THE

RW-40

PRELIMINARY MANUAL

of

INFORMATION

FOREWORD

This manual provides a preliminary description of the Data Processing Central AN/FSQ-27 (XW-1) designed and developed for the Rome Air Development Center under contract AF 30(602)-1814. The information contained in this report describes the characteristics, capabilities, and equipments of the AN/FSQ-27 (XW-1),(RW-40),in a general manner. An appendix outlining the RW-40 instructions and their functions is included as a part of this report.

The RW-40 is an integrated data processing system specifically designed for high speed data handling and simultaneous execution of multiple problems. The RW-40 design features a form of modularity which permits the rapid shift of equipments to handle multiple problems, new problems, or a varying workload. This modularity also provides a capability for growth, so that as the tasks increase and greater capacity is needed, the data processing system can grow in suitable increments with minimum cost and disruption.

Capability for growth has been extended through the use of several identical computers working simultaneously in the system. These Computer Modules operate in an extremely flexible manner with the various modules used in the system. Different configurations of equipment can automatically be connected together through an electronic switching system to service many independent problems at the same time or to concentrate upon urgent or massive tasks. Thus the computing capacity can be allocated in a highly flexible manner to meet the specific data processing needs of the user.

Although each module used in the RW-40 Data Processing System has built into it the highest reliability consistent with the "State of the Art," multiplicity of modules also has important reliability implications. For example, continuous operation is possible even in the event of a computer malfunction since one computer failure does not render the entire system inoperative. Instead another computer module takes over the assignment and the system continues operation at a slightly reduced capacity during the maintenance interval. The use of identical modules also facilitates maintenance since the parts are identical and easily changed.

TABLE OF CONTENTS

TABLE OF CONTENTS (CONT'D)

RW-40 DATA PROCESSING SYSTEM

1.0 GENERAL DESCRIPTION

The RW-40 represents a most significant advancement in data processing concepts based on a fundamentally different approach to the data processing problem. This forward step manifests itself in a new and sophisticated organization of system components in which more than one Computer Module can be utilized simultaneously to perform a single application. For multiple applications, the multiple computer approach offers great advantages in speed and flexibility since the workload can be apportioned among the various computers. In the RW-40, this apportionment of the workload between Computer Modules takes place automatically through the use of stored programs.

The RW-40 Master-Slave concept enables any single computer module to perform the role of "master" computer, and automatically monitor, schedule, and control the other Computer Modules which are designated "slave" computers. This affords an efficiency in operations and equipment utilization never realized previously. These special techniques are achieved in the RW-40 because the Computer Modules are able to communicate with one another in a rapid and automatic fashion. The unique organization of modules provides advantages in higher speed operation, more efficient equipment utilization, and greater flexibility and reliability, even though the system cost is much lower than existing large-scale computing systems. Further, data processing capability can be added as required in the form of additional modules rather than entire systems, and thus added capability is obtained in the RW-40 with minimum cost. The RW-40 is the first data processing system to simultaneously employ a number of Computer Modules for executing single or multiple applications and, therefore represents the first truly modular data processing system.

The RW-40 arithmetic capacity can be increased by adding Computer Modules in the system configuration. These Computer Modules are high-speed general-purpose computers and function as the arithmetic and control units of the system. As mentioned previously, any one of these can be assigned the role of master computer and thus exert automatic control over system operations. Communication between Computer Modules is a vital requirement for a truly modular data processing system, and this is accomplished in the RW-40 by means of high-speed magnetic core transfer buffers (Buffer Modules), an electronic switching network (Central Exchange) and generalized sense and interrupt instructions.

The Buffer Modules are capable of independently gathering records from magnetic tape or drums. Each Buffer Module can load or unload 1024 full words of information for transmission to or receipt from any storage

device in the system, and this data transfer takes place without intervention by the Computer Modules. In fact, the Buffer Module may also serve as an extension of the Computer Module memory, since data stored in the Buffer Memory is accessible as rapidly as data stored in the computer internal memory.

Because the features employed in the RW-40 system are so different from those adopted in conventional systems, traditional system parameter comparisons based on add time, memory capacity, etc., are quite misleading. For example, an RW-40 configuration consisting of two Computer Modules, a Buffer Module, and a Drum Module, provides much greater capacity at much lower cost than the traditional large-scale computer consisting of a single large arithmetic unit and 32,000 words of core memory. This greater capacity is accomplished in spite of the fact that the RW-40 configuration provides only 12,000 words of storage (of which 8000 words are drum storage). The power of the RW-40 configuration arises from the efficient parallel arithmetic capacity and buffered use of the drum storage, or, in more general terms, from the unique organization and use of the modules.

Figure 1 is an illustration of an RW-40 System showing the interrelationship of system modules. Each system component can communicate with all other system components by means of the Central Exchange, which functions very much like a telephone switchboard and is under the control of both the Computer and Buffer Modules.

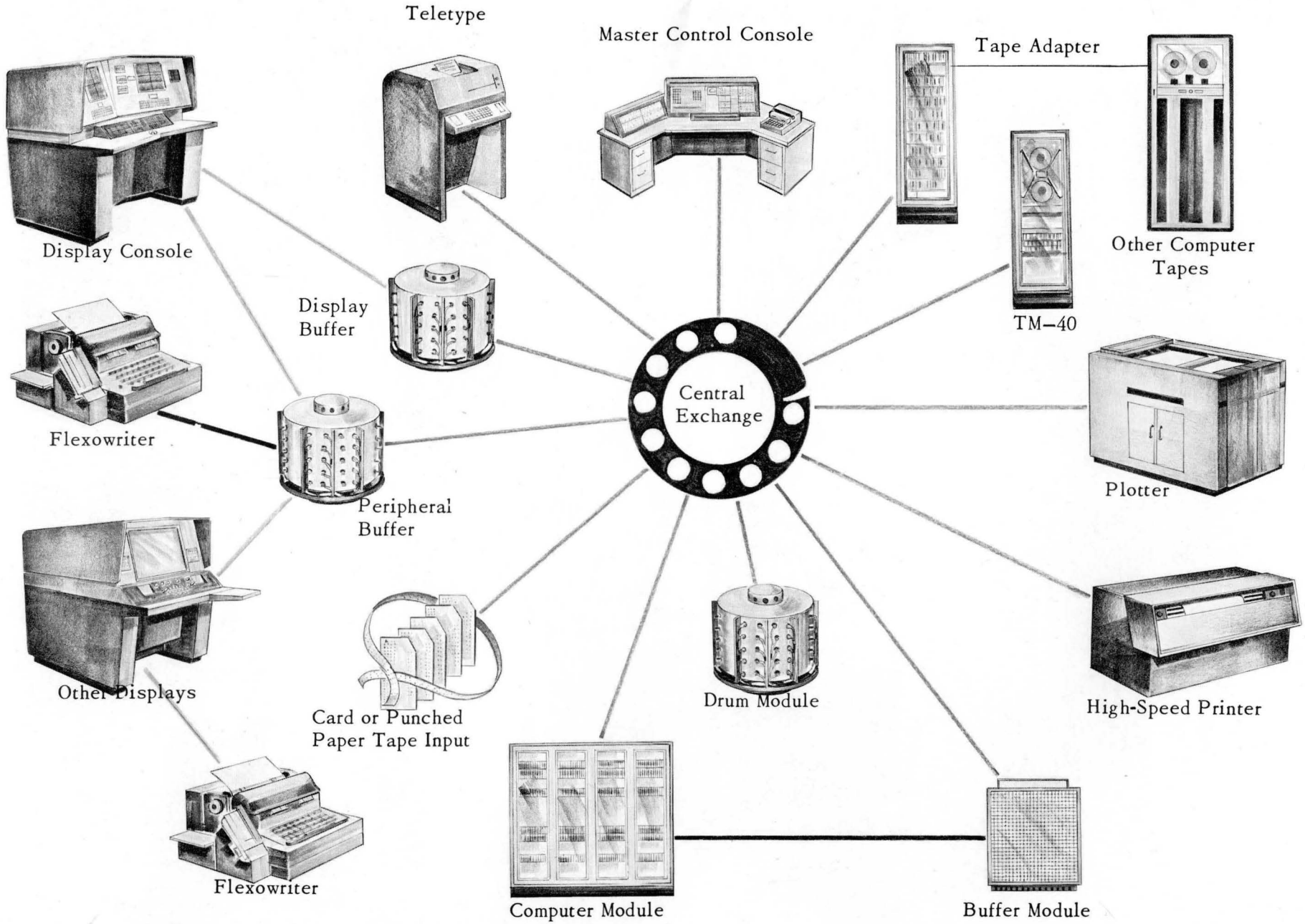


Figure 1. Typical RW-40 Organization

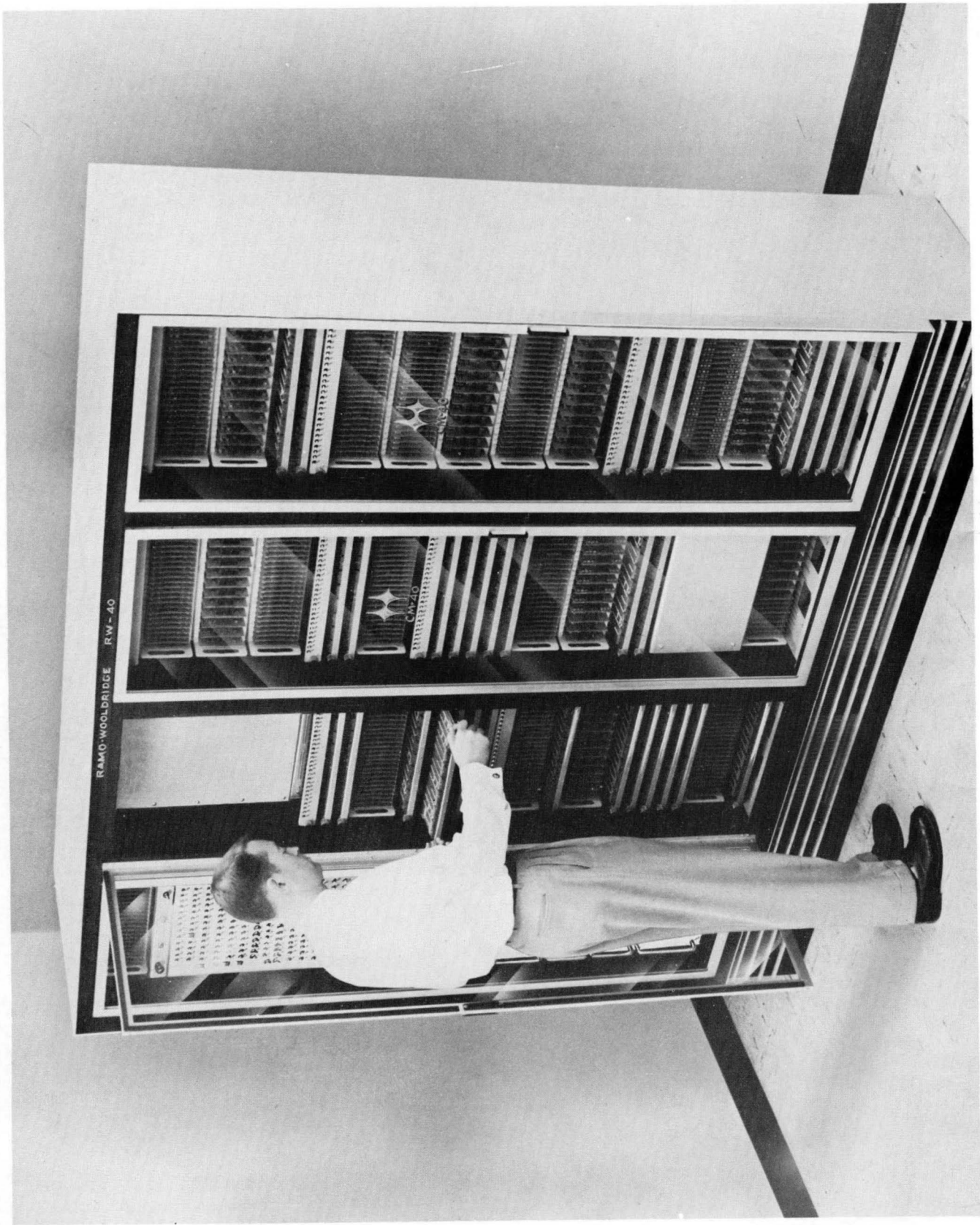


Figure 2. RW-40 Computer Module

2.0 EQUIPMENT DESCRIPTION

This section briefly describes the characteristics and capabilities of the equipment which comprises the RW-40 DATA PROCESSING SYSTEM.

COMPUTER MODULE, CM-40

The CM-40 (Fig. 2) is a high-speed general-purpose digital computer which is housed in a four-bay cabinet approximately 7 feet high, 8 feet wide, and two feet deep. It is designed to integrate into a system which contains other RW-40 system components such as Buffer Modules, Tape Modules, Drum Modules and other Computer Modules. The characteristics and capabilities of the CM-40 are described in the following paragraphs:

1. High-Speed Operations

There are 32 internal instructions for arithmetic and logical operations. These instructions are 26 bits long and have two 10-bit addresses and a 6-bit operation code. The two addresses are utilized in such a way as to be nearly as efficient as three-address instructions in most computations. A typical instruction time, including both memory accesses, is 40 microseconds. Five general external (input-output) instructions are provided.

2. Memory

The CM-40 has a 1024-word random access magnetic core memory. The read-write cycle time is 10 microseconds. Stored words are 28 bits long--26 bits of information and 2 parity bits.

3. Interrupt Capability

Automatic interrupt of a CM-40 is controlled by the masking action of an internal Sense Register which is under program control. Interrupts may be due to "master" computer intervention, alerting signals from external system modules, and internal conditions such as overflow. An interrupted CM-40 may be programmed to process the condition that caused the interrupt. It then returns to its normal sequence of operations at the point of interruption.

4. Switching Capability

A CM-40 may connect itself through Central Exchange to any available Buffer Module, Tape Module, Drum Module, Peripheral Buffer, or Display Buffer. Connections are made within the Central Exchange. All modules communicate over standard cables of 40 wires each.

5. Interrogation Capability

A CM-40 may test the status of any system module. Status information includes "ready" signals and locally detected parity errors.

6. Master Control Capability

Any CM-40 may be designated as a "master" computer to direct other system modules on a priority and availability basis. Designation of the master computer may be made by the human supervisor at the Master Control Console.

BUFFER MODULE, BM-40

The BM-40 (Fig. 3) is a high-speed magnetic core buffer designed to handle the parallel information format that is used by tapes and drums. It is capable of loading or unloading up to 1024 full words of information to or from any storage device without computer intervention.

It communicates with the computer either by a block transfer of information or by individual operands. Its purpose in the system is threefold: First, it provides a buffer between the computer, and magnetic tapes and drums. Second, it provides a versatile data handling device from which data processing complexes may be formed. Third, the Buffer Module augments computer internal storage as an extension of the random access memory providing automatic indirect address sequencing. In this mode the computer directly manipulates the buffer which serves as an external device for the computer.

The Buffer Module serves to improve computer access to files by providing off-line block transfer to and from Tape Modules and Drum Modules. The buffer can execute single instructions as provided by a computer, or it can execute a sequence of internally stored instructions. Self-instruction enables the buffer to communicate with tapes and drums without requiring continuous control by a computer. Self-instruction capabilities include positioning of tape, search for block numbers on tape, search for addresses on drums, and read or write tape or drums.

A Buffer Module consists of two independent buffers packaged in a three-bay cabinet. The pair of buffers in one cabinet may be used together to provide continuous data transfer between a Computer Module and a Tape Module.

DRUM MODULE, DM-40

The DM-40 (Fig. 4) is a one-bay cabinet containing a magnetic drum with a storage capacity of 8192 words. A DM-40 may communicate with any Computer Module or Buffer Module through the Central Exchange.

The DM-40 contains a word computer which is compared with the requested first word address specified by the instruction. The module which is connected to the DM-40 senses the word counter to effect the comparison. Average access time to the first word is 8-1/2 milliseconds. The information transferred, after encountering the first desired word, is all successive words up to the length specified by the

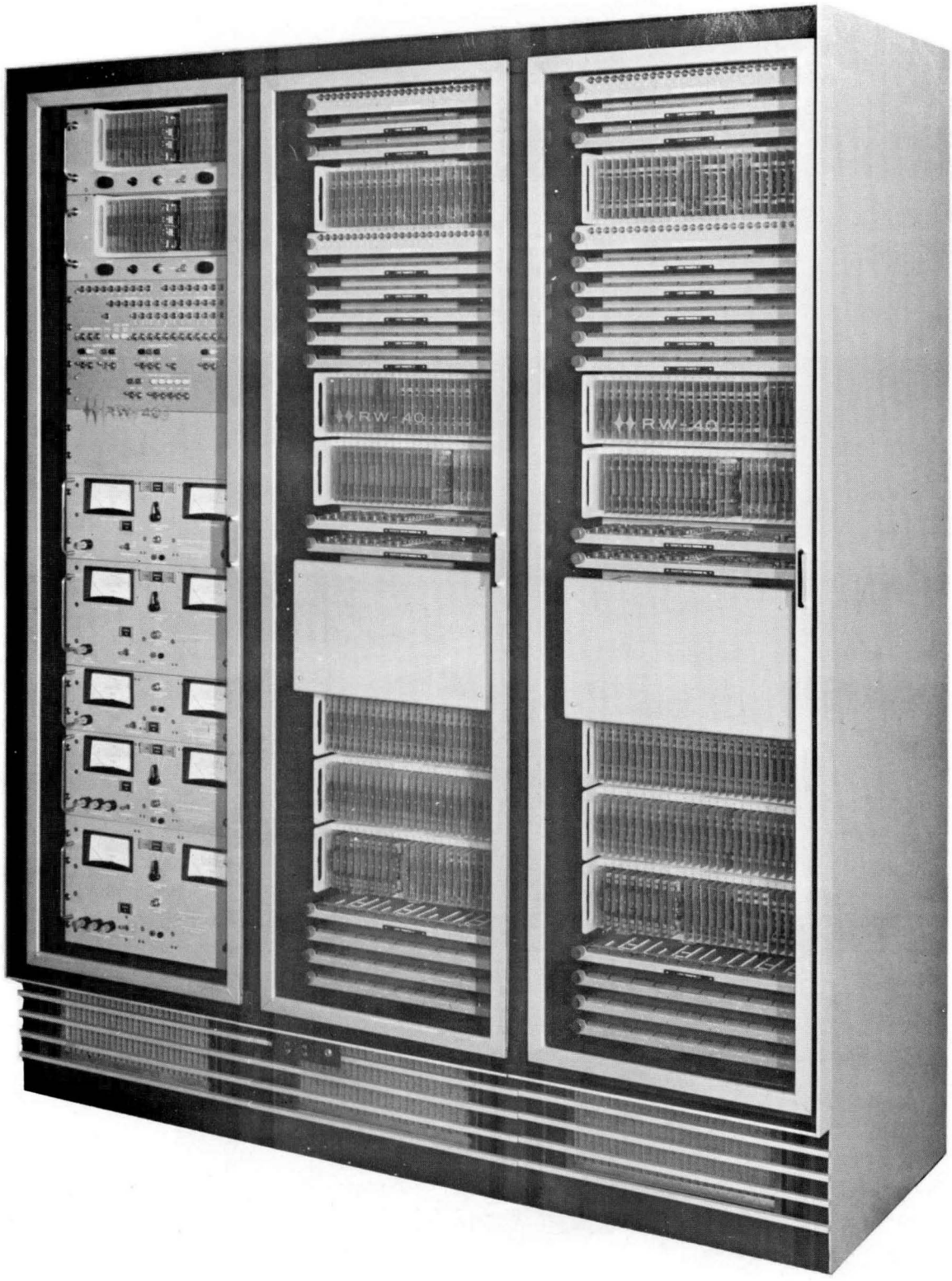


Figure 3. RW-40 Buffer Module

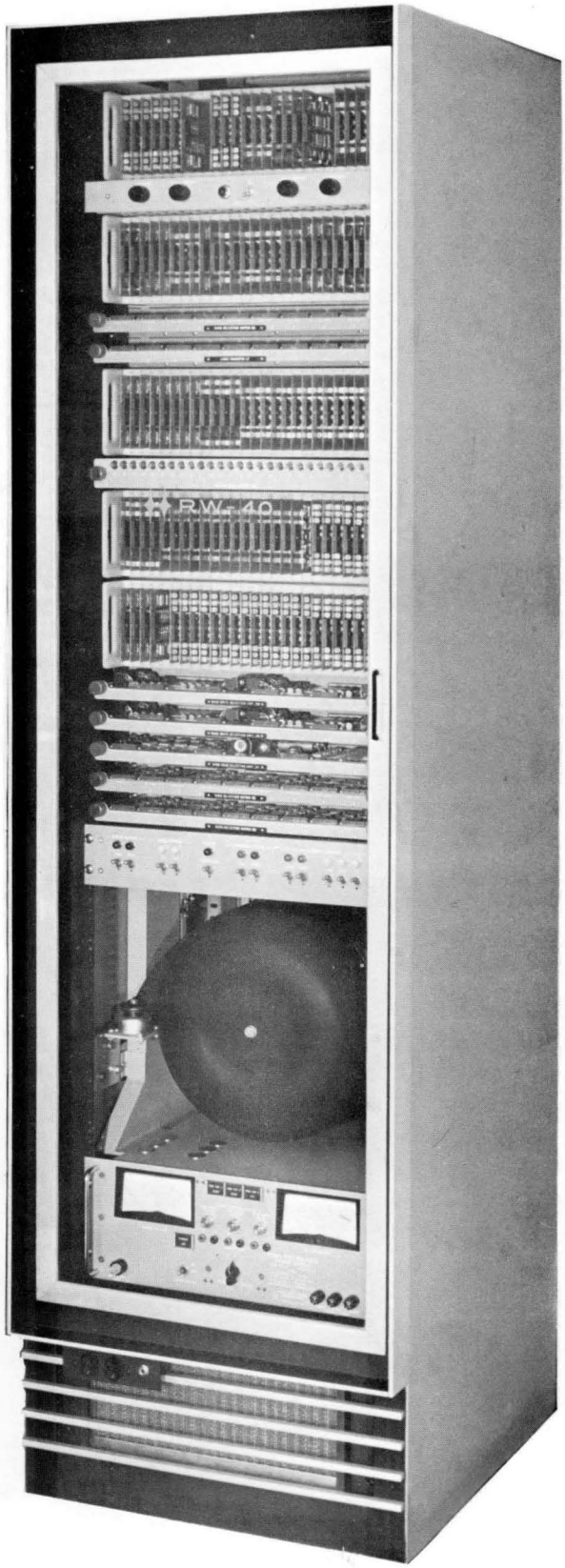


Figure 4. RW-40 Drum Module

instruction. Completion of transfer is detected by the connected module. The transmission rate is 60,000 full words per second.

Digital-to-analog and analog-to-digital conversion equipment of the type available with the RW-300 control computer can be readily incorporated for use with the Drum Module.

TAPE MODULE, TM-40

The TM-40 (Fig. 5) consists of an Ampex FR-300 tape transport, control power supply, logical and clock circuits, amplifiers, and power supplies. The transport and control power supply provide the tape handling capabilities, while the other circuits control and perform the storing and sensing of information onto and from the magnetic tape. The TM-40 is a one-bay cabinet. It communicates through the Central Exchange with either Computer Module or a Buffer Module. The communication includes both transport control lines and information lines.

One-inch wide mylar tape is used. Sixteen channels are available--2 are used for clocking purposes and the remaining 14 channels are used for a parallel half-word and associated parity bit. Tape speed is 150 inches per second, and the recording density is 200 bits per inch. The information rate is 15 full words per millisecond.

Blocks recorded on tape can be variable in length up to 1024 words. A full word-block number precedes the information in each block to permit selective reading. Single blocks imbedded in a tape file of other blocks can be overwritten by blocks of the same length. Information on either side of the overwritten block is unaffected.

Two head stacks permit automatic verification of each block as it is written. That is, readback parity errors are automatically detected during the writing process. This permits the avoidance of drop-out areas on a tape and the determination of excessive wear in the oxide coating while the data is still available in the computer for recording elsewhere.

FLEXOWRITER

Each Flexowriter accepts paper in excess of 11 inches width for printing and is equipped with an edge card reader and punch for reading and punching standard 7/8-inch paper tape or edge cards. Read and punch rates are approximately ten characters per second. Flexowriters are cable-connected to a Peripheral Buffer (PB-40). Special control switches and indicator lights are provided on the Flexowriter to aid the operator in this communication. A large number of Flexowriters can be accommodated in the RW-40 system.

A seven-channel code is used with the two most significant bits designating zone, such as letters, numbers, or Flexowriter functions. The next four bits are binary coded magnitude bits, and the last bit is an odd parity bit.

Figure 5. RW-40 Tape Module

PERIPHERAL BUFFER, PB-40

The PB-40 assembles slow manual requests and disperses results for typing without loss of valuable computer time. The PB-40 employs a magnetic drum identical to that in the DM-40. It buffers digital inputs from Flexowriters and keyboards of remote consoles, and buffers digital outputs to plotters and Flexowriters.

The storage on this drum is divided into 32 input channels and 32 output channels for connection with up to 64 separate sources of digital inputs and outputs. Each channel is capable of storing 256 alpha-numeric characters. Any Computer Module or Buffer Module can connect itself to a PB-40 through the Central Exchange.

DISPLAY BUFFER, DB-40

The DB-40 continuously regenerates display information without computer intervention. The computer is only required when a display is to be changed. The DB-40 is an output buffer which provides recirculating digital information for external displays such as cathode-ray tube displays. The DB-40 contains a magnetic drum which is identical to that of the DM-40 and has sufficient circuits to service eight display units with the equivalent of 1024 words at a 15 cycle repetition rate.

HIGH-SPEED PRINTER, PR-40

The PR-40 provides printed outputs of 80 characters per line at 900 lines per minute. Input to the PR-40 may be accomplished by magnetic tape for off-line use or by computer for on-line use via a buffer. The PR-40 is capable of producing up to six legible copies at the stated speeds.

Fifty different characters can be specified for each printed column. Paper feed format control is achieved by punched paper tape, permiting electronic selection of several different line skipping formats.

PLOTTER, PL-40

The PL-40 contains a plotting surface which is 30 inches by 30 inches. A partial vacuum holds the paper in place. Paper can be as small as 8-1/2 by 11 inches, or it can also be supplied from a roll.

The plotter provides three basic types of operation:

1. drawing lines which are continuous functions of x and y d. c. voltages,

2. drawing straight-line segments between specified points (25 per minute),

3. plotting symbols (50 per minute).

The first two types of operation require a pen head on the plotting arm; the last type uses a 12-symbol head. Heads are interchanged manually. There are three input sources to the plotter:

1. x and y d. c. voltages (on-line)
2. contact closures data (on-line)
3. punched paper tape data (off-line).

The voltage source is developed from computer output data by digital-to-voltage conversion. The contact closures source is produced by the Peripheral Buffer and has characteristics similar to the output of a punch paper tape reader. The punched paper tape source provides an off-line capability.

MASTER CONTROL CONSOLE, CC-40

The CC-40 provides the necessary link between the data processing system and its human supervisors. The console provides information to keep the supervisor informed of the status of the system. It employs a Flexowriter as a means of printed communication. The Master Control Console provides the means of changing the master computer assignment in case the master computer fails to operate correctly with the master computer program due to master computer failure.

Maintenance panels are provided as part of the individual modules or as accessory test equipment and are not duplicated at the Master Control Console.

CENTRAL EXCHANGE, CX-40

The CX-40 provides all the communication paths between modules of the system. Its size is dependent upon the number of required paths. Features of the CX-40 are as follows:

1. The Exchange functions at electronic speeds.
2. The Exchange is controlled by both Computer and Buffer Modules.
3. All transmission paths are of standard format.
4. Unidirectional transmission is employed.
5. The Master Computer can designate which devices can be connected to a given Computer or Buffer Module.

TAPE ADAPTER, TA-40

The TA-40 permits the conversion of magnetic tapes which have been prepared by some other data processors to the RW-40 tape format. The Tape Adapter provides the capability to read or write magnetic tapes for this other system; however, the CM-40 makes all format changes by the use of an interpretive program.

3.0 SYSTEM COMMUNICATIONS

Communication between the RW-40 and its operators is accomplished by means of a wide variety of extremely versatile devices such as display consoles, keyboards, line printers, typewriters, and plotters.

Within the RW-40 itself, communication is under the control of the modular Central Exchange, a modern electronic variant of the telephone relay exchange.

Communication with other systems is greatly facilitated by the flexibility of the RW-40 input-output buffering, permitting the use of whatever external equipment may be needed for a particular application; for example, teletype communications networks.

MAN-MACHINE COMMUNICATIONS

The Display Console

The Display Console is designed to handle the major share of man-machine communications. It is a general-purpose device that can be used to perform either analytic or simulated operational functions. The display and control functions of this console are independent; the loop between these functions is made through the Central Data Processor so that the operator has direct access to the automatic data processing power of the system. The display and control functions are dependent upon the computer programs for meaning and are thus modifiable for any number of applications without equipment change. Typewriters, high speed printers, and plotters are available in the system as secondary means of communication where hard copy outputs are desired. A Flexowriter can also be used to simulate inputs from remote communication links, such as teletype.

The Display Console can be logically broken down into display elements and command control elements. The display elements of the console consist of two 17 inch cathode-ray tubes (CRT) used for graphical displays. In addition to dots and lines, fourteen special symbols can be generated on this display. A seven inch CRT is used to display alpha-numeric information. Alpha-numeric and numeric indicators are used to display the coordinate scale and identity on the large display tubes and indicate the range of these coordinate systems. Special indicator lights are located on the console to display the command control operations being carried out.

The command control elements, as presently designed, consist of some fixed program keys which can be used to either modify displays or carry out standard analytic routines. In addition, there is a set of keys whose functions is defined by any one of 64 possible program cards. When one of these program cards is inserted in the space allocated on the console

it activates a specific computer program which controls the function of all the associated keys. Each program card specifies its own computer program. A joy stick control is provided to position a set of cross hairs on either large display tube, while a light gun provides the capability for referencing any point displayed on these tubes. A decimal keyboard is included on the console for the insertion of numeric information into the computer, and a Flexowriter can be included for the insertion of alpha-numeric information.

COMMUNICATIONS WITHIN THE RW-40 SYSTEM

Modular Central Exchange

The Central Exchange is a high-speed electronic switching network which interconnects all elements of the system. The computers and serial transfer buffers can control the Central Exchange by means of the function output instruction. In many respects the Central Exchange is similar to a telephone crossbar relay exchange--its principal difference is in its electronic switching speed.

The electronic switching network which interconnects elements of the system also has a modular design. This permits augumentation of control and switching points as required, in the same way that trunks can be added to a telephone central when additional capacity is needed. Individual switch points are on printed circuit cards which can be added to the switch cabinet as required, or removed to permit switch maintenance to proceed while the switch is in use. Switches of various sizes are available and may be cascaded to produce additional configurations.

COMMUNICATION WITH OTHER SYSTEMS

The Tape adapter provides for additional off-line communication with other computing or data processing systems, by enabling the RW-40 to interpret magnetic tape recorded by these other systems.

Punched card input and output can be provided in two ways depending upon the volume of data to be transferred by cards. For low volume applications commercially available card to punched paper tape and punched paper tape to card equipment is available. The punched paper tape input-output is buffered by the Peripheral Buffer so there is no loss of computation time. For higher card volumes commercially available card to magnetic tape units may be employed with the magnetic tape entry accomplished via the Tape Adapter.

Thus, the flexibility of the RW-40 is as apparent in its ability to communicate with other systems as in its ability to communicate with human operators.

4.0 OUTSTANDING DESIGN FEATURES OF THE RW-40

The RW-40 system was designed specifically to meet the most stringent data processing requirements, e.g., applications, which have requirements for processing speed and system expandability so extreme that they could not be met by any configuration of conventional computational equipment. The solution to these problems was found to lie in a modular, multiple-computer approach to data processing--an approach originating with the RW-40.

As will be shown in this section, the RW-40 offers speed and flexibility far beyond anything previously available; but in addition to this, the unique design concepts embodied in the RW-40 have enabled remarkable advances in programming convenience, reliability, and system economy as well. No other commercially available equipment can match these features.

FEATURES PROVIDING FOR RAPID DATA PROCESSING

Multiple Arithmetic Capacity

The RW-40 Computer Modules have been designed to permit parallel arithmetic operations with a minimum of interference. Parallel arithmetic units permit computation rates in excess of those required by the most stringent real-time applications. This rapidity of operation is accomplished by providing each computer with an adequate memory for its exclusive use, while at the same time providing Buffer Modules which can independently gather data and distribute results without computer control. These buffers may then be switched at electronic speeds to work with a computer. This provides additional memory capacity which the Computer Module can access as rapidly and as conveniently as the computer's internal memory. Information from a Computer Module can be stored in the Buffer Module. The buffer can then be released from the computer for subsequent acquisition by another computer or for distribution of the results under control of the Buffer Module itself.

The parallel arithmetic units permit many interesting new programming economies. For example, in simulating a control system operating in a complex environment, it is very convenient and economical to program the operation of the control system for one computer and to program the environment for another. The simulation consists of operating the two programs simultaneously, the employing the transfer buffers to shuttle results between the two machines. If at some later time the environment requires modification, the program for the control system need not be revised; it can in fact operate with either environment program.

Parallel Input-Output System

Rapid data processing is greatly facilitated by a completely buffered input-output system which provides a convenient, flexible means for

incorporating elements of the system to meet changes in the processing tasks or loads.

Digital messages and analog signals are accepted and dispatched by the RW-40 without interrupting the operation of any of the Computer Modules and without interfering with one another. System components such as the Buffer Module, Display Buffer, and Peripheral Buffer have their own control capability and are high speed magnetic core and drum storage elements. The Computer Modules are only concerned with placing information into or extracting information from these buffers, and not with the actual input or output function itself. The input-output buffers, therefore, provide for flexible interconnection with displays, or elements of the RW-40, or other systems without costly loss of computational capacity while servicing these devices.

FLEXIBILITY

Modular Construction

Future expansion of facilities to accommodate more complex systems is facilitated by the modular design of the RW-40 system--the only completely modular system available. This system was designed especially to accommodate data processing requirements which might increase almost indefinitely. The design of the RW-40 permits adding modules as required, or the regrouping of modules for more effective utilization. Programming techniques have been evolved, employing the master-slave concept, which permit these system modifications with a minimum of programming inconvenience.

The modules have, in addition, been designed to provide the most computational capacity for the lowest cost, by having each module provide system capability in a particular area. For example, it is possible to add arithmetic capacity without appreciably increasing the memory or the input-output system if this should prove to be desirable.

Expandable Memory

The RW-40 system memory can be autmented by adding computers, magnetic core buffers, magnetic drums, or magnetic tapes. Thus the memory capacity and speed of the RW-40 system can be tailored to match the exacting requirements of each application; memory can be added without directly increasing the arithmetic capacity of the system.

The magnetic core transfer buffer (BM-40) furnishes the capability for indirect addressing. Indirect addressing enables memory additions without a large address structure within the word length of the Computer Module. This form of indirect addressing eliminates the computation time required with index registers, for address modification. The Buffer Module also increases the accessability of data stored in the slower drum and tape memories since the buffer gathers the records independently for subsequent high speed utilization by the computer.

Real-Time Display System

Display modules provide a powerful means for controlling and/or assessing results of system performance.

The RW-40 Display Console provides extremely flexible computer output displays and contains keyboards for display manipulation or control of the simulation. All aspects of the display as well as the keyboard are controlled by computer programming. The repetitive regeneration of the display is automatic within the Display Buffer and the Display Console itself.

A general computer program is available which provides display capability with existing RW-40 hardware modules (Display Buffer, Display Console, and Peripheral Buffer) and makes use of existing programs for utilization of these modules.

RELIABILITY

All RW-40 modules are mounted compactly into uniform bays. Each bay is 84 inches tall, 23 inches wide and 26 inches deep. A glass door on the front of each bay makes all circuits easily accessible for maintenance. Each bay contains a forced air blower. No other special air-conditioning is required within the cabinet.

A small number of standard printed circuit insert cards are used throughout the system.

Where possible, components are selected because they have a long history of reliable operation in similar circuits. If new types of components are used, they are selected after evaluation of life-test data, careful study of the component itself to determine possible causes of failure, and consideration of the manufacturer's reputation for producing reliable components. Transistors are used throughout the system in order to achieve the maximum possible system reliability and only the most advanced solid-state components were incorporated into the design of the RW-40. The advantages of this approach are apparent from the speed with which conventional computing systems are being converted to solid-state circuits. RW computer systems have obtained mean times between failures measured in hundreds of hours by these techniques. The design criterion for the entire RW-40 system was a mean time between electronic failure of at least 500 hours. Solid state circuitry requires less power and hence imposes a far smaller heat load on the air conditioning system; also less floor space is required than for an equivalent conventional system.

The use of parallel modules further reduces the change of a system failure, since the system can operate at reduced capacity while one module is being repaired. With conventional computation systems a failure in the arithmetic unit or the memory usually disables the entire system.

MAINTAINABILITY

In the RW-40 Data Processing System, active circuits and logic circuits are built on plug-in replaceable subassemblies. Consequently, when a fault is located it can be corrected by plugging in a properly functioning subassembly . Fault location is facilitated by the use of many readily accessible test points. Also included are special automatic test routines, marginal checks and provisions for manual control to effect one-step operation. Forced setting of key flip-flops, and visual indication of the contents of all registers are among the features designed to facilitate maintenance of the computing system.

The modular construction of the RW-40 allows off-line maintenance of individual modules without shutting down the entire system. This allows the technician to perform corrective maintenance without the pressures precipitated by the demands to return to on-line performance. Also the use of identical modules (interchangeable parts) facilitates the ease and therefore speed of maintenance.

ECONOMY

The RW-40 system provides simultaneous arithmetic and tape search capacity in excess of the largest conventional systems, at a fraction of the cost of these systems.

The modular approach permits optimizing the design of each module to provide the most versatile performance at the most economical price. For example, the RW-40 approach to memory utilization through use of transfer buffers permits the computer to employ a small address structure for large capacity memory systems. The short address makes possible the powerful two address instruction format within a word size which is long enough for computation accuracy but short enough to permit high execution speeds. A single RW-40 computer is faster in executing general scientific computations than many conventional computation systems of considerably greater cost. The speed factor in favor of the RW-40 system becomes arbitrarily large when additional computers are added to the system.

PROGRAMMING CONVENIENCE

All of the basic routines necessary for the operation of a computing center are in existence. For this type of programming the computation system provides a two-pass mnemonic compiler system, the necessary input-output routines, the basic subroutines, the diagnostic routines such as selective output, trace and dump routines.

The RW-40 system is a unique data processing system which permits many new programming techniques. For example, many computations divide naturally into independent but related parts. These independent

parts may be programmed for separate computers within the RW-40 system. The separate computations may proceed simultaneously with resultant low execution time. New programming techniques are developing methods of employing more than one computer for economic high speed solution of problems which do not separate naturally into independent parts. As a specific example, it is possible to employ two computers to evaluate a single polynomial in little more than half the time required with a single computer.

A large number of general-purpose input-output programs are completed for use with the Peripheral Buffers, Display Buffers, Display Consoles, keyboards, typewriters, line printers, etc. Because of the newness of the system, however, many advanced techniques remain to be exploited.

APPENDIX
DETAILED INSTRUCTION LIST

INTRODUCTION

This appendix contains a listing of the instructions for the RW-40 Computer Module together with a functional explanation of each instruction and the associated execution time. The instruction list is not final; minor changes, additions, or deletions may take place. However, these changes will not effect the capability of the RW-40 Data Processing System other than to further improve the speed or flexibility of operations. The instruction list together with execution times presented in this appendix should be helpful in evaluating the computational capability of the RW-40 System.

Functional Notation of Instructions

g	First Core Memory address specified by instruction
G	Word in address g
h	Second Core Memory address specified by instruction
H	Word in address h
l	Address in Core Memory specified by Program Counter
L	Word in address l
CM	Word in Core Memory
DM	Half word in Drum Memory
TM	Half word in Tape Memory
P	Contents of Program Counter
X	Word in Instruction register
E	Word in Exchange register
A	Word in A-register, or Accumulator
B	Word in B-register
T	Contents of Time Counter
S	Contents of Sense register

Subscripts used with letters correspond to registers and counters to denote digit position. The subscript "n" designates a single digit in an arbitrary position. The subscript "n" increases with digit significance, or from right to left, within a register. The subscript "s" designates the sign digit. Subscripts "f," "g," and "h" designate fields, or groups of digits.

Examples:

A_n n-th digit of A, n = 1 to 26

B_s sign of B

E_g g field of E

E_m magnitude of E

A_c carry in to A

A_o overflow digit

E_p parity digit

$\vee$ logical sum or "or;" e.g., $B_n \vee E_n$, B_n or E_n (n-th digit of B or n-th digit of E). The logical product or "and" is written as: $G_n A_n$, G_n and A_n (n-th digit of G and n-th digit of A).

$\longrightarrow$ "replaces;" e.g., E$\rightarrow$X is read "word in Exchange register replaces word in Instruction register."

' "prime;" indicates complementation; e.g., E_n' the complement of E_n, E' one's complement of E.

FA full adder

HA half adder

I_n n-th interrupt condition

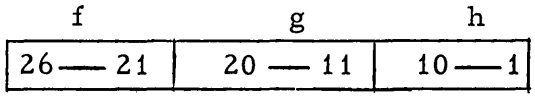
A/C Arithmetic and Control Unit of the Computer

$S_n I_n$ is n-th sensed interrupt condition (the "and" of S_n and I_n).

SI is word consisting of digits equal to $S_n I_n$

Instruction Word

Each instruction word contains 26 bits. The six most significant bits of the word are the operation code and are designated as the f field of the instruction word. The remaining 20 bits are divided into two 10-bit address fields which are designated the g and h fields as shown in the diagram, below.



Instruction Word

The bits in the g field of the instruction are used to specify one of the following: (a) address of the operand memory read cycle; (b) the magnitude, direction, and other options of a shift instruction; (c) conditions on which to transfer control; and (d) the address of the first word in a group of successive words in the core memory for an input-output instruction.

The bits in the h field of the instruction are used to specify one of the following: (a) the address of the operand that is read from the core memory on the second-operand memory cycle; (b) the address of the result that is written into the core memory on the result memory cycle (the h field of the instruction may be used to specify both of the above in one instruction); (c) the address of the next instruction in a jump-type instruction; (d) the number of words to be transmitted in a parallel input-output instruction.

Internal Instructions

An internal instruction obtains its operands from the core memory of the Computer or Buffer Modules or from registers of the Arithmetic

and Control Section. Other instructions are input-output instructions. As many as four core memory cycles may be initiated by one internal instruction. These memory cycles are designated as follows: (a) instruction cycle, (b) first-operand cycle, (c) second-operand cycle, and (d) result cycle.

Instruction Cycle

The instruction cycle is the memory cycle that reads the instruction word into the E-register. It is then transferred to the X-register. The operation code in the f field of X logically specifies the operations to be performed. The instruction cycle is initiated at the end of every instruction after the address register, P, has been advanced by one (or replaced for jump instructions) and transferred to the g field of the X register.

First-Operand Cycle

The first-operand cycle is the memory cycle that reads the word, G, specified by the g field of the instruction word to the E-register.

Second-Operand Cycle

The second-operand cycle is the memory cycle that reads the word, H, specified by the h field of the instruction word to the E-register. The h field of the X-register is transferred to the g field before this memory cycle is initiated.

Result Cycle

The result cycle is the memory cycle that writes the result of the instruction into the word addressed by the h field of the instruction. Before the result cycle occurs, the result is located in the E-register. The h field of the X-register is transferred to the g field before this memory cycle is initiated. A parity bit is generated during the first half of the result memory cycle.

Most of the arithmetic-type internal instructions have three modes of operation. The mode depends upon the order and number of memory cycles initiated by the instruction. These modes are as follows:

Replace Mode H * G→H

All four memory cycles are initiated. The A-register is cleared (or set to plus zero) at the end of the instruction cycle. The first operand, G, and second operand, H, are read and placed in the A-, B-, and E-registers, according to the instruction executed. The arithmetic operations occur and the result is located in the A- and E-registers. The B-register holds the remainder or least significant half of the result when these results occur. The result in E is written into address h in the core memory.

Hold Mode H * G→A

All memory cycles except the result cycle are initiated. The operations are similar to the replace mode except that the result is located in the A-register and is not written into memory.

Store Mode A * G→H

All memory cycles except the second-operand cycle are initiated. The A-register is not cleared and its contents are operated upon similarly to the second operand in the replace mode. The result is located in the A-register and address h.

Variations in the normal execution of an internal instruction occur when the g field, h field, or both fields, of an instruction word are all zeros. A zero address will not allow the corresponding operand memory cycle to be initiated. The operand is considered as plus zero by the computer.

An overflow indication may be the result of an attempt to add two numbers whose sum is 26 bits in magnitude, and therefore exceeds the capacity of the A-register. In this case, the stored result consists of the least significant 25 bits of the sum. An attempt to divide by a number, G, equal to or less that H or A causes the overflow indicator to be set. In this case, the division operation is not performed and the instruction is ended with the dividend, H or A, stored in the A-register.

At the programmer's option, an overflow indication may result on left shift operations when the specified magnitude of the shift causes significant one's in the A-register to shift out.

Add and subtract instructions are accurate to the least significant bit. The quotients of the divide instructions are rounded-off to the 25th bit. The roots in the square-root instructions have 24 accurate bits with the 25th bit always set to one. The products of the multiply instructions are double length with the most significant half in A and least significant half in B. Only the most significant half of the product is automatically stored in h of the memory. No automatic round-off occurs on multiply instructions. Round-off may be programmed with the shift instruction described below.

The following brief description of each internal instruction indicates operational steps that automatically occur in the arithmetic unit on each instruction. Tables summarizing the results of each instruction follow the descriptions.

Direct Addressing of a Buffer Module

Internal instructions may obtain one or both operands from a Buffer Module and may replace or store the results of the instruction in a Buffer Module. This is programmed by making the address fields, g or h, of the instruction all one's in the field that is selected to address the Buffer Module. Obtaining an operand from a Buffer Module adds approximately two microseconds, for each operand, to the time of each internal instruction.

Reservation of all one bits of the address fields of the instructions eliminates word address 1,023 of the main memory as an addressable operand. The operand obtained in the Buffer Module is the word addressed by the read address register of the BM. This register advances by one each time an operand is obtained from the BM. That is, operands are obtained from the Buffer Module in sequence starting with the first word of a block in the BM. If the h field of the internal instruction is all ones

on a Replace Mode, the result of the instruction replaces the last operand obtained from the BM. If it is a Store Mode internal instruction that addresses the BM, the result is stored in the address of the write address in the BM. That is, the result is stored in sequence at the end of the block of words previously stored in the BM. These BM registers, read address and write address, may be preset by the programmer with an output instruction.

Computer Internal Instructions

Replace Add H + G→H, A

G is copied to A; H is held in E; E is added to A with the result placed in A; A replaces H; B is not modified. Overflow indication is possible. If g = 0, H replaces A. If h = 0, G replaces H and A. If g = 0 and h = 0, A is replaced by + 0.

Hold Add H+G→A

G is copied to A; H is held in E; E is added to A with the result placed in A; B is not modified. Overflow indication is possible. If g = 0, H replaces A. If h = 0, G replaces A. If g = 0 and h = 0, A is replaced by + 0.

Store Add A + G→H, A

G is held in E; E is added to A with the result placed in A; A replaces H; B is not modified. Overflow indication is possible. If g = 0, A replaces H. If h = 0, G is added to A as above but is not stored in memory. If g = 0 and h = 0, no useful operation occurs.

Replace Subtract H - G→H, A

G is copied to A and the sign of A is reversed; H is held in E; E is added to A with the result placed in A; A replaces H; B is not modified. Overflow indication is possible. If g = 0, H replaces A. If h = 0, -G replaces H and A. If g = 0 and h = 0, A is replaced by + 0.

Hold Subtract H - G→A

G is copied to A and the sign of A is reversed; H is held in E; E is added to A with the result placed in A; B is not modified. Overflow

indication is possible. If g = 0, H replaces A. If h = 0, -G replaces A. If g = 0 and h = 0, A is replaced by +0.

Store Subtract A - G→H, A

G is held in E; the sign of A is reversed and E is added to A with the result placed in A; then the sign of A is reversed and A replaces H. B is not modified. Overflow indication is possible. If g = 0, A replaces H. If h = 0, G is subtracted from A as above, but is not stored in memory. If g = 0 and h = 0, no useful operation occurs.

Replace Absolute Subtract |H| - |G|→H, A

G is copied to A and the sign of A is set negative. H is held in E and the sign of E is set positive. E is added to A with the result placed in A. A replaces H. B is not modified. Overflow indication is not possible. If g = 0, the absolute value of H replaces A. If h = 0, the magnitude of G with a negative sign replaces H and A. If g = 0 and h = 0, A is replaced by +0.

Hold Absolute Subtract |H| - |G|→A

G is copied to A and the sign of A is set negative. H is held in E and the sign of E is set positive. E is added to A with the result placed in A. B is not modified. Overflow indication is not possible. If g = 0, the absolute value of H replaces A. If h = 0, the magnitude of G with a negative sign replaces A. If g = 0 and h = 0, A is replaced by +0.

Store Absolute Subtract |A| - |G|→H, A

G is held in E and the sign of E is set positive. The sign of A is set negative. E is added to A with the result placed in A. The sign of A is then reversed and A replaces H. B is not modified. No overflow indication is possible. If g = 0, the absolute value of A replaces H. If h = 0, the difference of |A| and |G| is computed as above, but the result is not stored in memory. If g = 0 and h = 0, the absolute-value of A is placed in A.

Replace Multiply H x G→H, A

G is transferred to B; H is held in E; A is replaced by +0. The h field of X is set to 24 and is counted down towards 0 for each successive addition of E into A and B or left shifts of A and B. The most significant half of the product is then transferred from B to A, and the least significant half of the product is transferred from A to B. A replaces H; overflow indication is not possible. If g = 0, then H, B and A are replaced by +0. If h = 0, or g = 0 and h = 0, A and B are replaced by +0.

Hold Multiply H x G→A

The same operations occur as in Replace Multiply except that the most significant half of the product is held in A and not stored in h. If g = 0 or h = 0, or both, A and B are replaced by +0. Overflow indication is not possible.

Store Multiply A x G→H, A

G is transferred to B and A is transferred to E. A is replaced by +0 and the 25 steps of successive adds and shifts occur. The most significant half of the product is transferred to A and replaces H. B contains the least significant half of the product. If g = 0, A and B are replaced by +0. If h = 0, the product A x G replaces A. If g = 0 and h = 0, A and B are replaced by +0. Overflow indication is not possible.

Replace Divide H÷G→H, A

H is transferred to A; G is transferred to E and is 1's complemented. B is replaced by +0. The h field of X is set to 24 and is counted down towards 0 for each successive addition of E into A and B or left shifts of A and B. The quotient is then transferred from B to A, and the remainder is transferred from A to B. A replaces H. If g = 0, an overflow indication occurs and H replaces A. If h = 0, or g = 0 and h = 0, A is replaced by +0. The result is rounded to 26 bits.

14 Hold Divide H÷G→A

The same operations occur as in Replace Divide except that the quotient is not stored in h.

Store Divide $A \div G \rightarrow A$

G is transferred to E and is 1's complemented. B is replaced by +0 and the 25 steps of successive adds and shifts occur. The quotient is placed in A and replaces H. The remainder is placed in B. If g = 0, an overflow indication occurs and A is not modified. If h = 0, A is divided by G as above, but the result is not stored in h. If g = 0 and h = 0, an overflow indication occurs and A is not modified. The result is not rounded to 26 bits.

Replace Square Root $\sqrt{H+G} \rightarrow H, A$

G is placed in A and H is placed in E; E is added to A with the result in A similar to the Hold Add instruction; A is transformed to B. The normal long-hand method of square root calculation is mechanized. The h field of X is set to 24 and is counted down towards 0 for each successive addition of E to A or shift of A and double left shift of B, which appends two significant bits of the radicand to the remainder in A. E contains the 1's complement of the root. E is then 1's complemented and transformed to A. A replaces H. The 25th bit of the square root is always set to 1. Overflow may result from the addition in which case the square root is not computed. If g = 0, the square root of H replaces A. If h = 0 the square root of G replaces H and A. If g = 0 and h = 0, A is replaced by +0.

Hold Square Root $\sqrt{H+G} \rightarrow A$

The same operations are performed as in Replace Square Root except that the result is not stored in h. Overflow may occur on the addition.

Store Square Root $\sqrt{A+G} \rightarrow H, A$

G is copied to E and E is added to A. The square root is computed as in Replace Square Root. The result replaces H and A. Overflow may occur on the addition. If g = 0, the square of A replaces H. If h = 0, the square root of A + G replaces A. If g = 0 and h = 0, the square root of A replaces A.

Replace Insert $HG' \vee AG \rightarrow H, A$

The Insert instruction is a logical-type instruction. The above logical expression may be read as follows: Place a 1 in the corresponding position of H and A when there is a 1 in H and a 0 in G, or a 1 in G and a 1 in A. This instruction may be used to perform the following functions:

Insert (or address modification) $HG' \vee AG \rightarrow H, A$

H = word to be modified

G = mask, $G_n = 1$ for every bit to be changed

A = new bits to be inserted in H

Extract (logical AND) $HG' \rightarrow H, A$

H = word to be modified

G = extraction, $G_n = 0$ for every bit of H desired

A = +0

or $AG \rightarrow H, A$

H = +0

G = extractor, $G_n = 1$ for every bit of A desired

A = word to be modified

or $AG \rightarrow H, A$

H = +0

G = word to be modified

A = extractor, $A_n = 1$ for every bit of G desired

Complement (either position or whole word) $HG' \rightarrow H, A$

H = mask, $H_n = 1$ for every bit of G to be complemented

G = word to be complemented

A = +0

Merge (logical OR) Inclusive OR $H \vee G \rightarrow H, A$

H = word whose bits are to be included

G = word whose bits are to be included

A = G

Exclusive OR $HG' \vee H'G \rightarrow H, A$

H = word whose bits are to be included

G = word whose bits are to be included

A = H' by a previous insert instruction

G is transferred to B; H is held in E and is 1's complemented. Transfer of words between registers without clearing the registers yields logical OR functions, and the operation that complements registers yields the logical AND functions. B contains the logical product of AG at the end of this instruction. The result is placed in A and H. No overflow is possible. If g = 0, H replaces A. If h = 0, the logical AND of A and G replace A. If g = 0 and h = 0, A and B are replaced by + 0.

Hold Insert $HG' \vee AG \rightarrow A$

The same operations are performed as in Replace Insert except that the result is not stored in h.

Store Insert $AG \rightarrow H, A$

G is copied to E; A is complemented; E is complemented; A is transferred to E without clearing E. This yields $E' \vee A'$ in E. E is complemented yielding AE = GA in E. E is transferred to A and H; B is not modified. No overflow is possible. If g = 0, H and A are replaced by + 0. If h = 0, or g = 0 and h = 0, no useful operation occurs.

Accumulate Add $A + H + G \rightarrow A$

G is copied to E; E is added to A; then H is copied to E, and E is added to A. Overflow is possible. The B-register is not modified. If g = 0, H is added to A and the result is placed in A. If h = 0, G is added to A and the result is placed in A. If g = 0 and h = 0, no useful operation occurs.

Accumulate Multiply $A + H \times G \rightarrow A$

G is copied to B, and H is held in E. A is added to the first partial product, hence A is added to the most significant half of the product. The most significant half of the result is placed in A. The least significant half is placed in B. Overflow is possible. If g = 0 or h = 0, no useful operation occurs.

Shift

The 10-bit g field of the instruction specifies the magnitude and type of shift as follows:

g

10	9	8	7	6	5 4 3 2 1
1 = double length (A and B) 0 = single length (A only)	1 = sign included 0 = sign not included	1 = rounded 0 = not rounded	1 = overflow indicated 0 = overflow not indicated	1 = left shift 0 = right shift	Binary coded magnitude of shift

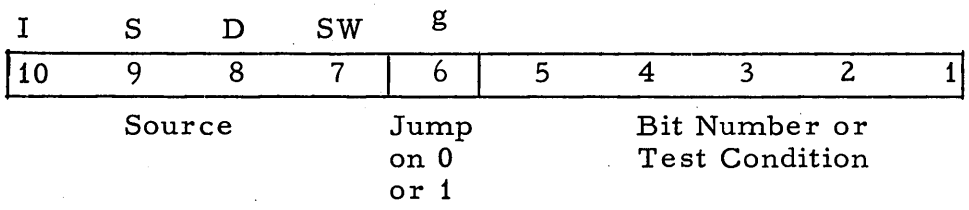
The h field of the instruction specifies the store address in memory or buffer, of the shifted result in A. If the h field of the instruction is all zeros, no store occurs, and the shifted result is left in A only. If the magnitude portion of the g field is made all 1's, a FLOAT operation occurs. All options except overflow indication are now possible on float operations. If a left float is given, the contents of A (and B if double length), are shifted left until a 1-bit appears in the sign position when sign is included; or in the most significant magnitude bit position of A when sign is not included. If a right float is given, the contents of A (and B if double length), are shifted right until a 1-bit appears in the least significant bit position of A. The floated result in A is stored in H, and the number of shifts is stored in A in the least significant end. The remaining bits of A are set to 0's.

Transmit g→h

Contents of g replaces contents of h without disturbing the accumulator (A). The contents of word 0 is treated like any other word, contrary to most other instructions.

Test Jump

The g field of the instruction is separated into three parts as follows:



Bits 7-10 specify source of the word to be tested and this word is loaded into the least significant end of A. Possible sources are as follows:

Bits 10	9	8	7	Source
1	0	0	0	Interrupt conditions, 19 bits
0	1	0	0	Sense Register, 20 bits
0	0	1	0	14 data and 3 control input lines
0	0	0	1	Conditional Jump Switches, 8 bits
1	1	0	0	Logical product of Sense Register and interrupt conditions, 20 bits
0	0	0	0	No input, test A, 26 bits

Bit 6 specifies whether a jump is to be made on a 0 or a 1 in the bit position specified. Bits 1-5 specify bit numbers, from right to left in A, or special conditions to be tested. A binary coded number from 0 to 31 allows testing of the following:

Bits 5 4 3 2 1	
0	No test, only input of source to A occurs
1-25	Bit in A to be tested
26	Test sign of A
27	Test overflow indicator and reset
28	Test parity error indicator and reset
29	Test control panel test light and reset
30	Test and conditional tape read indicator and reset
31	Test program error indicator and reset

The overflow indicator may be set by all three modes of Add, Substract, Divide, and Square Root, and by the Accumulate Multiply instruction. Overflow indication is optional on left shift instructions. The parity error indicator may be set by parity errors occuring on operands from memory or on input from an external device. The control panel test light may be set by a programmed Function Output instruction. The conditional tape read indicator is set with a conditional Data Input instruction when the search condition is not met. The program error indicator is turned on by detection of unused command codes or by attempting to obtain operands from a Buffer Module when it is not connected or ready.

Link Jump

The contents of the address counter, P, which holds the address of the next instruction in sequence, is inserted into the h field of the word addressed by g, and H is taken as the next instruction. If g = 0 an unconditional jump to word H occurs; A and B are unchanged.

Tally Jump

The word in address g is tested to see if it is +0 or -0. If G = +0, the next instruction is taken in sequence and G is not changed. If G = -0, H is taken as the next instruction and G is not changed. If G > +0, a one in the least significant place is subtracted from G and H is taken as the next instruction. If G < +0, a one is added to G and the next instruction is taken in sequence.

If the tally number, G, is positive it represents the number of jumps. If the tally number, G, is negative it represents the number of times no jumps occur. If h = 0, G is counted down each time the instruction is used until G = 0 and the next instruction is always taken in sequence. If g = 0, an unconditional jump to word H occurs; A and B are unchanged.

Load A

The g and h fields of the instruction are transferred to A. The six most significant bits of A (including sign) are set to 0.

Insert "S"

The g and h fields of this instruction are used as a mask to insert ones or zeros into S. A is preloaded with a previous instruction (e.g., load A). A one-bit in the g or h field corresponds to the bit in S that will be changed. S will be changed to a zero or one corresponding to that position in A. This instruction is logically similar to the Insert Hold instruction and may be represented by the logical equation:

$$S = AX \vee SX'$$

The S-register is a 20-bit register which corresponds to 13 external interrupt conditions and 6 internal interrupt conditions such as sign, overflow, parity error, etc. The most significant bit of S which corresponds to the 20th bit position in A, numbered right to left, is the master interrupt bit. If this bit in S is set to zero, no interrupts may occur. The 20 least significant bits of A are used to insert new bits in S. This instruction does not permit an interrupt when it is being executed.

The 13 external interrupt conditions are selected by a switching instruction that selects interrupt conditions from a centrally stored location.

Store A, B

The contents of the B-register are stored in address g. The contents of the A-register are stored in address h. If the g field of the instruction is all 0's, A is stored in h only. If the h field of the instruction is all 0's, B is stored in g only. If both g and h fields of the instruction are all 0's, A and B are interchanged.

Compare Jump

The word in A is compared with the word addressed by g. If $A \leqslant G$, h is taken as the next instruction. If $A > G$, the next instruction is taken in sequence. If g = 0, a test for A = + 0 is made; if A = + 0, a jump to H occurs; otherwise, the next instruction is taken in sequence. A and B are unchanged.

Approximate Operation Times

Table 1 shows operation times for all of the internal instructions of the Computer Module.

Table 1. INTERNAL INSTRUCTIONS: Approximate Operation Times

Number	Operation	Time (usec)
1 — 0	Replace Add	40
2 1	Hold Add	33
3 2	Store Add	33
4 3	Replace Subtract	43
5 4	Hold Subtract	33
6 5	Store Subtract	33
7 6	Replace Absolute Subtract	43
8 7	Hold Absolute Subtract	33
9 10	Store Absolute Subtract	33

Table 1. INTERNAL INSTRUCTIONS: Approximate Operation Times (cont'd)

Number		Operations	Time (usec)
10	11	Replace Multiply	85
11	12	Hold Multiply	78
12	13	Store Multiply	85
13	14	Replace Divide	132
14	15	Hold Divide	125
15	16	Store Divide	125
16	17	Replace Square Root	175
17	20	Hold Square Root	167
18	21	Store Square Root	165
19	22	Replace Insert	48
20	23	Hold Insert	42
21	24	Store Insert	48
22	25	Accumulate Add	38
23	26	Accumulate Multiply	82
24	27	Transmit	30
25	30	Shift (Float)	27-67
26	31	Test Jump	27-67
27	32	Link Jump	30
28	33	Tally Jump	42
29	34	Load A	10
30	35	Insert S	10
31	36	Store A, B	22
32	37	Compare Jump	25

External Instructions

External instructions are those instructions that transmit or receive data over fourteen information lines, 13 data bits and a parity bit. These lines are switched to the external source through the Central Exchange.

There are five external instructions of a general type. The instruction code does not specify the type of module that is to transmit or receive data. The Computer Module automatically knows which device it is connected to and executes the instructions accordingly. The external instructions are described as follows:

Function Output

The least significant half, 13 bits, of this instruction is transmitted over the information lines to the external device that is currently switched to the computer or to the Central Exchange to change the switch. These 13 bits are called the function code and they constitute a command to the external device. The format of these half-word commands are different for each type device; however, there are two general types. These are function codes for a Buffer Module and function codes for all other external devices. Function codes for non-buffer devices use the three least significant bits of the 13-bit code to designate the type of device which is to receive the command. These codes are as follows:

Code 3 2 1	Connected Device
0 0 0	Not assigned
0 0 1	Tape Module
0 1 0	Drum Module
0 1 1	Peripheral Buffer
1 0 0	Display Buffer
1 0 1	Printer Control
1 1 0	Central Exchange
1 1 1	Not assigned

Addressing the Central Exchange, with bits 1, 2 and 3, is used only for selecting the connection Table Master Control in order to override a previous switch setting or to select the assignment Table Control. A Function Output instruction for a normal change in switch settings addresses the Central Exchange by a one-bit in bit 20 of the instruction.

The fourth least significant bit of the function code on non-buffer commands is used to reset error indicators.

The function code format for the Tape Module, Drum Module, Peripheral Buffer, and Buffer Module are as shown in Tables 2, 3 and 4.

The tape command consists of a coded operation specification, plus an independent specification of certain aspects of tape handling.

Table 2. Tape Module Function Code Format

Bit Number	Interpretation
1	1
2	0 — Mandatory, indicates Tape Command
3	0
4	1 = Reset Parity and Dropout Indicators
5 and 6	0 0 Read
	0 1 Erase
	1 0 Replace
	1 1 Write
7	1 = Search Data 0 = Search Block No.
8	1 = Set Forward Direction
9	1 = Set Reverse Direction
10	1 = Start or Continue
11	1 = Stop
12	1 = Single Block 0 = Full Tape
13	1 = High Speed* 0 = Slow Speed

*Tape reverts to slow speed at the completion of any motion

All drum commands cause a new band number to be inserted into the band selection circuitry. They also initiate a search of the specified type unless a read check is started. The drum normally transmits status except during searches, data transfers, and read checks.

Table 3. Drum Module Function Code Format

Bit Number	Interpretation
1	0
2	1 — Mandatory, indicates Drum Command
3	0
4	1 = Reset Parity Indicator
5	1 = Prepare to Write; 0 = Read
6	1 = Start Read Check*
7	1 = Search Data; 0 = Search Word Number
8	Not used
9	Not used
10	Not used
11	X
12	X — Band Number
13	X

*Bit 5 should indicate a read when starting a read check.

The Peripheral Buffer accepts drum commands and control signals for the transmission of data to or from high-speed devices.

Table 4. Peripheral Buffer Function Code Format

Bit Number	Interpretation
1	1
2	1 — Mandatory, indicates PB Command
3	0
4	1 = Reset Parity Indicator
5 and 6	0 0 Send Status of Indicated Group
	1 0 Send General Status
	0 1 Clear Availability Flip-Flop of Indicated In-Out Channel
	1 0 Set Availability Flip-Flop of Indicated In-Out Channel

Table 4. Peripheral Buffer Function Code Format (cont'd)

Bit Number	Interpretation
7	X — Input-Output Channel Code
8	X — Input-Output Channel Code
9	X — Input-Output Channel Code
10	X — Input-Output Channel Code; X — Status Group Code
11	X — Input-Output Channel Code; X — Status Group Code
12	X — Status Group Code
13	X — Status Group Code

The function code for the Buffer Module uses the three most significant bits, 11 through 13, to specify a particular function to be performed. The remaining 10 bits specify a number to be used as an address or length of a block transfer. The five least significant bits of the function code may also be used to set the instruction register, I, or the status register, S, of the Buffer Module. The coded functions that a Buffer Module may be commanded to perform are shown in Table 5.

Table 5. Buffer Module Function Code Format

Code 13	12	11	Interpretation
0	0	0	Not assigned
0	0	1	Set I or S
0	1	0	Set Length
0	1	1	Set Write Address
1	0	0	Set Read Address
1	0	1	Switch and Set Read Address*
1	1	0	Start Self-Instruction and Set Read Address*
1	1	1	Stop Self-Instruction and Set Read Address*

*A zero address field indicates the read address is not to be changed.

The Function Output instruction is also used to perform functions internal to the computer. These functions are termed self-instructions. An example of a self-instruction would be to set "on" the control panel test light. Bits 20 and 19 of the instruction specify the type of Function Output. Bit 18 specifies whether the computer should wait in the instruction until it is signaled that the function code is accepted or not. The word structure of the Function Output instruction is as follows:

26-21	20-19	18	17-14	13-1
Op. Code	↓ ↓	|	Not used	Function code

20-19	18	
0 0		Send function code to external device
0 1		Self-instruction
1 0		Send function code to Central Exchange
	↓ 0	Terminate instruction if signal returned says either function code accepted or function code not accepted
	1	Terminate instruction if signal returned says function code accepted

Data Input

This instruction accepts blocks of full words from the module that is connected to the information lines and stores these words in the computer memory.

The g field of the instruction designates the address in memory of the first word. If a search is required for Tape Modules, the word in address g is the block number, previously stored. The word in address g always must hold the drum sector number, 10 bits, when inputs from a drum are desired. This word in g is not examined if a buffer is connected to the computer. If no search on tapes is desired, this word in g must be set to all zeros.

The h field of the instruction designates the number of full words to be read in. If h is all zeros, only a search is performed. If h is all ones, the whole length of a block on tapes will be read in and the length of the block will be stored in A at the end of the input operation.

Data Output

This instruction outputs blocks of full words from the computer memory to the module connected to the information lines. The g and h fields of the instruction are used similar to the Data Input instruction.

Two modes of tape write, Data Output, instructions are available. These are write and overwrite which are preset by a Function Output instruction. In the write mode two clock channels are recorded. In the overwrite mode one channel of clock information is recorded. This mode permits replacement of any block of information by a new block of the same length.

Conditional Data Input

This instruction is useful with Tape Modules only. It allows tape search without tying up the computer between block numbers and interblock spaces. The g and h fields are used as in Data Input. The word in g must be the block address.

When this instruction is executed, the first tape block number is compared for equality with the word in g. If they are equal, the block is read in, according to h. If they are not equal, the conditional read indicator is turned on and the computer takes the next instruction in sequence. The tape continues to move forward and the computer can be warned before the next block number is available. The instruction should be repeated for each block number until the search and read are completed.

Flexowriter Communication

The Computer Module can communicate with a User Module. The seven data output lines of the User Module are connected to the six least significant bits of the A-register; and the seven data input lines of the User Module are connected to the six most significant bits of the

A-register. Parity is verified on input and six data bits are transferred into the A-register. Parity is automatically generated on output from the Computer Module.

Interrupt Capability

The computer allows an interruption of its current program by any of a number of conditions which may exist throughout the system. The configuration of bits in the S-register of the computer specifies the conditions which are allowed to interrupt the computer program. The S-register is loaded by a computer instruction, or by an external device. The aggregate of system conditions which may cause interruption is designated. A typical condition involves many more bits of I than bits of S. Therefore, a bit S_n normally specifies a field I_n corresponding to a certain type of interrupt condition. In a system with full flexibility the correspondence between S_n and I_n is variable, perhaps by plugboard. The members of a field I_n are designated I_{ni}.

Any time a condition I_{ni} comes on while there is a one in the corresponding S_n, an interruption occurs. At the end of each instruction, the computer inspects to see if an interruption has occurred. If one has occured, the next instruction is in address 000. When the computer has been programmed for interruption, there will be a Link Jump instruction in 000 which stores the return address for the interrupted program, and provides entry to a subroutine which stores A and B, finds out what caused the interruption, and does whatever is necessary to resolve the interruption.

Upon interruption, the computer enters an interrupt status, indicated by a flip-flop in A/C, and remains there until it returns to the program which was interrupted. Further interruptions are locked out during the intervening "interruption subroutine." However, after the initial interruption is resolved, the test of interrupt conditions is repeated, and if a second interrupt has occured, it is resolved also. This loop is repeated until all interruptions have been cleared out. Then A and B are restored, and a Jump instruction re-enters the interrupted program.

By executing a Test Jump instruction, program branching on condition I_{ni} may be effected under program control independent of the S-register. This instruction transfers to H on conditions specified by g.