different locations at the same time. The problem is to integrate the existing and developing technologies into a complete, reliable, and economical system.

2. DESIGN CONCEPTS

New and carefully integrated system designs were needed for both hardware and software to approach the objectives of the next generation information processing service. Hardware features had to be provided to enable software supervisors to achieve their goals with required efficiency. Software systems had to be developed to exploit these hardware capabilities and to provide new levels of user service. As a result there evolved the GE-645 Computer System which uses the Multics Operating System. The following paragraphs outline the resulting design and functional concepts established for the overall GE-645 Computer System with the Multics Operating System.

The tasks which the man with a problem requests to be performed will normally take very little computer time to execute. Thus each user can proceed in the solution of his problem without being delayed by the computer. He is allowed to interact with the computer as it achieves the results he desires. The vast resources of the system are his to command. While using these resources, he has the impression that he is the sole user.

Once he has learned a few simple rules and system commands, the man with a problem can guide the computer through the steps of his problem solution from the keyboard of his remote terminal. If he is a novice in programming, he can obtain a useful knowledge of one of the simpler programming languages in a few hours. If he is an experienced programmer and wants greater power and flexibility, he can employ any of the widely used languages in the industry. Even when the GE-645, operating under Multics, is interacting with users at remote terminals, it can concurrently serve users with batch processing work.

When operating under control of Multics, a user is not concerned about the size of his program or data storage. Multics automatically moves unused parts of his program to secondary storage. Hence, both the information currently in core memory and that which Multics has placed in secondary storage are available to the user. In spite of the actual physical location of user programs and data, they appear to him as if they were all in core memory.

Multics allows a user to declare the degree of privacy to be observed in the use of each of his files. He may declare a file to be private, public, or accessible only to a specific list of other users. He can further restrict the use of a file by declaring the manner in which it may be accessed. For example, a file containing a program could be made public for purposes of execution but private for purposes of reading and modification.

A user may find it convenient to use subprograms or data which are provided by Multics or made available to him by other users. In fact, some programs and data files may be

needed concurrently by many users. Multics enables many users to share programs and data files. In essence, the system provides storage facilities analogous to the combined resources of a public library (public system files), the library of co-workers (shared files), and a personal library (private files).

Multiprogramming allows adequate service to be provided to all classes of users. It permits processing part of one program, then part of a second, then part of a third, etc. This allows multiple programs to receive service in rapid succession giving the appearance that they are all being executed simultaneously. If this technique were not used, each user would need to await the completion of many other user programs before having an opportunity to utilize a processor. Without multiprogramming, interactive users would experience intolerable delays.

The total processing capability of the system is increased by having multiple processors execute multiple programs concurrently. This technique is known as multiprocessing.

The control of the processors is directed from one user program to another by Multics. Every time a new program is placed in execution, special registers are initialized which prevent damage to information belonging to any other user or the operating system.

Large memories are made available in the GE-645 hardware. Through their use, many programs and their data may be stored concurrently awaiting execution. Even with the large memories available on the GE-645, it is generally impossible to allow the programs and data of all users to reside in core memory throughout their execution. The number of users that can be serviced by the GE-645 is made large by providing a high-speed fixed-head magnetic disc unit for moving information rapidly into and out of core memory. The number of users that can be serviced is also increased by employing the techniques of paging and segmentation.

In the concept of paging, a program is divided into pieces of equal size called pages. When the program is in execution, only its most active pages reside in core storage. Its less active pages are stored in the secondary storage hierarchy.

In the GE-645 the management of paged programs is facilitated by special hardware which makes the pages appear to be in contiguous locations even though they may be widely separated in core memory. The hardware also keeps track of which pages have been altered while they are in core memory so that unaltered pages need not be rerecorded in secondary storage when they are removed from core memory.

Segmentation allows a program to be divided into separate parts called segments. Each segment can be thought of as a separate core memory with its own origin and maximum size. One of the important features of segments is that they may be shared among many users. Hence, only a single copy of a program or data file need exist if these are properly stored in the segment being accessed by many users.

A second important property of segments is that they allow a user to collect in effectively separate memories information with like characteristics. For example, some or all of the unalterable information of a program can be brought together into a "read only" segment. This not only protects the information from inadvertent modification but also collects information which will never need to be rewritten to secondary storage during the paging operation.

Modular hardware lets the user reconfigure a system to fit the changing needs brought about by growth. A system can

consist of multiple memory, processor, and input/output controller modules. The modular hardware lets the user assemble a combination of modules that most closely fits his current need. This reconfiguration can take place without providing an excess of unwanted modules. For example, one processor module can be added when additional processing is needed, or one input/output controller and/or peripheral devices can be added when greater input/output capability is needed.

Modular software is a direct result of segmentation, and makes it practical to alter the operating system to fit computing needs. It also allows the operating system maintenance group to make appropriate improvements and modifications that are of benefit to the user. The same programming standards are used in Multics as in user programs so that maximum consistency and flexibility is achieved within the system.

3. SYSTEM SUMMARY

The previous chapters have discussed the general problems facing the user, the objectives of the GE-645 system, and the design concepts which enable the system to solve the user's problems. This chapter contains a brief view of the total hardware-software system followed by descriptions of the hardware and its components, and the software system and its components. Should the reader desire more information than is contained in this chapter, he may refer to Part II.

TOTAL SYSTEM

The GE-645 is a large-scale, high-performance, binary computing system that operates with the Multics operating system, several language processors, and a wide variety of application packages. Most of the equipment is located at the computer center, while terminals are placed in locations convenient for the users of the system.

HARDWARE

The hardware consists of four types of modules, peripheral and terminal equipment, and common carrier facilities for the terminals. The four module types are processor, generalized input/output controller (GIOC), system controller (memory), and extended memory. A full range of peripherals is available, from perforated tape units to large, high-speed discs. Terminals that may be used range from slow speed teletypewriters to complex display consoles and computers. Terminals are connected to the system with standard, presently-available communication equipment.

A typical hardware system organization is shown in Figure 1. In this example, two processors, four system controllers, two GIOCs, and one extended memory module are combined to form a nine-module system, together with a system configuration console. The figure shows the interconnections between the nine modules and the way that the peripherals and terminals are connected to the system.

The system configuration console establishes the basic parameters of the system configuration that make it possible for the modules to communicate with one another through the system controllers. The SCC also provides the means for rapid re-configuration of these parameters and for initiating the bootloading operation for placing the system in operation. A teletypewriter can be connected to the SCC to provide a printed copy of the "conversations" between the operator and the operating system.

The processor modules do the computation and decision-making in the system. Many registers and an associative (content-addressable) memory are provided in each processor to facilitate rapid execution of powerful instructions. Performance is also enhanced with instruction

overlap: that is, address preparation for the next instruction and fetching of subsequent instructions continue while the current instruction is being executed. The instruction repertoire is large and varied, and has 404 operation codes currently defined. (This leaves 108 operation codes reserved for future upward growth.) A wide range of conventional instructions is supplemented with many special-purpose instructions and several instructions specifically for character manipulation. Address modification includes conventional register modification and multilevel indirect addressing plus several variations particularly aimed at character manipulation and operating on stacks. All hardware manipulation related to segmentation and paging is done by a processor.

The GIOC modules handle all input/output for peripherals and terminals. (Note that the extended memory module is independent of the GIOC.) Input/output operations on a given GIOC are performed concurrently but independently of each other. The detailed actions to be performed during an input/output operation are specified in a list of control words stored in memory. Once a processor sets up the control words and initiates an input/output operation, the GIOC obtains the control words from memory and transfers data without any further processor action.

System controller modules contain the system's high-speed storage, the system's time clocks, and serve as control communication paths between the processor, GIOC, and extended memory modules. A system controller module consists of 32k, 64k, or 128k 36-bit words with a storage cycle time of one microsecond. The system can move data in groups of 6, 9, 18, 36, or 72 bits into or out of memory. The system clock consists of a calendar clock and an alarm register. The clock is located in a system controller module so that all processors can use the same clock. Thus, all processors use the same time base and perform time calculations in a consistent manner. Each system controller also contains a group of interrupt cells. These cells are used by a processor, GIOC, and extended memory module to notify the operating system of the occurrence of special events.

Memory is addressed in an interlaced manner. When a block of data is transferred into or out of memory, the data accesses are distributed among the system controller modules. This prevents a data transfer to the extended memory module, for example, from putting a heavy load on just one system controller and thereby effectively preventing other modules from using that system controller.

The extended memory module is a fixed-head disc which is used as an extension of memory. The capacity is four million words. The disc performance characteristics were carefully designed to match it to the GE-645 system. The

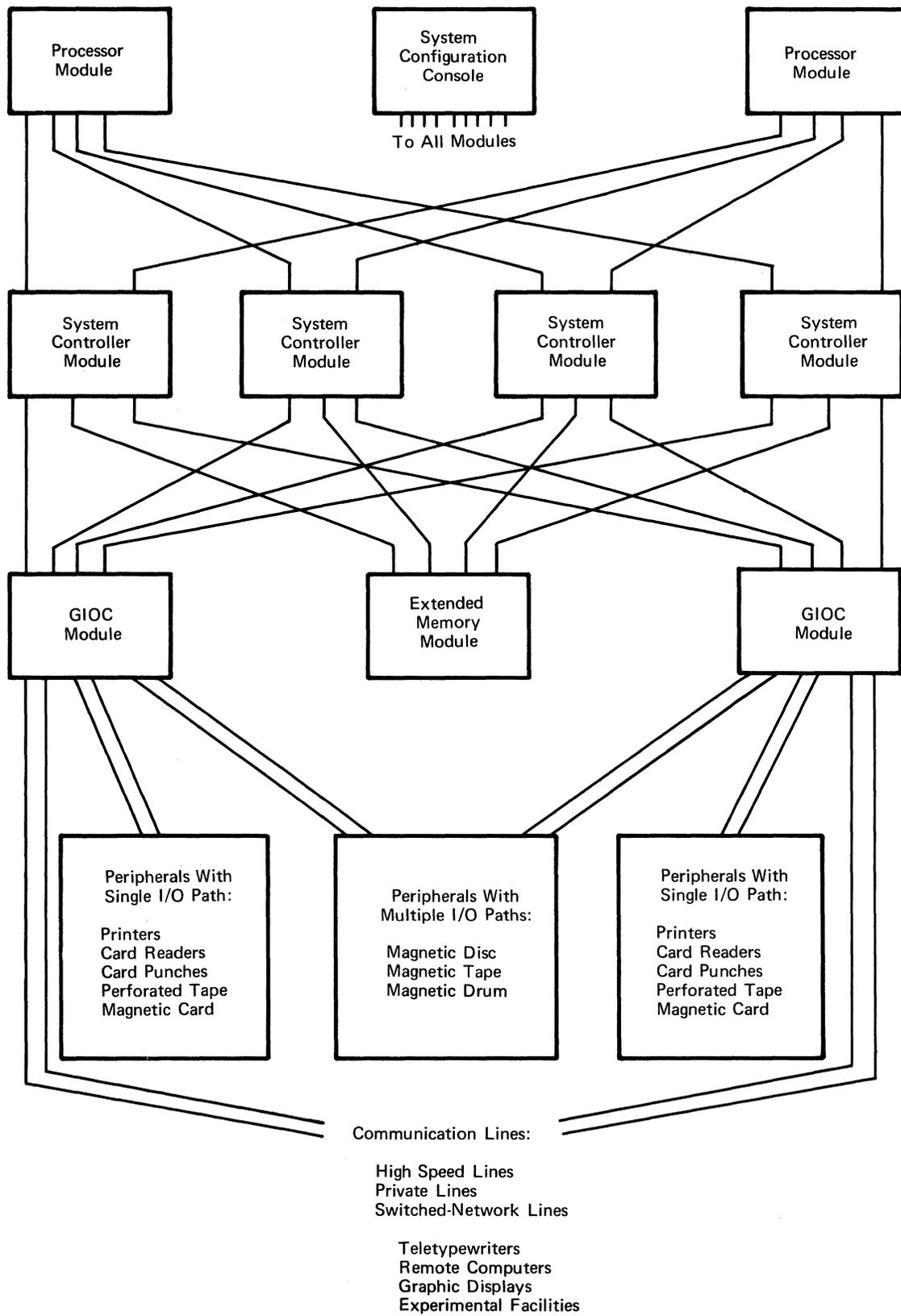


Figure 1. Hardware System Organization

block sizes used for information transfer are the same size as the pages handled by the processor: 64 or 1024 words. A transfer rate of 470,000 words per second was chosen to minimize the time required to move a block of data, yet not be so fast that memory is overloaded when the extended memory unit is transferring data.

The kinds of peripheral and remote terminal equipment associated with the GE-645 are:

- Large moving-head disc
- Magnetic drum
- Magnetic card
- Magnetic tape
- Card reader
- Card punch
- Printer
- Perforated tape
- Typewriter-like devices
- Other computers (for both normal I/O and for experimental facilities.)
- Graphic displays

The system shown in Figure 1 has nine modules. The system can be expanded to a maximum of sixteen modules: eight system controller modules and a combination of processor, GIOC, and extended memory modules.

The separation of the major system functions into several modules provides hardware redundancy in case of equipment failure and facilitates system growth. The proper quantities of the various types of modules may be combined to create a system that is well suited to the workload. Peripherals and terminals also have this modular property. Peripherals which operate at high speeds and require little operator attention are connected to both GIOCs to improve system performance and provide more than one path in case a GIOC should ever malfunction.

SOFTWARE

The GE-645 software consists of the Multics operating system, language processors, application packages, and compatible GE-625/635 series programs. The general capabilities of the operating system will be described briefly.

OPERATING SYSTEM

Multics (Multiplex Information and Computing Service) is an operating system being developed from research by Massachusetts Institute of Technology, Bell Telephone Laboratories, and General Electric. It draws upon the design and operating experience gained with CTSS (Compatible Time-Sharing System) at the Massachusetts Institute of Technology Computation Center and Project MAC.

The four major elements of Multics are the supervisor, the command system, the file system, and the input/output system. The operational features and objectives of these major elements are briefly described below.

The supervisor determines the sequence in which various user programs and the Multics modules serving them are executed. Each user is allotted a fair share of available processor time by the supervisor. The supervisor contains the scheduler. Under normal circumstances the scheduler guarantees that interactive users may proceed at full speed and that batch processing jobs are completed before the specified deadlines. The supervisor also measures and records the amount of system resources expended by each user.

The command system examines the input from a user's terminal looking for a command and its arguments, such as "typeout mydata". When a command and its arguments are found, they are changed from user-oriented format into hardware-executable format. The module that executes the command is then called. The command system can be used by batch users by substituting a file containing commands for a terminal.

The file system frees the user from concern over the physical location of any of his information. He refers to his programs or data by name. The file system stores and catalogs them in its storage hierarchy. If the program or data is not in core memory when it is referenced, it is automatically retrieved and made available. The file system allows all users to retain as much information as they wish in machine-accessible secondary storage. In most cases the use of removable media such as cards and magnetic tape is unnecessary. It moves little-used information to devices with longer access time to allow ample space on faster devices for more frequently-used files. The file system provides information to a user when he requests it and protects it from accidental destruction. At the same time the file system allows files to be shared among authorized users.

The input/output system uses the GIOC to perform all reading and writing of information to peripheral and terminal devices. It provides a way for users to communicate with specific devices in cases where the device-independent input/output of the file system is not suitable. The input/output system performs the code conversion, queuing, and buffering needed by specific devices.

PART II.
GE-645 SYSTEM DESCRIPTION

4. HARDWARE SYSTEM CHARACTERISTICS

This chapter discusses several overall system features and implications so that the reader may better understand the total hardware system. The topics covered in this chapter include hardware system organization, various system configurations, and connecting input/output devices to the system.

HARDWARE SYSTEM ORGANIZATION

Considerable modularity is provided in the system for several reasons. The user may select various modules and peripherals in order to tailor his system to his particular work load. Modules and peripherals may be added to a system at any time without changing the users' programs or the software. Only configuration tables in Multics need to be changed. The user system may also grow in a different way. An existing module or peripheral may be replaced by a higher performance module or peripheral of the same type, yielding improved system performance. Considerable hardware redundancy also exists in a typical system so that if, for example, a processor module should malfunction and have to be removed from the system, a processor module is still left in the system so that the system can continue operation.

The system in Figure 1 has multiple paths to many of the modules, peripherals, and terminals. Multiple paths between modules increase system performance by permitting memory interlacing. (See Chapter 5, System Controller Module, for details.) Multiple paths to peripherals also increase system performance. For example, both GIOCs can perform input/output operations with a disc subsystem simultaneously. Another significant advantage of multiple paths becomes evident when one considers hardware malfunctions. Most components of the system have more than one path to them; therefore, the malfunctioning of one component does not isolate any multiple-path component.

Peripherals such as printers and card readers are ordinarily connected to only one GIOC because they are allocated to only one user at a time. An operator can, for example, move an operation from a printer on one GIOC to a printer on a different GIOC if the first printer or its GIOC should have a problem. If desired, a single input/output path peripheral can be connected to both GIOCs. This requires another hardware unit, a peripheral switch, which is discussed later in this chapter.

The paths between modules, peripherals, and terminals pass through "ports" in modules and "data channels" in GIOCs. Ports and data channels are terms for the hardware at the boundary of a module and are illustrated in Figure 2.

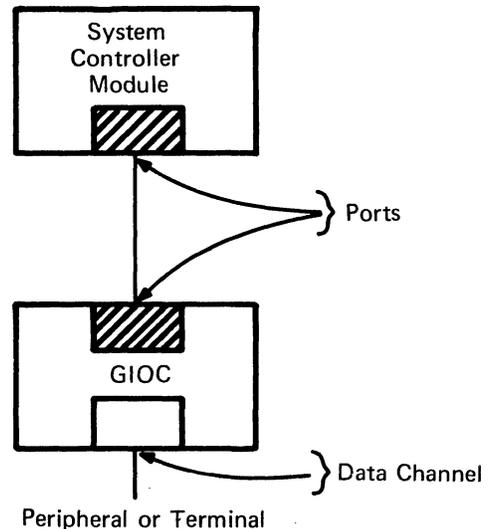


Figure 2. Distinction Between Ports and Data Channels

All types of modules – processor, GIOC, system controller and extended memory – can have up to eight ports. Thus, the maximum number of modules in a system is sixteen: eight system controller modules, and eight modules of some combination of processors, GIOCs, and extended memory units.

Multiple processors, GIOCs and extended memory modules (EMMs) allow simultaneous functions to be occurring, however a priority scheme must exist for access to a system controller. The ports on a system controller are lettered A through H and A has a higher priority than B, B higher than C, etc. The order of active module priority is established by the order in which the cables are connected to the system controller. For best system operation the priority order should be EMM, GIOC, processor. The extended memory module is highest because it has a high transfer rate and delay can result in transfer timing errors. The GIOCs are next for the same reason, except the transfer rates are lower. Under normal circumstances the processor does not experience transfer timing errors.

Control of the system involves the system configuration console and one or more teletypewriters. The system configuration console displays a summary of the configuration status of every module at the computer center. Portions of the display will be of interest to the operator. Another major function of the system configuration console is the provision of a way to initialize the system and start the bootloading operation from one central point.

Although a teletypewriter is not physically part of the system configuration console, it is a necessary part of the total system control. More than one teletypewriter may be used, with messages for certain units sent to a teletypewriter that is physically located near those units. Thus, printer messages would be typed near the printers and messages for overall control of the system would be typed near the system configuration console. The printed record of the conversations between the operator and the operating system can be quite valuable in monitoring system performance and in troubleshooting certain malfunctions. Each of the teletypewriters is connected to a GIOC through a standard communication line interface. Chapter 9 covers the system configuration console.

Processors, GIOCs and extended memory modules all form 24-bit absolute addresses. This permits use of over

16 million words of memory. Memories of this size are not available for initial GE-645 systems. The 16-million-word addressing capability exists to facilitate future growth of the system.

SYSTEM CONFIGURATIONS

Most of the discussions so far in this manual have centered around Figure 1. Although this figure represents a typical system, there are obviously many other possible system configurations. Some of these are shown in Table 1, where the middle column indicates the quantities that might be present in the typical system of Figure 1. Since the number of terminals that may be connected to the system is not directly reflected in the amount of hardware present in the computer center, what is shown in the table for terminals is the number of terminal-type input/output channels in the total system.

TABLE 1
SYSTEM CONFIGURATIONS

	Small	Quantities Typical Fig. 1	Large
System Configuration Console	1	1	2
Processor	1	2	4
GIOC	1	2	3
System Controller			
Number of modules	2	4	8
Total capacity (Words)	128k	256k	1024k
Extended Memory Unit			
Number of modules	1	1	1
Total capacity (Words)	4096k	4096k	4096k
Fixed disc (words)	33M	67M	134M
Magnetic card (words)	---	113M	226M
Magnetic tape handlers	4	16	32
Printers	2	4	6
Card Readers	1	2	3
Card Punches	1	2	2
Perforated Tape	---	1	2
Channels for teletypewriters	64	192	384
Channels for voice-grade communication lines for remote terminals such as DATANET*-760 GE-115	---	12	18

k = 1024
M = million

* Reg. Trademark of General Electric Company

It is quite important that the reader realize that the quantities shown in Table 1 illustrate only three of the many possible system configurations. The actual quantities to order for a particular installation depend strongly on the workload and applications to be run at that installation. One installation may need more mass storage and fewer terminal channels than the quantities shown, while another installation might need more printers and magnetic tapes and fewer terminal channels. A dedicated time sharing system might require more terminal channels and disc storage but fewer tapes, printers and card equipment. Also, the functional configuration of a particular installation can be changed from hour to hour from the system configuration console.

CONNECTING INPUT/OUTPUT DEVICES TO THE SYSTEM

There are several ways to connect both peripherals and terminals to a GE-645 system. These are briefly discussed to show the reader how peripherals and terminals may be connected to best suit his needs.

Figure 3 illustrates the connection of single-channel peripherals to a GIOC. Both single device (card reader) and multiple device (magnetic card) subsystems are shown.

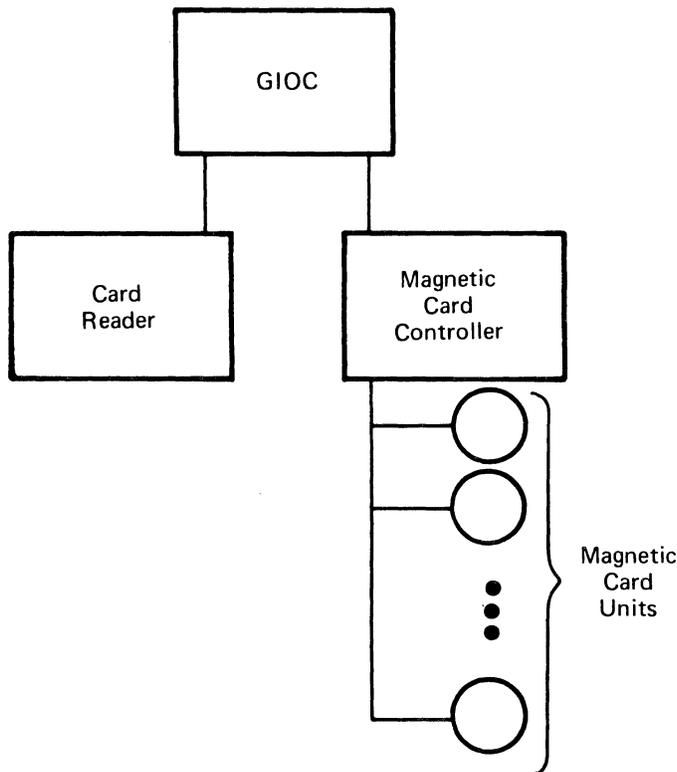


Figure 3. Single-Channel Peripherals Connected to a GIOC

System performance is improved by using more than one input/output channel for multiple-device subsystems. An example is the dual channel magnetic tape subsystem shown in Figure 4. Two independent operations may take place simultaneously, and either GIOC may use any tape handler.

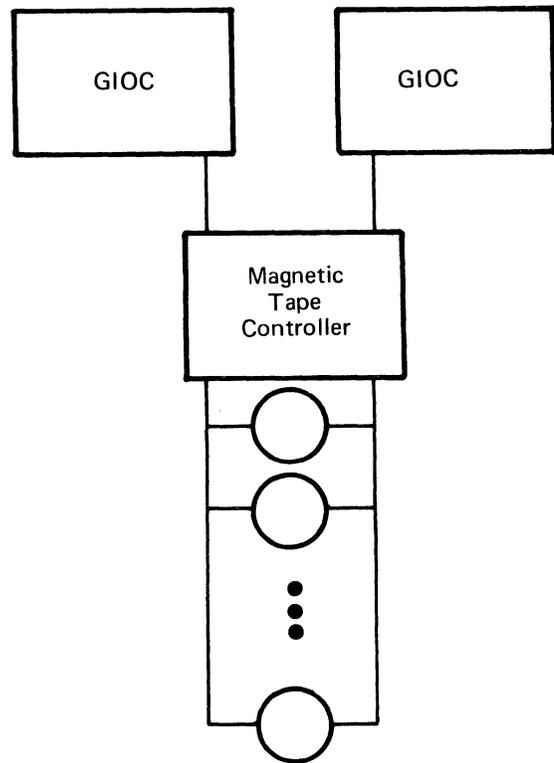


Figure 4. Multiple-Channel Peripheral Connected to two GIOC's

Additional flexibility is available by using a peripheral switch. Details on peripheral switches are given in Chapter 10. Briefly, a peripheral switch provides a rapid way to disconnect a peripheral from one GIOC and connect it to another GIOC and vice-versa. Two examples of this are given in Figures 5 and 6. With these basic elements more complex switching configuration (i.e. cross bar) can be produced to meet specific site requirements.

Terminals may also be connected to the system in several ways. In general, the user at a terminal will dial into the system in order to become connected. There are many possible paths through the switching network. Depending on the path taken, the user may be connected to either GIOC. And since many of the paths go to the same GIOC, his connection may be to any one of several channels within the GIOC. In order to dial in, the user must have a data set in his terminal. The general data set usage is illustrated in Figure 7.

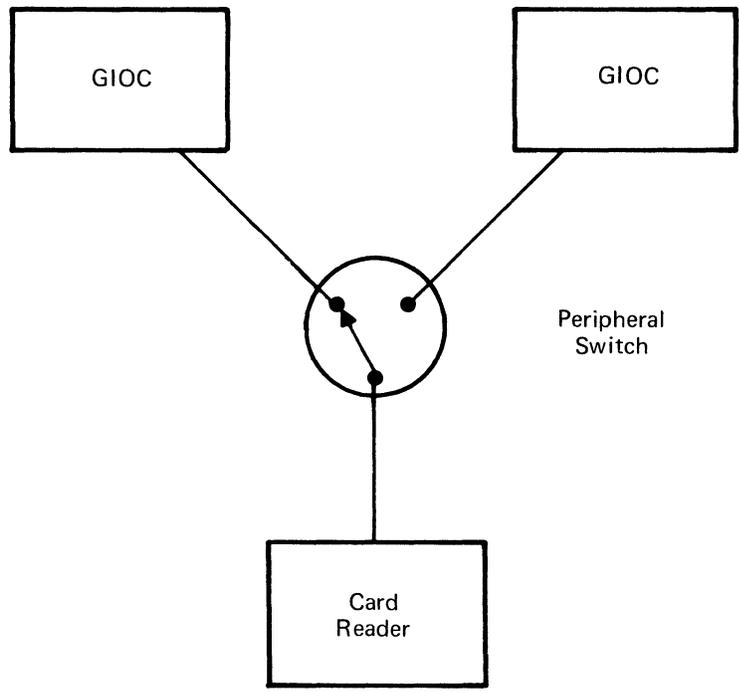


Figure 5. A Peripheral Switch and Two GIOC's

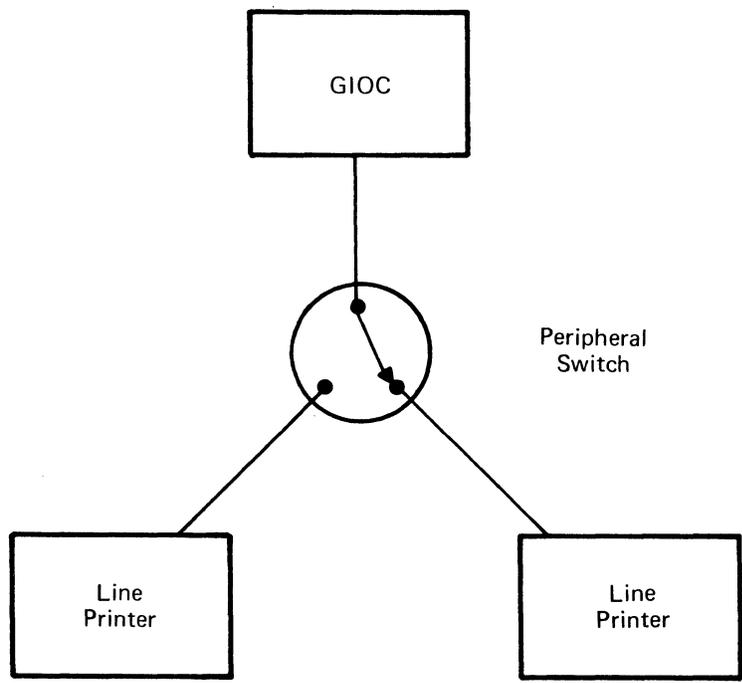


Figure 6. A Peripheral Switch and Two Peripherals

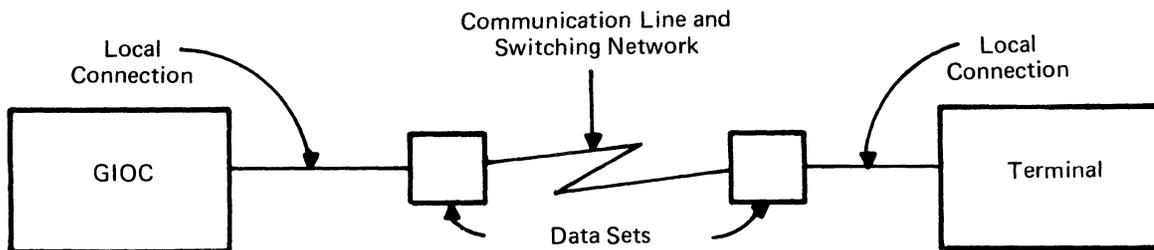


Figure 7. General Data Set Usage

A distinction must be made between the number of terminals which have the capability of being connected to the system and the maximum number of terminals which may be connected at any given time. The former may be far larger than the latter. As an example, suppose that a system is capable of connecting and operating 200 terminals simultaneously. There could be 2,000 terminals which, at one time or another, are able to be connected to the system. However, as long as not more

than 10 percent of these users wish to be connected to the system at any given moment, there is no problem or conflict.

Normally a terminal is connected to a GIOC through a common carrier switching network as has been discussed. In certain situations, the terminal may bypass the switching network and utilize a private line (see Figure 8).

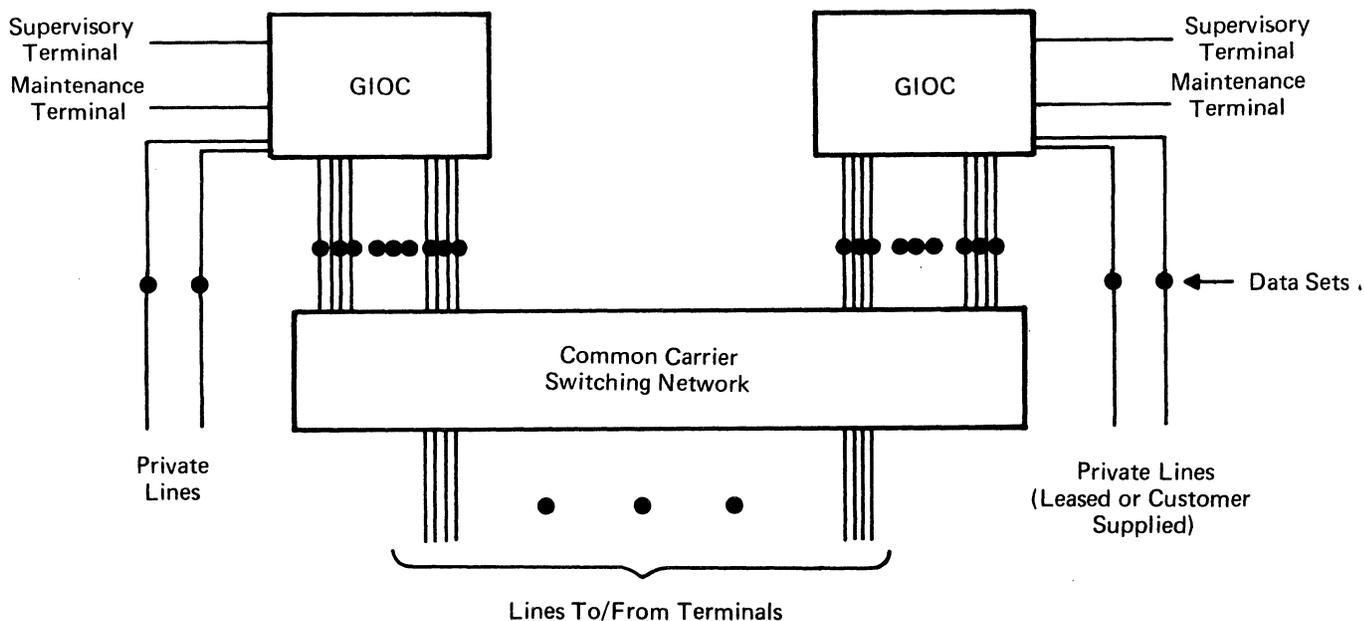


Figure 8. Bypassing the Switching Network



5. SYSTEM CONTROLLER MODULE

The system controller modules serve as the center for communications between the other modules in the GE-645 system by providing the following functions:

- Core memory for storage of instructions, control words, and data.
- Central point for forwarding control signals from one active module to another.
- System clock for providing data and time-of-day information, together with the ability to interrupt a processor module at a predetermined time.

Each system controller has up to eight ports for connection to processors, GIOCs and extended memory modules. One of these ports is designated as the control port, and the processor connected to that port is called the control processor. Thus each system controller has a control processor.

STORAGE FUNCTION

A system controller may contain 32k, 64k, or 128k of 36-bit words (plus parity), and has a cycle time of one microsecond. Either one or two words can be read or written in one memory access. It is also possible to store one 6-bit or 9-bit character in a word without disturbing the other characters in the same word.

In order to store or retrieve information, a processor, GIOC, or extended memory module sends a command, and address, and the necessary data to the appropriate system controller. The system controller executes the command and either stores the received data or sends the desired data to the requesting module. The requesting module performs the conversion from relative to absolute addresses, when this conversion is necessary, before the address is sent to the system controller.

Five commands are used by the processors, GIOCs, and extended memory controllers for data transfer:

Read Restore, Single precision (not used by extended controllers)

Read Restore, Double precision

Clear Write, Single precision

Clear Write, Double precision

Read Alter Rewrite (used only by processor)

The Read Alter Rewrite command makes it possible for a processor to read and alter the contents of a memory

location during a single memory cycle, so that no other active unit can gain access to that location while it is being altered. This is a very useful characteristic of the GE-645 in a multi-processor environment.

A typical GE-645 system has more than one system controller. Memory access requests are distributed among the various system controllers by a memory interlace technique. This equalizes the load among the system controllers and increases system performance by decreasing the competition and queuing of requests for the same physical system controller. Although the circuitry that controls interlacing is located in the processors, GIOCs, and extended memory modules, interlacing is discussed in this chapter because of its close connection with the storage function.

There can be two-way interlace, four-way interlace, or no interlace. The choice is a basic parameter of the system configuration, normally established at the system configuration console. Memory addressing by the hardware is actually done by pairs of words, so it is more correct to speak of pairs of memory addresses. Therefore, with four system controllers, A, B, C, and D, and no interlacing, sequential pairs of addresses fall in succession into system controller A until it has been filled, then start filling system controller B, etc. Two-way interlace has sequential pairs of addresses located so that the first pair is in system controller A, the second pair in B, and the third pair in A, etc. With four-way interlace, the location of sequential pairs of addresses rotates among the four system controllers — A, B, C, D, A, B, C, D, etc. The effect of the types of interlace is summarized in Table 2, with system controller A arbitrarily selected as the initial system controller.

CONTROL SIGNALS

CONNECT INSTRUCTIONS

When a processor executes a Connect instruction, the system controller which contains the Connect Operand Word immediately forwards the Connect Operand Word and a Connect signal to the module specified in Connect Operand Word. The recipient module may be a GIOC, an extended memory unit, or another processor.

INTERRUPT CELLS

A group of 32 interrupt cells are located in each system controller. These interrupt cells are used by the GE-645 hardware modules to notify Multics that some event has occurred which needs attention. An interrupt cell may be set by a GIOC, an extended memory unit, a processor, or by the system clock. When an interrupt cell is set, the system controller notifies its control processor of

TABLE 2
EXAMPLE OF TYPES OF INTERLACE

Word Pair	Type of Interlace					
	None		Two-Way		Four-Way	
START+ 0, 1	System Controller	A	System Controller	A	System Controller	A
START+ 2, 3		A		B		B
START+ 4, 5		A		A		C
START+ 6, 7		A		B		D
START+ 8, 9		A		A		A
START+ 10, 11		A		B		B
START+ 12, 13		A		A		C
START+ 14, 15		A		B		D

this fact, along with the cell number, so that the source of the interrupt can be identified. The control processor stops what it is doing and jumps to a software routine designed for handling that particular interrupt cell. Multics can temporarily inhibit the interrupting action of any interrupt cell or cells with the aid of a 32-bit interrupt mask register that is also located in each system controller, or by making use of the interrupt-inhibit bit in each instruction.

SYSTEM CLOCK

The system clock consists of a calendar clock and an alarm register. The calendar clock is a 52-bit binary counter which counts at one microsecond intervals, providing a capacity greater than 142 years without overflowing. The program can read the contents of the calendar clock with a precision of one microsecond. The clock accuracy is better than 5 parts per million (3 seconds/week) and can be improved by using an external stable reference.

The 142-year capacity of the calendar clock makes it possible for the software to operate on Universal Time. To accomplish this the General Electric field engineer must set the calendar clock to the number of microseconds since midnight, January 1, 1901, Greenwich Mean Time. Although a processor can read the calendar clock at any time, provision for setting it under program control is deliberately omitted, so that programming errors or hardware malfunctions outside the clock cannot destroy the current time.

The alarm register has the same 142-year capacity and can be set by the software. The value in the alarm register has a resolution of 64 microseconds and is continuously compared to the changing time in the calendar clock by special hardware in the system clock. When the calendar clock reaches the time preset in the alarm register, an interrupt cell is set. This causes a program interrupt and notifies Multics that the desired instant of time has arrived.

6. PROCESSOR MODULE

Most of the advanced capabilities of the GE-645 which are seen by its users are provided by the processor. The processor module has full program execution capability and conducts all actual computational processing within the GE-645 system. The processor performs instruction fetching, address preparation, memory protection, data fetching and data storing. These functions are overlapped to provide the highest rate of instruction execution.

The GE-645 Processor (CP8031) contains all the general features of the GE-635 Processor except for Base Address Register and the two associated instructions (See Appendix A) plus the following additional features:

- hardware for handling segments and pages.
- hardware to generate 24-bit memory addresses.
- an associative memory to speed up address generation for segments and pages.
- ten program addressable registers used in segment and page address preparation.
- more extensive address modification capability.
- instructions to handle the segmentation and paging hardware and the system clock.
- several levels of memory access permission.
- several new processor faults.
- a modified Timer Register to also count memory accesses.
- a maximum of eight ports for connection to System Controller modules.
- logic for remote configuration from system configuration console.
- storage for up to four instructions within the processor in various stages of address preparation and execution.

MODES OF OPERATION

The processor has three modes of operation: absolute, master and slave. All instructions are available in absolute and master modes. Most, but not all, of the instructions are available in slave mode. General users are restricted to the slave mode and hence are prevented from executing any instructions that will damage other programs or Multics. Privileged instructions such as those which operate upon the descriptor base register, the system clocks, and input/output devices are available only in the absolute and master modes.

The full segmentation and paging capability of the processor is used in the master and slave modes for fetching instructions and operands. The addressing in the absolute mode does not use any of the segmentation and paging capability except for a limited case of operand references.

REGISTER DESCRIPTIONS

The program-accessible processor registers are:

<u>Quantity</u>	<u>Name</u>	<u>Length</u>
1	Accumulator Register	36 bits
1	Quotient Register	36 bits
1	Exponent Register	8 bits
8	Index Registers	18 bits each
1	Timer Register	24 bits
1	Instruction Counter	18 bits
1	Indicator Register	18 bits
1	Descriptor Base Register	29 bits
1	Procedure Base Register	18 bits
8	Address Base Registers	24 bits each

The Accumulator (A) Register is used:

- As an operand register for all classes of single-precision (36 bit) operations. For floating-point operations, the fixed point part is in the A register and the exponent is in the Exponent Register.
- For address modification. Each half of the A register can hold an index value. The left half is called A Upper (AU) and the right half A Lower (AL).

The Quotient (Q) Register is used in the same ways as the A register, with the halves called QU and QL.

The AQ Register, which is the combined A and Q registers, is used as the operand register for all double-precision (72 bit) operations. For floating-point operations, the fixed point part is in the A or AQ register and the exponent is in the exponent register.

The Exponent Register holds the exponent in floating-point operations with the fixed point part in either the A or the AQ register.

The Index Registers are used during address modification. These registers can also be used as half-precision fixed point operand registers.

The Timer Register (TR) counts a 64 kHz clock (15.625 microsecond interval) as in the GE-635, or can be switched to count the number of memory access cycles. Multics uses the timer register to count memory accesses for a measurement of processor utilization. The register can be set to an initial value which, when counted down to zero, causes a processor fault in the slave mode. This register should not be confused with the system clock in the System Controller Module.

The Instruction Counter (IC) partially specifies the next instruction to be executed. Details will be discussed in the segmentation portion of this chapter.

The Indicator Register contains a group of indicators that specify or hold certain processor states.

The various indicators are:

- Zero
- Negative
- Carry
- Overflow
- Exponent Overflow
- Exponent Underflow
- Overflow Mask (can inhibit overflow recognition)

Tally Runout

Parity Error

Parity Mask (can inhibit parity error recognition)

Absolute Mode (use only IC for instruction fetching)

The Descriptor Base Register (DBR), Procedure Base Register (PBR), and Address Base Registers (ABRn) are used in performing relative addressing. Their description is postponed until segmentation is discussed later in this chapter.

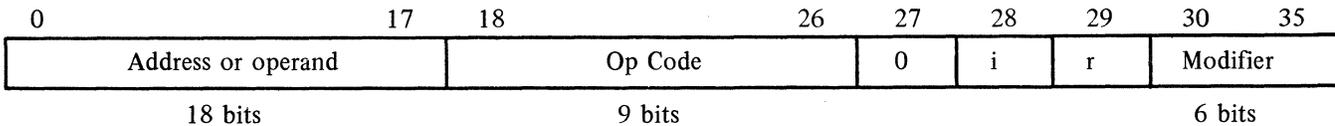
INSTRUCTION REPERTOIRE

For the GE-645, there are a possible 512 instructions of which 404 have been assigned. This leaves 108 operation codes for upward growth. A list of the assigned instructions is included in Appendix A.

The GE-645 instructions are listed and described according to operations for: data movement, arithmetic, logical, comparison, control, shifting, and special operations. Many of the GE-645 instructions are familiar to experienced programmers of large scale computers. In addition to the familiar ones, there are other instructions to facilitate: segmentation and paging, saving and restoring of registers, character handling, decision making, and list processing.

INSTRUCTION FORMAT

Most of the instructions have the following format. A few special instructions have a format adapted to their unique functions.



The address or operand field specifies information used in addressing an operand or contains the operand.

The operation code field specifies the operation to be performed and the registers to be used.

Bit 27 is not used and must be zero for upward compatibility with future systems.

Bit 28 is used under certain conditions to inhibit a program interrupt.

Bit 29 specifies how the address field is to be interpreted. See Base Registers, Page 22.

The modifier field specifies the kind of address modification to be used. Possibilities include: (1) no modification, (2) which register to use, (3) indirect or not, and

(4) several special-purpose modification methods that significantly increase the power of the instructions.

DATA MOVEMENT OPERATION

These instructions move data between the system's core memory and the processor registers, indicators, and associative memory. They include half, single, and double-precision operations. Also 6 and 9 bit characters can be handled.

ARITHMETIC OPERATIONS

These instructions include fixed and floating point; half, single, and double-precision; and placement of results in registers or memory.

LOGICAL OPERATIONS

These instructions include single and double-precision, and placement of results in registers or memory.

COMPARISON OPERATIONS

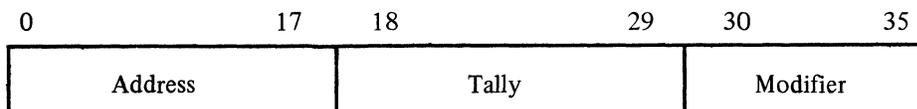
These instructions include logical, fixed, and floating-point comparisons on half, single, and double-precision operands. The results of a comparison are saved in certain indicators for later testing by control instructions.

CONTROL OPERATIONS

These instructions include normal transfers, conditional transfers, and some special instructions which simultaneously transfer control and restore various registers.

SHIFTING OPERATIONS

These operations include shift and rotate instructions using the A, Q, and AQ registers.



There are four classes of address modification: (1) register, (2) register then indirect, (3) indirect then register, and (4) indirect then tally. These are described briefly below. Modification registers are the 8 index registers, plus the AU, AL, QU, QL, and IC registers. Direct operand modifiers may be used to treat the address directly as the operand.

REGISTER MODIFICATION (R)

An effective address is produced by adding the contents of a modification register to the contents of the instruction's address field.

REGISTER THEN INDIRECT (RI)

The indicated register modification of the address is performed (as with R) to obtain an indirect word, then the modification specified in the indirect word is conducted. Address modification in an indirect word is specified in the same manner as in an instruction word.

INDIRECT THEN REGISTER (IR)

The indirect word is obtained using the original address, then the modification specified in the indirect word is conducted. Upon completion of the indirect addressing string the indicated register modification in the last IR word (instruction or indirect word) encountered is performed.

SPECIAL OPERATIONS

An important instruction in this class is the Connect, which initiates all input/output operations and provides a method of communicating between processors. Another group of instructions enables one or two instructions to be repeated without having to continually fetch the instruction(s) from memory. Other instructions provide a way to execute one or two instructions which are not in the sequential string of instructions being executed, provide entries to the operating system, convert a binary number to its binary-coded-decimal equivalent, and convert a Gray Code word to a binary equivalent.

ADDRESS MODIFICATION

A comprehensive set of address modification features includes both modification of addresses by the contents of registers and multi-level indirect addressing. The general format of the indirect words is:

INDIRECT THEN TALLY (IT)

The indirect word is obtained using the original address. Then one of the 15 possible sequences below is performed.

Mnemonic

- I Indirect – The address field of the indirect word is used for the operand address without further change. Tally and modifier fields are unused.
- ID Increment Address, Decrement Tally – The address field of the indirect word is incremented by 1 and the tally is decremented by 1 after the address field is used for the operand address. The modifier field is unused.
- AD Add Delta – This is the same as ID except delta is added to the address. Delta is the value in the modifier field of the indirect word.
- IDC Increment Address, Decrement Tally and Continue – This is the same as ID except the address field in the indirect word specified another indirect word and the modifier field is interpreted in the normal manner.
- DI Decrement Address, Increment Tally – The address field of the indirect word is decremented by 1 and the tally is incremented by 1 before the address field is used for the operand address. The modifier field is unused.

- SD** Subtract Delta — This is the same as DI except delta is subtracted from the address. Delta is the value in the modifier field of the indirect word.
- DIC** Decrement Address, Increment Tally and Continue — This is the same as DI except the address field in the indirect word specifies that of another indirect word and the modifier field is interpreted in the normal manner.
- CI** Character from Indirect — The indirect word specifies an address and a character position which are used for an operand character address without change. The character size (6 or 9 bits) and position is specified by the modifier field. The tally field is unused.
- SC** Sequence Character — The character position (in modifier field) is incremented by 1 and the tally is decremented by 1 after the indirect word address field and character position are used for the operand character address. When the character position tries to exceed the maximum value (3 or 5) it is reset to 0, and the address field is incremented by 1. The character size is also specified in the modifier field.
- SCR** Sequence Character Reversed — This is the reverse of SC. The character position is decremented by 1 and the tally is incremented by 1 before the indirect word address field and character position are used for the operand character address. When the character position tries to go negative it is set to the maximum value (3 or 5), and the address field is decremented by 1. The character size is also specified in the modifier field.
- F** Fault 1 — This modifier causes Processor Fault Tag 1. (See Interrupt and Fault section of this chapter.)
- F2** Fault 2 — This modifier causes Processor Fault Tag 2.
- F3** Fault 3 — This modifier causes Processor Fault Tag 3.
- ITS** Indirect to Segment — This will be discussed in the Segmentation section of this chapter.
- ITB** Indirect to Base — This will be discussed in the Segmentation section of this chapter.

SEGMENTATION

Segmentation has been used in some form or other throughout the fields of computation and information processing for several years. In the GE-645, this familiar

concept takes on added power and increased significance. In the past, program segments have generally consisted of subprograms and/or storage areas, and were combined into a single program by a loader prior to execution. This method of operation provided several important advantages over compiling or assembling entire programs prior to their execution. These advantages are preserved in the GE-645 segmentation, and other capabilities are added. Some of the added capabilities are:

- The ability for the same copy of a segment to be simultaneously shared by many users without making special prearrangements.
- The ability to allow individual segments to vary dynamically in size without influencing the addressing of other segments.
- The ability to address programs larger than available core memory.
- The ability for each segment to possess unique attributes to prevent misuse.

Significant among the advantages of segmentation is the facility it supplies for use of pure procedures. These are procedures which are not altered during their execution. As used in the GE-645, a segment can be identified as pure procedure and invoked by any user to perform its function on his own data. This provides two economies in the multi-user GE-645. First, only one copy of a pure procedure need exist to be shared by all its users; second, pure procedures constitute an important class of information that need never be rewritten to secondary storage.

It is often desirable to write a program that can operate on variable size data arrays. In previous computer systems it has been necessary to use complex overlaying techniques or to place an upper bound on the size of arrays that could be handled by such programs. In the GE-645, variable arrays can be assigned to separate segments so that each can vary in size from one up to 218 words without any special provision being made by the computer users.

A GE-645 program consists of a variety of segments, each of which has a unique name. These names are converted to a segment number of Multics at execution time. The user may view each of his segments as if it were stored in an independent core memory. Each segment has its own origin which can be addressed as location zero. The size of each segment may vary without affecting the addressing in other segments.

The words contained within a segment are addressed by an 18-bit segment address. If a program consists of several segments, the GE-645 refers to any word

in any segment according to a segment number and a segment address. Figure 9 represents schematically the GE-645 memory topology. The individual segments of a program are shown as consisting of independent variable

size storage arrays. In analogy to a rectangular coordinate system, it suggests that the location of any word may be specified by two coordinates: a segment number and a segment address.

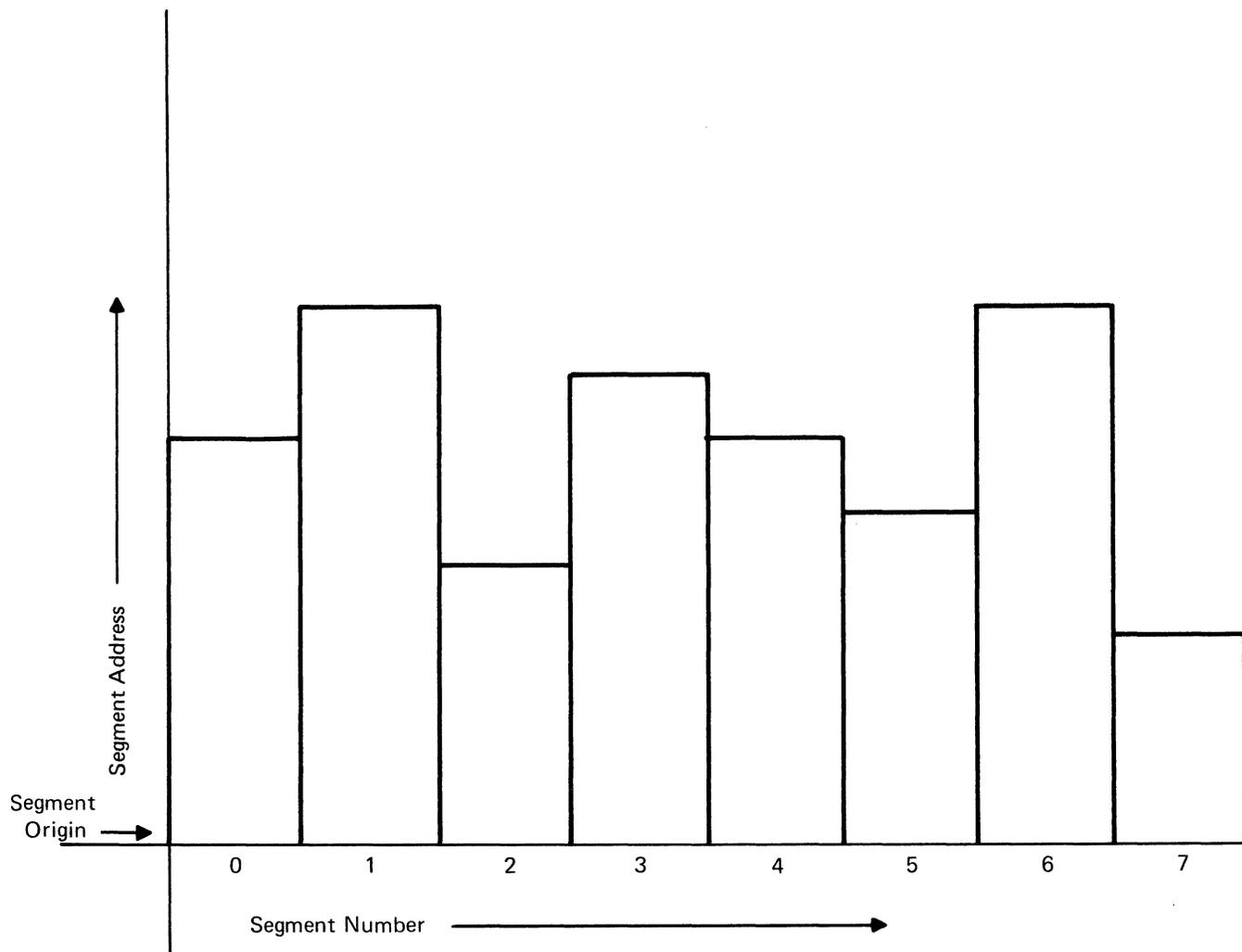


Figure 9. Two-Coordinate Addressing in the GE-645

DESCRIPTOR SEGMENT

A descriptor segment is the processor's means of relating program references to absolute memory locations. Multics provides a descriptor segment for each program. The descriptor segment contains one word called a segment descriptor word (SDW) for each segment being used by the program. Each SDW in the descriptor segment indicates the absolute location of the origin, and gives size and control information about the segment to which it corresponds (See Figure 10).

The segment number used in locating a segment refers to the location of SDW in the descriptor segment. This SDW contains the segment origin. When a reference is made using a two-coordinate address, the processor retrieves the segment origin from the descriptor segment and adds the segment address to it. The result is the absolute address of the desired word. This is illustrated in Figure 10 and in numerous examples which follow. The segment address is compared to the segment size to verify that the reference is within the specified allowable address range of the segment.

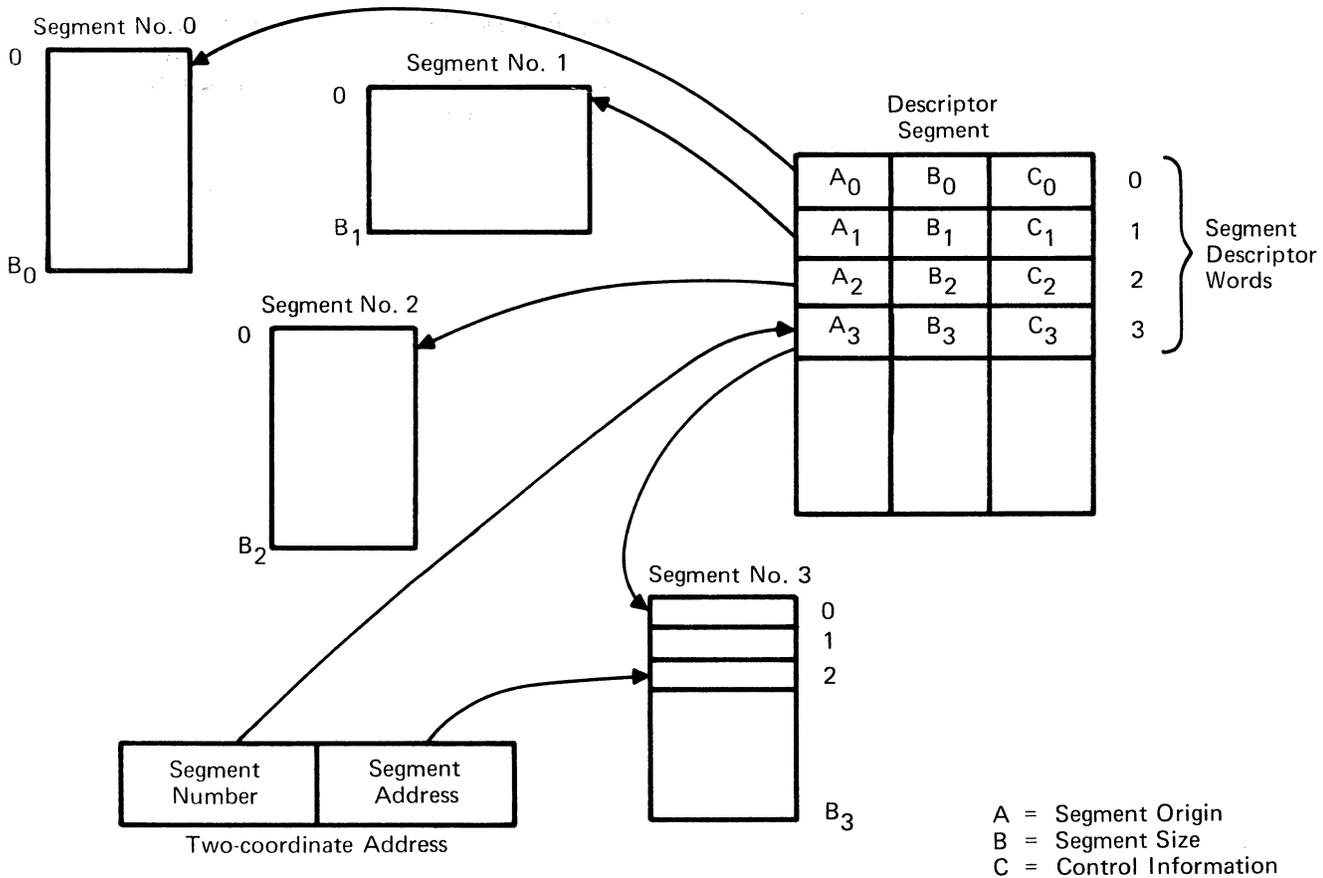


Figure 10. The Descriptor Segment Facilitates Two-Coordinate Addressing

Since the descriptor segment can be up to 2^{18} words long, the two-coordinate addressing of the GE-645 allows a program to reference up to 2^{18} segments.

BASE REGISTERS

The location of the descriptor segment for the process in execution is always known to the processor. Its origin and size are stored in a special high-speed register called the descriptor base register (DBR). (See Figure 11.) When control of the processor is switched from one user to another, the DBR is reloaded to point to the descriptor segment belonging to the new user. In this way all of the relative addressing information for all of the segments accessible to the program in execution are changed by simply changing the contents of one register.

The DBR, like the descriptor segment, is accessible only to Multics. This means it cannot be manipulated in order

to gain unauthorized access to segments of another program. Also, storage belonging to another program cannot be accidentally damaged.

A user may utilize segments in various ways depending upon his objectives. A simple way of using segments is to place instructions in one class of segments called procedure segments, and variable information in another class of segments called data segments. The GE-645 provides convenient methods for performing necessary references between such segments.

There are eight address base registers (ABR's) in each processor. They may be used to specify the segment number portion of a two-coordinate address as in Figure 12. If bit 29 of an instruction is set to 1, this signals the processor to interpret the three high-order bits of the address field as an address base register number. The ABR will contain the segment number to be addressed, and bits three through seventeen of the instruction (y in the figure) will contain the segment address.

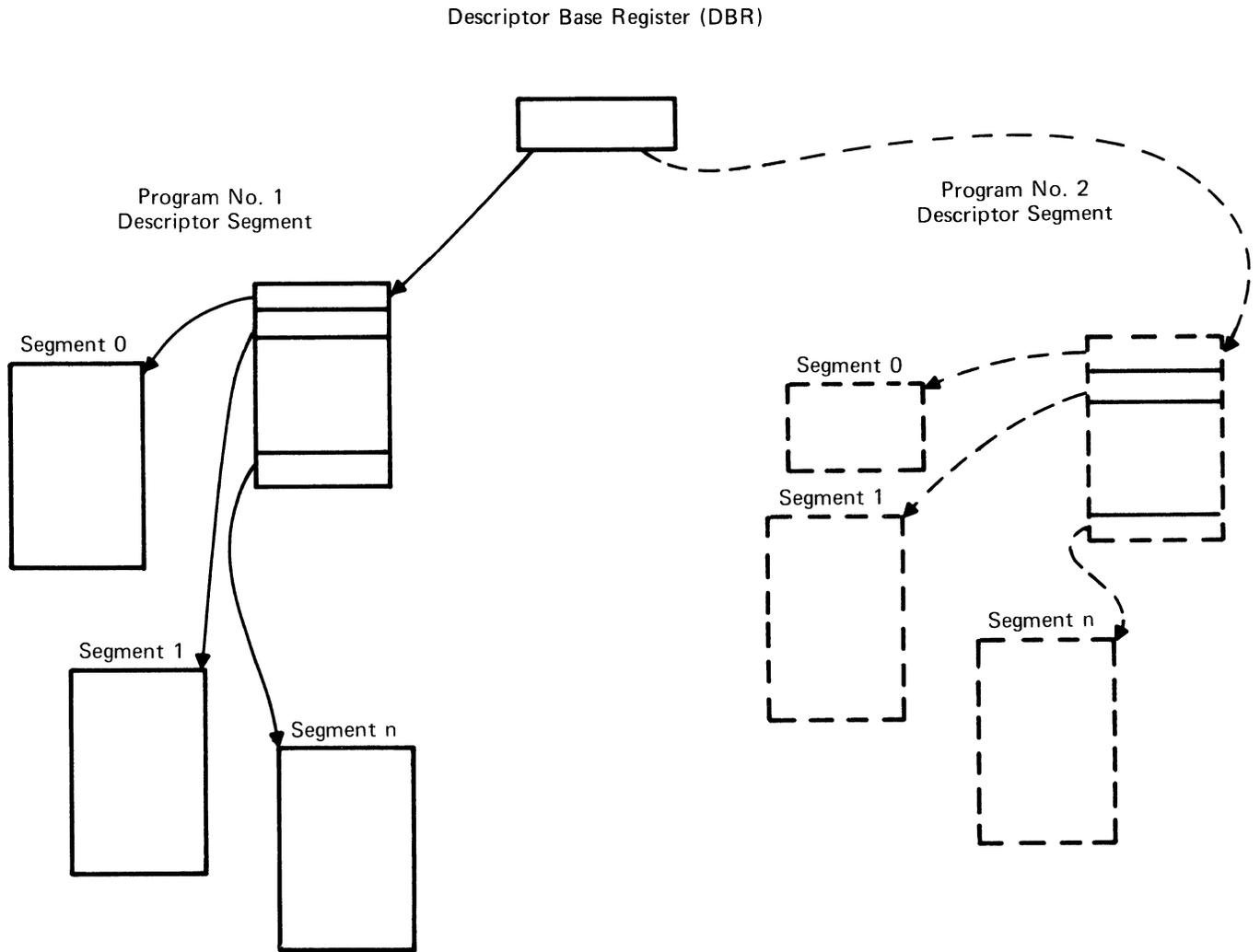


Figure 11. The DBR Determines which Segments the Processor May Access

The procedure base register (PBR) contains the segment number of the segment from which the processor is to retrieve instructions. The PBR functions exactly like an ABR except the PBR is never referenced explicitly in a user program. The processor automatically refers to the PBR in the execution of every instruction. The instruction counter (IC) contains the segment address and the PBR contains the segment number of the two-coordinate address of the next instruction. See Figure 13.

The PBR is automatically changed to contain the segment number of the target segment when a transfer between segments is performed. Figure 14 illustrates this process. In this case ABR2 contains the segment number which is placed in the PBR as part of the intersegment transfer operation.

INDIRECT TO PAIR MODIFIERS.

An additional method of addressing between segments is provided in the GE-645. This is the indirect to pair (IP) address modifier. The IP modifier has two variations: indirect to segment modification (ITS), and indirect to base modification (ITB).

If an instruction refers indirectly (IR or RI) to a word located in an even addressed location in core memory and if the indirect word contains the bit configuration corresponding to the ITS modifier in its modifier field (bits 30-35), then the following occurs (See Figure 15).

1. Bits 0 through 17 of the indirect word are interpreted as a segment number.

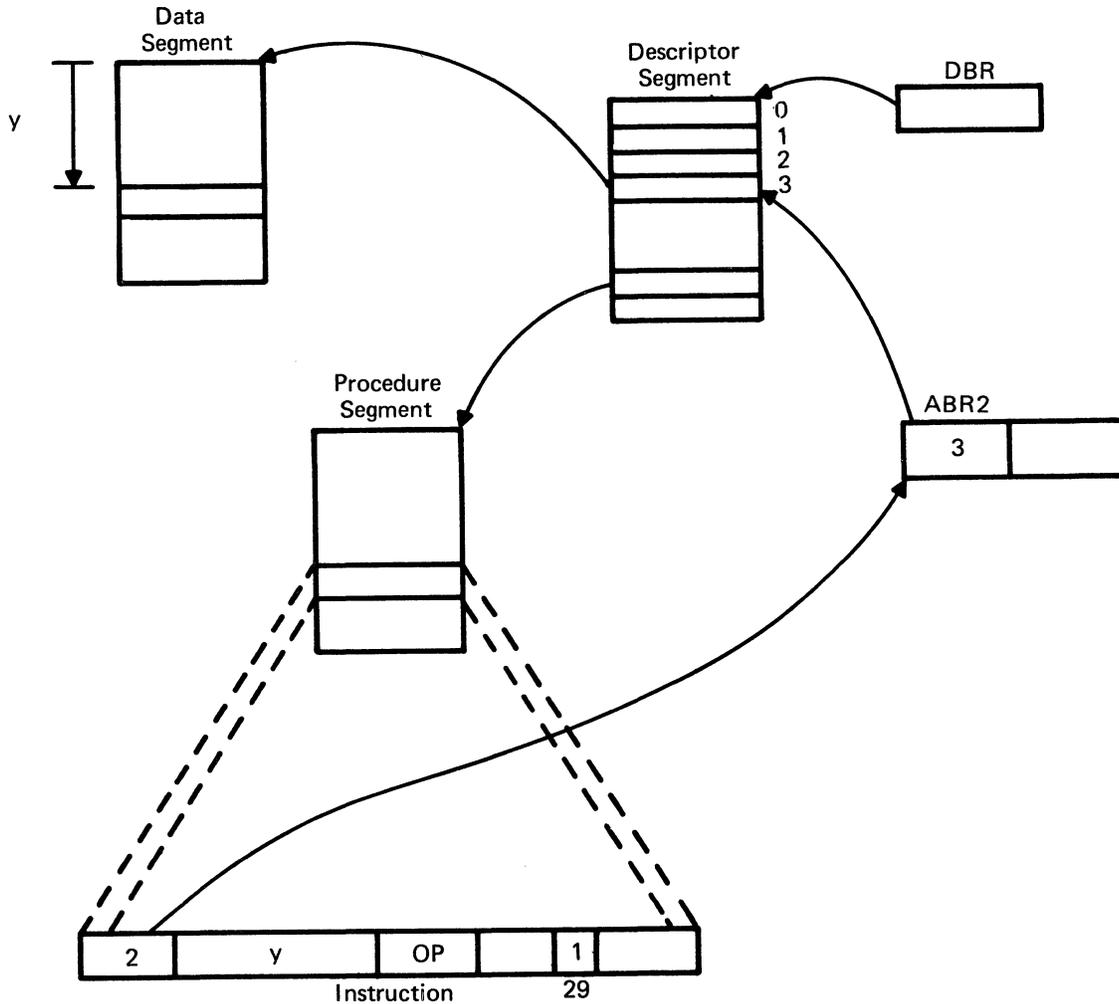


Figure 12. An ABR is Used in Specifying a Two-Coordinate Address to Refer to Another Segment

2. Bits 0 through 17 of the word following the indirect word are interpreted as a segment address.
3. Bits 30 through 35 of the word following the indirect word are interpreted as a modifier to be applied to the segment address. Figure 15 illustrates the use of an ITS pair in performing a reference between segments.

ITB modification is similar to ITS modification in its use of indirect words. However, the interpretation of the even-odd pair is as follows (See Figure 16):

1. Bits 0 through 2 of the indirect word are interpreted as an ABR number.

2. The ABR contains a segment number to be used in relative addressing.
3. Bits 0 through 17 of the word following the indirect word are interpreted as a segment address.
4. Bits 30 through 35 of the odd word following the indirect word are interpreted as a modifier to be applied to the segment address.

BASE PAIRS.

Address Base Registers (ABR's) may be used in pairs to provide a local origin within a segment. This is useful in various applications as an additional level of index-type

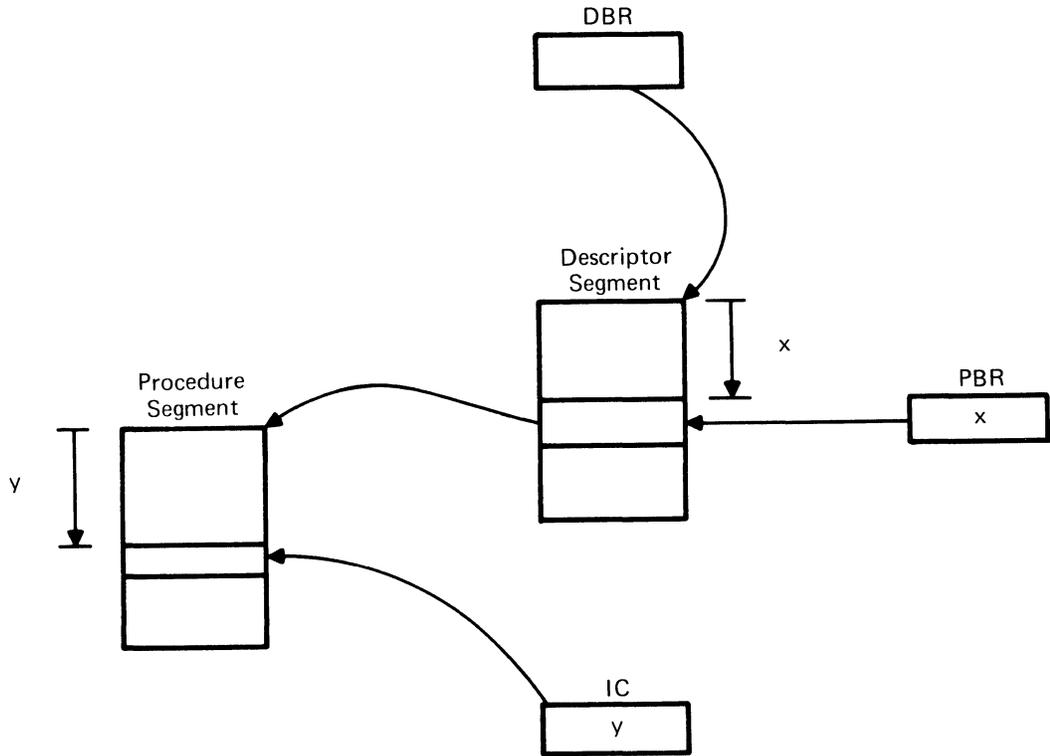


Figure 13. PBR Designates the Segment from which Instructions are Fetched

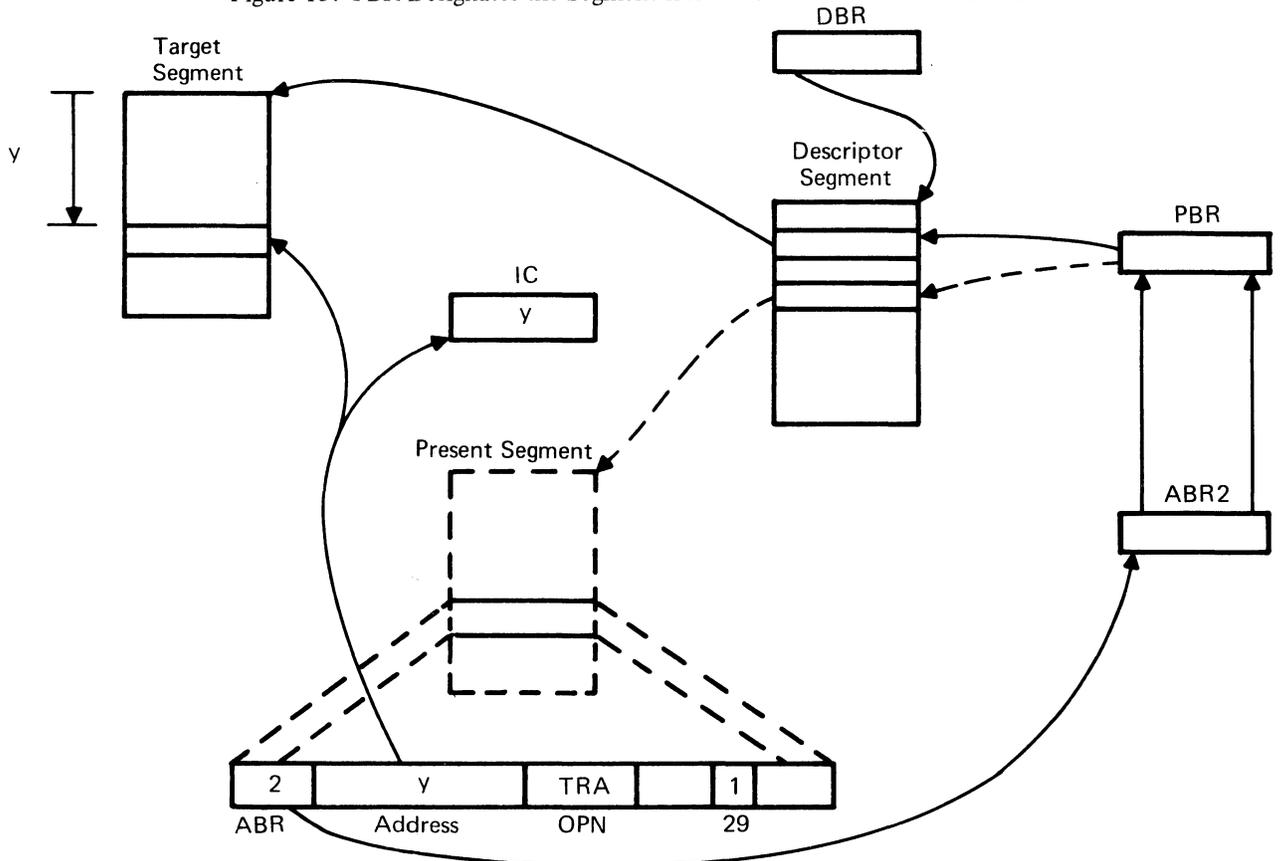


Figure 14. A Transfer Instruction Causes the Procedure Base Register to Be Changed

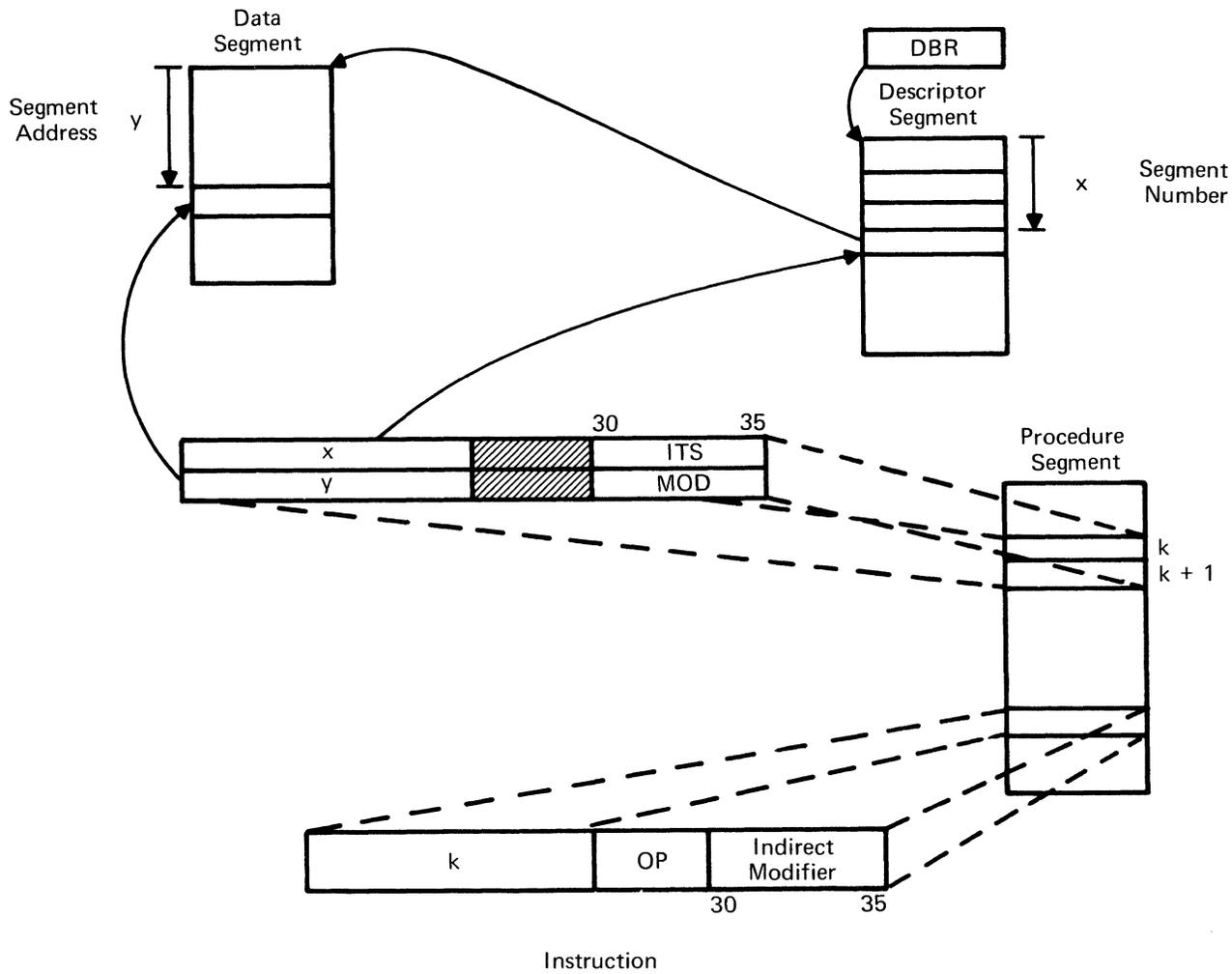


Figure 15. An ITS Pair is Used to Refer to Another Segment

modification. It is particularly useful in the maintenance of pushdown-popup stacks.

Figure 17 illustrates the use of a base pair. The base number specified in bits 0 through 2 of the instruction point to a base register which is designated as "internal" in its control field. The contents of this base register will be added to the address of the instruction providing a displacement from the segment origin. Three additional bits in the control field of the internal base register designate a second ABR containing the segment number of the target segment. This second ABR is designated "external" in its control field.

SHARED SEGMENTS:

Two or more programs may make common use of certain segments. For example, several users may share common data bases and pure procedures, such as compilers and

utility routines. With the demonstration of appropriate authorization, Multics allows a user to access a common segment by placing a corresponding segment descriptor word (SDW) in his descriptor segment. Figure 18 shows the descriptor segments of three programs sharing a segment named XYZ. After Multics has entered the SDW's in the three descriptor segments, the three programs all refer to the common segment.

Figure 19 illustrates one use of shared segments. Here a single pure procedure operates on a separate data segment for every program in a uniprocessor system. Figure 20 illustrates the use of the same pure procedure segment by two processors in a dual processor configuration. In this case both processors could be simultaneously executing the same instructions on behalf of two different programs. And in general, any number of processors could be making simultaneous use of a pure procedure.

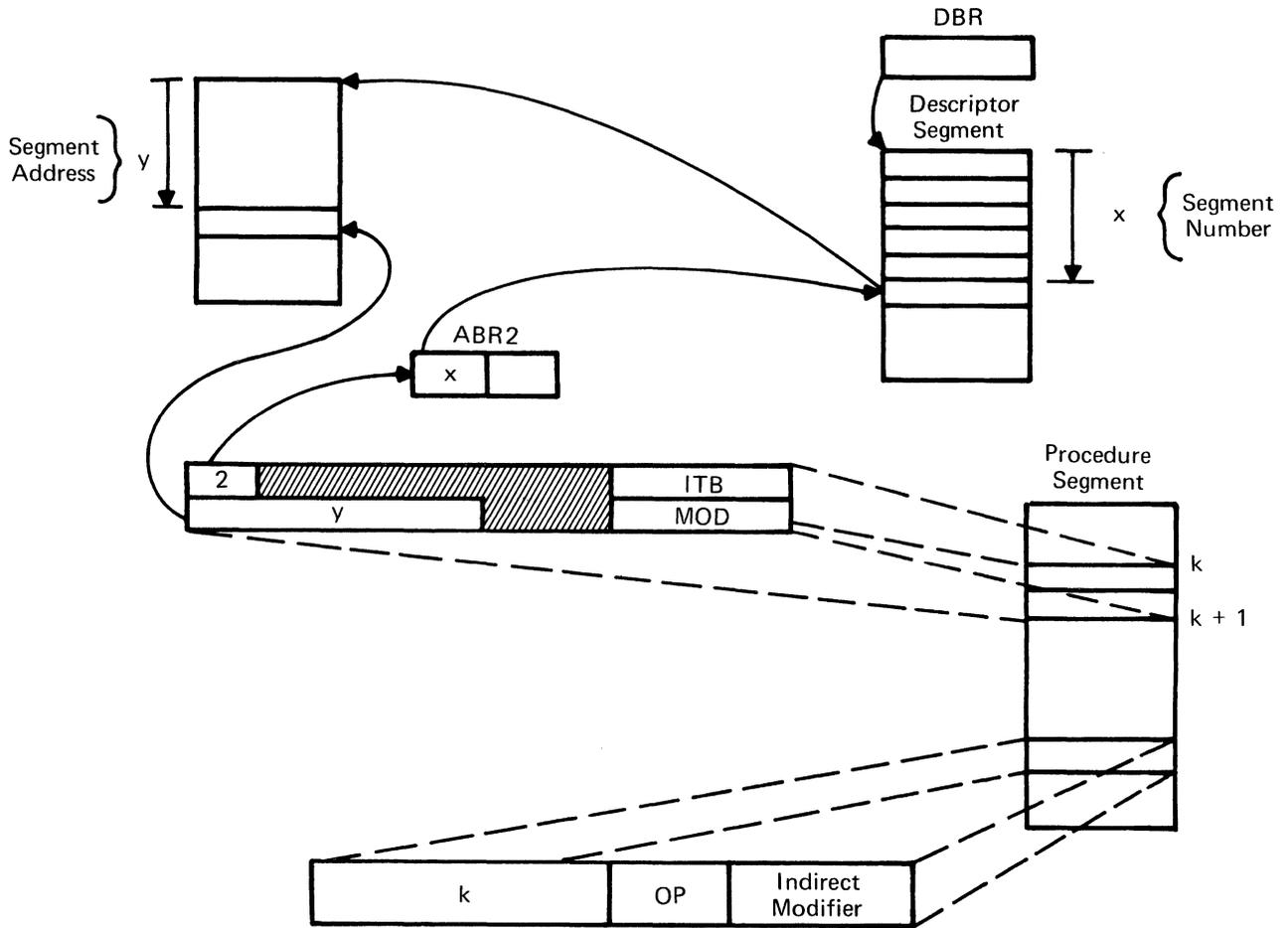


Figure 16. An ITB Pair is Used to Refer to Another Segment

PAGING

GE-645 users consider their programs to consist of named segments without regard for their absolute locations in core memory. In fact, only a small portion of most segments must be in core memory at one time. Instead, almost all of a segment can be stored in a relatively inexpensive secondary storage devices in locations known to Multics. Only the relatively active portions of each segment must reside in core memory.

The GE-645 processor considers core memory to consist of blocks of 64 or 1024 words. Each block begins in an absolute address, which is 0 modulo 64 or 0 modulo 1024.

A segment similarly consists of blocks of 64 or 1024 words called pages. Any pages of a segment may now be placed in any available core memory blocks of appropriate size. The GE-645 relative addressing capability allows such pages to be addressed as if they are physically contiguous even though they are in widely scattered absolute locations. The above ideas are shown in Figure 21. Note: Pages in core memory still exist in secondary storage.

SEGMENT DESCRIPTOR WORD CONTROL FIELD

In describing segmentation, the origin and size fields of the SDW have been discussed. In describing paging, the control field of the SDW will be discussed. Two bits of the control field describe whether or not the corresponding segment is paged and if so whether the pages are 64 or 1024 words long.

If a segment is nonpaged, the complete segment is located in contiguous core memory addresses. In the usual case, a segment is paged. All of its pages are the same size, either 64 or 1024 words. The address field of its SDW specifies the origin of a page table, and not the segment origin as previously described.

Each word in a page table indicates the absolute location of the block in core memory to which the corresponding page is assigned; the first word locates the first page of the segment, the second word locates the second page, etc. The individual entries in the page table are called Page Table Words (PTW's). The PTW's contain a control field as well as a page origin. (See Figure 22.)

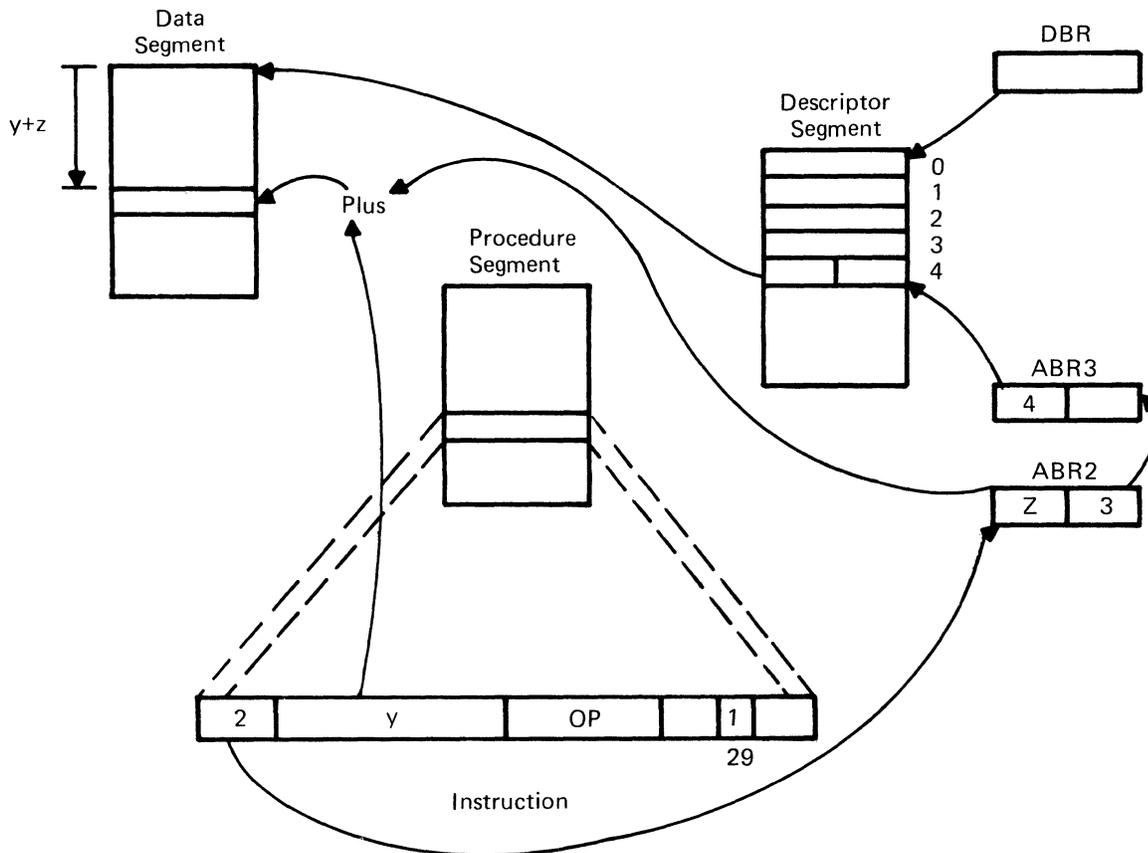


Figure 17. Operation of a Base Pair

SEGMENT ADDRESS PARTITIONING.

When paging is used, each segment address is partitioned into two parts by the processor before it is used for address selection: a page number and a word number.

The processor partitions the segment address so that each entire page can be addressed by the word number field. This requires the word number to be 10 bits long for 1024-word pages and 6 bits long for 64-word pages. Figures 23 and 24 show segment address partitioning for two sizes of pages. Eight bits are used for page number; thus a segment can be divided into a maximum of 256 pages.

PRODUCING THE ABSOLUTE ADDRESS.

With this understanding of segmentation and paging, the actions of the processor in selecting an address from a page can be traced. See Figure 25. First, the segment

number is determined and used to obtain a segment descriptor word (SDW) from the descriptor segment as described under segmentation. The SDW is now examined and found to refer to a paged segment. The segment address is partitioned into a page number and word number. The page number is added to the page table origin obtained from the SDW to locate the page table word (PTW). Finally, the word number is added to the page origin obtained from the PTW to produce the absolute memory address of the desired word.

PAGED DESCRIPTOR SEGMENT.

As previously mentioned most segments are paged. In fact, descriptor segments too can be paged. A two-bit control field in the descriptor base register specifies paging and page size in the same way as do the corresponding bits of the SDW's. If paging is specified, the descriptor segment has a page table which is used exactly as any other segment page table. See Figure 26.

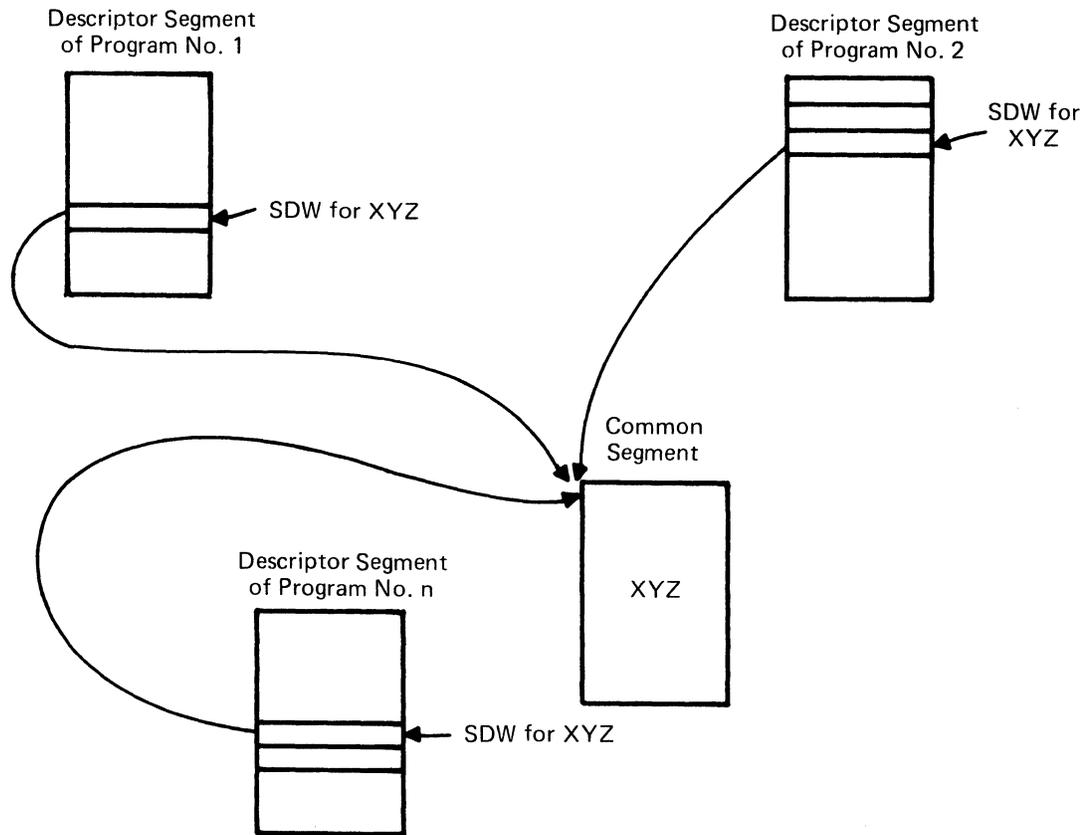


Figure 18. Several Descriptor Segments May Refer to a Common Segment

ASSOCIATIVE MEMORY

The procedure just described involves several memory accesses to obtain the final operand. This sequence must be done once to locate the page, but only once. When the page's origin is determined, it is automatically saved in the processor's associative memory, along with the segment and page numbers. When a later access to the same segment number and page number is made, the absolute page origin is provided immediately from the associative memory. Hence, repeated accesses to the descriptor segment page table, descriptor segment, and page tables are unnecessary. Through incorporation of the associative memory, the powers of segmentation and paging are made available to GE-645 users while high processor performance is maintained.

The associative memory consists of 16 high speed content addressable registers. Each register has a usage counter. The contents of the register with the greatest elapsed time since last usage are replaced when new information is added to the associative memory. A subsequent reference

to such a page requires that the absolute page origin be obtained once again from the SDW and PTW in core memory.

FAULTS AND INTERRUPTS

The system must respond promptly when an event has occurred which needs servicing or when a hardware malfunction is detected. Faults are conditions detected within a processor, while interrupt cells are set in the System Controller which in turn notifies its control processor. There are 32 faults in each processor. The 32 interrupt cells in a system controller can be set by a GIOC, extended memory module, system clock or a processor. Faults have priority over interrupts and all interrupts and low priority faults can be inhibited by setting bit 28 in the instruction word for instructions in master mode procedures.

Fast response to faults and interrupts is obtained by having a unique pair of locations set aside for each fault and interrupt condition. The locations associated with

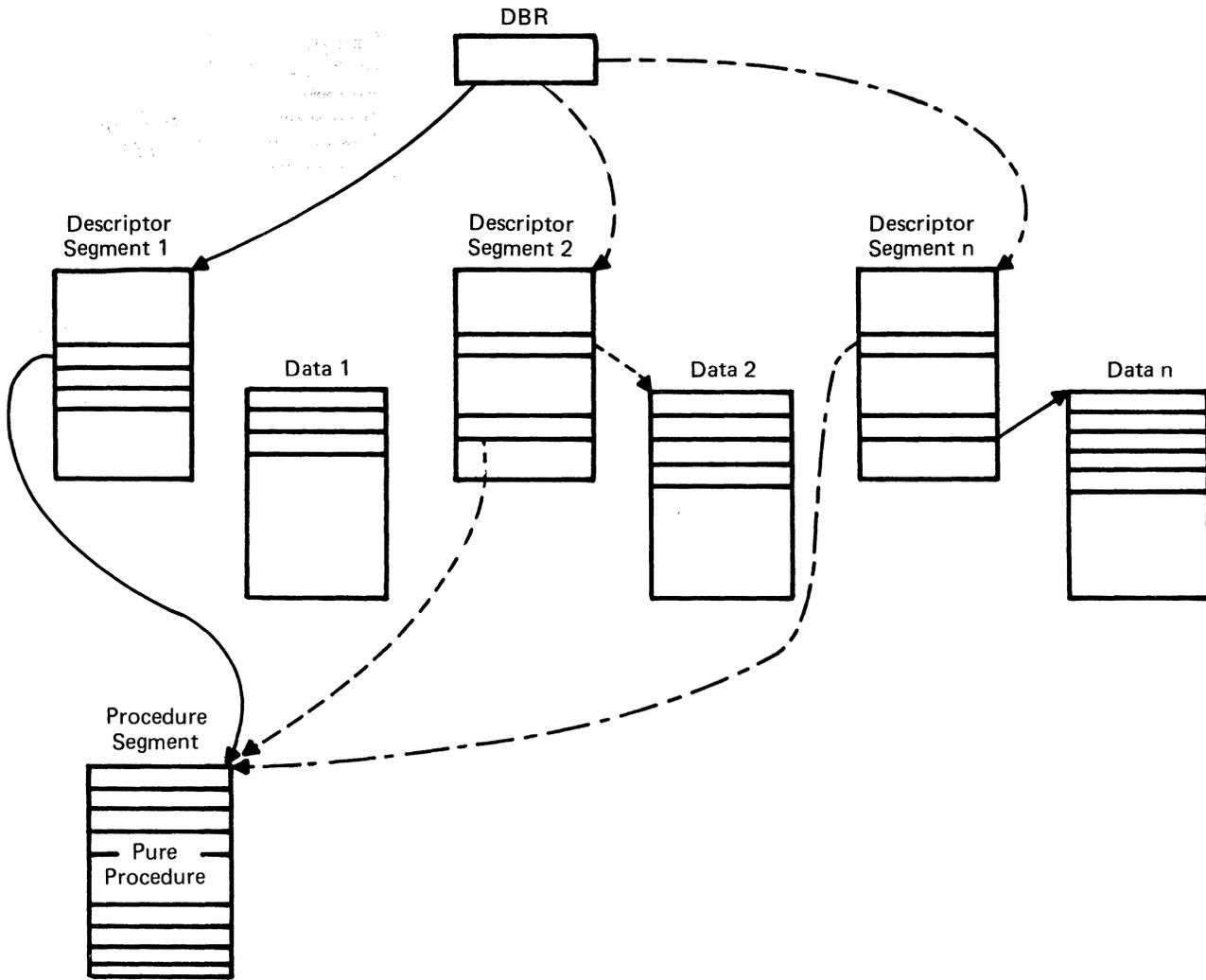


Figure 19. Common Pure Procedure Alternately Serves n Programs in a Uniprocessor Configuration

faults are called the fault vector and the locations associated with interrupts are called the interrupt vector. The base address for the fault vector is one of the basic parameters of the GE-645 system configuration established from the system configuration console. The interrupt vectors for each system controller are located immediately after the fault vector in memory.

The processor response is the same for a fault or an interrupt: the processor stops what it is doing and executes the pair of instructions in the location associated with that specific fault or interrupt. Multics has stored a pair of instructions here which save the processor's status and transfer to a routine that saves the processor's registers and services the fault or interrupt. At some point the interrupted routine will have the

registers and processor status restored and then continue as though it had not been interrupted.

Many of the processor fault conditions are deliberately or inadvertently caused by program and do not involve any hardware malfunction. Other faults are definitely caused by some hardware malfunction. In either case, a Multics routine is available to service the fault. All of the processor faults are listed with a short description in order of priority.

START UP – Caused by the POWER ON button being depressed; works in conjunction with a previous shutdown fault.

EXECUTE – Caused by EXECUTE button being depressed.

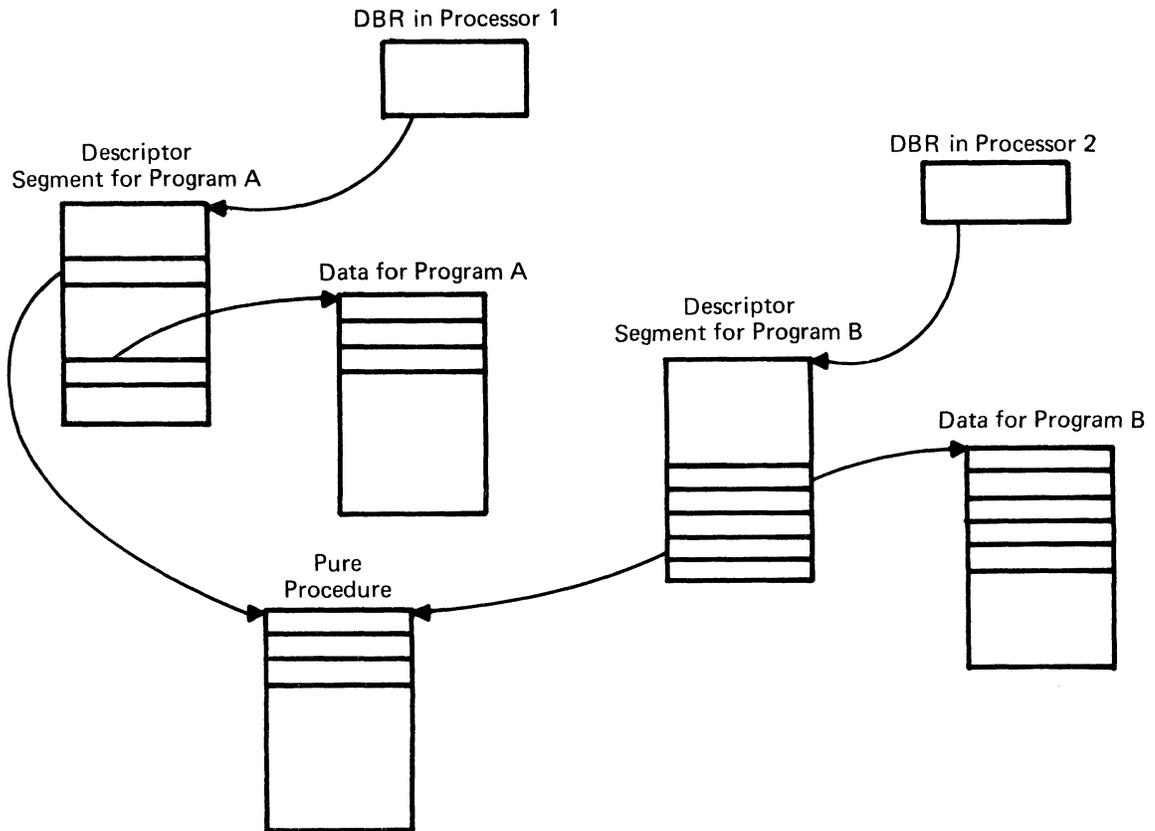


Figure 20. Common Pure Procedure Used Concurrently by Two Processors in Dual Configuration

TROUBLE – Caused by problems with Execute Double (XED) or Store Control Unit (SCU) instructions.

OPERATION NOT COMPLETE – Caused by excess delay in the processor, usually because of communication problems with system controller.

DIVIDE CHECK – Caused by a division instruction with a divisor equal to zero.

OVERFLOW – Caused by the generation of a number with a magnitude too large (or too small) for the registers. This fault can be inhibited by an overflow mask bit in the Indicator Register.

PARITY – Caused by parity error being detected during the reading of a word from memory. This fault can be inhibited by a parity mask bit in the Indicator Register.

ILLEGAL STORE COMMAND – Caused by the processor issuing a Connect (CIOC) instruction to a system controller port which is masked.

LOCKUP – Caused when interrupts are inhibited for more than one to two milliseconds.

ILLEGAL PROCEDURE – Caused when the program attempts to violate its access rights. This includes execution of GE-645 privileged instructions in slave mode, slave program access of address base registers reversed for use by Multics, undefined operation codes, address outside segment bounds, and access type not permitted by access control bits.

GE-635 COMPATIBILITY – Caused when 635 LBAR or SBAR instructions are executed.

GE-635/645 COMPATIBILITY – Caused when a slave program attempts to execute a privilege instruction that exists in both GE-645 and GE-635.

MASTER MODE ENTRY 1

MASTER MODE ENTRY 2

MASTER MODE ENTRY 3

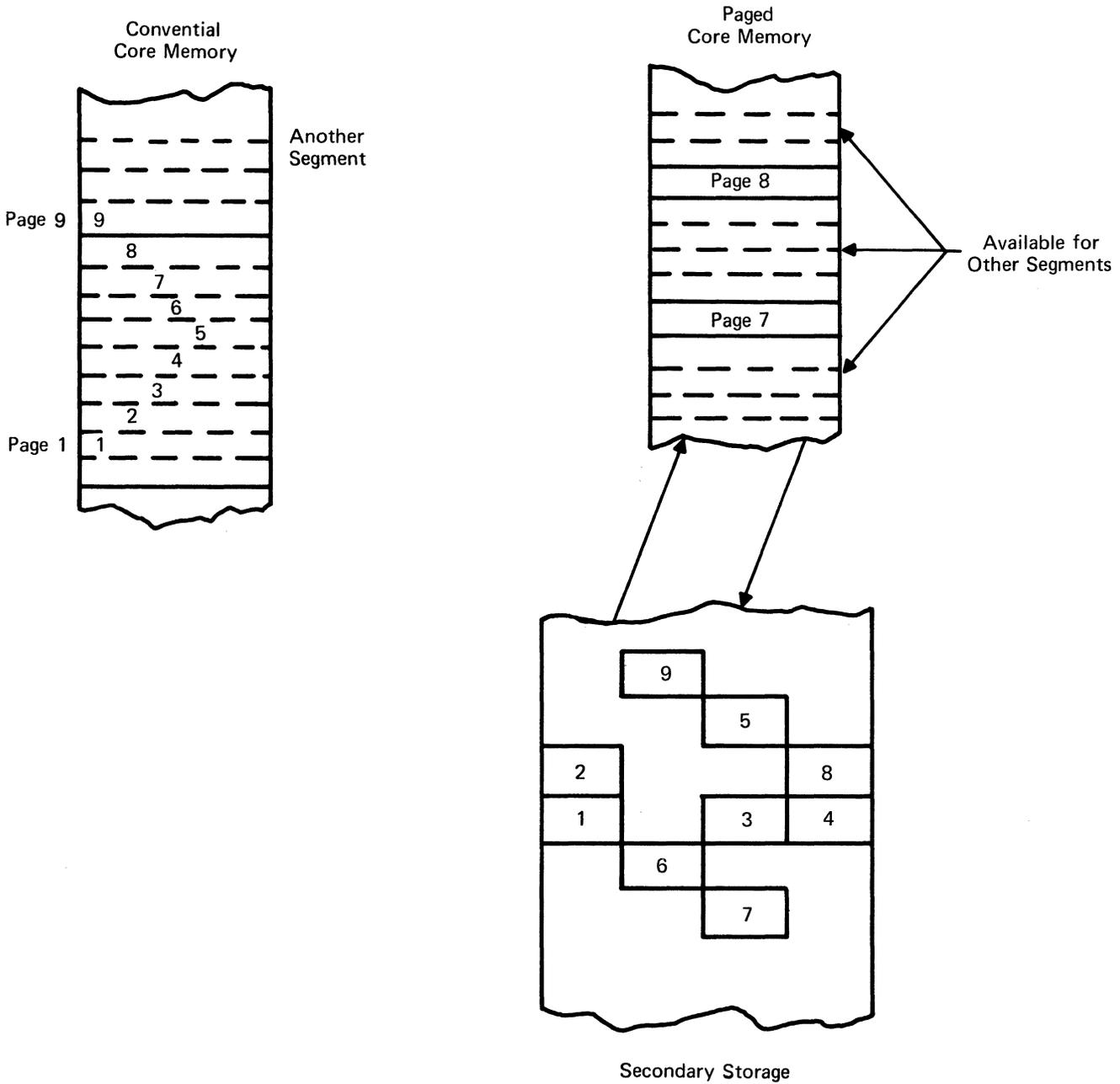


Figure 21. Frequently-Used Pages Occupy Any Available Blocks of Core Memory

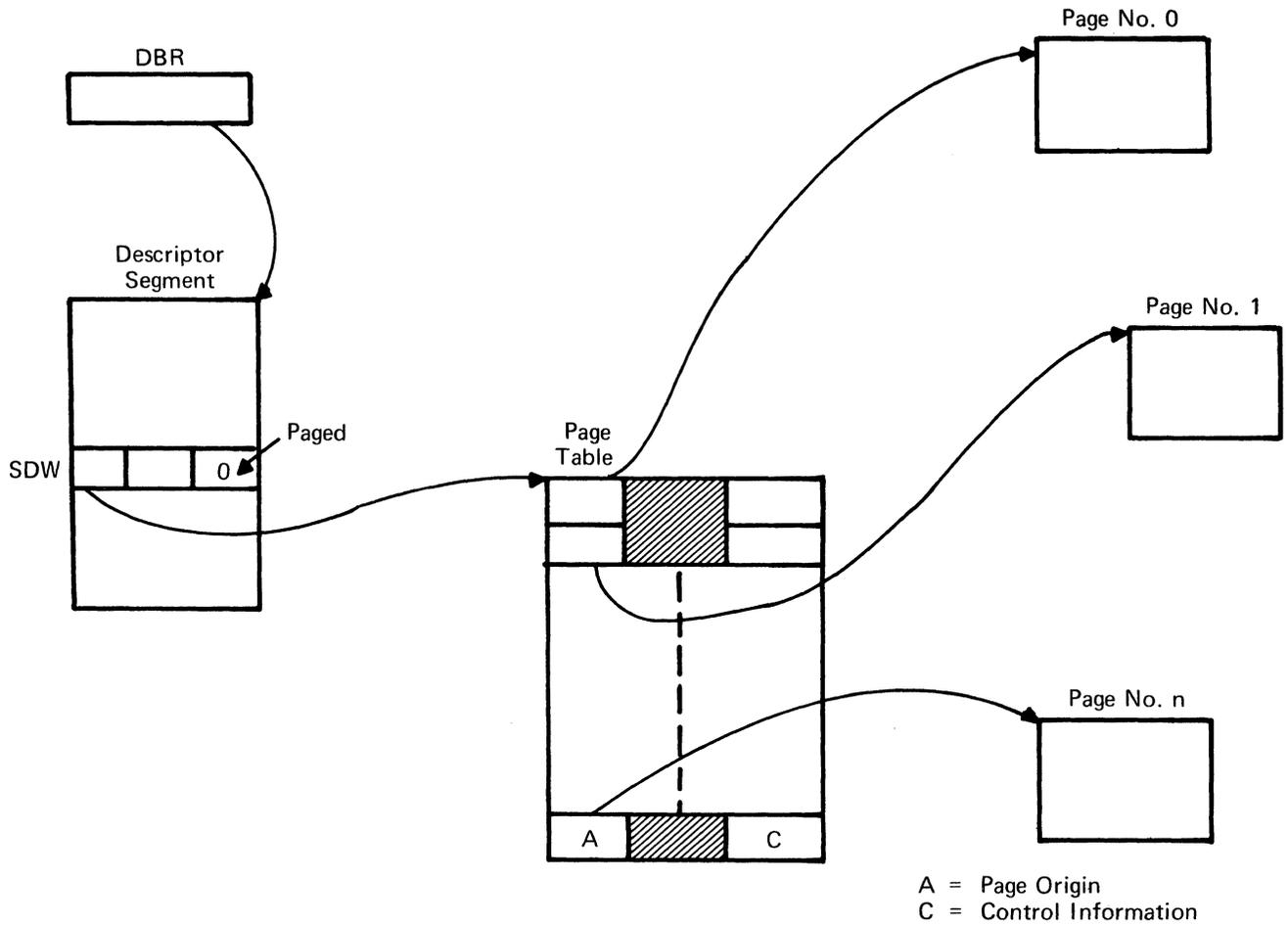


Figure 22. Page Table Words Specify Absolute Core Memory Origins

MASTER MODE ENTRY 4

DERAIL

These 5 faults are each caused by the corresponding program instructions and are used for slave entry into master mode programs.

FAULT TAG 1

FAULT TAG 2

FAULT TAG 3

These faults occur when the associated fault tag tally designator is encountered in an Indirect Then Tally (IT) address modifier field.

ILLEGAL DESCRIPTOR – Caused by improper combination of control bits in either a segment descriptor word or page table word.

DIRECTED FAULT 0

DIRECTED FAULT 1

DIRECTED FAULT 2

DIRECTED FAULT 3

DIRECTED FAULT 4

DIRECTED FAULT 5

DIRECTED FAULT 6

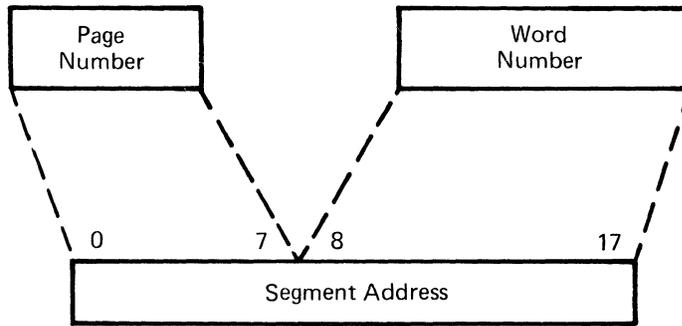


Figure 23. Partitioning of Segment Address for 1024-Word Pages

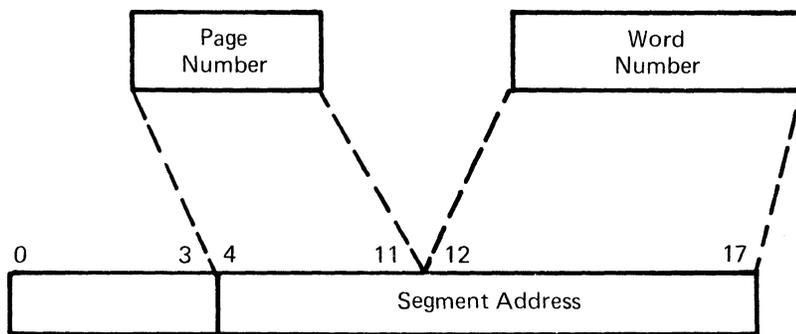


Figure 24. Partitioning of Segment Address for 64-Word Pages

DIRECTED FAULT 7

These faults occur when the appropriate control bit configuration is used in a segment descriptor word or a page table word. These faults are used to let Multics know when a program tries to access a segment or page that is missing from core, or that requires service metering, or special access control.

CONNECT – Occurs when the processor receives a connect signal from another processor through a system controller.

TIMER RUNOUT – Occurs when the contents of the Timer Register reach zero.

SHUTDOWN – Occurs about one millisecond before power fails.

PRIVILEGED OPERATION

As mentioned earlier in this chapter, the processor has three modes of operation: absolute, master, and slave. All instructions are available in absolute and master modes. Most, but not all, of the instructions are available in slave mode. General users are restricted to the slave mode

and hence are prevented from executing any instructions that will damage other programs or Multics. Privileged instructions such as those which operate upon the descriptor base register, load the alarm clock, and initiate input/output devices are available only in the absolute and master modes.

Whenever a fault or interrupt occurs, the absolute mode is entered. This causes no trouble since faults place the processor in control of Multics which, like the hardware, can be assumed to be debugged. Instructions in the absolute mode can be inhibited from being interrupted. Only in the absolute mode may the absolute addresses of core memory be referenced by their true values. The relative addressing features associated with segmentation and paging can be activated optionally in the absolute mode for referring to operands.

Whenever a processing unit is not in absolute mode, it will be in master or slave mode and use relative addressing in fetching instructions and operands. The mode is determined by the control fields of the SDW and the PTW of the procedure segment page from which the processor is fetching instructions. When the control field designates the segment to be a master procedure, the processor is in master mode. As such it is eligible to exercise all the absolute mode privileges except that of referring to absolute

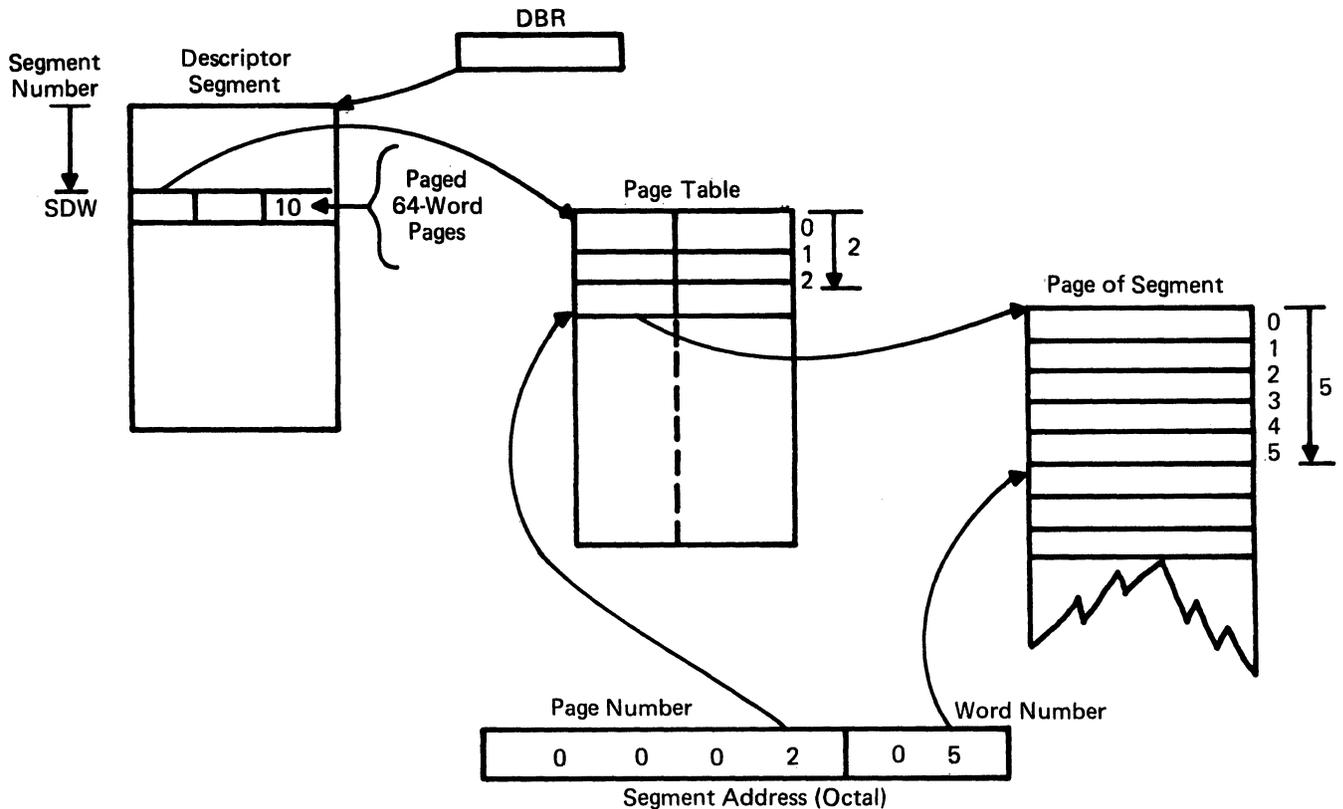


Figure 25. Selection of a Word From a Paged Segment

memory addresses. In addition to their other properties, master mode procedure segments, when entered from a slave mode instruction, may be entered only at segment address zero. This provides a method for Multics to verify the validity of such entries and prevent misuse of a master mode segment to the detriment of the system and its users.

If the segment is designated to be a slave procedure, interrupt inhibition indicated in individual instructions is ignored and attempted execution of any privileged instruction results in a fault.

ACCESS CONTROL

In addition to the modes of the processor which determine certain privileges, access to segments can be further controlled. The foregoing has introduced the notion that segments can be classified as master or slave procedures; they can also be classified as data or "execute-only" procedures. These classifications are made as the third and fourth variations of the segment type portion of the SDW control field. If a segment is classified as data, it cannot be accessed for instructions. If a segment is classified as an execute-only procedure, it can be accessed only for

instructions. A slave procedure can enter an execute-only segment only at segment address zero. This is the same restriction as for entering master mode segments. In all other ways, execute-only segments are executed in the same way as slave procedures.

Access to all types of segments may be further qualified by the use of two additional and independent bits in the control field of the SDW. These are the write permit and master access bits. They may be used to prevent modification of a segment and to permit access to a segment only by master procedures. Either or both of these qualifications may be applied to any type of segment.

All pure procedures, regardless of their mode, should be consistent with their definition and use, and have write permission disallowed. In this way, attempted improper use by one user cannot diminish the usefulness to others.

Similarly, many users might find the general availability of certain data, e.g. tables of physical constants, to be valuable for their common use. However, if a single user stored an incorrect value in a table, its potential usefulness to all would be destroyed. Such eventualities can be eliminated by proper use of the write-permit bit and

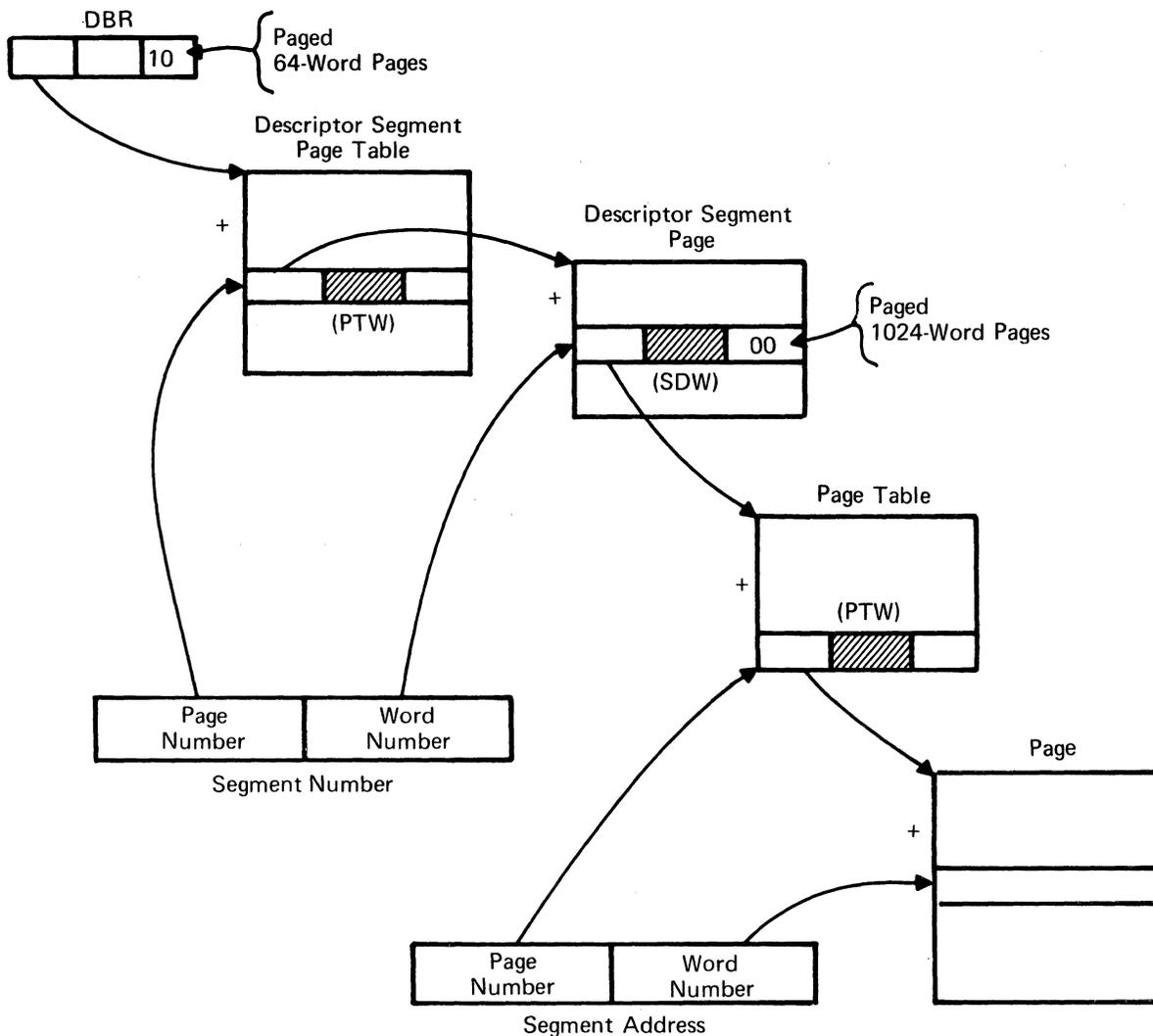


Figure 26. Selection of a Word from a Paged Segment Using a Paged Descriptor Segment

the declarations available through Multics which control setting of the write-permit bit.

The features of the three processor modes of operation are summarized in the following chart.

	MODES		
	Absolute	Master	Slave
Can privileged instructions be executed?	Yes	Yes	No
Does bit 28 of an instruction inhibit an interrupt?	Yes	Yes	No
Type of Address for instruction fetch	Absolute	Relative	Relative
Type of Address for operand fetch	Absolute or Relative	Relative	Relative
Access to other segments and pages is controlled?	No	Yes	Yes

7. GENERALIZED INPUT/OUTPUT CONTROLLER

The generalized input/output controller (GIOC) interfaces peripherals and communication lines with the memory of the 645 system, and is capable of operating a large number of devices of almost arbitrary variety and speed. The GIOC is controlled by information stored in memory, with access to the memory being shared with the other active modules in the system. Data transfers between I/O devices and memory are accomplished by the GIOC while the processors continue to run programs. I/O transactions are controlled by lists of control words prepared by Multics and stored in memory. When an I/O transaction has been completed, or when special conditions are detected, the GIOC informs Multics by causing a program interrupt.

The GIOC consists of two, three, or four cabinets, depending on the number of modular units, called adapters, that are required by the specific configuration of peripherals and remote terminals.

The functional division of the GIOC is illustrated in Figure 27. The modular functional building blocks are the various types of channels, the adapters, and the GIOC controller.

CHANNELS

The functional entities that the Multics I/O system deals with in a GIOC are called channels. There are four types of channels:

- Data channels – control the transfer of data.
- List channels – obtain new control words for associated data channels.

<u>Channel Number</u>	<u>Channel Type</u>	<u>Control Word</u>
0, 1, 2, 3	Status channel	Status control word (SCW)
4 - 7	-	Reserved
8, 9, 10	Connect channel	Instruction pointer word (IPW)
11 - 15	-	Reserved
Even from 16 to 4094	List channel	List pointer word (LPW)
Odd from 17 to 4095	Data channel	Data control word (DCW)

DATA CHANNELS

Data transfers between the memory modules and a peripheral subsystem or communication line are controlled by a data channel. Each peripheral subsystem or communication line connects to a separate data channel, which provides the necessary data buffering and interface hardware. Because of

- Connect channels – distribute instructions to list and data channels.
- Status channels – report occurrence of status events in connect, list, and data channels.

There are two classes of data channels – indirect and direct. All list channels, connect channels, and status channels are indirect.

Each channel is controlled by a control word. The control word for an indirect channel resides in core memory. Thus a data transaction, or “data service”, requires additional memory accesses to obtain the control word. A direct channel avoids the additional accesses by handling the control word in the channel hardware.

Each channel has a mailbox, consisting of a pair of 36-bit words in core memory. The mailbox for an indirect channel contains the active control word for the channel, while the mailbox for a direct channel is used for making the updated control word from the direct channel available to Multics at the conclusion of an I/O transaction.

The mailboxes are located in a block of memory starting at an address called the “GIOC base address.” This address is one of the basic parameters of the GE-645 system configuration established from the system configuration console. A pair of 36-bit words from this block is assigned to each channel in channel-number order. The channel number also determines the type of channel and the interpretation of the control word as shown by the following table:

the diverse requirements for interfaces and buffering there are a number of different types of data channels, each type being for use with a different type of adapter. Data channels are intended for two-way nonsimultaneous communication. Two data channels must be used for applications requiring two-way simultaneous communication.

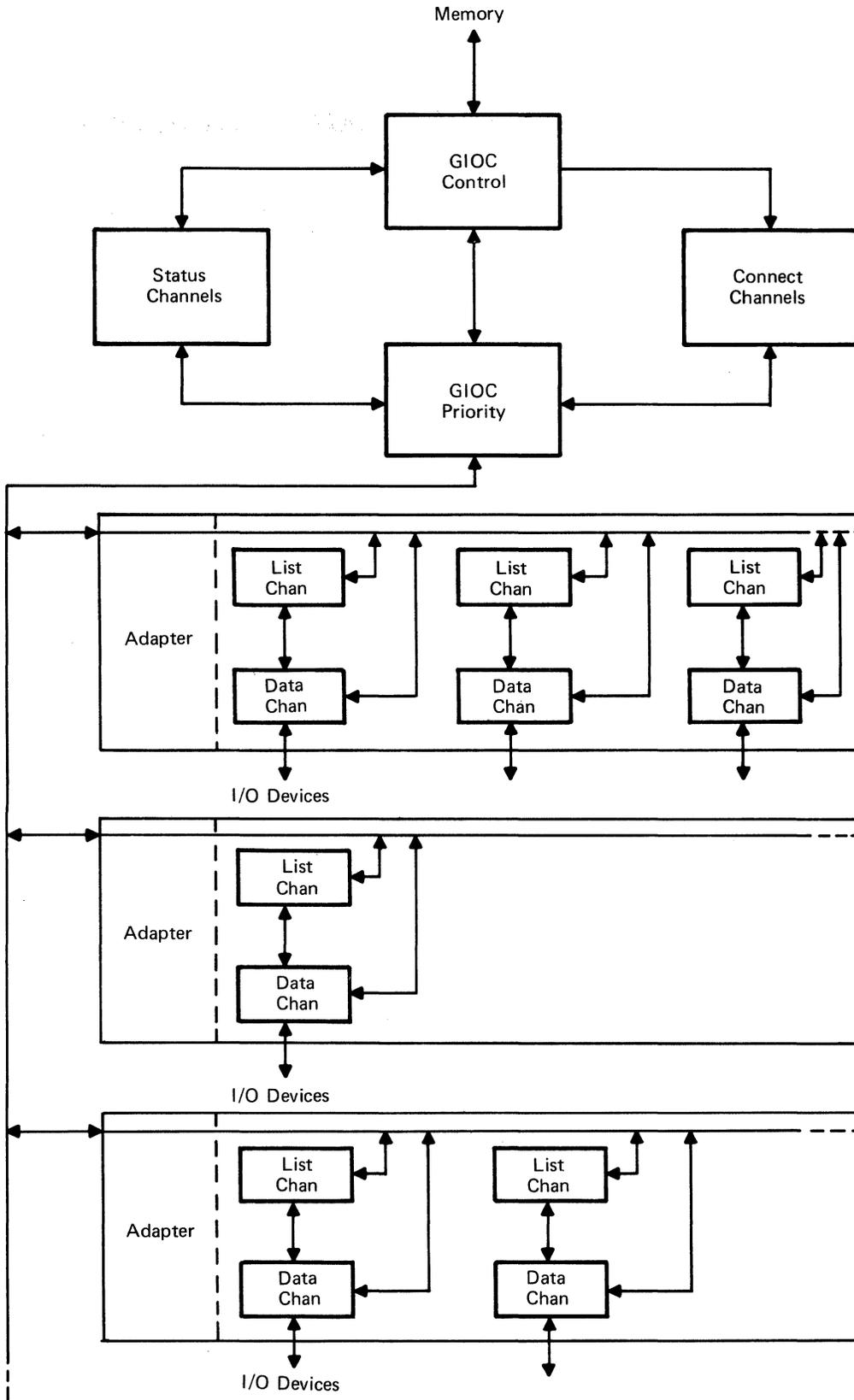
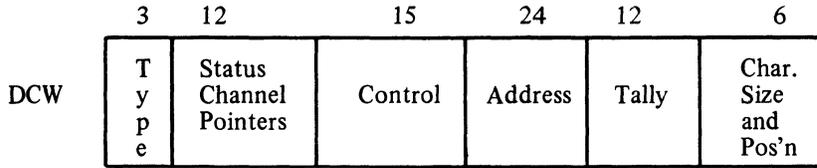


Figure 27. GIOC Functional Organization

The control word for a data channel is called a data control word (DCW), and has the format shown below.



Five types of DCWs are available for use with indirect data channels, and three of these can also be used with direct channels.

The microcode DCW defines the starting address and tally (number of characters or words) in a block of memory for input or output data, and can be used with any data channel.

The control character DCW defines the starting address and tally in a block of memory for input data, with provision for detection of a specific character or class of characters, and can be used only with indirect data channels.

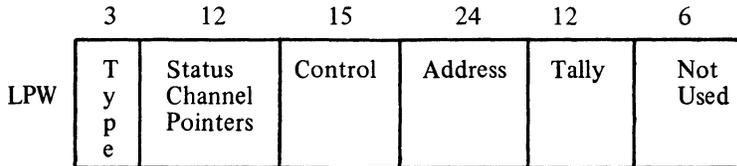
The transfer DCW defines the starting address and tally (number of DCWs) in a block of memory containing additional DCWs, and can be used with any data channel.

The instruction DCW specifies an instruction to be issued to the data channel at a preplanned point in the data transfer sequence. Some form of instruction DCW is available for all data channels.

The literal DCW defines a specific pattern of output data which is to be transmitted repetitively for a specified number of times (tally), and can be used only with indirect data channels.

LIST CHANNELS

Each data channel has an associated list channel which is responsible for obtaining new DCWs for the data channel when the previous DCW has been used up. The control word for a list channel, called a list pointer word (LPW), is generated and placed in the list channel mailbox by Multics. The format of the LPW is shown below.

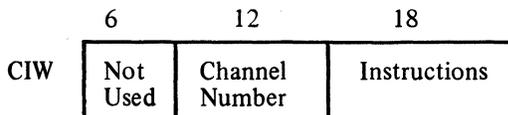


The LPW defines the starting address and number of DCWs (tally) in a list of DCWs which are to be used by the associated data channel. The DCW is placed in the mailbox of the associated data channel by the GIOC unless the data channel is direct. For direct channels the GIOC retains the DCW in the data channel hardware.

The channel numbers of a list channel and its associated data channel are always an even-odd pair of channel numbers.

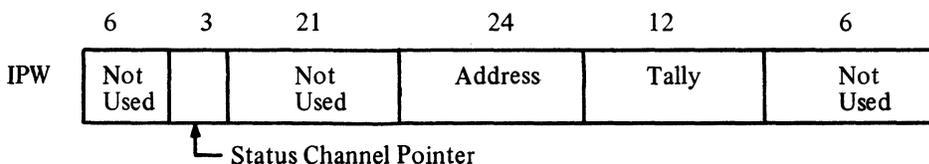
CONNECT CHANNELS

Each of the three connect channels is responsible for the distribution of channel instruction words (CIW) to data channels and list channels. CIWs are used to initiate or change the operation of data channels, and have the format shown below.



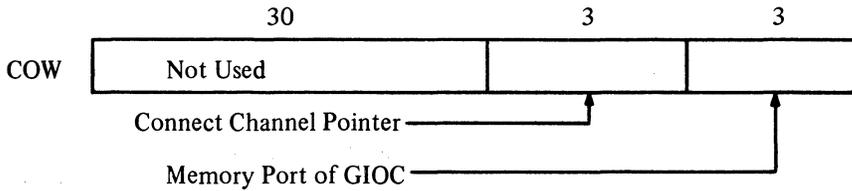
The control word for a connect channel, called an instruction pointer word (IPW) is generated and placed in the connect channel mailbox by Multics. The format of the IPW is shown below.

The control word for a connect channel, called an instruction pointer word (IPW) is generated and placed in the connect channel mailbox by Multics. The format of the IPW is shown below.



The IPW defines the starting address and number of CIWs (tally) in the list of CIWs that is to be distributed. The operation of a connect channel is initiated when a Connect instruction (CIOC) is executed by a processor and the

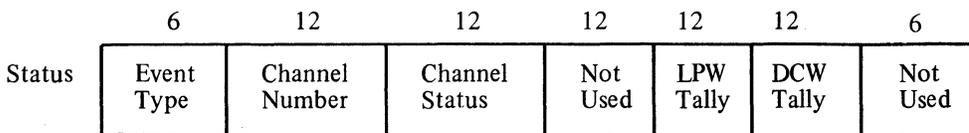
operand word from the CIOC instruction is sent to the GIOC. The connect operand word (COW) has the format shown below.



STATUS CHANNELS

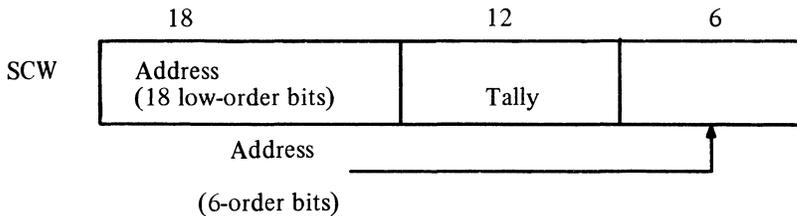
Each of the four status channels is responsible for reporting the occurrence of significant status events in the I/O sub-

system to Multics. When a status event occurs, the appropriate status channel forms two 36-bit words of status having the format shown below.



The control word for a status channel, called a status control word (SCW), is generated and placed in the even word

of the status channel mailbox by Multics and has the format shown below.



The SCW defines the starting address and the number of words in a block of memory where the GIOC is to place status words after the block currently in use has been filled. When a block has been filled by the GIOC, it obtains the definition of the next block by moving the even word of the mailbox into the odd word. The even word is then set to zero so that Multics is informed that another new block must be defined before the remaining block has been filled.

ADAPTERS

Adapters are modular units which can be chosen to fit the requirements of the peripheral and remote terminal equipment. Each adapter includes one or more data channels and the associated list channels. Each data channel provides an interface suitable for connection to a particular type of input/output device.

Each of the status channels corresponds to a particular interrupt cell in the system controller which contains the GIOC base address, and the appropriate interrupt cell is turned on every time that a status event is reported, so that one of the processors will be interrupted and made aware of the status event in the GIOC.

Since there are a number of different interfaces required by various types of input/output devices there are a number of different types of GIOC adapters. Each type of adapter provides for the handling of one or more devices of a particular type. The following adapters are available.

SUMMARY

The arrangement of mailboxes and the way that control words from the mailboxes define areas of memory for use by the GIOC is shown in Figure 28.

- High performance peripheral adapter (HPC600)
- Direct disc adapter (DDA600)
- Custom direct adapter (CDA600)
- Indirect peripheral adapter (IPA600)
- Teletypewriter adapter (TTA600)

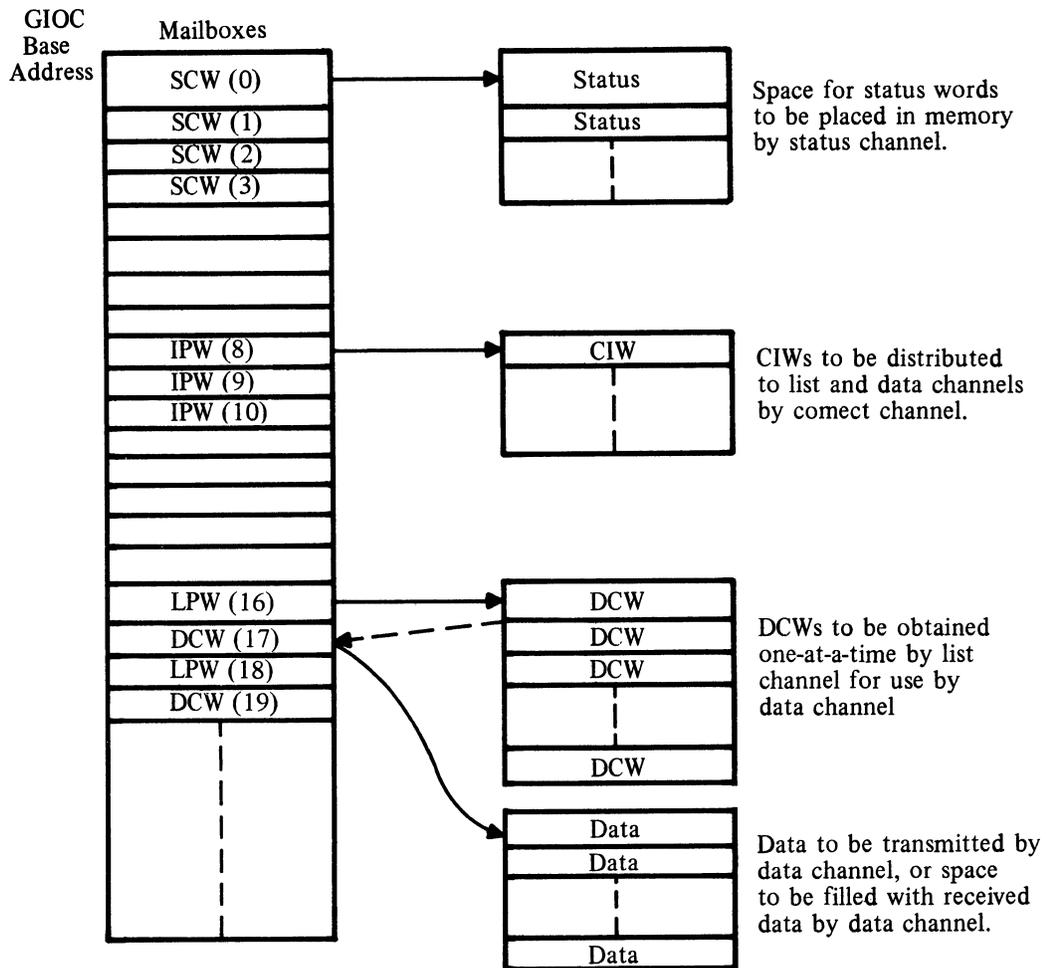


Figure 28

- Character asynchronous adapter (CAA600)
- Character synchronous adapter (CSA600)
- Dialing adapter (DGA600)

HIGH PERFORMANCE PERIPHERAL ADAPTER (HPC600)

The high performance peripheral adapter provides one direct data channel for connection to a high speed peripheral subsystem such as a magnetic tape controller, magnetic drum controller, or mass storage subsystem. Two high performance peripheral adapters are required for connection to a dual magnetic tape controller. Data transfer to or from memory usually involves 72 bits at a time.

DIRECT DISC ADAPTER (DDA600)

The direct disc adapter provides two direct data channels for connection to a disc storage controller. Each data transfer to or from memory involves 36 bits.

CUSTOM DIRECT ADAPTER (CDA600)

The custom direct adapter provides one direct data channel for connection to special high speed interfaces requiring transfer rates of up to 250,000 36-bit words per second. Data transfer to or from memory usually involves 72 bits at a time.

INDIRECT PERIPHERAL ADAPTER (IPA600)

The indirect peripheral adapter provides up to six indirect data channels for connection to low speed peripheral subsystems such as a card reader, card punch, line printer, or perforated tape subsystem. Each data transfer to or from memory involves six bits.

TELETYPEWRITER ADAPTER (TTA600)

The teletypewriter adapter provides up to 32 indirect data channels (in groups of eight) for connection to Bell System 103 series data sets. Plug selection provides for any of the standard bit rates between 45 and 200 bits per second, and for 5-, 6-, 7-, or 8-level codes. All channels within one adapter operate at the same bit rate and code level. Each data transfer to or from memory involves six or nine bits.

CHARACTER ASYNCHRONOUS ADAPTER (CAA600)

The character asynchronous adapter provides up to three indirect data channels for connection to Bell System 202 series data sets. Plug selection provides for any of the standard bit rates between 150 and 2400 bits per second and for 5-, 6-, 7-, 8-level codes. All channels within one adapter operate at the same bit rate and code level. Each data transfer to or from memory involves six or nine bits.

CHARACTER SYNCHRONOUS ADAPTER (CSA600)

The character synchronous adapter provides up to three indirect data channels for connection to Bell System 201 series data sets. The bit rate (up to 2400 bits per second) of each channel is dependent on timing signals from the attached data set and is, therefore, independent on each channel. Plug selection for each channel provides for 5-, 6-, 7-, or 8-level codes and for definition of the synchronization character. Each data transfer to or from memory involves six or nine bits.

DIALING ADAPTER (DGA600)

The dialing adapter provides up to eight indirect data channels for connection to Bell System 801 series automatic call units so that the 645 system can originate calls to remote terminals on dial telephone facilities. Each data transfer from memory involves six or nine bits, and provides one digit for dialing.

GIOC CONTROLLER (DC8031)

All work done by the GIOC is partitioned into units of service so that a single service can be completed with no more than five accesses to memory. Services are performed one-at-a-time, except that a direct data channel can temporarily preempt service from an indirect channel.

Each request for service has a specific priority associated with it. Each adapter uses one or more levels of GIOC hardware priority, and assigns specific types of service requests to each of the levels which it uses. All data and list channels within an adapter may share the same priorities. Logic within the adapter ensures that only one channel at a time is serviced.

The three connect channels are each assigned a separate priority so that instructions can be issued at any of the three priority levels. Similarly, the four status channels are each assigned a separate priority. Each status channel also has a priority for program interrupt i.e., the interrupt cell it sets in the system controller, that is completely independent of its GIOC hardware priority.

STATUS EVENTS

The status channels are responsible for reporting the occurrence of significant status events in the I/O subsystem so that appropriate action can be taken by Multics. Five classes of status events are detected by the GIOC:

- Internal signal
- Exhaust

- Terminate
- External signal
- Emergency

Internal signal status indicates the detection of a special control event, as defined by the control field in a DCW or LPW.

Exhaust status indicates that the block of memory defined by a DCW, LPW, or IPW has been used up.

Terminate status indicates that control of data transfer operations in a data channel has been returned from the GIOC to Multics.

External signal status indicates the detection of significant status events, other than termination, in the device that is connected to a data channel.

Emergency status indicates the detection of either a hardware malfunction or a software error in setting up control words for the GIOC. All emergency status events are reported through status channel zero.

The control words for data channels, list channels, and connect channels include status channel pointer fields which allow Multics to designate the status channel to be used for reporting terminate, exhaust, external signal, and internal signal status events. In choosing a particular status channel Multics effectively chooses a particular interrupt cell that is to be set when the status event is reported. Reporting of any class of status event can be prevented by placing a zero in the appropriate status channel pointer field.

Thus, under program control, the same class of status event occurring on two different data channels can be assigned to two different levels of GIOC hardware priority and correspondingly different levels of program interrupt priority. This allows optimization of the real-time effect of any event upon any other queued events.

OPERATION

Before the operation of a data channel in the GIOC can be initiated Multics performs the following operations:

- A list of one or more DCWs that will control the data channel is generated.
- An LPW, defining the location and length of the list of DCWs, is generated and placed in the mailbox for the associated list channel.
- A CIW that will initiate the operation of the list channel is generated and included in a list of CIWs intended for other list channels.
- A IPW, defining the location and length of the list of CIWs, is generated and placed in the mailbox for one of the three connect channels.

After this has been accomplished Multics executes a Connect instruction (GIOC) which initiates the operation of the connect channel in the GIOC. The connect channel distributes the CIWs, thereby initiating the operation of the list channels.

The list channel obtains the first DCW from the list, and presumably this DCW is an instruction DCW which initiates data transfer. This also causes the list channel to obtain the

next DCW, defining the block of memory to be used for the data transfer.

The data transfer proceeds under the control of the GIOC. When the I/O operation has been completed, or when the GIOC detects a status event which may require the attention of Multics, one of the status channels is used to store status in memory and to set an interrupt cell in one of the system controllers, so that one of the processors will be interrupted and made aware of the status event.

8. EXTENDED MEMORY MODULE

The Extended Memory Unit (EMU302) is used as an extension of core memory. Segments and pages of programs flow between memory and the EMU under the control of Multics. Information used frequently remains in core memory, while information needed less often remains in the extended memory unit where it can be obtained quickly when needed.

ORGANIZATION

The extended memory module consists of two units: a controller and a rotating storage unit. The controller obtains control words from memory, interprets them, and reads or writes the desired information from or onto the rotating unit. The storage unit is actually a fixed head magnetic disc unit.

Data is organized in sectors of 80 words, 64 of which are data words and 16 are in the guard band. Words in the guard band are used to store parity for the 64 data words and for testing storage unit operation without disturbing the data words. (This testing feature is discussed later in this chapter.) A track set of 16 read/write heads simultaneously reads or writes a sector.

The extended memory module has a storage capacity of 4 million words. The storage unit is organized into 4096 tracks (read/write heads), and therefore, 256 track sets, with 256 sectors in each track set.

PERFORMANCE CHARACTERISTICS

The average transfer rate between the extended memory unit and core memory is 470,000 words per second. Data is always transferred as pairs of words, that is, 72 bits per memory access, with four words transferred every 6.7 microseconds. At this rate, 32k words are transferred in 70 milliseconds allowing for guard bands. The storage unit rotates at 1725 rpm which gives an average latency of 17.4 milliseconds. Since there is a guard band at the end of each sector, there is a small amount of time between the end of one sector and the beginning of the next sector. Figure 29 represents conceptually a small portion of the storage unit's surface. Each rectangle represents an 80-word sector, which includes a guard band.

Assume that sector A is being written. After writing the last of the data information, new parity is recorded in the guard band, but the other guard band information is not changed. The extended memory controller can obtain a new control word from memory, interpret that word, select a different set of read/write heads, and change from the writing to the reading mode before the beginning of sector B comes under its set of read/write heads. Thus it is possible to eliminate the latency by building the list of control words in the proper order. Sectors can be identified

by their angular position and by the track set used in reading and writing. In figure 29, A and C start at the same angular position, and B and D start in the next angular position. C and D are read or written by the same track set.

OPERATION

The EMU differs from the GIOC in that, under normal circumstances, it never terminates its operation and hence does not have to be repeatedly connected. A list of data control words (DCW's) is continually updated by the operating system. Each DCW specifies a command, a core-storage address, an EMU address, and the location of the next DCW. The EMU address may be either one 64-word sector or sixteen consecutive sectors with a total of 1024 words. These storage areas correspond to the page sizes used by a processor.

The extended memory portion of Multics receives requests from other parts of Multics, determines what angular sectors are involved, makes up the DCW's and inserts the DCW's in a list. The DCW specifies a storage area by track set, angular position, and number of sectors. The latency is minimized by the order in which the DCW's are placed in the list.

Instead of the order corresponding to the order in which the requests are received by the operating system, it is determined by the angular position of the storage areas. Thus, when the storage unit reaches a new angular position, the next DCW is for a sector at that position.

At times, the storage unit is ready to service the list of DCW's before requests have been received for sectors in every angular position. In this situation, the operating system uses a DCW which specifies that no data transfer is to take place. This allows the storage unit to maintain continuous operation by eliminating a disconnect and then a connect.

A current status word (CSW) is stored in memory as the storage unit begins "executing" a DCW. If an error of any kind is detected by the extended memory module, an abnormal status word (ASW) is stored in core memory. The contents of the ASW identify the error. In addition to storing an ASW, the controller initiates a program interrupt by setting an interrupt cell in a system controller module.

Abnormal status words are stored in a queue of 32 words. When the end of the queue is reached, the extended memory controller automatically starts over at the beginning of the queue's storage area. Hardware is provided in the controller so that Multics can indicate which ASW's it has serviced and, hence, which ASW locations the controller may use for storing new ASW's. If the controller reaches

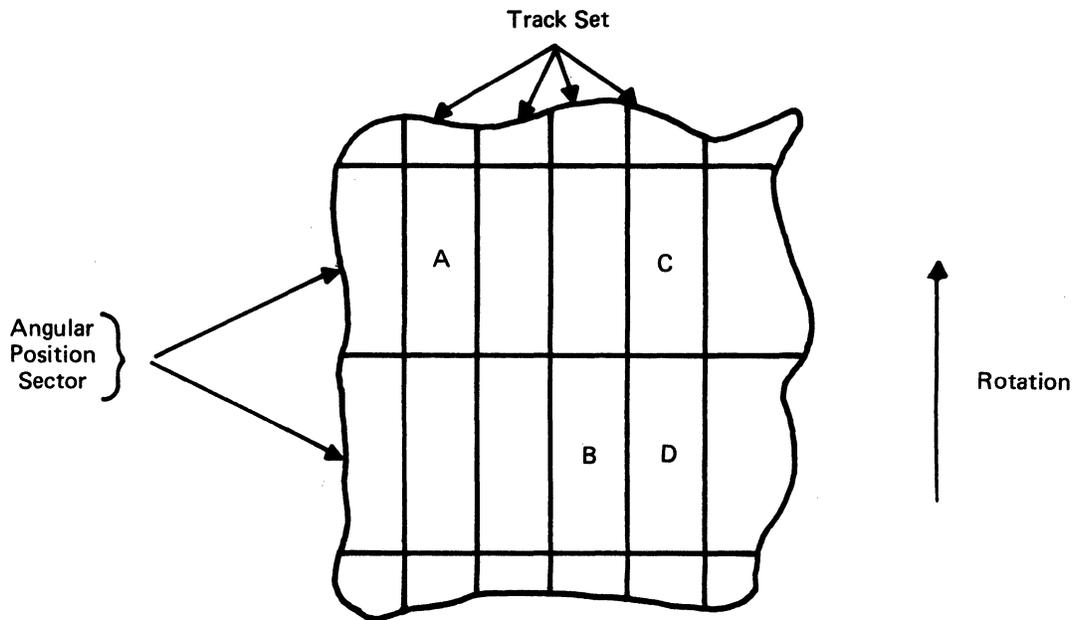


Figure 29. Successive Sector Capability

a point where it would store an ASW on top of another ASW that Multics has not yet used, it disconnects and signals this event by a program interrupt.

The extended memory controller also can store an ASW and cause an interrupt even though no error is detected. This is done with specific DCW commands, such as "read and interrupt". Thus, when that DCW is used, the operating system knows where the extended memory module is in the list of DCW's. This may be used to warn the operating system that the controller is near the end of the DCW list and that more DCW's will be needed soon.

When a program interrupt is needed, the extended memory controller indicates the relative seriousness of the interrupt by setting one of three possible interrupt cells. The actual cells used are determined by a patchboard in

the extended memory controller. One interrupt cell is used to indicate that the selected DCW has been used. A second interrupt cell is used when a data error is detected. The third interrupt cell is used if control problems are detected. In all three cases, the accompanying ASW contains bits which further define the reason for the interrupt.

TEST MODES

Special test modes enable extensive test and diagnostic programs to be run without disturbing the user's data. There is a special test sector of eight 36-bit words within the guard band of each sector. When testing, data can be transferred between the test sector and core memory. Every addressable sector recorded on the storage unit can be accessed, and associated logic and recording electronics can be checked without altering or disturbing user data.

9. SYSTEM CONFIGURATION CONSOLE

The System Configuration Console (SCC600) is a central console from which the major modules of the GE-645 System can be reconfigured and the start-up process can be initiated. The following features are included.

- Static card reader to input system configuration information.
- A Card Number Verification (CNV) channel to permit software verification of a particular configuration.
- Major Module status display.
- Configuration data display.
- System Initialization and Bootload controls.
- Console MASTER/SLAVE Select.
- Connections for Operators Teletypewriter Station (OTS).
- Space for common carrier voice communication set.
- System Emergency Power Off Button.

In a large system it is possible to use 1 SCC's with one designated master and one slave.

The console is designed for normal operator use from a seated position, but is arranged for efficient operation from a standing position. The console is 44 inches high, 32 inches deep and 80 inches wide. (Figure 30)

RECONFIGURATION

The System Configuration console connects to all major modules in the system. Provisions are made to connect 8 system controllers, 4 GIOC's, 4 processors and 2 extended memory modules. This is more than a maximum configuration since the number of active modules (processors, GIOC's and EMM's) can not exceed 8 in any system.

Each major module contains a number of switches that must be set properly to specify a particular consistent system configuration. There is a maximum of 473 switches in a system composed of 8 system controllers, 4 GIOC's, 3 processors and 1 EMM. The setting of these switches can be very time consuming and very error prone. To speed up the reconfiguration of a system the setting of all these switches are punched into a single 80 column data card. A static card reader on the SCC is used to read the card

and the switch settings are transmitted to latching relays in each of the affected modules. Each module has a LOCAL/REMOTE switch which must be placed in the REMOTE position to use or store data in the latching relays. When the switch is in the LOCAL position the local switches are used.

A card number verification (CNV) channel is provided to allow the software to determine the number of the last reconfiguration card used (this is stored in the console in latching relays) and the card presently in the card reader. These numbers are 9-bit identification numbers assigned to each card. To initiate a reconfiguration the following steps are necessary.

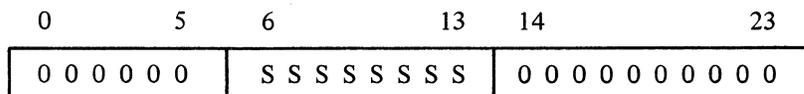
1. Turn the reconfiguration key switch to the ENABLE position.
2. Check that the OPERATE indicator is illuminated. This indicates the console is ready, not in a test mode.
3. Insert the reconfiguration data card in the card reader and engage the handle.
4. The card number of this card and the last reconfiguration card are displayed as two 3-digit octal numbers.
5. Set the mode to MASTER or SLAVE.
6. The card data can be examined using the configuration display on the console.
7. Press the ENABLE button. This button has a plastic hood to prevent accidental activation.
8. Press the LOAD button. This button also has a plastic hood. The reconfiguration data on the card is now transferred to the latching relays in each module. The duration of this cycle is 2.5 seconds.

RECONFIGURATION DATA

The data necessary to reconfigure a system can be divided into 3 categories.

Active Module Data (Processor, GIOC, EMM)

1. Base Address is the starting absolute address of a control region for each active module. The eight S bits of the 24 bits shown below are specified by the SCC; all others are zero.



COMPATIBLES / 600

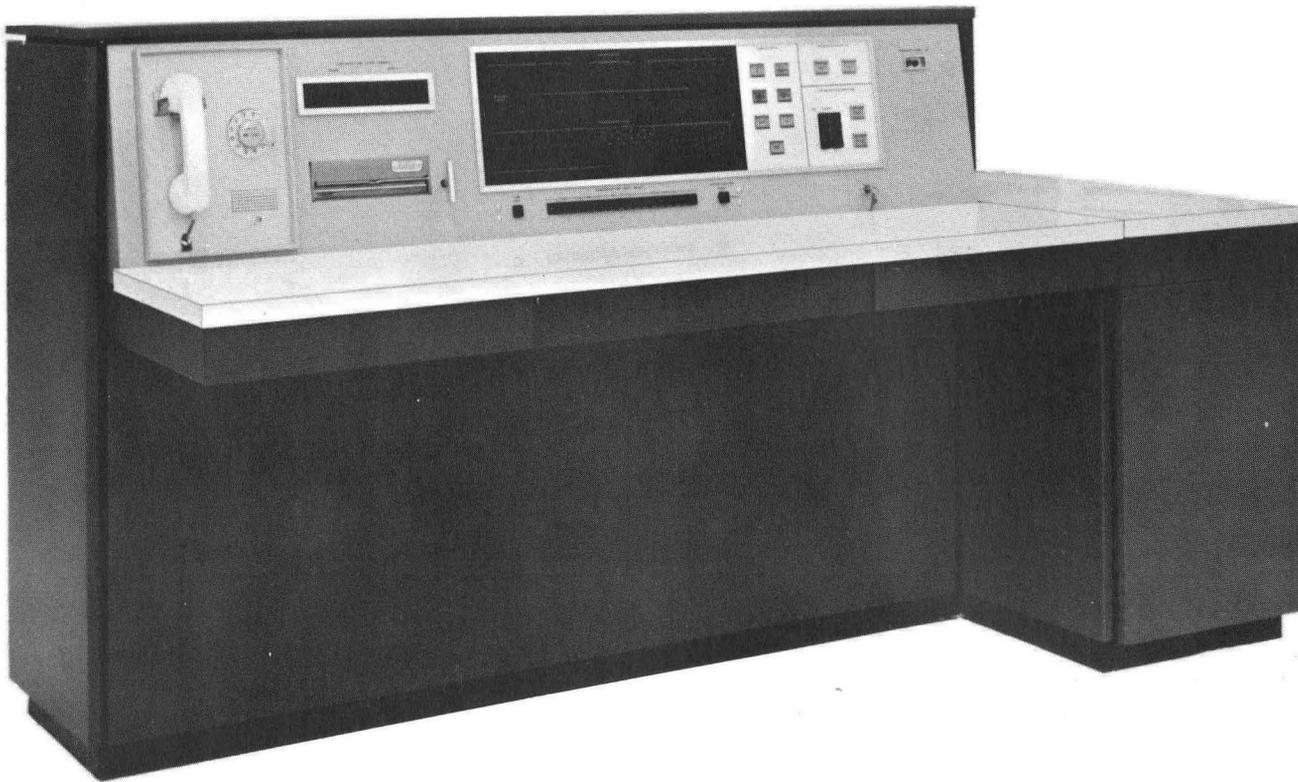
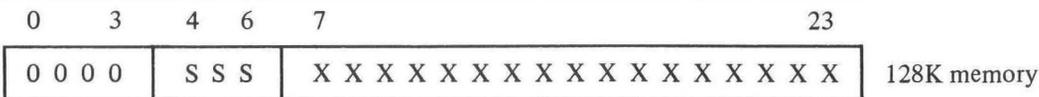
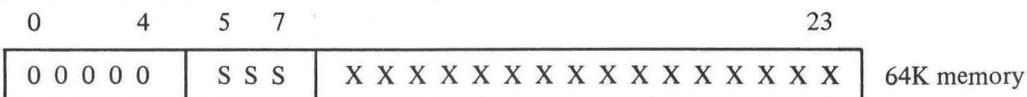
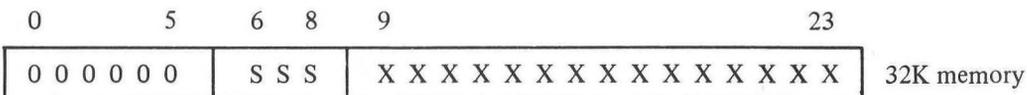


Figure 30. System Configuration Console

2. Port Enable/Disable Control permits any of the eight memory ports to be disabled in a particular active module.

3. Port Block Number is a 3-bit number to specify the three S bits above the X bits shown below, necessary to address all the words in a system controller. All bits above the S bits must be zero.



This allows multiple system controllers to be connected to each active module. The active module decides which port to use for a particular address. Each active module requires a port block number for each port connected to a system controller.

can be coded (none, 2 phase, or 4 phase). These 2 bits replace a 3 position switch at the active module.

GIOC BOOTLOAD BASE ADDRESS

In addition to the other active module reconfiguration data sent to the GIOC, the Bootload Base Address must be supplied for each GIOC. The Bootload Base Address (12 bits) is used during a system start-up to load a fixed program from a GIOC into memory.

4. Interlace control consists of 2 bits for each pair of active module ports connected to system controllers (i.e., A & B, C & D, etc.) so that the type of interlacing



SYSTEM CONTROLLER DATA

1. Port Mask Controls consist of 2 types of data. One bit is used to enable/disable the port mask register in a system controller for all ports. One bit per port is used as an ON/OFF control. This makes a total of 9 control bits. The order of control priority on each port is ON – mask – OFF. That is ON overrides the mask, but the mask register overrides an OFF condition. The mask register is program settable. In the local mode there is a 3-position switch for each port.
2. Control Processor Port Selection designates one of 8 ports which is notified when an interrupt cell is set in the system controller. A rotary switch is used in the local mode.

MASTER/SLAVE RECONFIGURATION

The SCC has the capability of defining a master and slave mode. This is not related to the master/slave mode of the processor. During a master reconfiguration any module's configuration can be changed and the input data card designates each module as belonging to either a master or slave subsystem. This data is stored in latching relays in the SCC. When the console performs a reconfiguration in the slave mode only those modules not designated master in the last master reconfiguration can be altered.

The mode of the console is changed by 2 push buttons on the console. In a two-console system the console that is first designated master locks the other into the slave mode.

SYSTEM STATUS DISPLAY

The system status display gives the operator an immediate indication of the configuration status of each module in the system.

A tri-color indicator is provided for each of the possible eighteen major modules (maximum of 16 at one time) that can be connected to an SCC. The display element for a non-existent module is not illuminated and appears blank with a dark background. The tri-color indicator display is ON-LINE (green), OFF-LINE (yellow) and LOCAL (red).

ON-LINE/OFF-LINE indication is supplied by the master/slave subsystem assignment established during the last master mode reconfiguration. The ON-LINE indicator is lit for a module having the same subsystem designation as the SCC mode (i.e., master subsystem modules are ON-LINE to a master SCC and OFF-LINE to a slave SCC).

The LOCAL indicator is illuminated when the SCC does not have control of a particular module's configuration.

Four PROCESSOR ACTIVE indicators indicate the status of the system processing activity. The indicator is green for each processor actively executing instructions other than the DIS (Delay until Interrupt Signal) instruction.

OPERATOR TELETYPEWRITER STATION

An Operator Teletypewriter Station (OTS) is provided as part of the SCC. Circuits are provided so the teletypewriter can be connected directly to 2 GIOC teletypewriter channels through a switch. These channels should be in different GIOC's (if available) to improve reliability. The switching circuits also allow the teletypewriter to be connected to two data sets for connection to the GIOC's in the normal manner. The actual teletypewriter is not included with the system configuration console.

SYSTEM INITIALIZATION AND BOOTLOAD

System Initialization and/or Bootload can be started at either the system configuration console or any GIOC. In either case the actual system action starts in a particular GIOC.

When initiated at the SCC a set of interlocking push-buttons are used to select one of two channels (A or B) and one of four GIOC's. The GIOC selected must be under control of that SCC (i.e., master or slave subsystem).

SYSTEM INITIALIZATION

A hooded SYSTEM INITIALIZATION pushbutton is located on the SCC control panel and is used for remote activation of the system initialization circuitry in the selected GIOC.

The activated GIOC sends an initialization signal to all system controllers on enabled ports. Each system controller initializes itself and sends an initialization signal out on all non-masked ports to the active modules. All active modules reset their control logic to an initial state. The initial state in each module terminates any action in process, clears all requests for action, resets all external devices and places itself in an inactive, but ready condition.

SYSTEM BOOTLOAD

A hooded SYSTEM BOOTLOAD pushbutton is located on the SCC control panel and is used for remote activation of the system bootload circuitry in the selected GIOC.

In selecting the GIOC, Channel A or B must be selected to correspond to the input channel connected to the peripheral device that contains the program to be loaded.

The selected GIOC activates the previously described system initialization process. The bootload program is sorted in diodes in the GIOC and consists of 64 words. The bootload program is loaded into core storage from the diodes addresses relative to the bootload and GIOC base addresses, after which the GIOC sets an interrupt cell in the system controller containing the GIOC base address. The bootload program is executed by the control processor for that system controller. The program sets up control words and initiates input on the selected channel (A or B). When the input terminates the bootload program checks status and transfers to the program just loaded.

10. PERIPHERAL AND TERMINAL EQUIPMENT

The external input/output equipment that can be connected to the GIOC through one or more of its channels has been broken into two broad categories: peripherals and terminal equipment.

Peripheral equipment includes the equipment local to the computer center that has a direct connection to the GIOC. Terminal equipment is the equipment that is generally remote from the computer center and is connected to the GIOC with communication lines using a data set.

PERIPHERAL EQUIPMENT

The peripheral equipment with a GE-645 system includes all the standard types used with a large computer system.

DISC STORAGE UNIT (DSU10F)

DISC STORAGE CONTROLLER (DSC11F)

DISC ELECTRONICS UNIT (DEU11F)

Card reader, card punch, line printer, paper tape reader and paper tape punch are available for paper input/output media. Magnetic storage devices include tape, disc, drum and cards. A switch unit is available standard for switching between multiple I/O devices and/or different GIOC's

Since the GIOC is designed on a modular basis, adapters to interface with a specific custom device can be designed as the need arises.

A description of some of the standard peripheral devices for the GE-645 system are included on the following pages.



CONFIGURATION

A subsystem consists of a controller, a disc electronics unit and from one to eight disc storage units. Each storage unit has 32 discs.

DATA FORMAT

Each disc has a positioner with 8 heads. The disc has 2 recording surfaces and 2 zones with 2 active tracks each. The positioner has 64 positions for a total of 512 tracks per disc. The inner zone tracks are divided into 21 data blocks and the outer zone tracks have 43 data blocks each. A block stores 192 characters (6 bits). A single disc can store 3,145,728 characters.

SPEEDS

Data Transfer rates per channel (average)

- 165,120 characters/sec. — outer zone
- 80,640 characters/sec. — inner zone
- 122,880 characters/sec. — track pair average.

File latency time (one revolution)

52 milliseconds

Access (seek) time

- 95 milliseconds — minimum (adjacent track)
- 174 milliseconds — average
- 248 milliseconds — maximum.

CAPACITY Block

	<u>6 Bit Characters</u>	<u>36 Bit Words</u>
CAPACITY Block	192	32
Inner Track (21 Blocks)	4,032	672
Outer Track (43 Blocks)	8,256	1,376
Position (4 Inner Tracks + 4 Outer Tracks)	49,152	8,192
Disc (64 Positions)	3,145,728	524,288
Unit (32 Disc)	100,663,296	16,777,216
Subsystem (8 Units)	805,306,368	134,217,728

CHECKING

- Block Parity
- Invalid Compare
- Transfer timing
- Record field overflow
- Busy controller or unit
- Write lockout
- Invalid Address
- Invalid position or sector
- Internal error
- Illegal operation code
- Unit off-line

FEATURES

The Controller has 2 simultaneous channels. Each channel has 2 non-simultaneous inputs. Two simultaneous positioner seeks can occur in each disc unit, with only one actual data transfer per channel at a time.

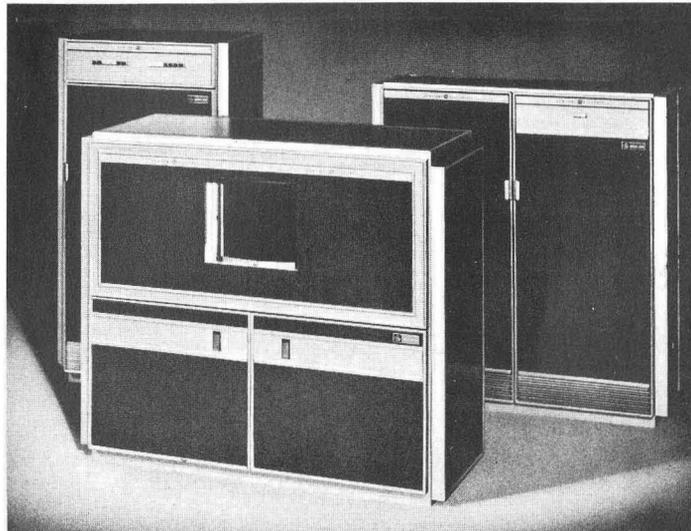
INTERFACE

The Disc Storage Controller (DSC11F) is a dual channel controller and uses a Direct Disc Adapter (DDA600), which is also dual channel, to interface with a GIOC. Two GIOCs can connect to one DSC11F.

DISC STORAGE UNIT (DSU204)

DISC STORAGE CONTROLLER (DSC200)

12 ADDITIONAL DISCS (OPT203)



CONFIGURATION

A subsystem consists of a controller and from one to four disc storage units. Each storage unit has up to 16 discs in groups of 4 discs.

DATA FORMAT

Each disc has 2 recording surfaces with 2 zones of 128 tracks each. The inner zone tracks are divided into 8 blocks and the outer zone tracks into 16 blocks each. A block stores 240 characters (6 bits + parity). A disc can store 1,474,560 characters.

SPEEDS

Data transfer rates (average)

41,700 Characters/second inner zone;
83,400 Characters/second outer zone;

File latency time (one revolution) 52 milliseconds. Access time:
225 ms average;

357 ms, maximum;

block on adjacent track 150 ms, maximum.

COMPATIBLES / 600

	<u>6 Bit Characters</u>	<u>36 Bit Words</u>
CAPACITY Block	240	40
Inner Track (8 Blocks)	1,920	320
Outer Track (16 Blocks)	3,840	640
Disc (256 Inner Tracks + 256 Outer Tracks)	1,474,560	245,760
Unit (16 Disc)	23,592,960	3,932,160
Subsystem (4 Units)	94,371,840	15,728,640

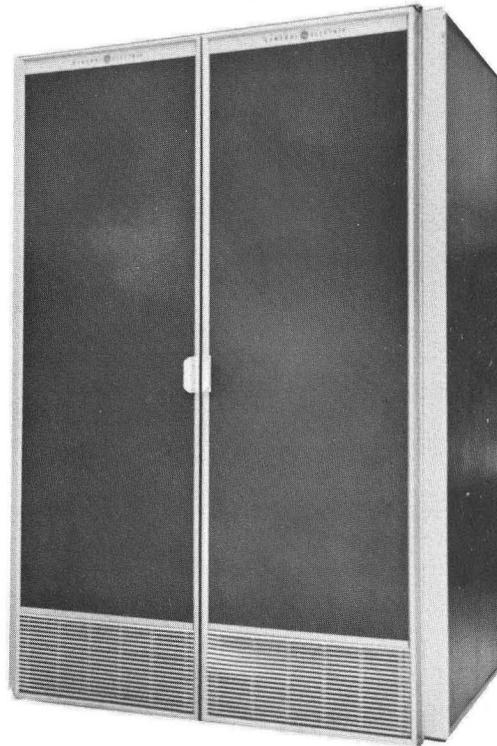
CHECKING

- Transfer timing
- Invalid control character
- Check character
- Buffer section committed
- Invalid device code
- Transmission parity
- Internal error
- Sector compare failure
- Invalid operation code
- Illegal buffer address

INTERFACE

The Disc Storage Controller (DSC200) is a single channel device and uses a High Performance Peripheral Adapter (HPC600) to interface with the GIOC.

MAGNETIC DRUM STORAGE UNIT (MDU200)
 MAGNETIC DRUM CONTROLLER (MDC201)
 ADDITIONAL DRUM STORAGE UNIT (ADS201)



CONFIGURATION

A subsystem consists of a controller and one or two magnetic drum storage units.

DATA FORMAT

Each drum has 284 data bands with 2 tracks each. A data band has 32 data blocks of 384 characters (6 bits). A drum can store 4,718,592 characters.

SPEEDS

Data transfer rates (maximum) 360,000 characters per second.

Drum latency (one revolution) 34 milliseconds.

	<u>6 Bit Characters</u>	<u>36 Bit Words</u>
CAPACITY Block	384	64
Band (32 Blocks)	12,288	2,048
Drum (384 Bands)	4,718,592	786,432
Subsystem (2 Drums)	9,437,184	1,572,864

CHECKING

- Invalid device codes
- Invalid operation codes
- Transmission parity

COMPATIBLES / 600

Record check character

Transfer timing

Block count

Writing voltage

Power failure

Cooling failure

Invalid drum address

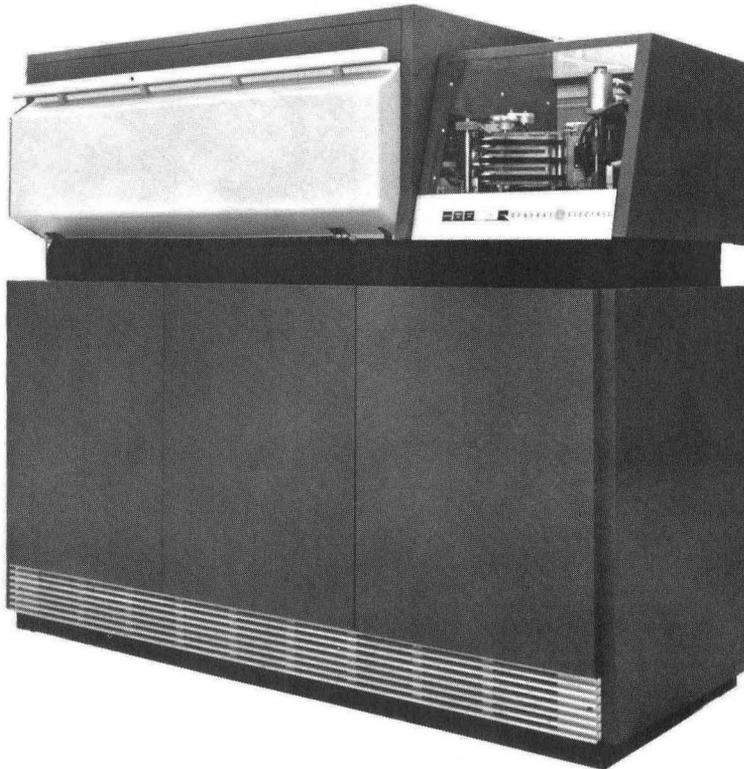
Proper drum rotation

INTERFACE

The Magnetic Drum Controller (MDC201) is a single channel device and uses a High Performance Peripheral Adapter (HPC600) to interface with the GIOC.

MASS STORAGE UNIT (MSU388)

MASS STORAGE CONTROLLER (MSC388)



CONFIGURATION

A subsystem consists of a controller and from one to four mass storage units. A storage unit has 8 removable magazines of 256 magnetic cards.

DATA MEDIUM

Data is recorded on one side of 16" by 4½" flexible magnetic cards.

DATA FORMAT

Information is recorded serially on 128 tracks extending the length of each card. Tracks are paired into 64 bands each divided into 4 longitudinal blocks.

COMPATIBLES / 600

A block stores 648 characters, (6 bits plus parity).

A card has 165,888 characters (256 blocks).

SPEEDS

Maximum transfer rate is 80,000 characters per second.

Maximum drum latency (one revolution) 60 milliseconds.

Access time without preselect is 460 milliseconds (average).

Access time with preselect is 185 milliseconds (average).

OPERATIONAL MODES

On line reading or writing.

CHECKING

Automatic read after write parity check.

Full address verification before card entry onto the drum.

Parity check during reads, including correction of every single bit error.

	<u>6 Bit Characters</u>	<u>36 Bit Words</u>
CAPACITY Block	648	108
Card (256 Blocks)	165,888	27,648
Magazine (256 Cards)	42,467,328	7,077,888
Unit (8 Magazines)	339,738,624	56,623,104
Subsystem (4 Units)	1,358,954,496	226,492,416

Each magazine has approximately the same storage capacity as 2 reels of 2400 feet, 800 bpi, 7 track, magnetic tape.

FEATURES

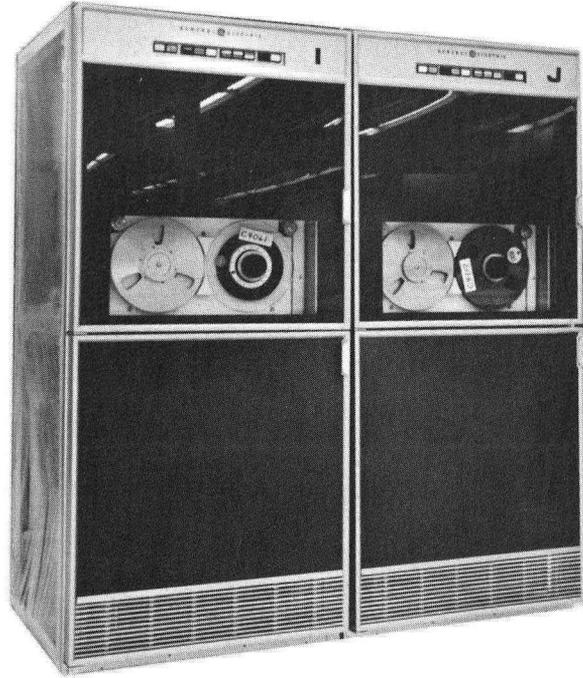
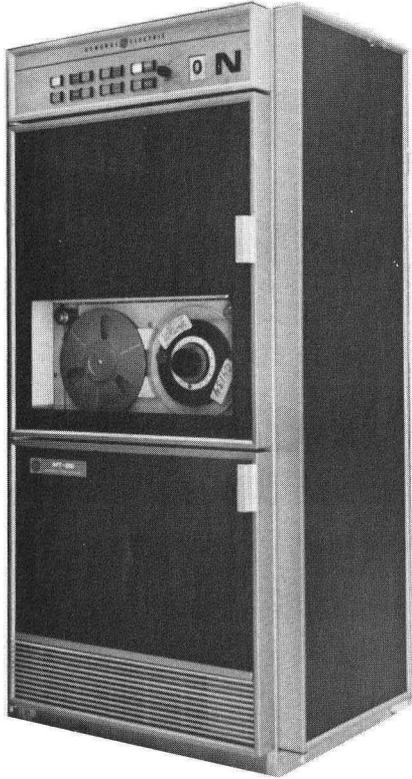
Simultaneous select and preselect operations on all units, therefore four simultaneous select operations on subsystems with four units.

A storage unit can read or write up to 256 blocks on one card by a single instruction.

INTERFACE

The Mass Storage Controller (MSC388) is a single channel device and uses a High Performance Peripheral Adapter (HPC600) to interface with the GIOC.

MAGNETIC TAPE SUBSYSTEM



A magnetic tape subsystem consists of one controller and one or more tape units. A GE-645 system may have more than one magnetic tape system.

DATE MEDIUM

Half-inch-wide, magnetic-oxide plastic tape, up to 2400 feet long, 7- or 9-track (compatible with American Standard Association codes).

DATA FORMATS

Binary (standard) and special decimal.

CHECKING

Transfer timing

Blank tape read

Transmission parity

Lateral parity

Missing character

Longitudinal parity

Bit detected during erase

FEATURES

Tape handlers are available in either 7 or 9 tracks, and operate with controller MTC404 and MTC400.

INTERFACE

Each channel of the magnetic tape controller requires a High Performance Peripheral Adapter (HPC600) to interface with a GIOC.

CONTROLLER CHARACTERISTICS

Controller Type No.	No. of I/O Channels	Maximum Number of Tape Handlers per Subsystem
MTC 404	2	16
MTC 400	1	8

Normally, the MTC 404 controller will have each of its two channels connected to a different GIOC.

CARD READER AND CONTROL (CRZ201)



DATA MEDIUM

80- or 51-column cards with upper left or right corners cut round or square. Score-edge cards can be read.

DATA FORMATS

Alphanumeric and column binary card code.

SPEED

900 cards per minute (80 columns)
1200 cards per minute (51 columns).

CHECKING

Card feed alert
Card synchronization
Read head alert
Card jam
Character validity (decimal mode)

COMPATIBLES / 600

TAPE HANDLER CHARACTERISTICS

Tape Handler Type No.	No. of Tracks	Tape Speed (ins. per sec.)		Recording Densities Available (bits per in.)	Data Transfer Rates for Various Densities (in thousands of characters per second)					
					8-bit characters			6-bit characters		
		Forwd.	Rewnd.		200	556	800	200	556	800
MTH412	9	150	300	200,556,800	30	83	120	40	111	160
MTH411	9	150	300	200,556	30	83		40	111	
MTH405	9	75	225	200,556,800	15	42	60	20	56	80
MTH404	9	75	225	200,556	15	42		20	56	
MTH311	7	150	300	200,556,800				30	83	120
MTH211	7	150	300	200,556				30	83	
MTH301	7	75	225	200,556,800				15	42	60
MTH201	7	75	225	200,556				15	42	

FEATURES

Hopper empty

Stacker full

2000-card hopper and stacker capacity.

Two output stackers, selected by the program.

Dual read heads: data read at two independent stations and compared.

Last batch control via LAST BATCH switch.

1000-card auxiliary stacker.

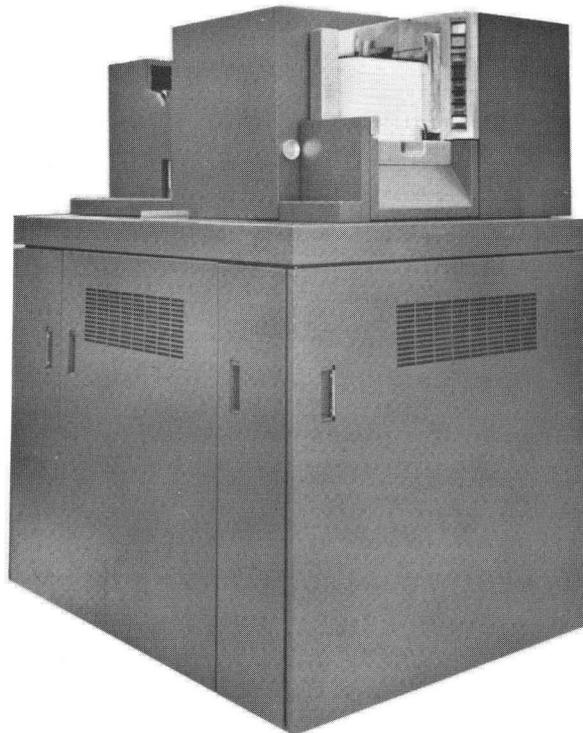
Continued operation while loading or removing cards.

Read alphanumeric and binary cards intermixed.

INTERFACE

An Indirect Peripheral Channel (IPC600) is used to interface the CRZ201 with the GIOC.

CARD PUNCH AND CONTROL (CPZ201)



DATA MEDIUM

Standard 80-column with round or square corners.

DATA FORMATS

Alphanumeric, edited alphanumeric and column-binary card code.

SPEED

300 cards per minute.

CHECKING

Card feed

Card synchronization

COMPATIBLES/600

Parity
Card jam
Hopper empty
Stacker full
Chad box not properly inserted
Chad box full
Read-after-punch
Auxiliary stacker full

1200-card stacker and 1200-card hopper capacities.

An auxiliary stacker of 100-card capacity. Cards are automatically directed to the auxiliary stacker when punch errors are detected.

Continued operation while loading or removing cards.

Extensive error monitoring for high-accuracy data transfers.

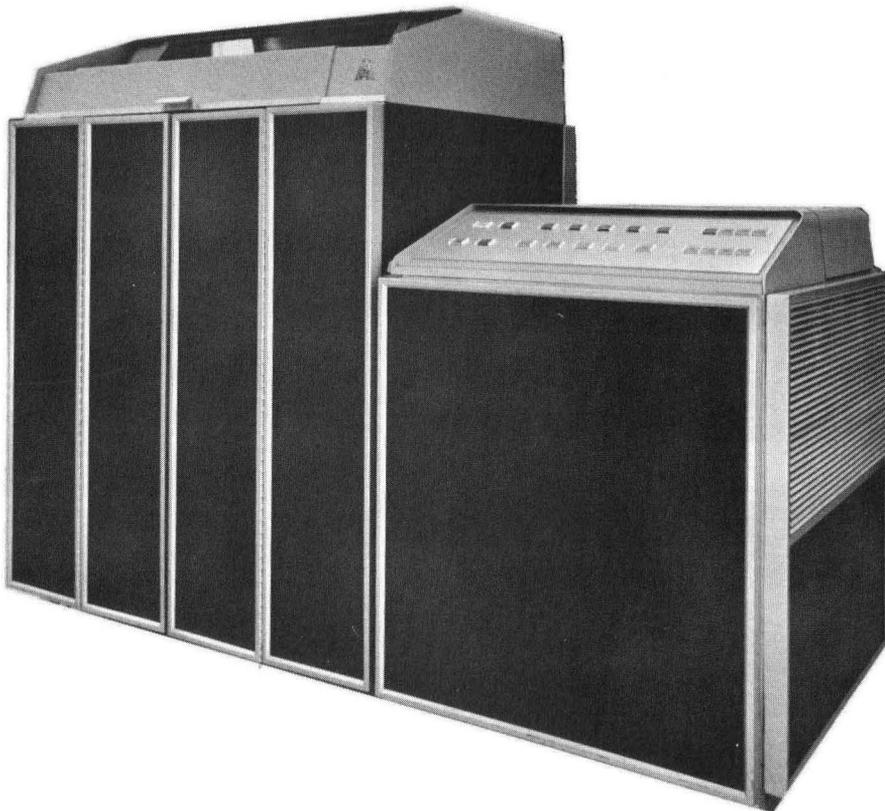
Automatic delay turnoff and halt.

An Indirect Peripheral Channel (IPC600) is used to interface the CPZ201 with the GIOC.

FEATURES

INTERFACE

ASCII EXTENDED CHARACTER SET PRINTER (PRT202)



COMPATIBLES / 600

DATA MEDIUM	Continuous forms, 3 to 19 inches wide, up to 22 inches long, 1 to 5 copies (original and 4 carbons).
DATA FORMATS	Standard ASCII printing character set. Vertical spacing is 6 lines per inch or 8 lines per inch and up to 136 characters per line.
SPEEDS	600 lpm, printing all 94 ASCII printable characters, 1200 lpm, printing a subset of 34 characters.
OPERATIONAL MODES	Edit mode allows column skipping and special slewing by count-down or VFU loop. Nonedit mode suppresses special editing functions.
CHECKING	Parity on input data Parity on VFU tape Low paper Out of paper Invalid operation code Channel busy Buffer overflow
FEATURES	Photoelectric sensing of the VFU tape for reliability. Switches for communication between operator and control program to position magnetically recorded input media. Separate VFU mechanism for each mode of vertical line density. Programmed control of slewing by VFU tape or countdown, includes top-of-page slew.
INTERFACE	An Indirect Peripheral Channel (IPC600) is used to interface the PRT202 printer with the GIOC.

PERFORATED TAPE SUBSYSTEM (PTS200)



DATA MEDIUM

Paper or Mylar polyester film laminate tape perforated with chad-type holes.

Reads and punches 5-, 6-, 7-, and 8-channel tapes, 10 characters per inch, in widths of 11/16, 7/8, and 1 inch. Recognizes all possible code combinations.

SPEED

Reader, 500 characters per second; punch, 150 characters per second.

OPERATIONAL MODES

On-line: punching fully controlled by the GIOC (reading controlled by the GIOC and the plugboard in the perforated tape subsystem).

Off-line: reading and punching controlled by the PTS200 Tape Reader/Punch control panel.

CHECKING

Output spool full

Tape breakage

Optional odd or even parity while reading

Odd parity check on all characters punched

FEATURES

Removable plugboard expands flexibility by providing ability to control input data format, check odd or even parity, delete specified characters, stop operation or indicate end of file upon detection of specified characters, and perform various logical functions.

COMPATIBLES / 600

Extensive error monitoring for high-accuracy data transfers.

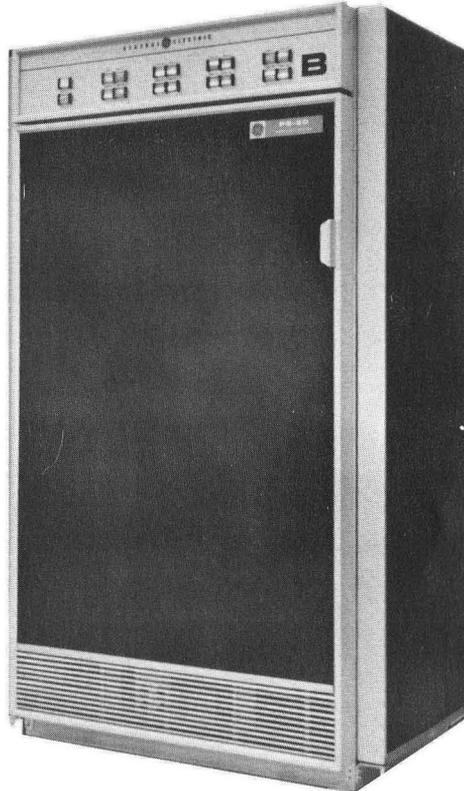
Photoelectric reading mechanism for accurate, reliable reading without tape wear.

Perforated Tape Reader (PTR200) and Perforated Tape Punch (PTP200) which make up the PTS200 can be installed separately.

INTERFACE

An Indirect Peripheral Channel (IPC600) is used to interface the PTS200 with the GIOC.

PERIPHERAL SWITCH CONSOLE (PSC200)



This peripheral switch console can switch peripherals and GIOC channels. This permits easy and rapid configuration of systems and permits convenient equipment substitution for maintenance purposes. The subsystem consists of a console which contains operator switches and from 1 to 16 switch units. Each unit can be used to connect a single peripheral subsystem to either of two GIOC's of either of two subsystems to one GIOC. Units are switched by the operator. Buttons on the control and indicator panel of the console control the switch units. A peripheral switch can be inserted in any peripheral line that connects to a High Performance Peripheral Adapter (HPC600) or an Indirect Peripheral Channel (IPC600) in the GIOC.

TERMINAL EQUIPMENT

Since the GE-645 System is designed to provide an information processing service capability, many of the users are remote from the actual computer center. To provide service at these remote locations the GIOC has adapters that interface with several types of data sets used for data transmission over common carrier communication lines (e.g., telephone or telegraph lines). Then by adapting a particular remote terminal to a compatible data set, data can be transferred between the remote device and the GE-645 System.

The Bell System Data Sets that can be connected to the GIOC and the associated adapter(s) used are shown in the following table.

COMPATIBLES / 600

Bell System Data Set	Type of Service	GIOC Adapters
103A, E	Switched Narrow Band	TTA600,CAA600
103F	Private Narrow Band	TTA600, CAA600
201A	Synchronous Voice Band	CSA600
202C, D	Asynchronous Voice Band	CAA600
801A, C	Automatic Call Unit	DGA600

Two General Electric terminal devices that can be used with the GE-645 System are the GE-115 Information Processing System and the DATANET-760* Keyboard/Display Subsystem. These are described in the following pages.

GE-115 INFORMATION PROCESSING SYSTEM



The GE-115 is ideal as a remote input/output terminal connected by communication lines to the GE-645. It can be taken off line and will perform as a card processing system. As a remote terminal to a larger computer, it provides direct access to the larger system without the media conversion often required at peripheral processors.

The GE-115 communicates with a GE-645 using a synchronous voice band data set (such as the Bell System 201 Data Set) through the DATANET-10* communication controller. The GE-115 system can be designed to fit the individual customer's needs. Besides the central processor, the system can include a card reader, card punch, line printer, and other peripheral equipment.

*DATANET is a reg. trade mark of the General Electric Company.

COMPATIBLES / 600

CENTRAL PROCESSOR

Memory Unit – 8.0 microsecond magnetic core

4096, 8192, 12288 or 16384 characters (or octets) (8 bits + parity)

Parity check on each character

28 primary arithmetic, logical editing and transcoding instructions

CONTROL UNIT

This unit fetches and interprets instructions from memory. It establishes connections with input/output units specified in the instruction. The control panel permits manual intervention to guide system operation.

Arithmetic Unit – Performs decimal and binary addition and subtraction operations. Decimal operation can be performed on 16 digit decimals and binary operations can be performed on fields up to 16 octets long.

CARD EQUIPMENT

One reader and 2 punches are available.

	Card Reader CRZ100	Card Punch CPZ101	Card Punch CPZ103
Speed-Max. (Card/minute)	300	60-200	300
Speed-Remote (Card/minute)	125	85	85
Reading (Punching) Method	Serial	Serial	Parallel
Hopper Capacity	500	1500	1200
Stacker Capacity	500	1500	1200

Remote Speed indicates the minimum number of complete cards that can be read or punched while transferring the resulting data to or from the communication line.

Line Printers – Two line printers are available (PRT100 and PRT110).

Speed: 300 (PRT100) or 600 (PRT110) lines/minute

Speed-Remote: 95 lines/minute

Character positions: 104, 120 or 136

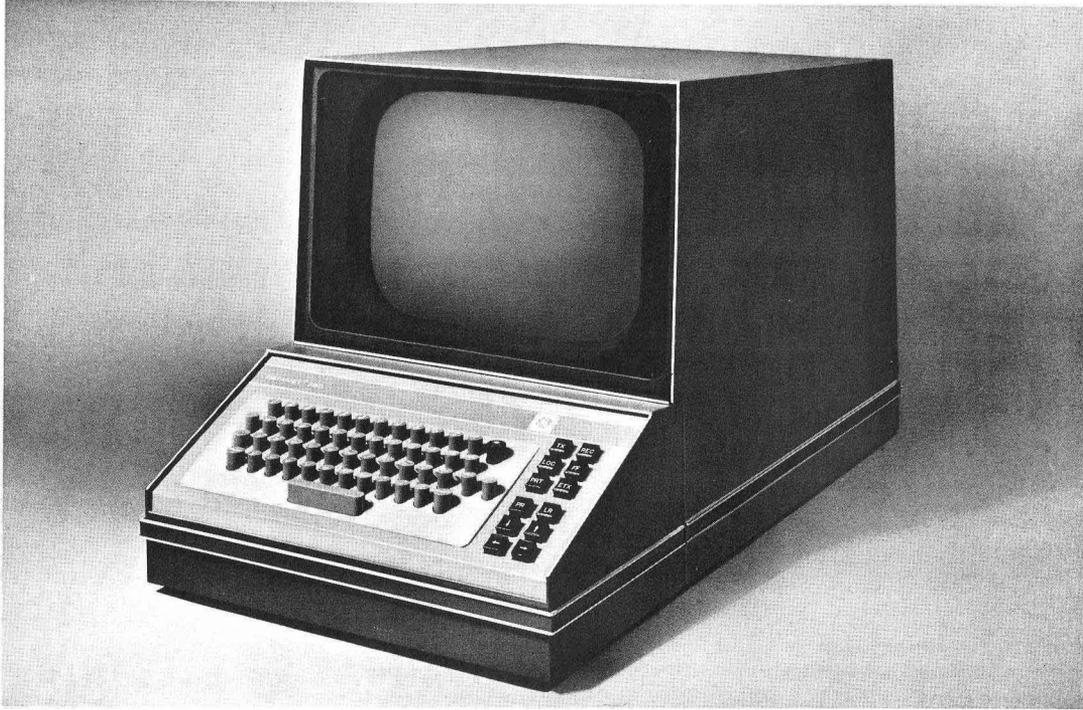
Characters: 64 standard General Electric Set

Spacing: 10 char/inch horizontal; 6 lines/inch vertical

Skipping: 12 (PRT100) or 60 (PRT110) inches/second

Form Width: 3 to 22 inches

DATANET-760* KEYBOARD/DISPLAY SUBSYSTEM



The DATANET-760* Keyboard/Display is an alphanumeric display system that provides rapid communication with computers from local or remote locations. It permits convenient entry and display of data or requests, transmission to the computer, and receipt, storage, and presentation of responses. The DATANET-760 consists of a Display Controller Unit and one or more TV-type Display Terminal Units. Up to 32 terminals, each of which may be at a remote location, may communicate with the GE-645 through the Display Controller Unit. Up to four page printers may be connected to each Display Controller Unit (in place of Display Terminal Units) to provide simultaneous hard copy of display data from any of the terminals.

The Display Controller Unit (DCU) consists of a Basic-Controller and up to four Terminal Memory Units, each of which can serve up to eight Display Terminal Units in simultaneous access. A Data Line Controller and multiple Page Print Controllers may be included on an optional basis.

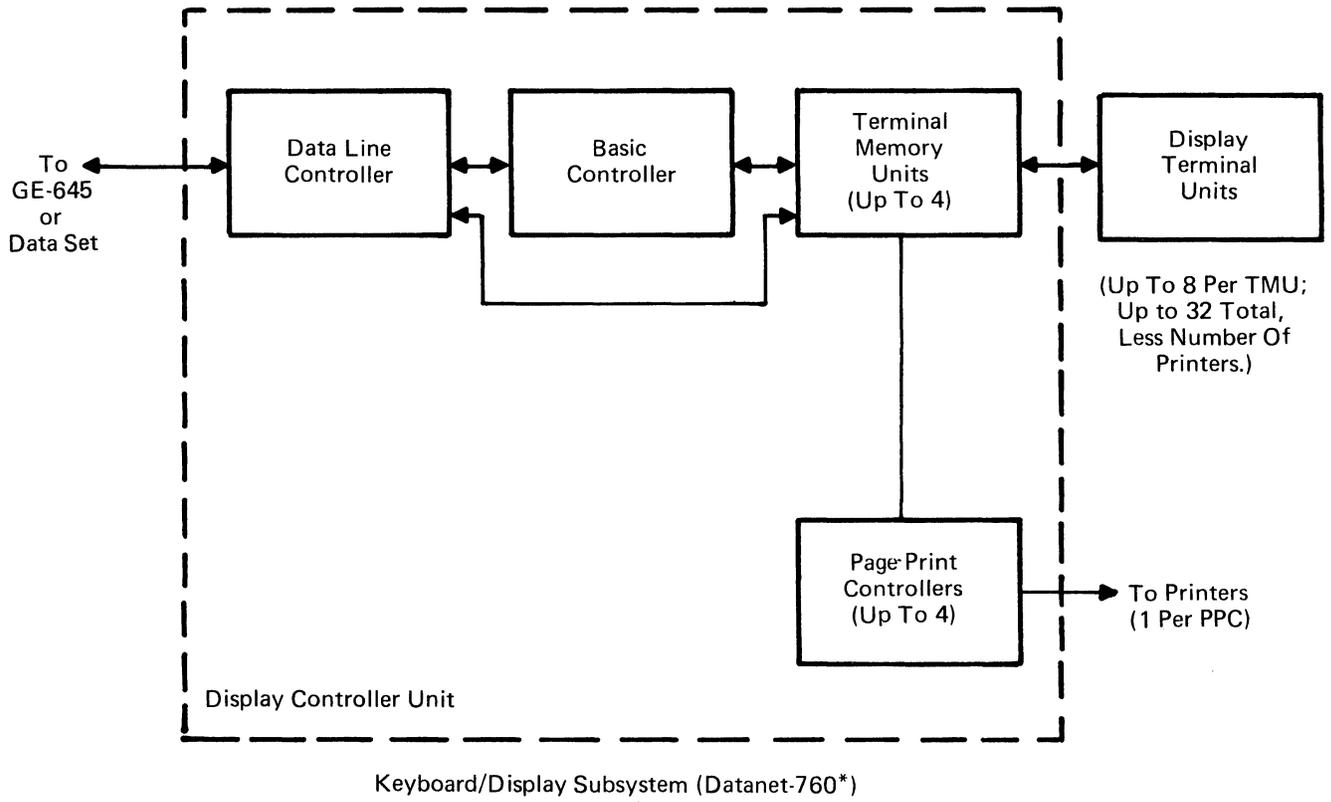
The Display Terminal Unit (DTU) consists of a standard industrial-quality TV monitor, supporting electronics, and a typewriter-like keyboard. Each DTU communicates with the GE-645 through the DCU via keyboard entries. Keyboard entries are converted to binary form and stored in the memory of the DCU. The coded characters in the memory are repetitively converted to TV video and, along with the synchronizing signals, are returned to the DTU. Selected portions of the stored information are transmitted on operator command to the computer, either by direct connection or by common carrier using Bell System 201 or 202 data sets. Only one line is needed for a Display Controller Unit.

The data display may be viewed at additional locations by coupling standard industrial-quality TV monitors to a Display Terminal Unit (DTU) or to the controller (DCU). The display can be projected for large-screen viewing by use of standard video projection equipment.

The presentation on the DTU is a fixed-format alpha-numeric display composed of up to 1196 characters and symbols stored in memory. The characters are arrayed in textual lines of 46 characters each. The number of lines in the display varies from 4 to 26 depending on the number of DTU's assigned to the Terminal Memory Unit in the DCU. The character repertoire consists of the alphabet, numbers, punctuation marks, and special symbols. Four of the special symbols allow horizontal and vertical lines to be drawn for added emphasis or generating simple diagrams, charts, or tables. In addition, a flashing code allows emergency or other important conditions to be emphasized.

*DATANET is a reg. trade mark of the General Electric Company.

COMPATIBLES / 600



11. MULTICS

The multi-user environment of the GE-645 will be controlled by the Multics operating system. The basic objectives governing the development of the operating system have been described briefly in chapters 2 and 3. In this chapter the general functional organization of Multics as it is being implemented, and some of its more detailed characteristics are explained.

CONCEPTS

USERS

Any person or any coding that calls a portion of Multics code is considered a "user" of Multics. Usually a user is a human user at a remote console. Other "users" in the true Multics sense are the Multics subsystems, modules, and procedures.

PROCESSES, SEGMENTS, AND FILES

The GE-645 storage hierarchy distinguishes between primary storage devices used in execution and secondary storage where data and instructions are kept before and after execution. In a similar way, Multics distinguishes between instructions in execution and when held in secondary storage.

The user may term his input to Multics according to his own conventions as a program, a job, or a run. However, once this input, consisting essentially of a sequence of actions, is in execution, it is known as a process. A process is defined as that sequence of actions running on and controlling a processor.

Processes are divided into linear arrays — ordered sequences of elements where each element is a machine word, character, or bit. These sequences of elements are called segments in primary storage. When held in secondary storage, segments are termed files. Each file must have a symbolic name.

Files are stored within the storage hierarchy according to various factors. In general, however, the particular device on which any file resides at a particular time is determined by the relative activity of the file. More active files are stored automatically on higher ranking devices than less active files.

PAGING

For execution, the Multics system may subdivide the segments of a process. These subdivisions made by Multics

are called hyperpages. Hyperpages consist of one or more hardware pages; thus each hyperpage is a multiple of 64 or 1024 words, depending upon the page size defined in the segment descriptor word. Users do not see the paging carried out by Multics; each user sees his program in execution as if it were the only program being run by the system.

Certain Multics segments must remain in core at all times for the system to operate properly. They are said to occupy wired-down core.

DISTRIBUTED OPERATING SYSTEM

In executing a user program, Multics combines the needed portions of the operating system — selected segments — with the user program segments in such a manner as to perform the task requested by the user. Thus a process in execution is neither a user program nor a collection of Multics system segments but a combination of both. This concept is called a distributed operating system, since system functions are distributed as needed within the process. A sophisticated Multics user can substitute his own system coding for Multics code in much of the system if he wishes. Further, the programs the user writes for execution become indistinguishable from system coding in Multics.

DYNAMIC LINKING

Since segments are brought into core memory from secondary storage only when needed in executing a process, the linkage of each symbolic reference between segments occurs the first time that the reference is encountered at execution time. Each segment of a program includes a linkage section which contains word pairs, called link pairs, representing the references to external segments from this segment. Before linking is performed for the first time, each link pair contains a modifier that causes a linkage fault (fault tag 2) when the external reference is initially made.

When a linkage fault occurs, a Multics module called the linker obtains pointer information from the linkage section and replaces the fault tag in the link pair with an ITS pair that points to the referenced location. The act of replacing the initial fault tag with an ITS pair is called linking. Once the linkage is initially created, any later external reference is handled without further calls to the linker.

PROTECTION OF PROCESSES

In any operating system and particularly in time-sharing, certain operating procedures are called many times by

different users. In traditional systems such shared procedures are usually protected from arbitrary modification by a user process because they are distinct from user coding and are executed separately. In Multics, operating procedures execute as part of the user process and are indistinguishable from it. Therefore, shared segments of Multics require a special means of protection against unauthorized access and modification.

Segments of Multics and of user processes exist and execute in a number of mutually exclusive subsets called rings. Conceptually these rings may be viewed as concentric with the inner containing those segments whose modification or destruction would harm all users and each outer ring containing segments whose importance to all users becomes less and less as we move outward from the innermost ring.

The outermost ring as initially seen by Multics, is that ring containing segments originated by users, i.e., any modification or destruction of segments in this ring affects one and only one user. However, a user may himself wish to protect some segments of his process from other segments of the process. To this end he may add additional rings of protection for segments of his process.

Within Multics, 3 concentric rings are distinguished. The innermost, called the hardcore ring or ring 0, is made up of those procedures common to all users and those procedures that interface directly with the hardware. Examples are the process exchange of the supervisor, the GIOC interface module of the I/O system, and procedures of the basic file system.

The next outer ring is called the administrative ring or ring 1. This ring contains most of the remaining segments of Multics. Since these procedures do not interface directly with the hardware they are somewhat less prone to cause a major system catastrophe if accessed in error.

The next ring is the user ring or ring containing segments of user programs and such Multics modules as the shell of the command system. Beyond this ring the user can add additional rings for segments of his programs that should be checked for valid access when calling inner rings.

As a process executes, it makes calls to other segments of the user code and to segments of the operating system. As long as the external references are confined to segments that exist within the same ring as that of the segment currently executing, no ring protection check is placed upon accessing the segments. For example, when the segment management module calls the search module to locate a file, no protection fault is generated since segments of both modules, the caller and the target, exist within the same ring. (This does not preclude protection within a given ring against an attempt to execute a data segment or access another user's procedure illegally. File system access control provides intra-ring protection.)

When a running procedure calls to a segment not in the same protection ring, a protection fault is generated. The fault interceptor then calls the gatekeeper module. The gatekeeper coordinates legality checking on the attempted reference. If satisfied that the called segment will permit

the reference, the called segment will begin to execute on the processor. In effect, the process is now executing in a different ring.

By analogy when a process calls a segment not in the ring in which it is currently executing, it rings a bell in the gate of a wall protecting the called segment. The gatekeeper answers, checks the credentials of the caller process, and if the reason for the call is valid, the gatekeeper permits the process to cross into the other ring. Therefore, the switching of execution of a process to a different ring is called wall crossing.

PURE PROCEDURES

The pure procedure contains no changeable data, is not altered during execution, and can be invoked by any user to perform its function on his own data.

All Multics Operating System procedures are pure procedures. Procedures can be shared by all Multics users. Other segments may be written that contain only data; Multics also accepts user-written segments that are partially procedure and partially data.

USER INTERACTION WITH MULTICS

This example discusses the principal steps that take place at the user at a terminal writes a program, compiles it, and executes it. The example is presented to help the reader understand the overall functioning of the major parts of Multics: (1) the supervisor, (2) the command system, (3) the file system, and (4) the input/output system. The use of these will be illustrated, and each will be described in the remaining sections of this chapter.

The user begins by dialing the number of the computer center. The supervisor reacts by logging in the user and building various segments that are needed by the operating system.

Now the user is ready to type in his program. To accomplish this he types in a command that tells the system he wants to build a file. The command system executes his command by giving control to an editing routine. He now types his program. As he types, the file system is placing his input in a file. The decisions of which pages to keep in core memory for his file are made by the file system. Control of the input/output hardware at the computer center is handled by the input/output system under the direction of the file system.

When the user finishes typing his program, he types a command that directs Multics to compile his program. The command system gives control to the compiler.

The compiler uses the file system to obtain the file of source language statements. The file system, and through it, the input/output system, is used to obtain additional parts of the compiler itself from secondary storage as the compilation proceeds. The principal output from the compilation is the object program. It will be in a file that is built as the compiler delivers its output to the file system. A message is typed when the compilation is done. The user

then types a command that directs the system to execute his program. (Assume that this program reads some data from the user's terminal, performs some calculations, and types the results on his terminal.) The command system interprets his execute command and, through the supervisor, causes his program to become a candidate for execution. When his program is placed in execution, it asks him for the input data. At this point, the program does not actually need a processor and the supervisor will assign the processor to some other user. While he is typing his data, the I/O system is taking care of the details of reading the input data.

When he is finished typing the input data, he indicates this to the system and an interrupt occurs. The supervisor receives control, analyzes the interrupt, and gives control to the input/output system. It verifies that the data was received satisfactorily and notifies the supervisor that the user's data is now available. The supervisor now gives control back to the program, and the program performs its calculations. The results are now ready to be sent to the user. His program calls on the input/output system to type the results.

To terminate his run, the user initiates the logout procedure. The supervisor summarizes resource utilization information for accounting purposes and takes the necessary steps to terminate the user's run. As time passes, and his program and data files are not used, the file system removes his files from memory to make room for active users. This example has merely touched on some of the functions performed by the various parts of the operating system. More information on the supervisor, command system, file system, and input/output system is supplied in the following sections of this chapter.

SUPERVISOR

The Multics supervisor acts in response to any information in the Multics system that affects the status of a process and/or processor. A process can exist in several states: running, ready, or blocked. A running process is one currently executing on a processor. A ready process is one that is not running but is held up awaiting availability of a processor. A blocked process is one awaiting an event (not necessarily imminent) in another process or in the external world, such as receipt of input from a device. The supervisor must not only keep track of what process is in what state and when to switch the state of a process, but it must also provide a method of communicating to a process the occurrence of significant events in other processes.

TRAFFIC CONTROL

Within the supervisor the set of procedures that make up the traffic controller perform the following major functions:

- A) Response to interrupts
- B) Response to faults
- C) Multiplexing of processors among processes.

All interrupts are passed directly to the interrupt interceptor module (IIM) by a transfer instruction in the processor interrupt vector. Interrupts are divided into two categories: system interrupts and process interrupts. System interrupts occur as a result of the reception of signals from I/O devices or from the calendar clock. These are signals from outside the Multics system and from outside the processor, directed to some process in the system (generally not the process running) and usually mean "start doing something to some process in the system". Process interrupts result from the reception of a signal from a processor and are directed to the process currently running. Process interrupts usually mean "change your execution state."

All faults are passed directly from the fault vector to the fault interceptor module (FIM). The FIM stores the processor state and calls the appropriate procedure to handle the fault. For example, "missing-page" and/or "missing segment" faults are handled by the basic file system which causes the segment or page to be read into core. After the fault is serviced, control is returned to the FIM which restores the machine conditions at the time of the occurrence of the fault.

Process exchange is the name applied to those procedures of the traffic controller that handle dispatching of processors among processes, scheduling of processes, and switching of processes. The process exchange is driven entirely by calls from other supervisor procedures, usually as a result of interrupts.

The basic hardware mechanism by which a processor switches from one process to another is the load descriptor segment base register instruction (LDBR). At the instant the descriptor segment base register (DBR) is reloaded, the processor sees the "core image" of a new process. However, the contents of the processor registers temporarily remain the same.

The process exchange maintains a "ready list" of all processes in the ready state in the order in which they are to be run. The ready list consists conceptually of pairs of entries; a process identification and a running time limit imposed by the scheduler.

The ready list is ordered and maintained by the scheduler procedure. In general, scheduler evaluates the process' request for priority in execution, comparing the request with present status of the ready list. Scheduler establishes a time limit from the process and places it in the ready list at its appropriate point. After a process is put on the ready list, it is run in turn on a first come-first served basis. However, if a process is known to have a high priority it can be given a favorable position on the ready list in accordance with its priority in relation to the priorities of the other processes on the list. Scheduler has the option of using a pre-emption interrupt to force a process to relinquish a processor to a process with a higher priority. When a process has exhausted its allotted time but has not completed execution, it is placed at the end of the ready list.

Processors are shared (multiplexed) among the processes. The traffic controller makes the multiplexed system appear to the user as if his process were the only one executing on

a processor at any given time. This technique shields the user from details of hardware management and handles the multiplexing of system resources among the users in such a way that the user need not be concerned with the problems of multiplexing processors. Included protection ensures that one user cannot affect another without prior agreement.

Since there are many processes and only a few processors, not all unblocked processes can be running. This is the reason for the existence of the ready state of a process. Intrinsic to the ready state in the "ready list" which lists all processes in the ready state as previously described.

The ready list contains the basic information to direct the dispatching of a processor when it is released by a process. A call to wake up a blocked process means "put it on the ready list". When a processor calls to block it means that the process is temporarily abandoning the processor upon which it is executing and that the first process on the ready list should be given control of the processor.

PROCESS CONTROL

Processes are allowed to be completely removed from core memory and to be subsequently returned to core without interfering with their programmed course. Process control is a general term applied to the means by which a process is brought into an active condition, loaded, unloaded, and returned to an inactive condition. Process control overlaps and interfaces closely with other supervisor modules and with the basic file system. In fact, many of the functions of process control are actually performed by the basic file system.

The various conditions of a process (i.e., active, inactive, etc.) are to be distinguished from the execution states (running, ready, blocked). The condition of a process is associated with the process control functions of the supervisor; the state is associated with the traffic controller function of the supervisor.

INTERPROCESS COMMUNICATION

To permit parallel processing, each user procedure and many Multics system modules execute as collections of separate processes, called process groups. Communication between processes is the function of interprocess communication.

The sending process places message information in a segment accessible to both itself and another process (the target process). The message may be either data, control information, or procedure. The target process reads the message information from the common segment. It is possible for the target process to suspend operation while awaiting a message. The sending process causes the target process to resume running when a message is in the common segment by calling wakeup.

The notion of an event is fundamental to interprocess communication. An event is anything recognized during the execution of one process that is of interest to another process. For example, the completion of the task of

collecting the characters of an input line from a typewriter is an event which might be recognized by a device manager process and be of interest to a working process. An event is a unique occurrence; it happens exactly once. If the device manager process of the example recognizes several successive line completions, each completion would be a separate event. The interprocess message is a signal from one process to another that an event has occurred.

RESOURCE MANAGEMENT

Since Multics is a system capable of meeting the computing needs of many varied users, it possesses an extensive resource-management facility which includes capabilities for measuring resource usage, charging for resources expended, regulating the use of resources, and evaluating the demands on the resources available. This function is performed by the modules that make up the accounting procedures in Multics; the accounting system is checked periodically by an auditing procedure. The accounting procedures are flexible in order to meet the needs of different installations.

Some resources of Multics are not easily shared among users. These resources are more effectively dedicated to a specific user for a given time period, despite the fact that the user does not make maximum use of the resource. The type of resources most often dedicated are detachable storage such as tape handlers and reels. However, resources which are normally shared may also be dedicated. For instance, a percentage of the capacity of a processor could be dedicated to a single user.

Dedicated resource management performs two functions:

- 1) the administrative function of reserving and allocating resources
- 2) the security function of protecting resources dedicated to one user from interference by another user.

Using dedicated resource management, it is possible to reserve I/O devices, the media for the devices, or a portion of a processor for a user and assign these resources at the specific time the user needs them.

COMMAND SYSTEM

Commands are calls to programs expressed in user-oriented language. Commands are often issued from the remote terminals of interactive users, but they may also be issued by other files for controlling batch processing applications.

Two essential parts of the command system are the "listener" and the "shell". The listener reads and controls the storage of messages arriving from remote terminals. The shell translates these console messages into calls for procedures. As its name implies, the shell is an interface between the user and the procedure being called.

The listener reads the message into a buffer. When it detects the end of the message, it calls the shell, using the buffer contents as arguments. The shell then translates the message into a call to a procedure. The shell breaks the

character string of the message into substrings according to the syntax for commands. It assumes the first substring to be a procedure name and interprets the remaining substrings as arguments (data) for that procedure. The shell converts the argument string to the form expected by the procedure named, creates a standard call for that procedure, and then transfers control to it. This procedure now has control until its execution is completed. Control is first returned to the shell, and then to the listener to receive the next message.

The shell can call any procedure for the user, providing the syntax of the command language has not been violated, and the procedure called is in a file which the user has permission to execute. The shell is itself a system subroutine, so that it may be called by any procedure. Therefore, it may be called recursively through any depth of nesting. This allows it to be used by any procedure to perform the standard terminal interface functions.

When a user types a command, he may want to include parameters which control the way a called program is executed. For example, a call to a compiler might contain a parameter to specify whether a listing file is to be created. Options of this type may be set permanently for a particular user or they may be set only for the duration of an interactive session.

Although the shell may call any procedure specified by the user, it is used mostly for calling system commands. A system command is a procedure available to all users. It is maintained by system programmers and appears in a special directory. This allows the user to call the utility procedures of the Multics system with minimum effort.

FILE SYSTEM

The file system is an integral part of a time-sharing or multiplexed system. One of the more stringent requirements of this environment is an on-line secondary storage complex in which all files are referred to symbolically (not by address) and in which movement of information is organized by the file system rather than the user.

Within Multics, a file is an ordered sequence of elements (with an element being a machine word, a character, or a bit). Files are created, modified, or deleted only through the file system, and at the level of the file system a file is formatless. All formatting is performed by Multics, or user supplied software modules outside of the file system. A file is known by its file name or names, and each file name is symbolic. File names can be common to a group of users, or a user can reference files through symbolic names known only to himself.

The Multics file system is divided into two parts:

- 1) The basic file system which manages files, moving segment pages into and out of core memory and to and from on-line secondary storage.

- 2) The multilevel and backup file system which moves infrequently used files downward to slower speed devices and which provides copies of all files on off-line devices, for retrieval and protection against destruction, either from user mishandling or hardware malfunction.

The file system is a memory system which gives the user and supervisor alike the illusion of maintaining a private set of segments or files of information for an indefinite period of time. This retention is handled by automatic mechanisms operated by the supervisor, independent of the complex of secondary storage devices of different capacity and access. Since files are referred to symbolically, the user is never aware of the movement of files through this complex.

Users' files can be completely private or may be accessed by other selected users. The file system allows files to be simultaneously read by automatically interlocking file writing. An interlock mechanism permits control over the degree of access allowed (e.g., a user may wish a file to be read but not written).

Files are of two basic types: directory files and non-directory files. Directory files have entries that point to and describe other files, both of the directory type and the non-directory type. Directory files are used to keep track of and provide information for referencing the files of the system. Non-directory files include all other files; pure procedure, data, linkage, etc.

FILE STRUCTURE

The structure of files in the file system provides the means for accessing secondary storage in a machine-independent and device-independent fashion. The user is aware only of symbolic addresses. All physical addressing is performed by the file system, unseen by the user.

The file structure can be looked at from three separate but interdependent views:

- 1) The directory tree, the logical file structure within Multics.
- 2) The storage hierarchy, the relative order of the various storage media accessible to the system.
- 3) The system skeleton, the basic organization of Multics.

DIRECTORY TREE

The logical organization of files in the Multics file system is a tree hierarchy. The root of the tree is a special type of file, maintained by the file system, known as a directory. A directory contains information which branches to (points to) other files. The files pointed to by branch entries can be directories themselves and contain branches to other files, which may or may not be directories.

STORAGE HIERARCHY

In most cases a user need not know how or where a file is stored. The user sees infinite storage capacity and all

files appear to be stored on-line. In general, only the file system knows which particular storage device holds a file, and it is the responsibility of the file system to insure that a file is available for processing when needed.

From a system viewpoint, there is primary storage and secondary storage for files. Primary storage is provided in the system controller modules, where file processing occurs. Secondary storage consists of the various file storage devices at the installation. The devices are ranked according to their relative speeds of access and transmission. Within secondary storage, devices are regarded as belonging to one of two systems: on-line and file backup.

The on-line storage system consists of those devices which are immediately accessible to the file system (e.g., extended memory unit, disc).

The file backup storage system consists of those devices with removable storage media (e.g., magnetic tape reels, magnetic cards, etc.).

SYSTEM SKELETON

The skeleton of the system, the information which supports Multics operation, is contained in the root directory of the tree hierarchy.

All directories reside in the hard core ring. Each of the components of the root directory have access restrictions associated with them which can differ from those for access to the root directory as an entity. In turn, the files pointed to by the root directory have their own access restrictions.

The root directory is made up of a number of individual directories. Each of these is a portion of the Multics system skeleton and points to files (or segments) containing the information necessary to perform Multics functions. Access to the root directory as an entity with the intention of writing is restricted to personnel at the Multics installation authorized to inspect and adjust accounting, billing, and personnel information.

BASIC FILE SYSTEM

The basic file system is concerned with the management of segments. It moves pages of segments into and out of core memory (primary storage) and to and from on-line secondary storage. It implements the hierarchical organization of segments into directories and includes a means for controlling the way in which a segment can be used. In conjunction with the multilevel and backup file system, it provides the user with complete independence from storage and maintenance consideration for files.

A user may reference data elements in a file explicitly through read and write statements, or implicitly by means of segment addressing. Although a file may sometimes be referenced as an input or output device, it can be referenced only through segment addressing. For example, a tape or teletypewriter cannot be referenced as a segment, and therefore cannot be regarded as a file by this definition.

Input and output requests directed to I/O devices other than files (e.g., tapes, teletypewriters, card readers) are

processed directly by a device interface module (DIM) which is designed to handle I/O requests for a given device. However, I/O requests which are directed to a file are processed by a procedure known as the file system interface module. This module acts as a DIM for files within the file system. Unlike other DIM's, the procedure does not explicitly issue I/O requests. Instead, the file system interface module accomplishes its I/O implicitly by means of segment addressing and by issuing declarative calls to the basic file system.

Whether a user references a file through the use of read and write statements or by means of segment addressing, ultimately a segment must be made available to his process. In managing segments, the basic file system:

- 1) Maintains the directories of existing segments (files).
- 2) Makes segments available to a process upon request.
- 3) Creates, truncates, and deletes segments.
- 4) Enforces the access control information of user's directories.

As a result of a user or user process referencing a file or a data element within a file, the data is made available by the basic file system through the following actions.

FILE RETRIEVAL FROM USER'S VANTAGE POINT

When a process refers to a segment for the first time, a linkage fault occurs. This initiates a search for the named segment in a per-user system data base, the segment name table (SNT). If the segment is not found, directories are searched to locate it and an entry is established for it in the SNT, and in another data base, the known segment table (KST). A segment with an entry in the KST is known to the process but not loaded. At this time, the file system determines the user's right to access the given segment. If the user receives access permission, a number is assigned to the segment and this information is passed back to the requesting process. No part of the segment has been brought into core, but it is now directly addressable.

Once a segment is known to a process, further reference to that segment causes a missing segment fault which initiates the following actions. The unique identifier for that segment is retrieved from the KST; then an entry is either found or created for the segment in the active segment table (AST) a system wide data base. A segment descriptor word is placed in the descriptor segment and a request to bring in pages of the segment is generated.

In order to do this, a page table must be created for that segment; this action may require that a page of core be removed to make room for the page table. The requested segment has yet to be brought into core memory, but now it has a page table.

As the original process tries further to complete its reference to the segment, a missing page fault occurs. After locating the page on secondary storage, a call to the appropriate device interface module (DIM) brings it into core memory. Loading this page in core may require the removal of some

other page belonging to this, or some other, process. This may involve copying a page onto secondary storage; therefore, a map of core is consulted to determine which pages can be removed. The request to remove a page is queued waiting on a DIM operation which may call the I/O system to perform the actual removal.

Since loading of a page, and possible copying of page onto secondary storage is time-consuming, the process generating the missing page fault is blocked until the page actually arrives in core. Then the process is awakened and final reference to the page completed. Future references to the same page will not usually cause a repetition of these steps.

FILE RETRIEVAL FROM SYSTEM'S VANTAGE POINT

All segments — either procedure or data — are referred to by a symbolic name. Only the basic file system knows of their physical locations in either memory or secondary storage. When a segment is first referenced, a module outside the file system — the linker module — gains control through a linkage fault.

The linker requests the segment management module to consult the SNT to see if it knows about the named segment. A segment name in a given process is a symbolic name by which that process may reference the segment and which corresponds to the name of an entry in the directory hierarchy.

If the SNT doesn't have an entry for the named segment, the segment management module calls the search module for advice on how to locate the segment. It then invokes directory control to manipulate directories and locate the segment. Once it is located, an entry is made for it in the SNT and control is passed back to the linker.

Figure 31 shows the interrelationships of software modules during the file retrieval process.

The solid lines indicate the flow of control through the use of formal calling sequences; arrows designate direction. The circles indicate some of the data bases (directories and various tables) used by the Multics system in its processing and dashed lines demarcate the file system from the rest of Multics.

The basic file system accomplishes its functions through the actions and interactions of these modules:

- 1) Segment control
- 2) Directory control
- 3) Access control
- 4) Page control
- 5) Core control
- 6) Device interface modules (DIMS)

Once the segment is located, segment control must establish the segment as known to the process by creating an entry for it in the KST, sorting a segment descriptor word in the descriptor segment and returning a segment number

to the linker. Before segment control can do this, access control must determine the user's right to access that segment. As the figure illustrates, segment control and directory control are the only file system modules which can be directly called by the user.

As the linker tries to make a reference to the segment, a missing segment fault takes it directly back to segment control. An entry for the segment is established in the AST in the manner described in the previous discussion; then page control is called to read in a page of the segment. It prepares the page table before returning control through segment control to the linker.

Page control is also invoked as a result of a missing page fault; this occurs when the calling process makes a further reference to the segment. Making use of the information kept in system segment tables (SST's), page control locates the page on secondary storage and calls the device interface module (DIM) for the appropriate device. All DIM's, except the extended memory unit DIM, interface with the I/O system at this point and must call the I/O system whenever they want a page transported to or from secondary storage. Core control is invoked to account for the availability and allocation of core space; it frees an area of core and the page is read in.

Once a segment is in core and is treated as part of a process the Process Exchange in the Multics supervisor monitors all functions in which it is involved.

MULTILEVEL AND BACKUP SYSTEM

The multilevel system is concerned with assigning files to storage devices, moving them to other devices, and providing storage space as required by files. The backup system provides for retrieval and protection of files against destruction by user mishandling or hardware malfunction. The backup system consists of a group of dump procedures that copy files periodically onto detachable storage and recovery procedures for restoring the most recent copies required. Procedures of the multilevel and backup system operate in the administrative ring while those of the basic file system operate in the hardcore ring.

MULTILEVEL STORAGE MANAGEMENT

The multilevel system controls the movement of files within the storage hierarchy by assigning the most active files to the faster devices and the least active to the slower devices. File activity is determined by a procedure of the multilevel system from information on the number of times the file has been accessed. This multilevel storage procedure is invoked each time a segment is activated, so that the procedure keeps an accurate account of access information.

Whenever a segment is to be moved, the multilevel system determines the target device on the basis of the segment's user access history.

The multilevel system also provides a means whereby a storage device does not become overcrowded with inactive files assigned at a time when they were being accessed more frequently.

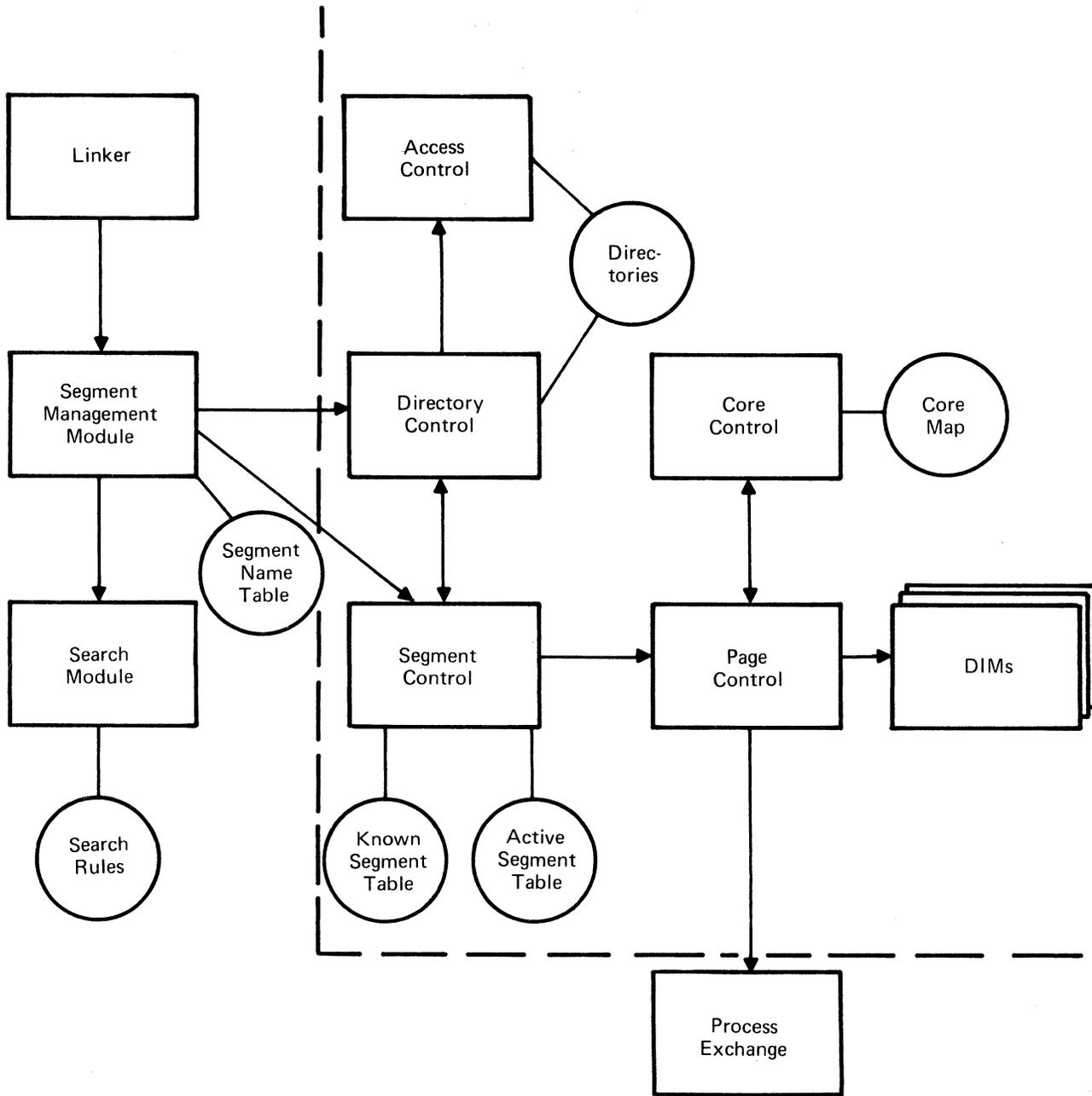


Figure 31. File Retrieval as Seen by the Basic File System

THE BACKUP FILE SYSTEM

The backup system protects all files against accidental destruction. A number of processes search up and down the file system hierarchy and copy all recently modified or recently created segments. Copies are made in duplicate on detachable storage which can be physically removed and saved. The standard backup processes enjoy freedom of the file system hierarchy; however, they can be denied access to segments that are highly secure.

Backup of the security information is effected by special processes with access to this type of information.

Copying file system data on backup storage is called dumping. There are three types of dumps within Multics; incremental, system-checkpoint, and user-checkpoint.

A set of procedures collectively called the incremental dumper operate constantly whenever the Multics system is functioning. It makes frequent passes over the file system

hierarchy, searching out segments and directory entries that have changed since the last dumping operation; then it directs the changed segments and directory entries to be dumped onto detachable storage. Incremental dumping produces a constant stream of output which represents all modifications to any part of the on-line storage system.

In the event of a catastrophe, the total reloading of all preserved incremental storage would be impractical. To minimize the time to recover after a catastrophe, periodic checkpoint measures are taken to reduce the total volume of backup storage. If on-line storage were destroyed, the following information would be required to restore the system to general use:

- 1) a set of necessary files used by components of the operating system

- 2) accounting files

- 3) a complete hierarchy skeleton (copies of all directory entries).

Periodic checkpoint dumping of this information provides recent versions of this data and allows an early return to normal system operation.

The user checkpoint dumper is a collection of procedures which run periodically, dumping segments used or modified since the last running time. Output of this type of dump is discarded as it is replaced by more current versions.

A comparison of dumper output is illustrated in the following chart.

DUMP	CONTENTS	TIME OF OCCURRENCE
Incremental	All segments and directories from time t_0 to time T.	Constantly during operation.
System Checkpoint	Just enough information to quickly restore the system to normal operation.	Determined by system strategy modules when essential files are changed.
User Checkpoint	Recently changed segments and directories.	Periodically; the time interval is set at system initialization time.

In the event of a catastrophe, file system information is reloaded into on-line storage from the detachable devices in the following order:

- 1) First, those segments and directory entries most recently dumped by incremental dumping are reloaded.
- 2) Second, the information dumped as a result of the most recent system checkpoint dump is reloaded.

At this point, normal operation of the system is recovered and the file system data can be reloaded under normal control of the Multics system. The third step to recovery from a catastrophe is to reload the remainder of the information dumped by the incremental dumper; and finally, the information accrued from the most recent user checkpoint dump is reloaded. During the recovery procedure, the data and time of file creation, or modification, is referred to by the backup file system which uses this information to ensure that only the most recent copy of the file remains in on-line storage.

It is often possible to salvage the contents of secondary storage without resorting to the reloading procedure outlined above. In this way the time required for recovery is reduced. The salvage procedure reads all the directories in the file system hierarchy and corrects information whenever possible. Since only directories and storage assignment tables are read, the salvage procedure can be run in a fraction of the time required to effect a complete reload.

Consolidation is effected whenever the amount of incremental backup storage becomes unwieldy. Segments dumped onto the area of storage allotted to them remain there for a limited time; then, they are removed from on-line to off-line storage by a removal mechanism. A transaction record which is accessible to the modules that perform consolidation ensures that although a segment no longer exists within secondary storage, it still remains known to the file system.

I/O SYSTEM

The I/O system contains the procedures by which processes communicate with external devices, and is designed to:

- A) interface user procedures with I/O devices in such a way that the user procedure is independent of any I/O device;
- B) provide a degree of modularity, consistent with operating efficiency, that allows new devices to be added with minimum coding effort by use of the I/O Table Compiler.

The I/O system works in conjunction with the GIOC to effect transfer of information to and from peripheral or terminal devices.

The I/O system maintains a list of connections between I/O names and physical devices. The I/O system takes the name from the read or write directive, locates the name in the connection list, and calls the appropriate modules within

the I/O system to perform the input/output. When the requested I/O has been initiated, the I/O system can call another module, go blocked awaiting a request, or return control to the calling process. The specific return action is based on information in the I/O system modules and on information in the connection list.

The Multics I/O system may be divided into two categories: hardcore I/O and device I/O. Hardcore I/O consists of the GIOC interface module (GIM). Device I/O comprises the remainder of the I/O system modules.

GIOC INTERFACE MODULE

The I/O routines that are most directly concerned with the manipulation of the hardware constitute the GIOC interface module (GIM). Calls to the GIM can cause it to list, define, alter, or connect, channels which interface the I/O system with the GIOC, and ultimately with the various devices. The GIM has the following functions:

- a) The GIM prepares lists of DCW's which, when decoded by the GIOC, cause the desired I/O function to be performed.
- b) The GIM associates particular GIOC channel numbers with the various devices.
- c) The GIM initiates activity on an I/O channel and passes status returns for that channel to the caller of the GIM.

The GIM is the most important module in the software interface to the GIOC. The GIM has control over all the facilities of the GIOC and, consequently, is located in the hardcore ring of the Multics system. The GIM is faced on one side with the hardware present in the GIOC and is faced on the other side with the software processes which use the hardware to control the amount and type of hardware activity.

Calls to the GIM are divided into two major categories which are:

- a) Calls to create the DCW, CIW lists
- b) Calls to use the DCW, CIW lists.

Calls in category (a) assign a data channel to a process, define the CIW to be used for that channel, create space for a list, and release space for a list. Calls in category (b) can change the contents of a list, activate a list, and process status words caused by active lists.

DEVICE I/O

In the next level of abstraction from the hardware, are routines called device interface modules (DIM's). For each device, there is at least one DIM. In some cases, there is more than one DIM for a given device. The modules at this level accept such calls as read, write, attach, and detach. The functions of the DIM's are:

- a) To provide a degree of uniformity in I/O calls to the various devices by generating pseudo DCW lists, indirectly through the GIM to the devices. These pseudo DCW lists are passed to a particular device in a particular sequence through the GIM, which changes them from symbolic to machine code and relays them to the GIOC mailbox areas. The meaning of a DCW list must be interpreted by the device itself.
- b) To provide for default error recovery and exception handling, such as retrying a write following a parity error (in writing tape) without user intervention.
- c) To perform code conversion when necessary.
- d) To perform read-ahead and write-behind for many of the devices. For example, the typewriter DIM can accept a user message before it has received a read request from the user's process.
- e) To perform (for devices such as tape) blocking of data into uniform physical record sizes, supplying an identifying prefix for each record.

For each type of device in the system there are usually two DIM modules: a device strategy module (DSM), and a device control module (DCM). Optionally, a third module may be provided, the code conversion module.

Many peripheral and/or terminal devices share common attributes and can be grouped into classes. Each device strategy module (DSM) in the I/O system is designed to function for a particular class of devices.

There is a device manager procedure (DMP) for each physical device. A DSM working process normally initiates I/O via a wakeup to the device manager process for the device in response to status information from GIM. The DMP may encompass one or more DCM's and can drive one, or many, devices.

The control of a particular device is the function of a device control module (DCM). The DCM converts the device capabilities assumed by the DSM into the capabilities of the device. This is accomplished by transforming DSM requests into physical I/O requests for the device.

APPENDICES

A. INSTRUCTION REPERTOIRE

The following table is a complete list of the instructions of the GE645 processor.

EXECUTION TIME NOTES

The listed execution times are the average time for a pair of instructions and are determined for the following general conditions:

- 1) The pair is preceded and followed by instructions of the same type.
- 2) Addresses are such that instruction fetch for the next instruction pair and operand fetches of the pair overlap.
- 3) Register (R) address modification is used. (No indirect addressing.)
- 4) All necessary descriptor words for segmentation and paging operations are available in the associative memory.

The average execution time for an instruction pair, is defined as the time interval between the start of address preparation for the even instruction of the pair, and the start of address preparation for the even instruction of the next pair. The average execution time then is one-half the execution time of the pair.

Some exceptions to the above definition of average exist.

- 1) Short load type instructions (i.e., LDA, LDAQ, ADA, ADAQ) require small enough processor operation time that overlap does not occur.
- 2) Store type (i.e., STA, STAQ, FST, ASA, etc.) instructions are preceded by a short load type instruction for timing purposes. Then the store type execution time is the pair execution time less the calculated short load type execution time.
- 3) For control type (i.e., TRA, TNC, XEC) instructions that fetch another instruction, timing is determined by the time interval between the start of address preparation for the control type instruction and the start of address preparation of the next instruction to be executed. This time is different depending on whether the next instruction is from an odd or even address so an average is used. Also the time is significantly less on conditional transfer instructions that do not transfer.

- 4) Base type instruction (LBRn, SBRn, EAPn, etc.) timing is different since they may effect the Associative Memory. The listed execution time is the time interval between the start of address preparation of one base type instruction and the start of address preparation of the next base type; plus, one-half the instruction address preparation time.

The execution time should be increased by the following factors for address modification, segmentation, and paging operations.

- 1) IR, RI modification: add 2.0 microseconds for each indirection and add 2.3 microseconds for each ITS or ITB indirection.
- 2) IT modification: add 2.0 microseconds if the contents of the indirect word are not changed and 2.7 microseconds if the contents are changed.
- 3) Segmentation and paging: add 1.6 microseconds for each SDW or PTW that is not already in the associative memory.

Instruction sequences containing several instruction types require additional time factors in order that the listed execution times can be used to calculate the actual execution time of the complete sequence. Some examples which need additional factors are listed below. Actual correction factors require a detailed knowledge of processor operation beyond the scope of this manual.

- 1) A transfer to or from a long load type instruction.
- 2) An instruction in location n which modifies instructions or registers used by instructions in locations n + 1, n + 2 or n + 3.
- 3) Entry into a fault or interrupt routine.

In the following table the following comments are concerned with the added description columns.

- 1) A "C" in the "Fault in Slave Mode" column indicates a conditional fault. The fault occurs if the referenced ABR is reserved for master mode only. (ABR bit 22 = 0).
- 2) Read - Alter - Rewrite (RAR) memory cycle instructions are identified because of their utilization in solving race problems between 2 processors.

DATA MOVEMENT

LOAD

		Average Execution Time (Microseconds)	n = 0 -7	Fault in Slave Mode	Special Instruction Format	Instruction not in GE-635	Use RAR Memory Cycle	Note No.
LDA	Load Accumulator	1.7						
LDQ	Load Quotient Register	1.7						
LDAQ	Load A-Q Register	1.9						
LDXn	Load Index Register n	1.7	x					
LDLXn	Load Index Register n from Lower	1.7	x					
LCA	Load Complement into Accumulator	1.7						
LCQ	Load Complement into Quotient Register	1.7						
LCAQ	Load Complement into A-Q Register	1.9						
LCXn	Load Complement into Index Register n	1.7	x					
FLD	Floating Load	1.7						
DFLD	Double-Precision Floating Load	1.9						
LDE	Load Exponent Register	1.7						
LDI	Load Indicator Register	1.7						
LDT	Load Timer Register	1.7		x				
LREG	Load Registers	6.3						
LDB	Load Address Base Registers	6.9				x		4
LBRn	Load Address Base Register n	3.0	x	C		x		
LDBR	Load Descriptor Base Register	3.0		x		x		
LDCF	Load Control Field	3.7		C		x		
LAM	Load Associative Memory	22.5		x		x		2
CAM	Clear Associative Memory	2.0		x		x		
EAA	Effective Address to Accumulator	1.5						
EAQ	Effective Address to Quotient Register	1.5						
EAXn	Effective Address to Index Register n	1.5	x					
EABn	Effective Address to Base Register n	2.0	x	C		x		
EAPn	Effective Address to Base Pair n	2.4	x	C		x		
LBAR	Load Base Address Register	2.0						1

STORE

STA	Store Accumulator	2.1						
STQ	Store Quotient Register	2.1						
STAQ	Store A-Q Register	2.6						
STXn	Store Index Register n	2.1	x					
STLXn	Store Index Register n in Lower	2.1	x					
STZ	Store Zero	2.4						

DATA MOVEMENT

STORE

		Average Execution Time (Microseconds)	n = 0 -7	Fault in Slave Mode	Special Instruction Format	Instruction not in GE-635	Use RAR Memory Cycle	Note No.
STCA	Store 6-bit Characters of Accumulator	2.1			x			
STCQ	Store 6-bit Characters of Quotient Register	2.1			x			
STBA	Store 9-bit Characters of Accumulator	2.1			x			
STBQ	Store 9-bit Characters of Quotient Register	2.1			x			
FST	Floating Store	2.1						
DFST	Double-Precision Floating Store	2.6						
STE	Store Exponent Register	2.1						
STI	Store Indicator Register	2.5						
STT	Store Timer Register	2.4						
STC1	Store Instruction Counter +1 and Indicators	2.5						
STC2	Store Instruction Counter +2	2.5						
SREG	Store Registers	8.6						
STB	Store Address Base Registers	8.3				x		
SBRn	Store Address Base Register n	2.2	x			x		
SDBR	Store Descriptor Base Register	2.2		x		x		
STPn	Store Pair in Address Base Register n	2.3	x			x		
SAM	Store Associative Memory	29.9		x		x		2
ZAM	Store Associative Memory Zero	2.8		x		x		2
STAC	Store Accumulator Conditional	3.7				x	x	
SCU	Store Control Unit	6.4				x		
SBAR	Store Base Address Register	2.0						1
STCD	Store Control Double	2.8				x		

ARITHMETIC INSTRUCTION

FIXED POINT ADDITION

		Average Execution Time (Microseconds)	n = 0 -7	Fault in Slave Mode	Special Instruction Format	Instruction not in GE-635	Use RAR Memory Cycle	Note No.
ADA	Add to Accumulator	1.7						
ADQ	Add to Quotient Register	1.7						
ADAQ	Add to A-Q Register	1.9						
ADXn	Add to Index Register n	1.7	x					
ADLA	Add Logical to Accumulator	1.7						
ADLQ	Add Logical to Quotient Register	1.7						
ADLAQ	Add Logical to A-Q Register	1.9						
ADLXn	Add Logical to Index Register n	1.7	x					
ASA	Add Stored to Accumulator	3.7					x	

ARITHMETIC INSTRUCTION

FIXED POINT ADDITION

ASQ	Add Stored to Quotient Register	3.7						
ASXn	Add Stored to Index Register n	3.7	x					x
AWCA	Add with Carry to Accumulator	1.7						
AWCQ	Add with Carry to Quotient Register	1.7						
ADL	Add Low to A-Q Register	1.7						
AOS	Add One to Storage	3.7						
ADBn	Add to Address Base Register n	3.3	x	C			x	

FIXED POINT SUBTRACTION

SDA	Subtract from Accumulator	1.7						
SBQ	Subtract from Quotient Register	1.7						
SBAQ	Subtract from A-Q Register	1.9						
SBXn	Subtract from Index Register n	1.7	x					
SBLA	Subtract Logical from Accumulator	1.7						
SBLQ	Subtract Logical from Quotient Register	1.7						
SBLAQ	Subtract Logical from A-Q Register	1.9						
SBLXn	Subtract Logical from Index Register n	1.7	x					
SSA	Subtract Stored from Accumulator	3.7						x
SSQ	Subtract Stored from Quotient Register	3.7						x
SSXn	Subtract Stored from Index Register n	3.7	x					x
SWCA	Subtract with Carry from Accumulator	1.7						
SWCQ	Subtract with Carry from Quotient Register	1.7						

FIXED POINT MULTIPLICATION AND DIVISION

MPY	Multiply Integer	7.1						
MPF	Multiply Fraction	7.1						
DIV	Divide Integer	14.1						
DVF	Divide Fraction	14.1						

FLOATING POINT ARITHMETIC INSTRUCTIONS

FAD	Floating Aid	2.0						
DFAD	Double-Precision Floating Aid	2.1						

Average Execution Time (Microseconds)	n = 0-7	Fault in Slave Mode	Special Instruction Format	Instruction not in GE-635	Use RAR Memory Cycle	Note No.
ASQ					x	
ASXn	x				x	
AWCA						
AWCQ						
ADL						
AOS						
ADBn	x	C		x		
SDA						
SBQ						
SBAQ						
SBXn	x					
SBLA						
SBLQ						
SBLAQ						
SBLXn	x					
SSA						x
SSQ						x
SSXn	x					x
SWCA						
SWCQ						
MPY						
MPF						
DIV						
DVF						
FAD						
DFAD						

<u>ARITHMETIC INSTRUCTION</u>		Average Execution Time (Microseconds)	n = 0 -7	Fault in Slave Mode	Special Instruction Format	Instruction not in GE-635	Use RAR Memory Cycle	Note No.
<u>FLOATING POINT ARITHMETIC INSTRUCTIONS</u>								
UFA	Unnormalized Floating Aid	2.0						
DUFA	Double-Precision Unnormalized Floating Aid	2.1						
ADE	Add to Exponent Register	1.7						
FSB	Floating Subtract	2.5						
DFSB	Double-Precision Floating Subtract	2.5						
UFS	Unnormalized Floating Subtract	2.5						
DUFS	Double-Precision Unnormalized Floating Subtract	2.5						
FMP	Floating Multiply	6.0						
DFMP	Double-Precision Floating Multiply	11.9						
UFM	Unnormalized Floating Multiply	5.8						
DUFM	Double-Precision Unnormalized Floating Multiply	11.6						
FDV	Floating Divide	14.5						
DFDV	Double-Precision Floating Divide	23.6						
FDI	Floating Divide Inverted	14.1						
DFDI	Double-Precision Floating Divide Inverted	23.2						
<u>MISCELLANEOUS ARITHMETIC INSTRUCTIONS</u>								
NEG	Negate Accumulator	1.5						
NEGL	Negate Long	1.5						
FNEG	Floating Negate	1.5						
FNO	Floating Normalize	1.5						
<u>LOGICAL INSTRUCTIONS</u>								
<u>AND</u>								
ANA	AND to Accumulator	1.7						
ANQ	AND to Quotient Register	1.7						
ANAQ	AND to A-Q Register	1.9						
ANX _n	AND to Index Register n	1.7	x					
ANSA	AND to Storage Accumulator	3.7					x	
ANSQ	AND to Storage Quotient Register	3.7					x	
ANSX _n	AND to Storage Index Register n	3.7	x					
<u>OR</u>								
ORA	OR to Accumulator	1.7						
ORQ	OR to Quotient Register	1.7						

LOGICAL INSTRUCTIONS

OR

Oraq	OR to A-Q Register	1.9					
ORXn	OR to Index Register n	1.7	x				
ORSA	OR to Storage Accumulator	3.7				x	
ORSQ	OR to Storage Quotient Register	3.7				x	
ORSXn	OR to Storage Index Register n	3.7	x			x	

EXCLUSIVE OR

ERA	EXCLUSIVE OR to Accumulator	1.7					
ERQ	EXCLUSIVE OR to Quotient Register	1.7					
ERAQ	EXCLUSIVE OR to A-Q Register	1.9					
ERXn	EXCLUSIVE OR to Index Register n	1.7	x				
ERSA	EXCLUSIVE OR to Storage Accumulator	3.7				x	
ERSQ	EXCLUSIVE OR to Storage Quotient Register	3.7				x	
ERSXn	EXCLUSIVE OR to Storage Index Register n	3.7	x			x	

COMPARISON INSTRUCTIONS

CMPA	Compare with Accumulator	1.7					
CMPQ	Compare with Quotient Register	1.7					
CMPAQ	Compare with A-Q Register	1.9					
CMPXn	Compare with Index Register n	1.7	x				
CANA	Comparative AND with Accumulator	1.7					
CANQ	Comparative AND with Quotient Register	1.7					
CANAQ	Comparative AND with A-Q Register	1.9					
CANXn	Comparative AND with Index Register n	1.7	x				
CNAA	Comparative NOT with Accumulator	1.7					
CNAQ	Comparative NOT with Quotient Register	1.7					
CNAAQ	Comparative NOT with A-Q Register	1.9					
CNAXn	Comparative NOT with Index Register n	1.7	x				
CMK	Compare Masked	1.7					
CMG	Compare Magnitude	1.7					
CWL	Compare with Limits	1.7					
SZN	Set Zero and Negative Indicators from Storage	1.7					
FCMP	Floating Compare	1.8					
FCMG	Floating Compare Magnitude	1.8					
DFCMP	Double-Precision Floating Compare	1.9					

Average Execution Time (Microseconds)	n = 0 - 7	Fault in Slave Mode	Special Instruction Format	Instruction not in GE-635	Use RAR Memory Cycle	Note No.
1.9						
1.7	x					
3.7					x	
3.7					x	
3.7	x				x	
1.7						
1.7						
1.9						
1.7	x					
3.7					x	
3.7					x	
3.7	x				x	
1.7						
1.7						
1.9						
1.7	x					
1.7						
1.7						
1.9						
1.7	x					
1.7						
1.7						
1.7						
1.8						
1.8						
1.9						

COMPARISON INSTRUCTIONS

		Average Execution Time (Microseconds)	n = 0 -7	Fault in Slave Mode	Special Instruction Format	Instruction not in GE-635	Use RAR Memory Cycle	Note No.
DFCMG	Double-Precision Floating Compare Magnitude	1.9						
FSZN	Floating Set Zero and Negative Indicators from Storage	1.7						

CONTROL INSTRUCTIONS

TRA	Transfer Unconditionally	3.0						
TZE	Transfer on Zero	3.0						3
TNZ	Transfer on Non-Zero	3.0						3
TPL	Transfer on Plus	3.0						3
TMI	Transfer on Minus	3.0						3
TRC	Transfer on Carry	3.0						3
TNC	Transfer on No Carry	3.0						3
TOV	Transfer on Overflow	3.0						3
TTF	Transfer on Tally Run out Indicator Off	3.0						3
TEO	Transfer on Exponent Overflow	3.0						3
TEV	Transfer on Exponent Underflow	3.0						3
TSS	Transfer and Set Slave	3.0		x				
TSXn	Transfer and Set Index Register n	3.0	x					
TSBn	Transfer and Set Base Address Register n	3.9	x	C		x		
RET	Return	3.7						
RTCD	Return Double	4.0				x		
RCU	Restore Control Unit	5.9		x		x		
MME	Master Mode Entry 1	2.0						
MME2	Master Mode Entry 2	2.0				x		
MME3	Master Mode Entry 3	2.0				x		
MME4	Master Mode Entry 4	2.0				x		
DRL	Derail	2.0						
XEC	Execute	2.0						
XED	Execute Double	2.0						

SHIFTING INSTRUCTIONS

ALS	Accumulator Left Shift	1.5						
ARS	Accumulator Right Shift	1.5						
ALR	Accumulator Left Rotate	1.5						
ARL	Accumulator Right Logical	1.5						
QLS	Quotient Register Left Shift	1.5						

SHIFTING INSTRUCTIONS

QRS	Quotient Register Right Shift
QLR	Quotient Register Left Rotate
QRL	Quotient Register Right Logic
LLS	Long Left Shift
LRS	Long Right Shift
LLR	Long Left Rotate
LRL	Long Right Logical

SPECIAL INSTRUCTIONS

CIOC	Connect I/O Channel
LACL	Load Alarm Clock
RCCL	Read Calendar Clock
SMIC	Set Memory Module Interrupt Cells
RMCM	Read Memory Module Mask Register
SMCM	Set Memory Module Mask Register
NOP	No Operation
RPT	Repeat
RPD	Repeat Double
RPL	Repeat Link
BCD	Binary to Binary Coded Decimal
GTB	Gray to Binary
DIS	Delay Until Interrupt Signal
RSW	Read Switches

Average Execution Time (Microseconds)	n = 0 -7	Fault in Slave Mode	Special Instruction Format	Instruction not in GE-635	Use RAR Memory Cycle	Note No.
1.5						
1.5						
1.5						
1.5						
1.5						
1.5						
1.5						
1.4		x				
2.6		x		x		
1.9				x		
2.0		x				
1.9		x				
4.0		x				
1.5						
1.4			x			
1.4			x			
1.4			x			
3.7						
9.9						
—		x				
1.4			x			

NOTE 1: Causes 635 Compatibility Fault.

NOTE 2: Execution Time shown is for absolute mode; add 3.5 microseconds for master mode.

NOTE 3: Execution times shown assume a transfer takes place. If no transfer, Execution Time is 1.6 microseconds.

NOTE 4: ABRn will not be loaded if ABRn(22) = 1 and slave mode set.

B. HARDWARE SUMMARY

In this appendix each modular hardware component of the GE-645 system is identified and described briefly.

PROCESSOR (CP8031)

The processor provides the program execution capability of the GE-645 system. It is a free-standing cabinet.

PROCESSOR PORT PAIR (CPP600)

One processor port pair for connection of the processor to two system controllers is basic equipment in each processor. Up to three additional processor port pairs can be added for connections to additional system controllers. All processor port pairs can be installed in the processor cabinet.

GENERALIZED INPUT/OUTPUT CONTROLLER (GIOC) (DC8031)

The GIOC controls input/output operations initiated by a processor. It is a free-standing cabinet.

GIOC PORT PAIR (MIP600)

One GIOC port pair for connection of the GIOC to two system controllers is basic equipment in each GIOC. Up to three additional GIOC port pairs can be added for connections to additional system controllers. All GIOC port pairs can be installed in the GIOC cabinet.

PRIORITY LEVEL GROUP (PLP600)

Two priority level groups are basic equipment in each GIOC. Each priority level group provides twelve priority levels for controlling the servicing of adapters and GIOC overhead channels. Up to six additional priority level groups can be added to meet priority requirements imposed by additional adapters. All priority level groups can be installed in the GIOC cabinet.

ADAPTER CABINET (CAB600)

One adapter cabinet to provide cabinet space for GIOC adapters is basic equipment with each GIOC. One or two additional adapter cabinets can be added to meet cabinet space requirements imposed by additional adapters. Each adapter cabinet is a free-standing cabinet.

INDIRECT PERIPHERAL ADAPTER (IPA600)

One indirect peripheral adapter, which provides the necessary control for up to six indirect peripheral channels, is basic equipment with each GIOC. Additional indirect peripheral adapters can be added to meet requirements for

additional indirect peripheral channels. All indirect peripheral adapters are for installation in adapter cabinets.

INDIRECT PERIPHERAL CHANNEL (IPC600)

Six indirect peripheral channels are basic equipment with each GIOC. Each indirect peripheral channel provides for the operation of one low-speed peripheral device, such as a card reader, card punch, printer, or perforated tape subsystem. Additional indirect peripheral channels can be added to meet requirements for multiple low-speed peripheral devices. All indirect peripheral channels are for installation in indirect peripheral adapters.

HIGH PERFORMANCE CHANNEL (HPC600)

One high performance channel is basic equipment with each GIOC. Each high performance channel provides for the operation of one high-speed peripheral subsystem, such as a magnetic tape, magnetic drum, or mass storage subsystem. Additional high performance channels can be added to meet requirements for additional high-speed peripheral subsystems. All high performance channels are for installation in adapter cabinets.

DIRECT DISC ADAPTER (DDA600)

The direct disc adapter provides for the 2 channel operation of the disc storage controller (DSC11F) and its associated disc storage units. More than one direct disc adapter can be used if there is a requirement for more than one disc storage controller. Direct disc adapters can be installed in adapter cabinets.

TELETYPEWRITER ADAPTER (TTA600)

The teletypewriter adapter provides the necessary control for up to four teletypewriter channel groups operating at the same speed. More than one teletypewriter adapter can be used if there is a requirement for additional teletypewriter channel groups or speeds. Teletypewriter adapters can be installed in adapter cabinets.

TELETYPEWRITER CHANNEL GROUP (TTC600)

Each teletypewriter channel group provides eight teletypewriter channels for half-duplex operation with telegraph grade lines. Teletypewriter channel groups can be installed in teletypewriter adapters.

TELETYPEWRITER CHANNEL EXTENSION (TTL600)

The teletypewriter channel extension provides additional capability to the channels of one teletypewriter channel group, allowing those channels to operate with data sets

such as Bell System 103A, 103E, and 103F. Teletypewriter channel extensions can be installed in teletypewriter channel groups.

TELETYPEWRITER SPEED OPTIONS

Each teletypewriter adapter requires one teletypewriter speed option to determine the speed at which the channels in that adapter are to operate. The teletypewriter speed option is installed in the teletypewriter adapter. The following speed options are available:

<u>Speed Option Type Number</u>	<u>Speed (bits/second)</u>
TTS605	45.5
TTS610	50
TTS615	56.9
TTS620	74.2
TTS625	110
TTS630	133.2
TTS635	150
TTS640	165

CHARACTER ASYNCHRONOUS ADAPTER (CAA600)

The character asynchronous adapter provides the necessary control for up to three character asynchronous channels operating at the same speed. More than one character asynchronous adapter can be used if there is a requirement for additional channels or speeds. Character asynchronous adapters can be installed in adapter cabinets.

CHARACTER ASYNCHRONOUS CHANNEL (CAC600)

Each character asynchronous channel provides capability for operation with data sets such as Bell System 202C and 202D. Character asynchronous channels can be installed in character asynchronous adapters.

CHARACTER SYNCHRONOUS ADAPTER (CSA600)

The character synchronous adapter provides the necessary control for up to three character synchronous channels. More than one character synchronous adapter can be used if there is a requirement for additional channels. Character synchronous adapters can be installed in adapter cabinets.

CHARACTER SYNCHRONOUS CHANNELS (CSC600)

Each character synchronous channel provides capability for operation with data sets such as Bell System 201A3 and 201B. Character synchronous channels can be installed in character synchronous adapters.

DIALING ADAPTER (DGA600)

The dialing adapter provides the necessary control for up to eight dialing channels. More than one dialing adapter can be used if there is a requirement for additional channels. Dialing adapters can be installed in adapter cabinets.

DIALING CHANNELS (DGC600)

Each dialing channel provides capability for operation with automatic call units such as Bell System 801A and 801C. Dialing channels can be installed in dialing adapters.

CUSTOM DIRECT ADAPTER (CDA600)

The custom direct adapter provides a direct channel for attaching a special customer interface. More than one custom direct adapter can be used if there is a requirement for additional channels. Custom direct adapters can be installed in adapter cabinets.

EXTENDED MEMORY CONTROLLER (EMC302)

The extended memory controller provides the necessary control for operation of one extended memory unit. The extended memory controlled is a free-standing cabinet.

EMC PORT PAIR (APP302)

One EMC port pair for connection of the extended memory controller to two system controllers is basic equipment in each extended memory controller. Up to three additional EMC port pairs can be added for connections to additional system controllers. All EMC port pairs can be installed in the extended memory controller.

EXTENDED MEMORY UNIT (EMU302)

One extended memory unit is basic equipment with each extended memory controller. The extended memory unit is a rotating storage device with a storage capacity of 4 million 36-bit words, and an average transfer rate of 470,000 words per second. The extended memory unit is a free-standing cabinet.

EMU COOLING UNIT (MCU302)

One EMU cooling unit, which provides necessary cooling for the extended memory unit, is normally required with each extended memory unit. The EMU cooling unit is a free-standing unit.

SYSTEM CONTROLLER (MM8040)

The system controller provides 32,768 36-bit words of core memory having a one microsecond cycle time, together with the means for allowing processors, GIOCs, and extended memory controller to send control information to each other. The system controller is a free-standing cabinet.

MEMORY PORT (OPT802)

Two memory ports for connection of the system controller to two modules, such as a processor, GIOC, or extended memory controller, are basic equipment in each system controller. Up to six additional memory ports can be added for connections to additional modules. All memory ports can be installed in the system controller.

INTERRUPT CELL GROUP (OPT815)

One interrupt cell group, consisting of 16 interrupt cells, is basic equipment in each system controller. The interrupt cells are used by processors, GIOCs, and extended memory controllers to cause program interrupts in a specific processor. One additional interrupt cell group can be installed in the system controller. Two interrupt cell groups, consisting of 32 interrupt cells, are required for Multics operation.

ADDITIONAL MEMORY MODULE (AMM600)

The additional memory module provides 32,768 36-bit words of core memory having a one microsecond cycle time. One additional memory module can be installed in each system controller. One additional memory module can also be installed in each auxiliary memory cabinet.

RECONFIGURATION OPTION (REC600)

The reconfiguration option provides the capability for reconfiguration of a system controller from a system configuration console. One reconfiguration option is required in each system controller if the system includes one or two system configuration consoles. The reconfiguration option can be installed in the system controller.

AUXILIARY MEMORY CABINET (CAB601)

The auxiliary memory cabinet provides cabinet space for additional core memory and for the calendar-alarm clocks. One auxiliary memory cabinet can be connected to each system controller. The auxiliary memory cabinet is a free-standing cabinet.

AUXILIARY MEMORY MODULE (AUM600)

The auxiliary memory module provides 32,768 36-bit words of core memory having a one microsecond cycle time for installation as the first memory module in an auxiliary memory cabinet.

CALENDAR-ALARM CLOCK (CLK600)

The calendar-alarm clock is a program readable calendar clock with a precision of one microsecond and a program settable alarm clock with an accuracy of 64 microseconds. One or two calendar-alarm clocks can be installed in an auxiliary memory cabinet.

MEMORY SWITCH (MSW600)

One memory switch is necessary in each auxiliary memory cabinet which contains both core memory and a calendar-alarm clock.

SYSTEM CONFIGURATION CONSOLE (SCC600)

The system configuration console provides a capability for semi-automatic system reconfiguration and startup. The system configuration console is a free-standing cabinet.

PERIPHERAL EQUIPMENT

CARD READER (CRZ201)

Each card reader connects to one indirect peripheral channel (IPC600), and is capable of reading 51- or 80-

column punched cards at the rate of 900 cards per minute. Two stackers are provided.

CARD PUNCH (CPZ201)

Each card punch connects to one indirect peripheral channel (IPC600), and is capable of punching 80-column cards at the rate of 300 cards per minute.

EXTENDED CHARACTER SET PRINTER (PRT202)

Each extended character set printer connects to one indirect peripheral channel (IPC600), and is capable of printing all 94 ASCII graphic characters in 136 columns at the rate of 600 lines per minute. Lines containing only the 34 most commonly used characters can be printed at the rate of 1200 lines per minute.

PERFORATED TAPE SUBSYSTEM (PTS200)

Each perforated tape subsystem connects to one indirect peripheral channel (IPC600), and is capable of reading at the rate of 500 characters per second and punching at the rate of 150 characters per second. The Perforated Tape Reader (PTR200) and the Perforated Tape Punch (PTP200) which make up the PTS200 can be obtained separately and used with an indirect peripheral channel.

AUXILIARY CONSOLE (CO8031)

Each auxiliary console connects to one indirect peripheral channel (IPC600) and provides an input/output typewriter. This console is used for operator communications with the software.

DUAL MAGNETIC TAPE CONTROLLER (MTC404)

Each dual magnetic tape controller connects to two high performance channels (HPC600) and provides for simultaneous access to any two of up to 16 magnetic tape handlers.

SINGLE MAGNETIC TAPE CONTROLLER (MTC400)

Each single magnetic tape controller connects to one high performance channel (HPC600) and provides for access to any one of up to eight magnetic tape handlers.

MAGNETIC TAPE HANDLERS

Any combination of the following magnetic tape handlers can be connected to a magnetic tape controller.

Tape Handler Type Number	Tracks	Speed Inches/sec.	Densities bits/inch
MTH201	7	75	200,556
MTH301	7	75	200,556,800
MTH211	7	150	200,556
MTH311	7	150	200,556,800
MTH404	9	75	200,556
MTH405	9	75	200,556,800
MTH411	9	150	200,556
MTH412	9	150	200,556,800

MAGNETIC DRUM CONTROLLER (MDC201)

Each magnetic drum controller connects to one high performance channel (HPC600) and provides the necessary control for operation of one magnetic drum unit and one additional drum unit. The magnetic drum controller is a free-standing cabinet.

MAGNETIC DRUM UNIT (MDU200)

One magnetic drum unit is basic equipment with each magnetic drum controller. The magnetic drum unit is a rotating storage device with a storage capacity of 786,000 36-bit words and a maximum transfer rate of 60,000 words per second. The magnetic drum unit is a free-standing cabinet.

ADDITIONAL DRUM UNIT (ADS201)

The additional drum unit is a rotating storage device with a storage capacity of 786,000 36-bit words and a maximum transfer rate of 60,000 words per second. The additional drum unit is a free-standing cabinet.

MASS STORAGE CONTROLLER (MSC388)

Each mass storage controller connects to one high performance channel (HPC600) and provides the necessary control for operation of up to four mass storage units. The mass storage controller is a free-standing cabinet.

MASS STORAGE UNIT (MSU388)

One to four mass storage units are connected to each mass storage controller. Each mass storage unit provides the capability of reading and writing information on flexible addressable magnetic cards which are stored in removable magazines. One mass storage unit with eight magazines has a storage capacity of more than 56 million 36-bit words. Information is transferred at the maximum rate of 13,333 words per second. Each mass storage unit is a free-standing cabinet.

PERIPHERAL SWITCH CONSOLE (PSC200)

Each peripheral switch console provides cabinet space for up to 16 peripheral switch units. The peripheral switch console is a free-standing cabinet.

PERIPHERAL SWITCH UNIT (OPT510)

One peripheral switch unit is basic equipment in each peripheral switch console. Up to 15 additional peripheral switch units can be added to each peripheral switch console. One peripheral switch unit provides the capability for manually switching a peripheral between either of two high performance channels or two indirect peripheral channels, or for manually switching a high performance channel or indirect peripheral channel between either of two peripherals.

DISK STORAGE CONTROLLER (DSC11F)

Each disc storage controller connects to one or two direct disc adapters (DDA600) and provides for simultaneous

access to 2 discs. Up to 8 disc storage units (DSU10F) can be accommodated by one disc storage controller. The disc storage controller is a free-standing cabinet.

DISC ELECTRONICS UNIT (DEU11F)

One disc electronics unit is basic equipment with each disc storage controller (DSC11F), and provides interface logic up to 8 disc storage units (DSU10F). The disc electronics unit is a free-standing cabinet.

DISC STORAGE UNIT (DSU10F)

The disc storage unit contains rotating disc storage devices with a storage capacity of 16 million 36-bit words and a transfer rate (averaged over all tracks) of 200,000 words/second per channel. Up to 8 disc storage units connect to a disc storage controller (DSC11F) through the disc electronics unit. Each disc storage unit is a free-standing cabinet.

DISC COMPRESSOR UNIT (DCU10F or DCU05F)

The disc compressor unit is basic equipment with the disc storage unit and supplies compressed air to disc storage units (DSU10F). The DCU05F can supply one, and the DCU10F can supply three disc storage units with air.

DISC STORAGE CONTROLLER (DSC200)

Each disc storage controller connects to one high performance channel (HPC600) and provides necessary control for the operation of from one to four disc storage units (DSU204). The disc storage controller is a free-standing cabinet.

DISC STORAGE UNIT (DSU204)

The disc storage unit contains 4 rotating disc storage devices with a total storage capacity of 983,000 36-bit words. The unit's capacity can be expanded to 3.9 million words by adding 12 additional discs. The transfer rate (averaged over all tracks) is 10,000 words/second. Up to four disc storage units connect to a disc storage controller (DSC200). Each disc storage unit is two free-standing cabinets (file and electronics).

12 ADDITIONAL DISCS (OPT203)

12 additional discs can be added to the disc storage unit (DSU204). These additional discs can be installed in the disc storage unit.

MOTOR-GENERATOR SETS

Motor-generator power sources are used for the major modules (processor, GIOC, EMM and system controller) to provide protection against line variation, momentary power outage and line noise. The motor-generator has a flywheel to carry through momentary power failures and provide sufficient power for an orderly shutdown in long power failures. Several 3-phase motor generator sets are available. The generator output on all units is 208V, 3-phase, 4 wire,

60 hertz. The motor input and generator power output are shown below.

Type Number	Line Freq. (hertz)	Line Voltage	Output Power (KVA)
MG8030	60	220/440	31.3
MG8031	60	440	62.5
MG8032	60	480	62.5
MG8033	50	380	62.5
MG8034	60	208	62.5

The motor-generator is a free-standing unit.

POWER SEQUENCER (OPT825 or OPT826)

A power sequencer is necessary to control each motor-generator set. The OPT825 unit controls a 60 hertz motor and the OPT826 unit controls a 50 hertz motor. Each power sequencer is a wall-mounted unit.

GENERAL  ELECTRIC