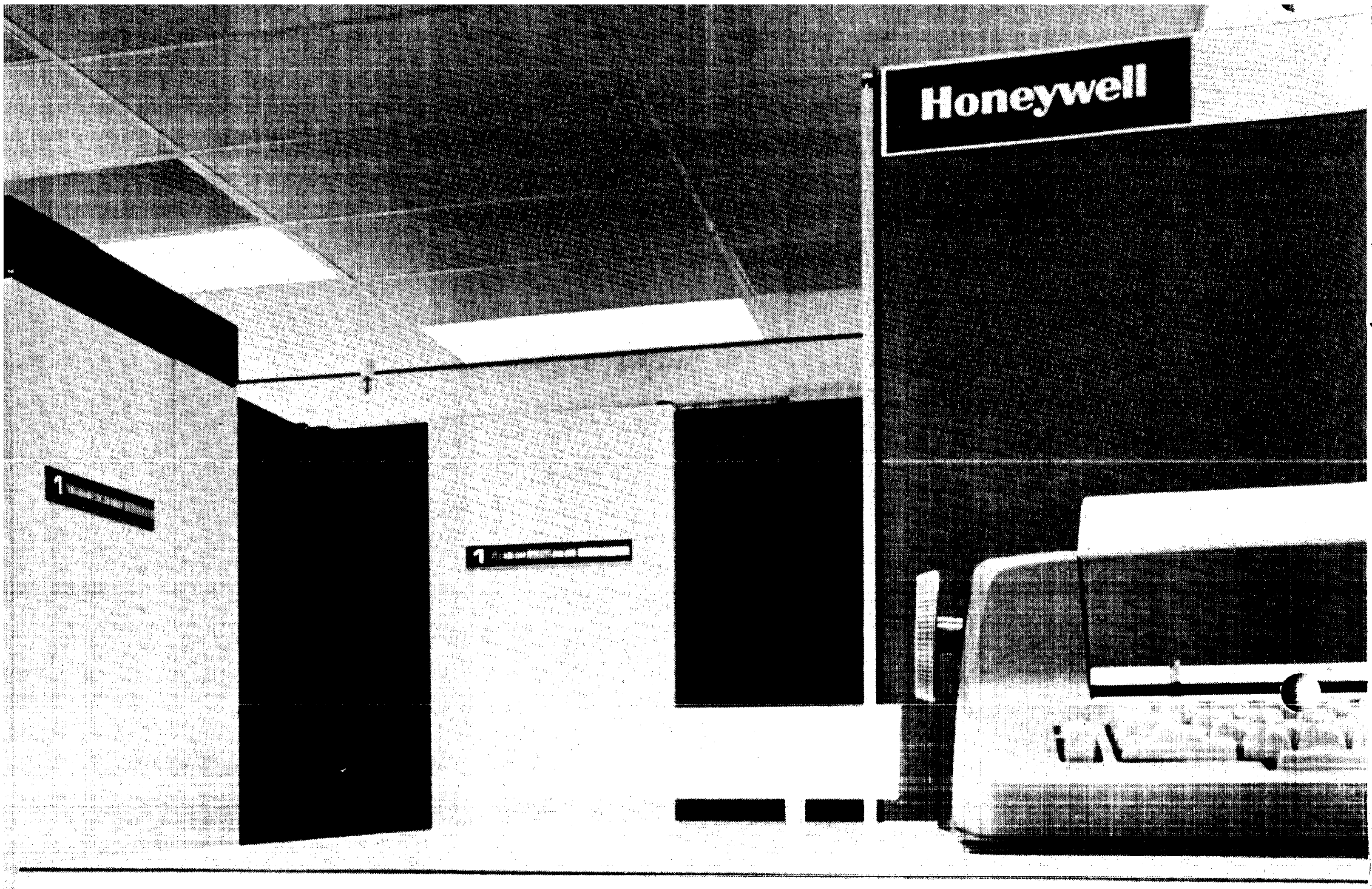
Honeywell

SERIES 6000 SUMMARY DESCRIPTION



Honeywell

SERIES 6000

SUMMARY DESCRIPTION

Table of Contents

List of Illustrations

List of Tables

Section I
Series 6000 Overview

The Honeywell Series 6000 is a family of large-scale, multi-dimensional information systems featuring high throughput, optimum use of system resources, and a wide spectrum of user-oriented capabilities.

The Series 6000 achieves an exceptionally high level of performance by operating in a total multiprogramming and multiprocessing environment. The system optimizes the use of its resources through concurrent processing in all dimensions – local and remote batch, remote access, transaction processing, and time sharing – all utilizing a common data base.

GROWTH AND FLEXIBILITY

Each member of the Series 6000 family is an optimum blend of powerful, state-of-the-art hardware and time-tested software.

- The functional modularity of the hardware provides an orderly, natural path of growth as the needs of a company's business grow. Multiple processors can be added, or more memory, or more data communications.
- Through it all, the user's programming investment is safe. The GECOS time-proven operating system is in control of all systems, from the smallest to the largest. No reprogramming is necessary, no matter what new combination of Series 6000 hardware best fits the growth needs of a business. GECOS operates in all Series 6000 models and all hardware configurations. All the capabilities of GECOS are available at each and every hardware level.

THE MODELS

Three models of the new family – Models 6030, 6050, and 6070 – are particularly well suited to mixed scientific, engineering, and business work loads. Price/performance in mixed-work load environments is outstanding. A few of the key features of these models are:

- Series 6000 Fortran – with the highest compile speed in the industry.
- Efficiency – the same fast Fortran compiler is used in both batch and time sharing.
- Data Base Management – a common file system is accessible in all dimensions – local and remote batch, remote access, time sharing, and transaction processing.

Three more members of the Series 6000 family – Models 6040, 6060 and 6080 – are specifically designed for heavy COBOL users. These models feature well over 100 new business-oriented instructions which provide powerful character string and decimal arithmetic capabilities. This extended instruction set closely correlates with the business source languages and significantly reduces program size (hence memory requirements). The result is faster and more economical processing of business oriented jobs.

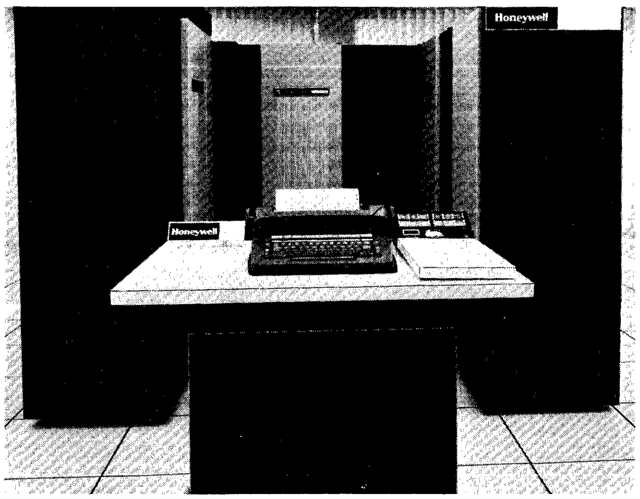


Figure 1. Model 6030 – Entry into the Series 6000 Family

The totally user-oriented features include:

- Full ANSI COBOL compiler
- Integrated Data Store (I-D-S), a powerful data base structuring technique
- Indexed-Sequential File Processing
- Unique Transaction Processing Executive

ADDITIONAL BENEFITS

Bundled	The Series 6000 systems come bundled – basic support services, an extensive software library, several language compilers, education and training, and applications packages are included in the basic system price.
Applications	Some of the application tools available are: Inventory Management System Parts Explosion System Automatically Programmed Tools (APT) Linear Programming (LP 6000) PERT/TIME and PERT/COST SIMSCRIPT Simulation Language Times Series Forecasting MATHPAC Subroutine Library
Languages	Language capabilities include ANSI COBOL, Fortran, ALGOL, JOVIAL, and a powerful macro-assembly language (GMAP), plus several problem-oriented languages and time-sharing languages.
Maximum Up-Time	Series 6000 systems are built for maximum up-time. With the latest in integrated circuit technology, improved packaging results in more compact and easily maintained hard-

ware. Circuit testers and maintenance panels expedite system diagnosis and help to isolate malfunctions quickly. The total on-line testing and diagnostic routines in GECOS allow temporary deallocation of central system modules, peripherals, or communication subsystems for automatic checkout while normal processing continues in other system elements.

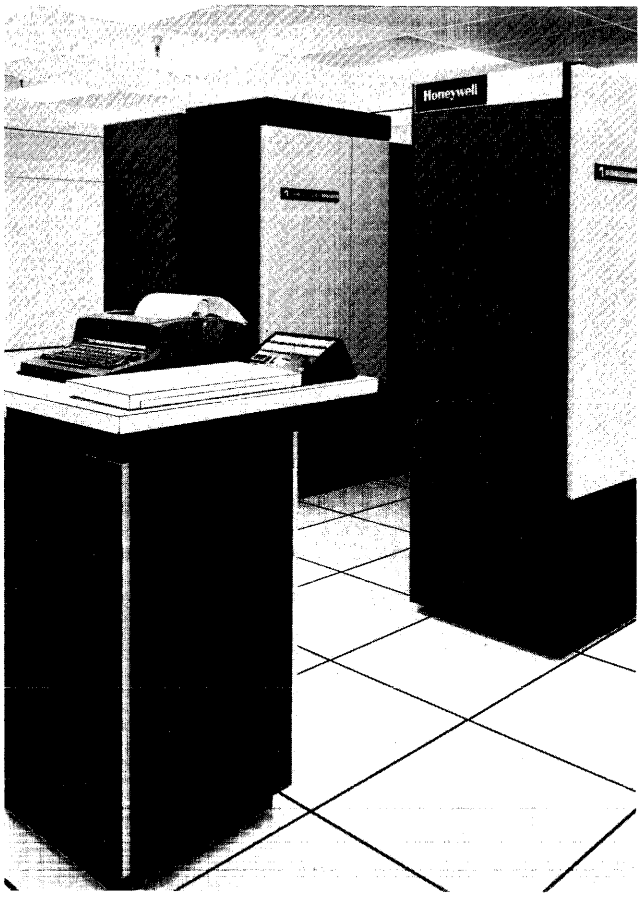


Figure 2. Model 6080 –Top Model of the Series 6000 Family

GECOS 6000

GECOS 6000 maintains the status of all system resources (peripherals, memory, and processors) and all user jobs in the system. Using the System Scheduler, the allocator queue accommodates a virtually unlimited number of jobs; these jobs can be entered into the system through multiple central and remote devices concurrent with the execution of jobs in the system. The jobs are dispensed to the system according to priorities and resource requirements. GECOS allocates system resources to jobs in the allocator queue in accordance with the priority of the job, and supervises the concurrent and simultaneous execution of as many programs (up to 63) as the configuration can accommodate. GECOS also controls the concurrent printing/punching of output from completed jobs. High-priority programs can be expedited by swapping out programs in execution.

GECOS also provides the programmer with a complete logical approach to problem solution. There are no constraints or unusual programming considerations imposed on the programmer because of the multiprogramming or multiprocessing environment in which his program will be executed. File processing is performed sequentially or randomly at the logical file level; the programmer need not be concerned with the physical characteristics and constraints of the peripheral device used nor with the organization of the file system.

COMMON FILE SYSTEM

The "heart" of a Series 6000 Multidimensional Information System is a centralized file system of hierarchical, tree-structured design, accessible by programs operating in any of the dimensions. Catalogs and files are secured by passwords. File access is controlled by GECOS. Several programs can read and/or can update a file concurrently. A unique design feature of the file system allows maintenance (repacking) routines to be "cleaning up" a portion of the file system while production routines in concurrent operation are accessing other portions of the system.

BATCH PROCESSING

GECOS 6000 provides a flexible, high-throughput batch processing environment. Up to 63 programs can be in concurrent execution. Incoming jobs are classified into a number of separately defined job streams, permitting each user to control his own priorities. Roll-out, roll-in capabilities allow for fast response to high-priority jobs or transactions. Batch jobs may be submitted from any local input device or remote terminal at any time, without operator intervention.

REMOTE PROCESSING

Concurrent remote processing capabilities can be added to Series 6000 systems by including one or more DATANET[1] 30, DATANET 305, or DATANET 355 Communications Processors in the configuration. Each communications processor permits a variety of terminal, transmission rate, and processing options. Remote processing capabilities include remote batch, remote access, transaction processing, and time sharing. GECOS remote access uses a reactive terminal interface which provides direct terminal access (through the communications processor) to the information system and to the common file system, facilitating the development of "on-line" terminal applications.

TRANSACTION PROCESSING

The Transaction Processing Executive opens the door to on-line, real-time data processing. Transactions (messages representing events in the user's business) can be entered via remote terminals. The Transaction Processing Executive interprets the transaction code contained in the message and calls the necessary application programs into execution to process the message. The output or acknowledgement will be returned to the designated terminal.

[1]Trademark.

TIME SHARING

A Time Sharing Executive, which utilizes the reactive terminal interface, provides Series 6000 installations with concurrent time sharing, featuring Fortran and BASIC, an easy to use, problem-solving language; and a powerful text-editing package to create, update, and obtain formatted printouts of textual data. Full upper/lower case ASCII character handling is provided.

Catalog structuring, file protection, file sharing, access control, and source/object file storage capabilities are provided through the file system.

The time-sharing batch capability permits time-sharing users to create and initiate batch mode programs and to scan or receive the batch output – all from a time-sharing terminal.

In addition, the structure of the Time Sharing Executive and the integration of time-sharing files in the file system facilitate user extension of the time-sharing system to provide further remote processing capabilities.

TOTAL ON-LINE TESTING SYSTEM (TOLTS)

The Total On-Line Testing System is composed of four major subsystems for peripheral, communications, main frame, and remote processing. This on-line testing system is part of a total maintenance and recovery concept. Eight concurrent diagnostic programs can operate with user programs under GECOS.

Section II
Hardware Overview

Honeywell Series 6000 systems provide processing and input/output capabilities across a wide performance range. Systems are tailored to the specific work load and processing environment of an installation through the selection of the appropriate system model and by the configuration of central system modules and peripheral devices. System models are differentiated by the speed of the central system components.

With a system model, further performance flexibility is possible through "functional modularity," or the selection of central system modules to match the work load needs of the installation.

The Comprehensive Operating Supervisor, GECOS, is the same for all models and configurations and provides multidimensional processing capabilities from the smallest to the largest system.

FUNCTIONAL MODULARITY

Series 6000 multidimensional systems employ a unique, new-design concept which provides complete flexibility in configuring the precise blend of processing, memory, input/output, and communications resources to perform efficiently any given scientific/data processing work load mix. This concept also facilitates major system extension without reprogramming or conversion.

The major system functions of processing, memory, input/output, and communications control have been separated into the following discrete functional modules:

- **Memory Modules** – to provide the required amount of core storage.
- **Processor Modules** – to provide the required amount of computational capability.
- **Input/Output Modules** – to control the required level of data input/output between memory and the peripheral and communications subsystems.
- **Communications Processor Modules** – to control data communications functions and provide service to remote users.

Multiple modules of each type may be configured on an information system to match the processing, memory size, memory access, and data input/output requirements of an

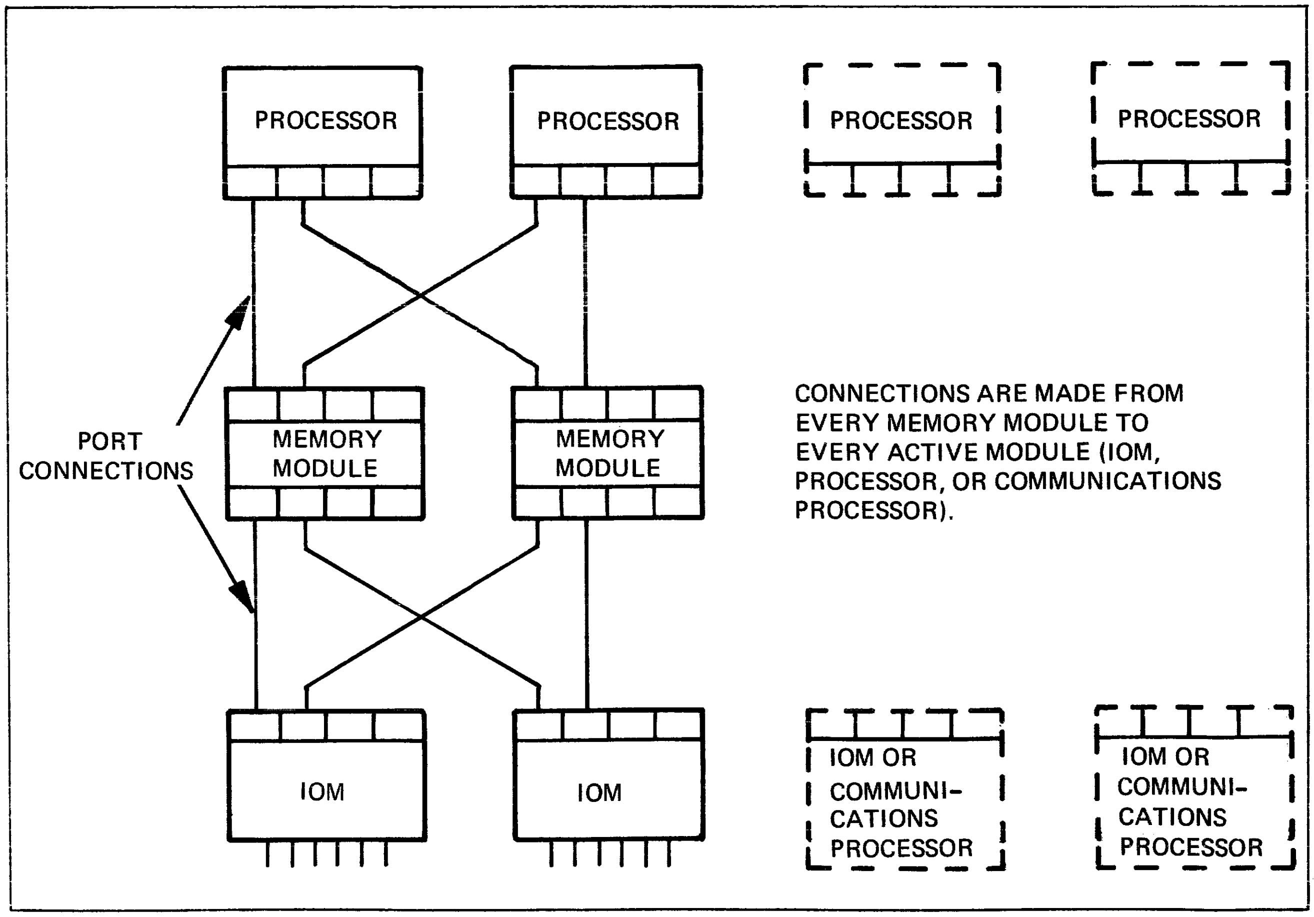

PROCESSOR
PROCESSOR
PROCESSOR
PROCESSOR
PORT CONNECTIONS
MEMORY MODULE
MEMORY MODULE
CONNECTIONS ARE MADE FROM EVERY MEMORY MODULE TO EVERY ACTIVE MODULE (IOM, PROCESSOR, OR COMMUNICATIONS PROCESSOR).
IOM
IOM
IOM OR COMMUNICATIONS PROCESSOR
IOM OR COMMUNICATIONS PROCESSOR

Figure 3. Functional Modularity

installation work load. This modular construction results in configurations that are tailored to the precise needs of an installation. System growth is readily accomplished by adding the appropriate modules as they are needed. Added reliability is a by-product of system extension, since all modules of a given type are identical and can provide backup for each other.

The operating system (GECOS) automatically adapts itself to control any standard equipment configuration.

Figure 3 illustrates the way the central system modules are interconnected to provide a high degree of functional modularity and backup capability. The solid-line figures represent a configuration consisting of two processor modules, two memory modules, and two input/output multiplexer (IOM) modules. Every processor and IOM connects through ports to each memory module; these connections permit access to the full range of processors, memory, and peripherals. The broken lines represent expansion capabilities. The system grows by adding more processors, memory, input/output multiplexer modules, or communications processors.

MEMORY MODULE

Each memory module is composed of a system controller and associated memory units. Series 6000 systems are "memory-oriented," permitting processor and IOM functions to execute asynchronously and simultaneously.

The memory module has neither program execution nor arithmetic capability, but acts as a passive system component. It serves the processor and I/O multiplexer modules which call upon the memory module to save or retrieve information or to communicate with other system components.

Each word in the memory module is composed of 36 bits plus a parity bit. For purposes of memory protection in multiprogramming, the memory is organized into blocks of 1024 words each. Each memory module may contain up to 131,072 words (128 blocks). Additional memory modules permit a maximum of 262,144 words per system.

The system controller has up to eight ports for connection to active modules and also contains 32 program interrupt cells. The eight ports have "wired-in" positional priority in the order of their numbers (0 . . . 7); thus, simultaneous requests are serviced in a predetermined manner.

Increased system throughput is achieved by operating the memory module and associated memory units on a 72-bit parallel basis. This corresponds to two instructions, two data words, or one double-precision fixed or floating-point number.

Systems with more than one system controller provide additional effective information rate, since each system controller operates independently and its functions can be overlapped with those of other system controllers.

Additional overlap is provided by the address interleaving feature of the Series 6000 systems. Address interleaving considerably reduces the possibility of the same memory unit being accessed in succession. Furthermore, the processor and system controller are especially designed to utilize memory accesses of two memory units in rapid succession. These two factors contribute to the higher access rates and effective memory cycle times of the Series 6000 systems. For example, Models 6070 and 6080 can have each of four memory units provide a complete memory cycle (read/write) of two full 36-bit words within a single basic cycle time of 500 nanoseconds – an effective rate of 62.5 nanoseconds per 36-bit word.

PROCESSOR MODULE

Series 6000 systems are highly modular, allowing the system configuration to be matched to the work load mix.
The Extended Instruction Set (EIS) processor is particularly well suited for a heavy business load, while the other models handle mixed work loads of business and scientific jobs.

Each processor module has full program execution capability and conducts all actual computational processing (data movement, arithmetic, logic, comparison, and control operations) within the information system. The processor, which communicates only with the system controller(s) and associated memory, consists of an operations unit and a control unit. The operations unit executes arithmetic and logical operations; the control unit performs instruction fetching, address preparation, memory protection, and data fetching/storing. Both units operate with relative independence and maximum overlap to provide the highest possible rate of instruction execution on the faster models.

The processors contain several special features that make significant contributions to the exceptional multiprogramming, high throughput, and rapid turnaround capabilities of the information systems.

These features are under the control of GECOS, which maintains automatic supervision and complete control of the multiprogramming/multiprocessing environment.

These features are:

Dual-Mode Operation. The processor has two modes of operation – master and slave.

- Master mode, reserved for GECOS, allows unrestricted access to all of memory, permits initiation of data input/output operations through the IOM(s), and permits the setting of control registers.
- Slave mode, used for the execution of all user programs, is also used by GECOS when appropriate. Slave mode operation restricts memory references to assigned program boundaries and causes all memory references to be relative to a base address register (BAR). Program execution time is strictly limited by a timer register. Also, program execution is limited to a subset of the instruction repertoire – control operations (such as input/output operations or setting the BAR and timer registers) cannot be executed in slave mode.

Dual-mode operation effectively restricts operating control of the information system's multiprogramming environment to GECOS.

Base Address Register (BAR). Each processor contains a base address register, which performs both address translation and memory protection functions in slave mode operation. (The BAR is not used in master mode processing.)

The BAR is set by GECOS prior to transferring control to a slave program. It contains the beginning address of the program (absolute) in memory and the number of 1024-word blocks assigned to the program. Program memory is logically and physically contiguous.

During slave mode execution, all memory addresses developed by a program are checked to ensure that the address is within the area of memory assigned to the program. If the developed address is within the program's area, the address is added to the beginning address value in the BAR to develop the true address, and the memory access is performed. If the address developed by the program is outside the area of memory assigned to the program, control automatically reverts to GECOS for appropriate action.

An important attribute of the BAR is the ability to move user programs in memory without address relocation merely by establishing a new BAR setting. This feature is used for program swapping and memory compaction.

Timer Register. Each processor contains a timer register, which initiates a program interrupt at the end of a pre-established interval of time. The interval is set by GECOS prior to giving control to a slave program. (The timer register can be set in master mode only.)

The timer register is used by GECOS to time programs for automatic termination, to prevent programs from monopolizing a processor, and to provide detailed accounting information on processor and peripheral use time.

Processor Faults. Sixteen special processing status conditions, termed "faults," cause interruption of sequential instruction exceution and transfer of control to one of 16 discrete fault vector locations for appropriate action by GECOS. Faults provide program control (e.g., arithmetic overflow), system control (e.g., timer runout or an attempt to reference outside of memory limits), and communication control (e.g., master mode entry).

Extended Instruction Set (EIS). The EIS models have processor instructions well suited for work loads with a predominance of business over scientific work. The EIS processor has all of the instructions of the other models plus many business-oriented features including: decimal arithmetic, powerful editing, mixed-mode operations, address registers, and extended instruction format (multiword instruction) with two or three addresses. For example, a single COBOL statement can be performed by a single instruction on the EIS processor but would require several instructions on a conventional processor. This not only reduces the memory required but also the execution time.

INPUT/OUTPUT MULTIPLEXER (IOM)

Each IOM module operates essentially as a stored-program device controlled by, and sharing memory accesses with, the processor modules. Data transfer operations are initiated by GECOS in master mode. (Data transfer operations cannot be initiated by a program in the slave mode.) Peripheral device operations are controlled by processor-prepared control word lists stored in the communications region in memory (referred to as "IOM mailboxes"). Data transfer operations are performed asynchronously with program processing.

Each IOM module in any configuration is directly coupled to each memory module, providing direct access to all of memory. Data transfer operations are controlled by lists of Data Control Words (DCW) which specify the areas of memory to/from which data is to be transferred. These DCW lists allow data to be gathered from, or distributed to, noncontiguous locations in memory.

Memory protection of data transfers is performed in much the same manner as in the processor. The BAR setting of the program requesting the I/O is inserted as part of the required instructions in the IOM mailboxes. Each DCW processed by an IOM is checked for address limits. If an out-of-bounds address is detected, the transfer is not performed and an appropriate interrupt is generated to the control processor. IOM communication with the processor is effected through the IOM mailboxes and through four discrete types of interrupts to one of 32 present interrupt vector locations. All data transfer and peripheral status conditions of interest are signaled to the control processor via the interrupt mechanism to maximize peripheral utilization and program throughput.

COMMUNICATIONS PROCESSORS

Communications capabilities are an integral part of the Series 6000 systems. Three data communications processors are available to provide the front-end processing functions – DATANET 355, DATANET 30, and DATANET 305.

The DATANET 355 Communications Processor is capable of servicing up to 200 remote terminals simultaneously. It is configured primarily on larger Series 6000 systems that have large communications requirements. Multiple DATANET 355s may be connected to a single system to permit even larger networks. The DATANET 355 connects directly to the system controller within each memory module. The communications link to mass storage for remote batch input/output functions is provided only by the DATANET 355 processor.

The DATANET 30 Communications Processor holds a long-time success record, fulfilling a variety of communications needs in the data processing industry. The DATANET 30 is currently available as a front-end communications processor for the Series 6000 systems to meet the needs of medium-size communications requirements. The DATANET 30 connects to a common peripheral channel of the Series 6000 Input/Output Multiplexer.

The DATANET 305 Communications Processor answers the need for small-to-medium communications requirements on a Series 6000 system. The DATANET 305 enables time-sharing functions for up to 12 teletypewriter users and remote batch processing for two users. The DATANET 305 permits growth compatibility to either the DATANET 30 or DATANET 355 when the communications requirements exceed those of the DATANET 305.

All three communications processors receive and process information from a variety of remote terminals for direct input to the System Scheduler, the Transaction Processing Executive, or time sharing or user application programs, and transmit output and control information to these terminals over common carrier communications facilities. As stored-program processors, all three communication processors provide the front-end processing benefits of more effective line handling discipline, greater terminal capacity and flexibility, less central processor overhead, standard system interface for remote input/output functions, and increased system availability to the remote terminal user. The division of the work load between the central processor and the front-end processor provides greater total system throughput.

SERIES 6000 CHARACTERISTICS

Table 1 gives the general characteristics of the various Series 6000 models. Section 1 points out that Models 6040, 6060, and 6080 have the Extended Instruction Set (EIS) processors, while Models 6030, 6050, and 6070, which are lower priced, do not.

TABLE 1. SERIES 6000 CHARACTERISTICS

	Model 6030/6040	Model 6050/6060	Model 6070/6080
Max. Memory Size in words (36-bit word)	131,072	262,144	262,144
Cycle Time (microseconds per two words)	1.2	1.2	0.5
No. of Data Channels	16	24	24
Max. Transfer Rate Per IOM (chars/sec.)	1.3M	3.7M	6.0M
Peripheral Capacity (subsystems)	16	24	24
I/O Compute Simultaneity	16	24	24
Programmable Registers	49/57	49/57	49/57
Floating Point	Yes	Yes	Yes
Memory Protect	Yes	Yes	Yes
Hardware Radix Conversion	Yes	Yes	Yes
Interleaving	No	2&4-way	2&4-way
Instruction Overlapping	No	Yes	Yes
Instructions per Second (max.)	340,000	550,000	1,400,000

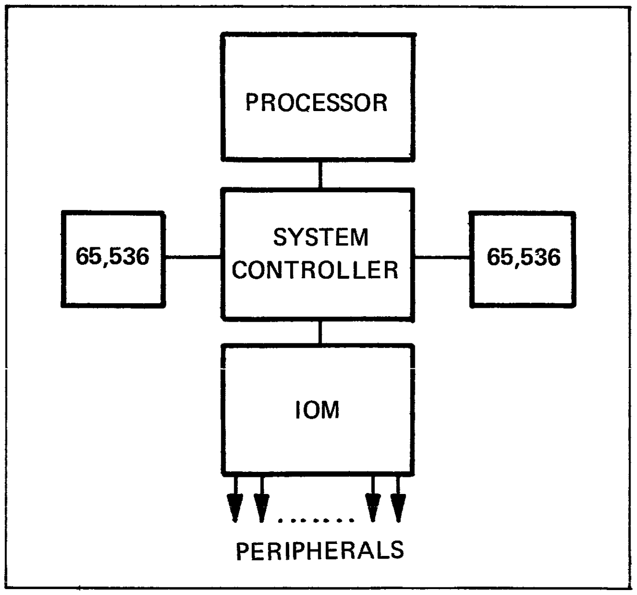


Figure 4. Basic System

Figure 5 shows the typical central modules for a multidimensional system. A basic batch-only system (see Figure 4) contains a single processor, a single system controller, and a single IOM with its peripherals, while the multidimensional system requires more memory, possibly two system controllers, and a DATANET communications processor. The communications processor is essential for all dimensions of the systems except local batch.

Figure 6 illustrates the central configuration of a multiprocessor system. Fundamental to Series 6000 operation is the fact that all active modules (processors, IOMs, and DATANET 355s) connect to all system controllers and thus have common access to memory and to the common data base. These connections permit GECOS to assign work to the available resources.

Although Figure 6 shows only two processors, one IOM, and one DATANET 355, the system allows as many as four processors, four IOMs, and three DATANET 355s, subject to the additional theoretical restriction of eight total active modules. It is unlikely that eight or more active modules would be required to perform the work load of a system.

Along with the functional modularity of the hardware configuration, there is complete program compatibility such that any program that is capable of running on one configuration can run on any other configuration (except for EIS programs, which must run on an EIS system).

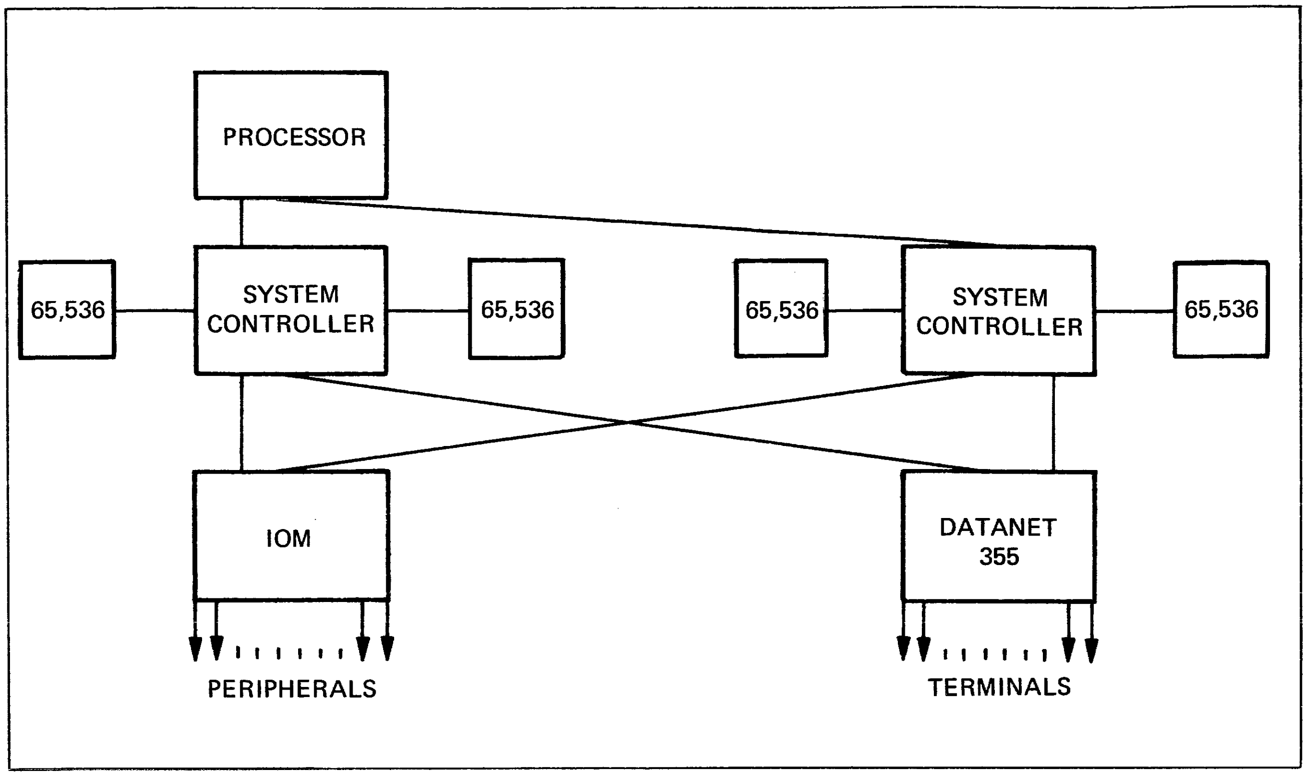


Figure 5. Multidimensional System

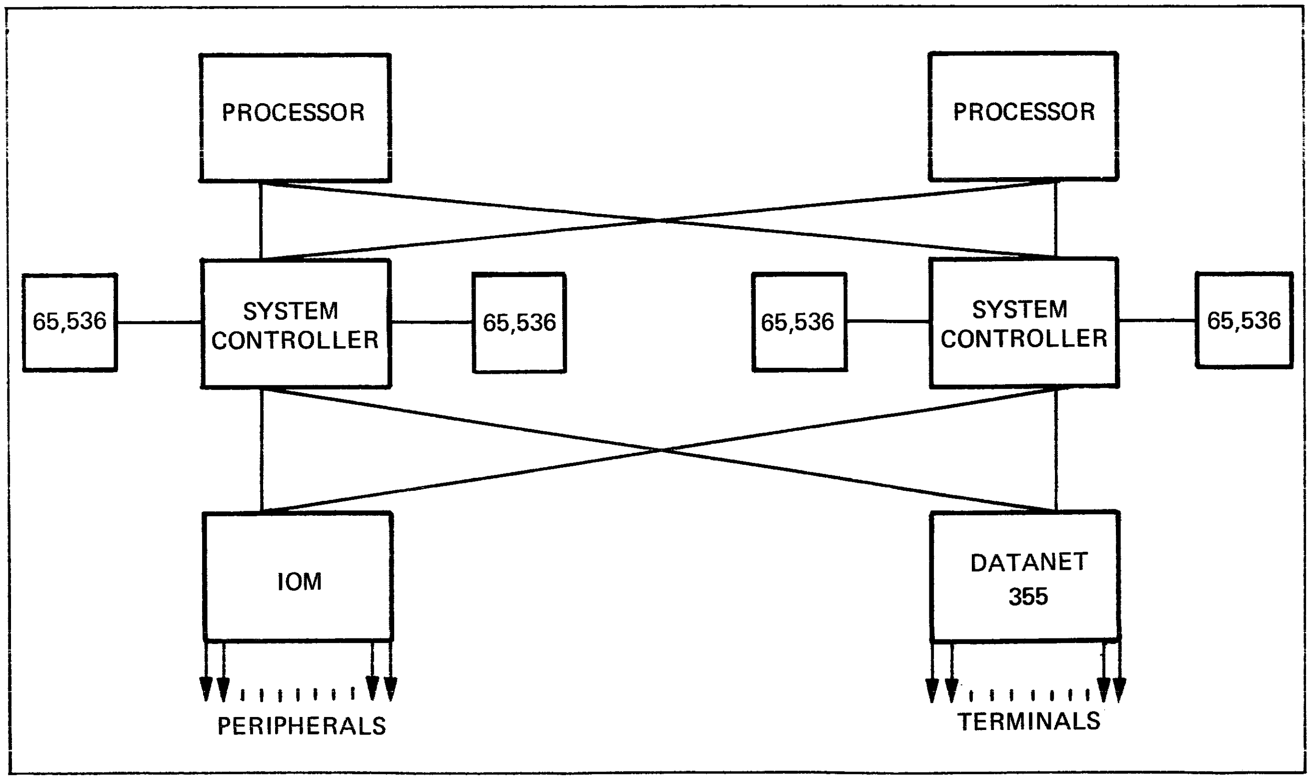


Figure 6. Multiprocessor System

Section III

General Comprehensive Operating Supervisor (GECOS) 6000

The General Comprehensive Operating Supervisor (GECOS) 6000 is the most versatile and proven executive system in the industry today. The Honeywell Series 6000 Information Processing Systems, with GECOS, incorporate all of the proven multiprogramming, multiprocessing, local and remote batch processing capabilities of its predecessors, *plus* the latest concepts of file system design, communications, and time sharing.

Under GECOS, the dimensions of batch processing, remote-access processing, transaction processing, and time sharing are integrated, thus creating a level of computational effectiveness and flexibility not attainable with multiple-system installations. This merging of processing dimensions in concurrent operation on a single system provides increased effectiveness and flexibility in two significant respects:

1. The respective amounts of batch processing, transaction processing, and time sharing can be tailored to individual installation requirements and can be dynamically variable throughout the processing day.
2. The file system (in which files from all processing dimensions are stored in a common file structure and which provides access to any file in the file system from any processing dimension) provides increased opportunities for data base effectiveness. The overhead of duplicate data bases and of transferring files between systems has been designed out of existence. Even more important, the centralized data storage provided by the file system is the necessary first step toward providing management with timely information upon which to base management decisions.

FEATURES OF GECOS

GECOS provides the user with a large number of benefits The major functional characteristics include the following:

- **Resource Management.** GECOS maintains current status of all system resources (processors, memory, and peripherals), allocating available resources to jobs with proven allocation algorithms for optimum resource utilization.
- **Job Scheduling.** GECOS, via the System Scheduler, accepts a virtually unlimited number of user jobs from multiple local and remote input devices simultaneously. Up to 63 jobs can execute concurrently or simultaneously (system resources permitting). GECOS also performs automatic output media conversion for completed jobs. GECOS utilizes the most advanced multiprogramming and multiprocessing techniques to maximize job throughput, minimize turnaround time, and optimize resource utilization.
- **Priority Allocation.** GECOS contains an effective job priority structure that allows "hot" jobs to be expedited. Priority jobs receive first consideration in resource allocation, which can cause the temporary suspension and removal from memory (swapout) of programs in execution to make room for a high-priority program.
- **Ease of Use.** GECOS provides the development programmer with a complete, logical interface to his program and file structuring. All physical system and peripheral device constraints and programming considerations are handled by GECOS. Remote locations interface directly with the central system without pre-established operator assistance or coordination. Concurrent time sharing places the full power of the information system at the fingertips of the engineer, the programmer, or the manager.
- **Data Base Management.** The file system provides a permanent on-line repository with millisecond access for data files from all modes of processing. Full user cataloging, password protection, access control, and file sharing capabilities are provided. Multiple programs in multiprogramming execution can read a file concurrently. I-D-S (Integrated Data Store, a Honeywell data management system) files can occupy space in the file system.
- **Security Control.** GECOS positively prevents all programs within the system from accessing unauthorized memory areas or data files. Programs in memory execute within firm boundaries. Data files are protected by passwords and positive checking of all data transfer operations. Remote users must be validated before they can access the system.
- **On-Line System Development.** The file system and the reactive terminal interface upon which GECOS Time Sharing is implemented provide a powerful, yet easily adaptable, base for the development of on-line terminal applications for management information systems (MIS) or information storage and retrieval.

BATCH PROCESSING

Job/Activity Definition

As a preface to describing the general flow of user jobs through GECOS, some definitions of terms are given below.

- **Multiprogramming.** The concurrent processing of many programs residing in core memory to maintain the highest possible amount of simultaneous input/output and to maximize processor utilization

- **Multiprocessing.** Two or more processors simultaneously executing programs in memory to gain greater throughput.
- **Activity.** A single program (such as a Fortran compilation or an object program execution).
- **Job.** A set of related activities that together constitute a logical computer application, and that must execute sequentially because of their logical relationship.

Job Flow

Local batch jobs, remote batch jobs, and transaction processor slaves are processed in the same manner by GECOS. Remote jobs differ only in the input and output routines that interface with the communications processor.

Batch jobs flow through five phases of GECOS in the course of processing:

- Job Input (Input Media Conversion)
- Activity Allocation
- Activity Execution
- Activity/Job Termination
- Job Output (Output Media Conversion)

The major functions of each phase are outlined below.

Job Input

User jobs enter the system simultaneously from multiple central and remote peripherals. Incoming jobs are placed in temporary files in the file system. These jobs can be entered directly to the allocator queue or via the System Scheduler.

System Scheduler

At the user's option, the jobs can enter the system via the System Scheduler rather than directly into the allocator queue. The System Scheduler is a powerful feature that enables various job priorities to be established and utilized in organizing the work load for the central system. The Scheduler can have a significant impact on job turnaround time. This effect is most pronounced if the higher-priority activities are relatively short in comparison to activities of lesser priority. This disparity is often encountered in practice; for example, transaction processing is usually more time-critical than batch processing.

Two main features are provided by the System Scheduler:

1. A virtually unlimited number of jobs can be entered in the system at one time.
2. An arbitrary number of job streams can be established in relation to user organizations served by the installation. Each organization can then establish priorities in the job streams to which it is assigned.

Two special scheduling streams are provided to facilitate "express" jobs and "hold" jobs. A "hold" job can have a "hold until" date and time associated with it; this is the date and time the job will be entered into its actual scheduling stream. "Express" jobs are those jobs that require a limited amount of system resources. The number of express jobs that can be run at any one time is determined by the user, as are the criteria for determining an "express" classification.

Jobs are scheduled using the following criteria. Scheduling streams are queued for an eligible job on a rotational basis. The highest priority job is selected from the queued stream; jobs of equal priority are selected on a first-in, first-out basis.

Allocator Queue

As the job sequence number card is detected for each job in the temporary file space, it is assigned an internal program number (1-63) and placed in the allocator queue according to its relative priority. All of the time-sharing jobs are considered as a single job of the 63 programs in the queue. Jobs entering the system may carry an initial priority established by the System Scheduler or the Transaction Processing Executive; if not, GECOS will calculate a priority for the job based on its resource requirements.

The major features of allocator-queue processing are:

- Multiple job input streams from a combination of central system and remote sources.
- The job image in the file system is preserved intact throughout job processing. A job or an activity within a job can be restarted when necessary.
- Each job entering the system is scanned for gross resource (peripheral and memory) requirements to ensure that the currently operational system configuration can accommodate the job. A single job that exceeds configuration capabilities is immediately deleted from the system with appropriate operator notification.
- Jobs entering the system are also screened for the estimated amounts of system resources required; these are compared to limits established by operations. Jobs exceeding these limits are run only with operator approval. These limits, changeable during processing, allow emphasis on small jobs.

Activity Allocation

The allocation phase of GECOS is activated (1) when a new job is placed in the queue by the job input phase and is a candidate for allocation, and (2) when resources are released that can be assigned to a new activity.

Allocation of system resources is performed in two subphases:

1. Peripheral Allocation
2. Memory Allocation

Peripheral Allocation Subphase. The queue is ordered according to job priority. Peripheral allocation routines examine each job in sequence, allocating peripherals to the current activity of a job.

Peripherals are not allocated to an activity until the activity's total peripheral requirements are available. Instead, the job is bypassed until the next allocation cycle in favor of lower-priority jobs whose requirements can be satisfied. If the job proves difficult to allocate, provision is made to block allocation of lower-priority jobs until sufficient resources become available for the job.

Special features of peripheral allocation are:

- *Balanced Peripheral Allocation.* File allocation on peripherals of a given type is balanced across all channels to which that type of peripheral is connected. On multidevice channels, file allocation is also balanced across all devices on that channel. This technique maximizes device and channel utilization and minimizes multiprogramming interference.
- *Hold Status for Demounted Files.* When a job requiring demounted files (e.g., magnetic tape) enters the system, GECOS notifies the operator of the files required. The job is not a candidate for allocation until the operator indicates that the files have been retrieved from the library.
- *Demounted File Request-by-Name.* The file system provides for cataloging demounted files. When a job in allocation requests such a file by name, GECOS retrieves the current reel number from the file system and notifies the operator that it is needed. This feature provides security for demounted files in that (1) the files can be protected by passwords, and (2) the correlation between file name and physical reel number is stored in the file system.
- *Named Device Request.* Programmers may request that a specific device be allocated to a job (instead of the normal method of specifying only the type of device needed). This feature is useful when a device with a special option is required (such as a printer with a special character set).
- *Dynamic Allocation of Implicit Files.* The implicit files used by software (such as the files used to hold user jobs or those used as intermediate files during compilation) are allocated dynamically. This technique handles variable-length files with effective storage utilization.

An activity is a candidate for memory allocation only after peripheral allocation is complete and the activity is ready for execution.

Memory Allocation Subphase. As defined previously, the concept of multiprogramming is based on the concurrent execution of multiple programs resident in memory. Those programs ready for execution utilize the processor, while others await the completion of input/output operations.

Effective multiprogramming demands full memory utilization, which in turn demands efficient handling of variable-length programs GECOS allocates memory in 1024-word blocks. A user activity may request any number of blocks, up to the full size of slave memory (total memory less the amount occupied by the resident portion of GECOS). The blocks allocated to an activity are contiguous. When an activity terminates, its blocks are made available for reallocation. When required, GECOS compacts the activities in memory (moves them together) to make multiple noncontiguous areas contiguous. With this technique, all available memory can be effectively utilized.

GECOS features a 3-level memory allocation algorithm based on job priority. An activity of a job with normal priority will be allocated memory only if a contiguous area of memory large enough to accommodate the activity is available. At a higher level of priority, the activities of other jobs in memory will be compacted, making noncontiguous areas contiguous so that the activity in allocation may be allocated. At the highest level of priority, activities of other jobs in execution will be temporarily removed from memory (swapped) to make room for the activity in allocation. Each time a job is thus bypassed for memory allocation, its priority is increased.

When memory allocation is successful, the activity is placed on the Dispatcher queue (a queue of all activities in memory that are ready for execution).

Activity Execution

During execution, each activity is executed under the supervision and control of the Dispatcher. The Dispatcher keeps as many system components as possible in simultaneous use. It accomplishes this by selecting the highest-priority activity that can make effective use of the processor and peripheral subsystems and transferring control to it.

This Dispatcher queue is dynamic. Activities are in the queue only if they are ready to execute. The queue is sequenced according to the priority of the activities for execution; the highest priority is at the top of the queue. Dispatching is, therefore, a very simple and straightforward mechanism of picking the top entry from the queue and dispatching to it. To maintain high utilization of physical resources, the Dispatcher maintains a list of user programs that require very little processor time, only enough to keep peripherals in full operation. The Dispatcher will select one of these for execution (a "courtesy call") if the user program at the top of the queue was not in a courtesy call. When dispatching to a user program, the Dispatcher sets the base address register (BAR) and the time register and sets slave mode in the process of transfer.

In multiprocessor systems, the Dispatcher selects activities from the same queue for each processor. All processors can execute both GECOS and user programs; however, only the control processor will respond to the peripheral interrupts. GECOS redispatches at the completion of interrupt processing. Since the Dispatcher queue may be modified in the course of interrupt processing, the priority of job execution is truly interrupt-oriented. This feature facilitates the development of responsive, event-oriented systems.

Swapping may occur as a normal function of activity execution, if the priority of a job in the input queue is sufficiently high to demand this service. GECOS performs a great many service functions for activities in execution that (1) the activities cannot perform directly from slave mode (e.g., input/output operation) or (2) that relieve the user programmer from detailed knowledge of system operation (e.g., calling in of overlays).

Centralized Input/Output. Input/Output (I/O) operations proceed simultaneously and asynchronously with processing under the control of the input/output multiplexer (IOM) module(s). I/O operations are initiated solely by GECOS in master mode. The hardware prohibits initiation of I/O operations from slave mode. The IOM modules execute I/O operations in accordance with control words prepared by GECOS and stored in an IOM "mailbox" area in memory.

In combination with the File and Record Control routines, GECOS provides a complete logical interface for file processing. All I/O requests are checked to ensure that the device or file is allocated to the requesting activity and that the data buffer is within the requesting program's memory allocation.

I/O requests are processed to the extent possible when received and are queued for each peripheral subsystem. These techniques allow interrupt processing to be reduced to a minimum and new I/O operations to be initiated quickly. GECOS I/O routines provide for device interchange. When an allocated peripheral device (e.g., card reader, printer, card punch, or magnetic tape) is operating at an error rate above a predetermined threshold the operator can request that a new device be allocated for the file being processed. The operator can then move the file to the new device and continue processing. GECOS keeps track of the number of files and the number of records within a file that have been read or written on magnetic tape and uses this information to reposition a magnetic tape after interchange.

GECOS provides a method of processing on disk that simulates the serial mode of processing normally peculiar to magnetic tape. This mode of processing not only provides for device independence but also allows the opportunity to reduce program setup time by eliminating the use of magnetic tape for some files.

Activity/Job Termination

An activity in execution may terminate normally or abnormally. The processing of each type is different.

- **Normal Termination.** Normal termination processing first looks ahead to the next activity (if there is one). If it is a compilation activity of the same type as that terminating, the new activity is merged with the present activity (initiated immediately) using the same resources used by the present activity. If the next activity is not the same, the operator is notified of files that require demounting. If a file in an activity is to be used in subsequent activities, a notation is made in the file system to save the file. An accounting record is written on the System Output (SYSOUT) file, itemizing the system resources used by the activity. The allocation phase of GECOS is notified that the resources used by the activity are available for reuse and that the next activity is a candidate for allocation.

  If end-of-job, the job is removed from the job stack.

  GECOS runs successive compilation activities of the same type as a single activity, thus avoiding the deallocation of resources at the end of a compilation only to reallocate the same type and amount of resources to the following activity.

- **Abnormal Termination.** Abnormal termination (abort) processing may be initiated by the activity, or by GECOS, when the activity tries to execute an illegal operation (e.g., an attempt to access memory outside its boundary).

  The aborting activity can at its option have a dump of its allocated memory written on the output collector (SYSOUT). The programmer may also define abort subactivities (useful for dumping data files) that are executed only when an abort occurs.

  As with normal termination, the resources used by the aborting activity are released for reallocation. Compilation activities following an aborted activity are executed. Whether subsequent object program execution activities are executed is a programmer option.

Job Output

GECOS includes a system output phase (SYSOUT) which consists of an output collection function and an output dispersing function.

Multiple output files (for printing and punching) from all activities in execution are collected in the file system, along with those generated by GECOS in the course of job processing (e.g., accounting reports and abort memory dumps). This mechanism avoids the necessity of dedicating peripheral devices to small-volume print or punch files.

The output-dispersing routines read the data collected in the file system and batch the output from many jobs on multiple printers and card punches. This provides a more effective utilization of peripherals.

Output printing and punching is automatically performed concurrently with the execution of other jobs in the system and the entry of still more jobs into the input stack.

MULTIPROCESSING

The effectiveness of multiprocessor systems is largely a combination of three factors:

- A sufficiently large number of activities in memory and ready for execution (as opposed to those waiting for I/O) to keep the processors busy in the average mix of jobs in the system.
- A sufficient number of memory accesses available to each processor to allow it to execute without delay.
- An operating system design that does not serialize processing each time a supervisory or service function is performed for an activity in execution.

The design of GECOS is especially effective in multiprocessor configurations. The following specific points are worthy of special mention:

- The organization and memory utilization of GECOS distributes user programs, tables, buffers, and service subroutines throughout memory, minimizing multiprocessor interference for memory accesses.
- A minimum of multiprocessor "gating" is required. (A "gate" is a software switch that prevents multiple processors from accessing a common table or entering a common routine if it would be harmful to do so.)
- Where interference would be likely, most major program service functions are performed in parallel by providing individual copies of the service subroutines to each program requiring the function.

REMOTE PROCESSING

Remote processing is available in a Series 6000 system by including one or more communications processors in the configuration. GECOS controls multiple communications processors, integrating remote processing with central-site processing in concurrent operation. Remote processing capabilities include remote batch, remote access, transaction processing, and time sharing.

Remote Batch

The full capabilities of the Series 6000 Multidimensional Information System are available to the remote batch user. Any job that can be entered directly at the central system can be entered remotely from a remote batch computer. A remote batch job differs from a local batch job only in the GECOS job input/output routines that interface with the communications processor. Once inside the central system, local batch and remote batch processing are identical.

Remote batch jobs have several options available for the output files they generate. These options are selected by control cards as follows:

- Return to sending terminal.
- Enter into the file system.
- Output at central site.
- Hold until terminal calls back.
- Send to another terminal.

Remote Access

The remote access capabilities satisfy the requirements for data base inquiry, remote data base management, data collection, or other "on-line" applications.

Remote access provides direct terminal access to a program in execution. The program, written in any batch programming language (Fortran, COBOL, GMAP, etc.) can be submitted as a job via local batch, remote batch, or time-sharing batch.

The terminal in remote access effectively becomes an online peripheral to the activity in process. The activity can send output to, and receive input from, the terminal or terminals.

Transaction Processing

The Transaction Processing Executive controls the concurrent execution of application programs in the GECOS multidimensional environment, providing on-line, real-time capability. Transactions are related to events in the user's business, such as order entry or customer file update. Some characteristics of transaction processing are:

- Transactions are submitted by the *end user* of the system (e.g., bank teller, stock clerk, sales manager) with no need for specialized knowledge of the information processing system or its operation.
- The transaction is initiated by the user. Subsequent actions are implicit to the transaction itself. No specific command language is required by the user.
- The scheduling of tasks to be performed is not the responsibility of the transaction submitter.
- The Transaction Processing Executive does not use system resources when the transaction system is not activated.

The executive operates as a privileged slave in the direct-access mode and will generally be in execution. Control over the execution status of the executive is from the operator's console. The application programs will not generally be in execution, but will be activated by the executive selectively upon receipt of transactions.

The transaction processing application programs are normal user application programs and may be written in any of the Series 6000 languages including the data base managers. The powerful resource management and service functions of GECOS are available to maximize throughput.

Time Sharing

GECOS Time Sharing is designed for installations that want to provide time-sharing service to their users without disruption of their batch processing commitments. The portion of the Honeywell Series 6000 system dedicated to time-sharing processing options is variable throughout the processing day.

All of the catalog structuring, source/object file storage, file protection, file sharing, and access control capabilities of the file system are available to the time-sharing programmer. A special catalog/file maintenance routine is provided to perform user catalog and file maintenance functions from a time-sharing terminal.

The Time Sharing Executive performs the functions of selecting, allocating, dispatching, and swapping time-sharing user programs. The executive is structured as a single privileged slave program operating under the control of GECOS. It, in turn, suballocates memory and subdispatches the processor to individual time-sharing user programs. In the process of subdispatching, the Time Sharing Executive establishes a new BAR setting around the user program to be executed, ensuring the integrity of other user programs in memory.

The Time Sharing Executive also performs various services for individual programs, including file system I/O, terminal I/O, and creation and modification of files, catalogs, and their security definitions. It also accounts for resources used by the individual time-sharing users.

Time-sharing user memory is allocated to individual user programs for execution. Several programs can occupy portions of this area. A program can be swapped to allow another user of higher priority to be allocated memory space.

GECOS Time Sharing provides standard time-sharing capabilities (BASIC, Fortran, and Text Editor); and, in addition, offers time-sharing batch and the capability of generating time-sharing systems through the load and execute subsystems.

GECOS Time Sharing is designed to encourage development of terminal applications, in keeping with the present batch mode philosophy in which the manufacturer supplies the operating system, the commonly used compilers, and utility routines, while the user develops the specific applications required for his business.

The structure of GECOS Time Sharing and the integration of time-sharing files in the file system facilitate user extension. Users can add primitives, commands, or subsystems to GECOS Time Sharing, or they can replace the Time Sharing Executive with an executive that satisfies their particular terminal application requirements.

COMMUNICATIONS/MASS STORE LINK

The communications/mass store link offers users of the Honeywell Series 6000 Information Systems an additional method of increasing the system availability and efficiency of remote batch processing. In its simplest terms, the communications/mass store link consists of a DATANET 355 and a Series 6000 central system sharing a mass storage subsystem (e.g., DSS270, DSS167, DSS180, DSS190) as shown below.

This configuration is designed to serve two objectives:

- Provide the capability for a remote batch terminal to transact its input and receive its output even when some modules of the Series 6000 system are not operational.
- Provide a more efficient mode of processing by minimizing the flow of input job streams and output through the central system processor and memory.

The communications/mass store link can be initiated at startup as an installation option. The amount of mass storage to be shared is also an installation parameter.

Input Functions. The system functions performed through the link will be roughly equivalent to those presently provided by remote batch input and by SYSOUT for remote terminals. Time sharing and remote access communications will operate directly through the memory interface to maintain reactive terminal interface response time. During normal operation, all central system modules (including the DATANET 355) are functioning; remote batch input will be received by the DATANET 355 and placed on the mass storage device following a terminal verification procedure conducted by GECOS. When it has completed placing a job on mass storage, the DATANET 355 will notify GECOS to request processing of the input file.

On occasions when the central system is not operational, password verification will be deferred until it becomes available. In this situation the DATANET 355 will maintain a file of jobs received to transmit to the central system when it again becomes available.

Output Functions. In the normal mode of operation, the Series 6000 system SYSOUT disperser performs its usual functions, up to the point where the actual transmission of SYSOUT data would begin. At this point a message is prepared for the DATANET 355 indicating the location of the report to be transmitted. When the DATANET 355 has completed transmission it will notify GECOS that the file

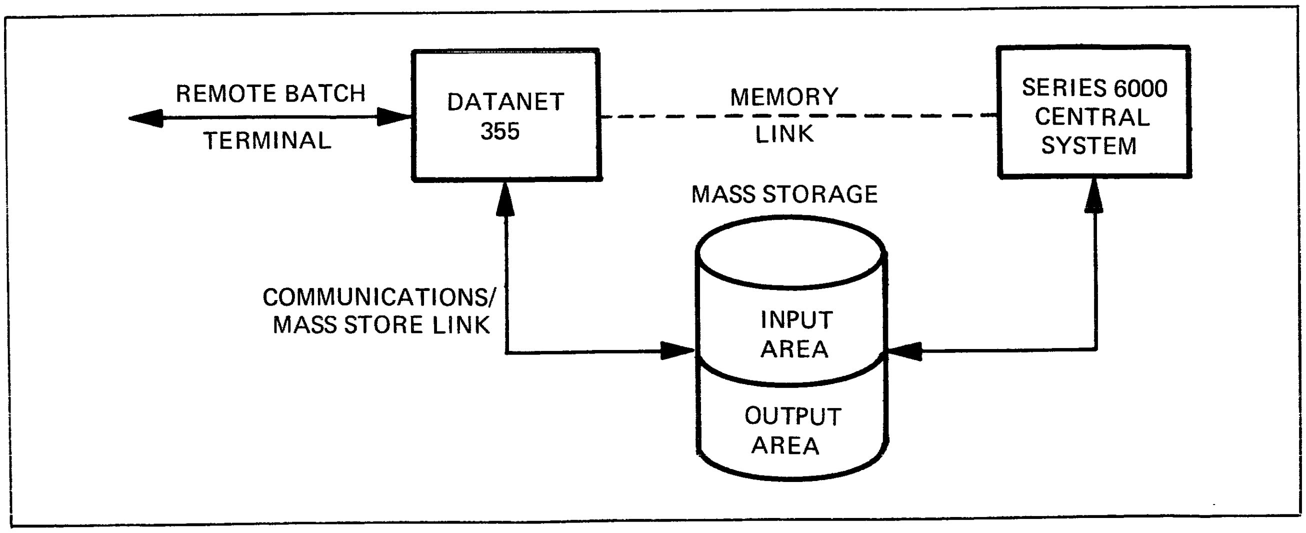


Figure 7. Communications/Mass Store Link

space in which the SYSOUT report was contained may be released.

If the central system is unavailable at the completion of the transmission, the DATANET 355 will log the message for transmission when it is again available.

Benefits. This hardware-software combination of a link from the communications processor to mass storage results in increased efficiency of system resource utilization. Communication data for mass storage is not required to pass through the central system; hence, the central processor is free to accept other work. The link eliminates redundant functions; the DATANET 355 can transmit to or from mass storage without requiring the central system to read and write the data. The link reduces Series 6000 processor and memory load for remote job handling. This provides additional capability for use by customer applications.

A primary benefit is an increase in system availability to the remote batch terminal user. Remote input/output can continue through the communications processor when the central system is off-line.

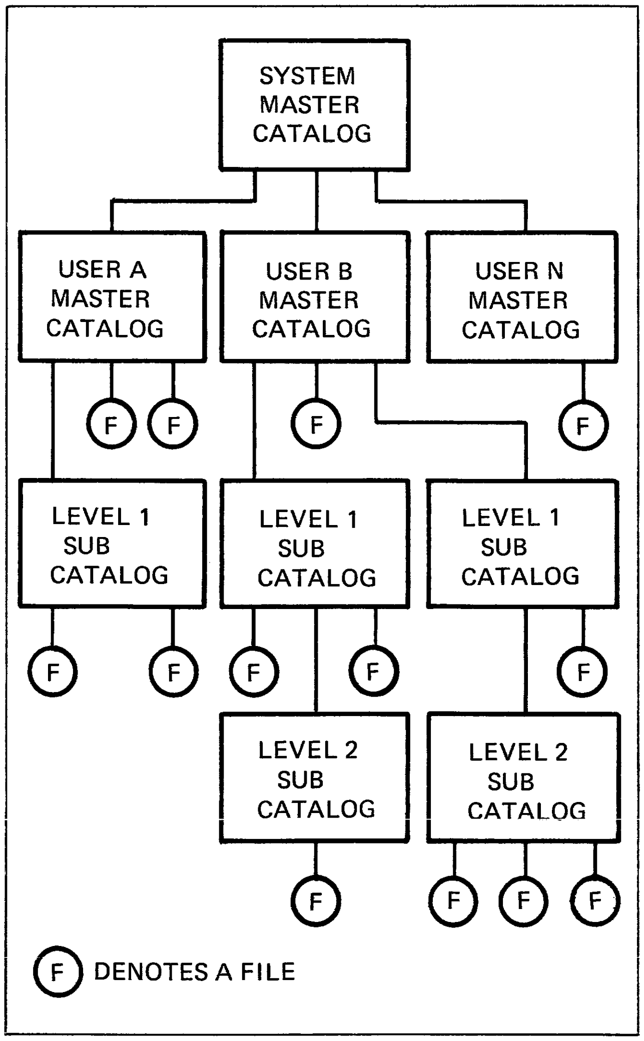


Figure 8. File System Structure

FILE SYSTEM

The file system is a hierarchical, "tree-structured" design with multilevel cataloging capabilities stemming from a system master catalog that lists all users known to the file system. Figure 8 illustrates the file system organization.

File sharing is also provided by the file system. Users can provide for general sharing of their files or can designate specific users who may access their files. They can also grant full read/write access or read-only access.

All catalogs and files can be protected by passwords. When a file or a catalog is protected by a password, the user must provide that password to gain access to the file.

The file system is completely transparent to the casual user; yet it provides complete subcataloging protection, access control, and file-sharing capabilities for the more sophisticated programmer.

The *type* of access that a user can have to a file is also controlled. A file can be allocated to several user programs concurrently for read only, but will be allocated to only one user program at a time for update. A user requesting read-only access will be prevented from writing on the file.

File processing under GECOS from a programming standpoint is completely logical. The programmer processes his files as a logical entity, either sequentially (as in tape processing) or randomly. The physical characteristics and restrictions of the device on which his file is physically located are handled entirely by GECOS.

A combination of hardware and operating system features prevents user programs from directly accessing random storage (and all other peripherals as well). A utility routine is therefore provided that will allow a programmer to maintain his catalogs and files.

The file system is a device-independent structure. The specific devices that constitute the file system may include a combination of Series 6000 disk devices. The file system catalogs and allocates the storage of all devices that are a part of the file system and provides for the movement of files between different types of devices.

File system maintenance is device-oriented. The catalog entries for a file are on the device on which the file resides. This allows the restructuring of a file system device (i.e., repacking the device after files have been purged) to be accomplished one device at a time, while production jobs continue to access other portions of the file system.

The file system accommodates files organized under the Integrated Data Store (I-D-S) file management technique. GECOS provides concurrent multiprogram access to I-D-S files. (For more information on this powerful file management concept, refer to the Series 6000 Integrated Data Store reference manual.)

GECOS also provides a journal file option for re-creating a file, if the integrity of the data base requires a history of all modifications, and the capability to rebuild that file.

TOTAL ON-LINE TESTING SYSTEM (TOLTS)

The Series 6000 incorporates a new, advanced concept in system availability. The major features of this concept are test and diagnostic visibility and on-line maintenance. Such aids as continuous parity from the peripheral media to the main memory, dynamic error thresholds, and central system processor fault register ensure early fault detection and data integrity in the Series 6000 system. On-line test and diagnostic programs operating concurrently with user programs maximize system availability. Automatic retry on input/output commands is designed to minimize the impact on the operating software of peripheral malfunctions. Greatly increased dynamic hardware visibility has been achieved by the inclusion of processor history registers that record the internal machine states of the last 16 steps performed. These history registers are dynamically unloaded and stored on media for a comprehensive trace of system operation and diagnosis.

The total Series 6000 system is oriented toward optimizing user availability concurrent with necessary maintenance functions. Various portions of the system can be devoted to routine preventive maintenance checks while running user programs. Spot diagnostics can exercise portions of the input/output system interlaced with, but not conflicting with, user input/output operation.

All of these advanced features are incorporated in the Series 6000 system to provide the greatest possible system availability. System interruptions are minimized by this error recovery and maintainability concept.

The GECOS operating software is integrally incorporated with the total maintenance and recovery concept. The Total On-Line Testing System (TOLTS) is composed of four major subsystems. These are: Peripheral On-Line Testing System, Communications On-Line Testing System, Main Frame On-Line Testing System, and Remote On-Line Testing System. This Total On-Line Testing concept is a Honeywell first in the computer business; it permits up to eight concurrent diagnostic programs to be operating with user programs in the Series 6000 systems.

Some of the major benefits of the Total On-Line Testing System (TOLTS) are:

- The test system provides a complete library of comprehensive on-line "test pages" designed especially for each system module or peripheral.
- It provides manual "test pages." These permit the maintenance engineer to design and execute his own test programs using the conversational Test and Diagnostic Language – concurrent with the user's operations.
- The test system is called in quickly from system mass storage.
- Eight test and diagnostic "test pages" can be run concurrently with user operations.
- It limits the amount of main memory used by dynamically allocating and releasing memory. Only the required amount is used.
- Master auxiliary or teletypewriter consoles can be used to call TOLTS into execution.
- All operational and error messages for the "test page" are directed back to the console that initiated the original request and to any other console for monitoring.
- Copies of the error messages can be directed to the system accounting file. Or, by completely bypassing the console, messages may be used and accumulated on the file for later analysis on demand.
- GECOS and TOLTS monitor all error status signals and notify the system of malfunction on a dynamic basis. Error thresholds are set; when they are exceeded TOLTS can automatically request test and diagnostic assistance or optionally print a message to the operator on the system console. This permits the rapid call-in of the appropriate on-line test and diagnostic program for further fault isolation.

An additional advantage is achieved by the Test and Diagnostic System in that tests are executed in the same environment as the user programs. Since TOLTS is on-line and an integral part of the total operating system, the user is able to establish a higher equipment confidence level.

The system reconfiguration capability permits any processor to become the control processor. This will permit an easy way to graceful degradation in a redundant system configuration without loss of user operation.

Peripheral On-Line Testing System

GECOS includes a comprehensive Peripheral On-Line Testing System, comprising an executive and a set of test and diagnostic routines. Special interfaces enable the diagnostic testing of peripheral devices concurrent with the production work load. Furthermore, GECOS accumulates recovered error statistics for continual measurement of peripheral device performance. Through subsequent analysis, problems can be detected and corrected before they become critical.

Communications On-Line Testing System

This new extension of the TOLTS permits test and diagnostic programs to be run on all DATANET 305s, DATANET 30s, DATANET 355s, High-Speed Line Adapters, Low-Speed Line Adapters, teletypewriters, DATANET 355 card readers, DATANET 355 consoles, and DATANET 355 GERTS input/output systems. Again, these tests can be under either local console or remote teletypewriter control. This Communications On-Line Testing System opens a new vista to system maintainability and availability.

Main Frame On-Line Testing System

This system extends the original OPTS-600 system on the Series 600 to the central system modules of the new advanced Series 6000 systems.

Main memory storage modules, system controller modules, control processor modules, input/output channels, and input/output multiplexer modules can be allocated to the Total On-Line Testing System concurrently with user operation. Now, for the first time, a large-scale multiprogramming, multiprocessing system can be maintained with minimal off-line maintenance.

Remote On-Line Testing System

For those systems that have remote terminal capability, TOLTS provides the ability to use a remote teletypewriter terminal as if it were a local system console. For those problems that require a maintenance specialist, it will no longer be required to wait for the specialist to travel to the malfunctioning site. Instead, by using a standard teletypewriter and the telephone network, the specialist can dial into the computer system and be automatically connected to TOLTS. The specialist will then have the full range of operating features of TOLTS programs plus his own designed programs available to him. All error messages for the module test will be directed to the local console for the site maintenance engineer, and to the remote teletypewriter for the maintenance specialist, with the additional ability to transmit copies of the TOLTS messages to still other teletypewriters for monitoring purposes. By getting first-hand knowledge about the malfunction via remote TOLTS, the specialist will be able to instruct the site maintenance engineer as to the corrective action to be taken. By resolving the problem in this manner, system down-time will be considerably reduced since the malfunctioning module is out of service for a shorter period of time. TOLTS provides the maintenance engineer with the capability of accumulating all TOLTS error messages on a dedicated system accounting file. The accumulation of error messages and related diagnostic data can be made available to the maintenance specialist via the remote teletypewriter console. This advanced system concept is the result of the continuing evolutionary maintenance techniques developed on the Honeywell Series 6000 systems.

OFF-LINE TEST AND DIAGNOSTIC SYSTEM

The Off-Line Test and Diagnostic System for the Series 6000 tests all hardware subsystems. The processors, system controllers, and memory units are normally tested in a single program environment so that problems can be detected and isolated easily. These programs are run sequentially, and each program tests a specified portion of the logic.

In addition to these main frame programs, a full set of peripheral programs is offered. The peripherals can be tested individually or in a systems environment. In the systems environment, the functioning of each peripheral is tested, as well as its interaction with other peripherals. To further simulate the normal software environment, main frame test programs can be executed simultaneously with peripheral programs in any mixture of processing and I/O desired.

All of these programs are designed for ease of use by the field engineer. In case of error, comprehensive error information is printed so that the field engineer can correct the problem immediately. For peripheral problems, a test and diagnostic language allows the field engineer to write his own testing sequences quickly and easily as further information about the problem develops.

In summary, Honeywell offers a full set of main frame and peripheral programs for comprehensive testing of the entire Series 6000 system.

Section IV
Software Overview

Honeywell Series 6000 Information Systems have complete software libraries, fully integrated under GECOS. A system library can be partitioned into four categories:

- Language Processors
- System Software
- Time-Sharing System
- Applications Software

LANGUAGE PROCESSORS

The Honeywell Series 6000 Information Systems incorporate a number of language processors, as described below.

Macro Assembler

The macro assembly program is a two-pass symbolic language assembler that provides the programmer with the convenience of coding in open-ended language or directly in machine-oriented symbolic instructions. The principal functions performed by the assembler are as follows:

- Translation of control and assembly-edit formatting pseudo-operations.
- Recognition and translation of addresses that are absolute or relative to subprogram origin, to common storage, to labeled or block common storage, and to externally defined symbols.
- Production of relocatable or absolute binary subprograms that can be combined at load time.
- Allowances for programmer-defined macro instructions at assembly time.
- Provision for accepting compressed symbolic decks plus any desired alter cards as input, and producing an updated compressed deck as output.
- Provision for a complete listing of assembled program, plus a symbol reference table as output.

Fortran Compiler

The Series 6000 Fortran is a totally new Fortran system which meets all ANSI standards. It is composed of three integrated but separate software entities. The first is the compiler; the second is the library of run-time modules that support the execution of the Fortran object programs; and the third is the time-sharing interface.

There is only one version of the Series 6000 Fortran compiler and this version is called to compile all programs, whether the program originates from local or remote batch, or from time sharing. Compatibility between source programs developed in one environment and used in another is ensured since the one compiler is doing the job for all environments. A collection of source programs may be compiled, some through time sharing, some through batch, and the object modules combined for execution in either environment.

System advantages include:

- One compiler for all dimensions
- A common library
- File may be in standard format
- Free-form format – with or without line numbers
- Multiple compilations within an activity (provided the options are the same for the collection of subprograms)

Fortran language features include:

- Memory-to-memory conversion (ENCODE/DECODE)
- List-directed formatted I/O
- Random file I/O
- Mixed-mode arithmetic
- Subscripts in any expression
- DATA initialization in any type statements
- END = clause in READ statements
- PAUSE with teletypewriter display
- ENTRY
- IMPLICIT statement
- PARAMETER
- CHARACTER
- T and R format specifiers
- ABNORMAL
- PAUSE and STOP with display
- Quoted character constants
- ERR = clause in READ and WRITE statements
- Switch variables
- Type statements with size-in-bytes notation
- FLD function – a built-in function which provides list string and field capabilities
- XOR function – complements the Boolean functions with an 'exclusive OR' capability
- Argument validation for built-in functions
- Null label fields in the arithmetic IF statement

COBOL Compiler

The Series 6000 COBOL Compiler is a full ANSI standard COBOL compiler, providing the maximum level of each functional processing module and of the Nucleus, as defined in USA Standard COBOL X3.23-1968. The compiler is one of the most complete COBOL implementations available.

The compiler generates fully annotated listings which include error analysis and diagnostics which are outstanding by industry standards. Detailed error messages are produced for ease of identification. Within the listings, detailed cross-reference messages for data-names and procedure-names are also produced.

COBOL capabilities include:

- The COBOL SORT verb is implemented and includes all features specified for ANSI COBOL.
- The Report Writer feature as specified in ANSI COBOL is included within Series 6000 COBOL.
- A language level debug is provided to allow the use of any statement in the COBOL language.
- An option is available under the SPECIAL-NAMES section which will cause the compilation to terminate after syntax checking is complete if errors are detected.
- REDEFINES: Series 6000 COBOL uses a chaining REDEFINES (B REDEFINES A, C REDEFINES B, D REDEFINES C., etc.)
- COPY. Series 6000 COBOL allows file renaming and copying of portions of the same source program containing the copy, as well as copying from the library file.

ALGOL Compiler

The Series 6000 ALGOL language encompasses the features of international ALGOL 60 plus improvements and extensions.

Some of the systems advantages are:

- A common library
- A common file system for data files
- Mixed source and object programs at compile time
- Overlays
- DEBUG option

Language extensions include:

- EXTENDED REAL type for extended precision real numbers
- Extended integer division operator
- Input/output according to Dr. Knuth's proposal in *Communications of the ACM,* May 1964

Special features of Series 6000 ALGOL are:

- Provides a flexible set of input/output functions that provide formatting, logical and physical record processing, and character-handling capabilities.
- Provides I/O procedure statements for logical record handling between memory and external devices. Statements are provided for transferring data in physical record blocks between memory and external devices, and for logical record character processing.
- Allows alteration of standard logical file definition by a job control card.
- Includes STACK tracing routines.
- EXTENDED REAL type provides machine double-precision, floating-point word capability.

JOVIAL Compiler

The Series 6000 JOVIAL compiler is the J3 version (1964 definition) including the J3X I/O definition.

Sort/Merge

The Series 6000 Sort/Merge Program is a highly efficient and responsive tool for sorting and merging data files. The program accepts a wide variety of data and task descriptions and dynamically adjusts itself to the individual task to provide the most efficient processing possible. The Sort function also features dynamic adjustment to the operating environment, so as to maximize resource utilization, especially during multiprogramming operations.

The Series 6000 Sort/Merge utilizes mass storage devices, tapes, or combinations of both, depending on the configuration of the system and the desires of the user.

The number of descriptive parameters have been minimized (no redundancy), yet flexibility of use is still provided. This flexibility is represented by the following items:

- Record size is from 1 to 4096 words (16,384 bytes or 24,576 characters).
- There is no restriction on key size.
- BCD, binary bits, words, double-precision floating-point, or single-precision floating-point fields may be used as keys.
- Key fields need not be contiguous nor in any special position within the record.
- Performs automatic key transliteration.
- Performs automatic record selection/deletion.
- Own-code interface is available.
- Has ability to allocate free memory dynamically at execution time.
- Spills onto tape if mass storage areas become full.
- Performs automatic memory sort for small data files.
- Keys may be mixed alphabetic and numeric.

Integrated Data Store (I-D-S)

The I-D-S file management system is a technique for organizing data in mass storage for direct-access or on-line applications. It is both an information-oriented file storage *technique* and a *language* to structure, maintain, and access data in that file.

I-D-S is an integral part of the information system and is integrated with the Series 6000 COBOL language processor. File structuring, file maintenance, and application programs can be written in COBOL.

I-D-S jobs are multiprogrammed; more than one job can be accessing the same file at the same time.

A journal can be maintained automatically during update program runs. It provides the "before and after" conditions and other information necessary to retrace the trail in the event of program or system error. Utility service routines dump and reload I-D-S files.

With I-D-S, there is no need for redundant information usually carried in two or more files; these files would be related through different "chains" to represent different relationships of the elements of the integrated file. Another feature of I-D-S is the simplified record processing made possible by the clearly defined relationships of the chains of the file. The advantage accrued here is lower cost in designing, programming, and checking out the system.

I-D-S Concurrent Access

Concurrent access is provided as an enhancement of Series 6000 I-D-S. It is the first step in a continuing I-D-S evolution toward fully system-controlled multiple access to a data base. This first step allows concurrent read and write access to I-D-S subfiles by several I-D-S jobs. It is directed primarily toward concurrent programs accessing independent data within the same subfile. As many as eight programs can be granted concurrent access to up to 36 subfiles.

This facility enhances the multiprogramming capability of the system by increasing the potential concurrencies of job execution.

Concurrent access control is provided on the I-D-S function level. That is, whenever a program executes a primary I-D-S verb, such as STORE, MODIFY, DELETE, etc., the subfile involved is locked from access by any other program until all operations necessary to perform the verb have been completed. This control prevents the execution of an I-D-S verb from being interfered with by the actions of a concurrent program. Concurrent access also enables each program to retrieve the most recent copy of data as updated by any program.

I-D-S will provide history profiles that identify each job that was executed, the subfiles that each job accessed, and the time during which each job was in execution. This history profile, the I-D-S journal map, is provided to facilitate error recovery by identifying concurrently executing jobs that share subfiles. The purpose of this step of I-D-S evolution is to provide a form of concurrent access for users who know their data base and applications well. The user can run update jobs concurrently when they know that the programs will manipulate different records of the data base subfiles.

I-D-S Data Query

The new Data Query System puts the Series 6000 data base information actively into the customer's business. The I-D-S comprehensive data base management system has provided a nonredundant method to organize and process business data. Now the Data Query System offers new, easier, quicker ways to use the I-D-S data base.

Data Query is designed for use by nonprogrammers. It is particularly effective for providing interactive, on-the-spot reports for market research, inventory control, sales summaries, production control, and many others.

The Data Query System is divided into two logical program groups serving two separate functions:

1. A series of programs to generate a file structure to be used by the on-line system for validating inquiries and structuring chain tables. This will be referred to as the translator.
2. A series of programs that are the on-line handlers of the inquiries and perform the service functions for the user. This will be referred to as the retrieval system.

The retrieval system operates under the Time Sharing Executive of GECOS. Therefore, it will operate with any terminal that is acceptable to GECOS Time Sharing, including Honeywell's Visual Information Projection (VIP) terminals.

The Data Query System is designed for operation primarily by nonprogrammers; it does not require the user to have specific knowledge of the data file structure. Above all, it is intended to provide the availability of selected data spontaneously (i.e., interactively), without this availability having been originally preplanned.

Indexed-Sequential Processor (ISP)

The Series 6000 Indexed-Sequential Processor (ISP) is a collection of subroutines that support an indexed-sequential file organization and access method. ISP allows the user to access direct-access files efficiently in either a random or sequential order. It provides an efficient, easy-to-use conversion aid for converting indexed-sequential programs for execution on Series 6000 systems. The ISP provides the following features:

- ISP allows the user to access the same file in either a random or a sequential order.
- ISP controls an indexed-sequential file by means of two separate, independent files: a key file and a data file. Since these files are separate, they need not be resident on the same device; direct-access device storage space can be more efficiently used.
- ISP allows the user to access an indexed-sequential file in both a random and a sequential order within the same program.
- ISP allows conversion of other manufacturers' indexed-sequential programs without requiring that the files or data base be redesigned or that the program logic be changed.
- The key field can be located anywhere within the data record and can be as long as the data record.
- The records are blocked into pages of 256 words; the data records can be as long as 256 words.
- ISP provides excellent facilities for converting indexed-sequential programs to Series 6000 systems. Since the files need not be redesigned nor program logic changed, ISP is easy to use and requires little programmer retraining.
- ISP has lower overhead requirements than more complex data base management systems The indexed-sequential file structures are simpler and the ISP routines are more specialized.
- ISP logically "inserts" records that are added to the file, rather than moving data records and physically inserting the added record. The new record is physically placed in an overflow area that is most likely on the same access area as the old record. This technique decreases the processing time to insert a record without increasing the access time to an inserted record.

SYSTEM SOFTWARE

The system software of prime importance is GECOS. It empowers the system hardware to become a multidimensional information system with the following modes of operation:

- Local Batch Processing
- Remote Batch Processing
- Remote Access Processing
- Time Sharing
- Transaction Processing

All these processing modes operate within the system concurrently and with an integrated file system (common data base), making the Honeywell Series 6000 a truly multidimensional information system.

GECOS encompasses many automatic and simplified programming aids. These aids include the following system software programs.

Loader

The loader not only loads a program but can:

- Link program segments and subroutines.
- Provide for overlays.
- Call in other programs from the library.
- Create file control blocks for I/O.
- Provide options for selected printouts during execution.
- Provide a load map.

File and Record Control

The use of file and record control relieves the user of the necessity for programming I/O routines. The user need only concern himself with the information content of his file; the processing device need not concern him.

Designed with modular construction for maximum flexibility, file and record control provides the user with the following:

- Ability to consider inputs and outputs as records and files of arbitrary lengths.
- Ability to GET and PUT logical records without regard to physical media, block size, or record size.
- Ability to interchange media or devices without reprogramming.
- Ability to improve performance through the use of record blocking without program change.
- Ability to input and output data to remote terminals.
- System software compatibility.
- Means of automatic error detection and correction.
- Standardized operating procedures.

Bulk Media Conversion (BMC)

Bulk Media Conversion performs conversion for either input or output that exceed limits set for GECOS system media conversion (GEIN or SYSOUT). BMC has no volume limitations; it does not unscramble mixed files, as does SYSOUT. BMC does handle successive files and performs media conversion for the following types of peripherals:

- Card reader
- Magnetic tape
- Magnetic disk
- Printer
- Card punch

System Editor

The System Editor is used to maintain system libraries in any one of three forms:

- Symbolic
- Object
- System (fast-load format)

Utility

The Utility package is a generalized system providing storage device processing capabilities. It permits copying, comparing, positioning, and printing. Utility is used mainly for operational and debug purposes. It resides in system storage and is called through GECOS, unconditionally by the $ UTILITY control card or conditionally by the $ ABORT control card in the case of an activity abort. Utility processes magnetic tapes or linked or random disk files as specified by user-supplied control cards.

TIME-SHARING SYSTEM

Time sharing, another dimension in the Honeywell Series 6000, brings the power of a large computer to the user via an inexpensive, on-line keyboard terminal. The Time-Sharing Executive is a privileged system program within GECOS. Time sharing comprises more than 20 subsystems. Some of these subsystems are described below under five category headings.

Programming Languages

BASIC (Beginner's All-Purpose Symbolic Instruction Code) is a fast, conversational, one-pass compiler especially adaptable to solving the day-to-day problems of the inexperienced computer user. It is easy to learn and easy to use. BASIC features include ASCII and binary file I/O, saving and executing object files, subroutine and function statements, plus many others.

Fortran is especially suited for solving scientific and engineering problems. It is particularly helpful to users who require a language more flexible and powerful than BASIC. Fortran is intended for expressing problems of numerical computation. In particular, problems composed of equations and dealing with many variables can be handled easily. Added commands simplify terminal I/O for the user. Series 6000 Fortran is a totally redesigned compiler that provides a number of significant enhancements in the areas of user convenience, performance, language capability, and system utilization. A program written in time-sharing Fortran can easily operate in the batch mode.

ABACUS is a subsystem that allows the time-sharing terminal to operate as a highly sophisticated desk calculator. Arithmetic statements are expressed in a Fortran-like manner and the evaluated results are printed after each carriage return. Included in the subsystem are commonly used functions (i.e., square root, trigonometric functions) and the commonly used constants (π,e, etc.). The subsystem has the ability to save the results of a calculation from expression to expression via symbolic name.

Time-Sharing Batch

Time-sharing batch is an interactive remote job entry facility providing the user at a time-sharing terminal all the capabilities of the batch dimension. This facility is one of the most powerful and flexible tools provided by any information system. It features program submission, debugging, inspection of output, and direct conversation with the batch program – all from the user's terminal.

CARDIN enables the user to create a job, enter it in the batch job stream for processing, take status checks on its progress, and receive parts of, or all, its output back at the terminal. The user can directly access a job that has been entered for terminal I/O, or he can abort a job if an error is detected. Also, the output of a job activity can be saved in the file system for later perusal.

BPRINT/BPUNCH, ASCBCD/BCDASC are utility and file media conversion commands, operable under the CARDIN subsystem, that permit printing and punching of time-sharing ASCII files at central site or remote computers. These commands further allow for conversion of files (from ASCII to BCD and from BCD to ASCII).

JOUT is a time-sharing facility for manipulating batch processing output. It permits directing selective activity reports to a terminal or to the central site printer.

SCAN provides commands to conversationally examine output of batch jobs saved on a permanent file. The time-sharing user can request a list of the compilation errors, print selected lines, and redirect the output to a local or remote printer.

RBUG is a conversational debug facility which permits the time-sharing user to halt the execution to display the contents of core locations and registers to set additional breakpoints, and then to continue or terminate the execution of the program.

File Editing, Inspection, and Maintenance

The EDITOR subsystem allows a user at a teletypewriter to enter text into a computer, edit it, store it, and retrieve it. The text may be of any type: letters, lists, manuals, or business records. EDITOR is especially useful for form letters, manuals, and inventories that often need individual items changed.

The RUNOFF subsystem (together with EDITOR) allows the user to specify the format in which a file is to be printed at the teletypewriter. Among RUNOFF's features are the ability to specify the number of lines on a page, the length of each line, justifications of right and left margins, page numbering, and margin size.

The FDUMP subsystem is a remote-terminal, word-oriented, file inspection maintenance facility for permanent files, regardless of their format. These files may have been generated in either the batch, remote batch, or time sharing environments. With the FDUMP subsystem the user can examine and manipulate from his remote terminal the content of any permanent file to which he has access.

The dataBASIC system is a terminal-oriented, data processing language which combines data base capabilities with BASIC-like procedural statements. The language permits the storage, maintenance, retrieval, and output of data on the basis of field names and associated field values. The dataBASIC system is an easy-to-learn and easy-to-use facility for structuring and accessing private data systems from a remote keyboard terminal.

ACCESS is a conversational, file space management subsystem that allows the user to create, delete, and modify file system catalogs, subcatalogs, and named files. The file *space* is manipulated with ACCESS, not the file content. Various types of permissions can be assigned to either the catalogs or the files. This system also allows the time-sharing user to access files that have been saved by others or that have been stored in the file system by means other than through time sharing (e.g., batch files).

I-D-S Data Query, operational under time sharing, performs data retrieval from an I-D-S structured data base. It comprises a translator to generate the file structure and data name dictionary and a retrieval subsystem to process the inquiries and dispense the output.

Time-Sharing System Extension and Maintenance

The LODX subsystem permits executing and debugging experimental subsystems in their normal environments. It is used to load a subsystem program from a permanent file into the time-sharing system for checkout purposes. When necessary, through TDS (Terminal Debug Subroutine), breakpoints can be set and octal patches can be applied to the subsystem during execution. The subsystem allows a thorough checkout of user system software before it is integrated into the command structure of the time-sharing system. The user can also scan the dump file of an aborted subsystem (SABT) by snapping portions of it at a terminal. He simply specifies the areas he wishes to inspect. These subsystems provide users with the capability of developing specialized subsystems that are not a part of the total time-sharing system offering.

MASTER is a system maintenance and monitoring facility for an authorized master user. He can obtain a status report of the time-sharing activity, monitor the user terminal activity, issue "all points bulletins," list the system master catalog, update the system master catalog, take snapshots, and temporarily patch any time-sharing location contents.

Time-Sharing Library

The Time-Sharing Library is a basic starter set of problem-oriented programs. The library includes the application areas of business and finance, management science and optimization, engineering, geometric and plotting, educational and tutorial, mathematics, and statistics. Emphasis is placed on key time-sharing, problem-solving application areas.

Time-Sharing Fortran Library Generator and Editor provide the user with the capability to produce his own load-time library of time-sharing Fortran subroutines. The Library Generator (TSLG) produces a library file of time-sharing Fortran subroutines, complete with directory, in a form that is acceptable to the time-sharing Fortran loader. The Library Editor (LIBED) allows the user to edit his files (add, delete, replace, or copy individual subroutines onto a master file) prior to using the Library Generator program.

APPLICATIONS SOFTWARE

Applications software is an integral part of the total information system's problem-solving, answer producing capability. The applications software packages described in the following paragraphs are representative of the applications software library.

Linear Programming System (LP 6000)

The Series 6000 Linear Programming System (LP 6000) is a comprehensive system that combines an extensive set of solution and postoptimal algorithms, a flexible language for user-system communication, and the latest advances in computational techniques and on-line system control. LP 6000, which is available in either local or remote batch modes, embodies the best linear programming techniques and solution algorithms with the latest advances and techniques of large system development, problem-oriented languages, and user problem control. Although the functions performed by LP 6000 are very complex, the system is easy to use, flexible, and comprehensive.

Automatically Programmed Tools (APT)

The Automatically Programmed Tools (APT) system for the Series 6000 is a comprehensive computer system and problem-oriented language for the automatic processing of descriptive data for products manufactured by numerically controlled machine tools. The system provides an effective and reliable means of achieving the ultimate benefits from numerically controlled manufacturing processes. Equally applicable to either local or remote batch processing, APT utilizes the versatility and capability of the digital computer to produce the machine-tool control medium automatically. With a time-sharing terminal using standard system software capabilities, APT can be used as a remote system. APT provides a manufacturing operation with the speed, flexibility, and savings necessary to use numerically controlled machine tools for a large variety of manufactured parts. (See Figure 9.)

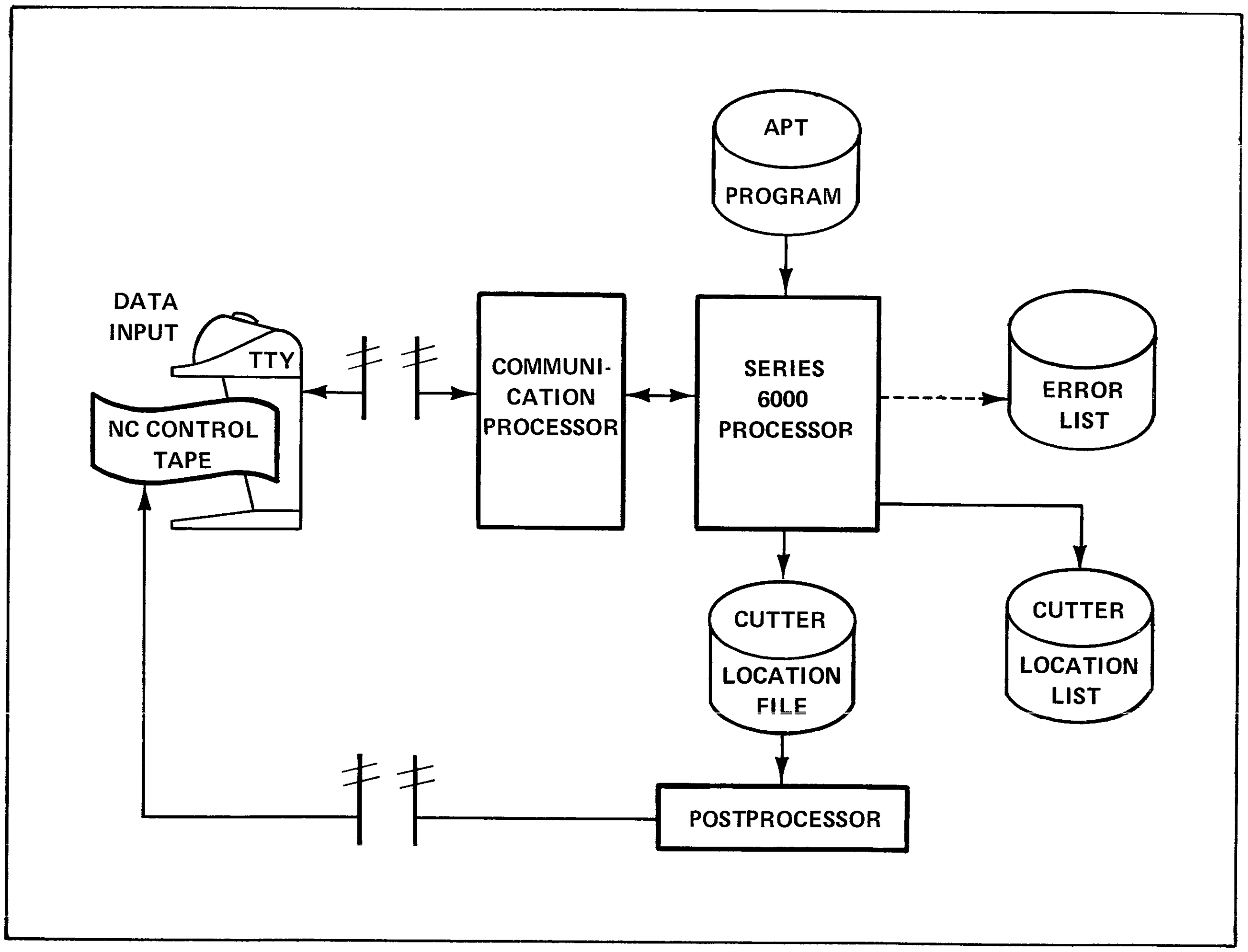


Figure 9. APT Remote Program Flow Chart

PERT/TIME

PERT/TIME is a system for schedule evaluation and project control. Consistent with dynamic project management, PERT/TIME provides new flexibility in planning and in reporting project progress. Written to meet all USAF specifications, PERT/TIME will operate efficiently in both local batch and remote batch modes. Working in conjunction with PERT/COST, PERT/TIME adds a new dimension to realistic, dynamic management. Important and critical facts are immediately available to managers for evaulation and consideration – an invaluable factor in management decision-making.

PERT/COST

The Honeywell PERT/COST system is used to develop realistic schedules and cost pictures, to allocate manpower and resources to better advantage, and to identify areas that might involve costly delays. Linked to the PERT/TIME system through an event time file, PERT/COST uses pertinent time information to indicate trouble spots and to demonstrate the costs of project delays.

SIMSCRIPT

Honeywell SIMSCRIPT provides the user with a simulation-oriented language that permits the translation of complex mathematical and logical models into meaningful simulation sequences. Simulation is used to analyze the behavior of these complex systems and to investigate the impact of alternate courses of management action on the system. A manager is able to increase the value of a system by pretesting many alternatives without undue cost or inconvenience. SIMSCRIPT brings a simulation language consisting of Fortran-like words, phrases, and symbols into the realistic world of management decision and control.

Time Series Forecasting

Time Series Forecasting permits better forecasting by business, resulting in better planning and more profitable operations. Figure 10 illustrates sales forecasting and analysis, which is one of the major applications of GECAST. The graph, plotted on a time-sharing terminal, shows the actual and predicted dollar sales over a 2-year period.

Time Series Forecasting is also a unique computer application for predicting the quantitative future behavior of any single variable; for example, business conditions (sales, prices, orders), social conditions (crime rate, population), economic conditions (GNP, national debt), or physical conditions (rainfall, air pollution).

Civil Engineering Package (CEP)

The Civil Engineering Package (CEP) encompasses eight problem-solving subsystems to provide civil engineers with quick and efficient solutions to many of their most common technical problems. These subsystems are general in form and may be easily modified to unique situations. The eight systems include:

- Earthwork
- Horizontal Geometry
- Traverse Analysis
- Slope Stability Analysis
- Curved Bridge Geometry
- Composite Beam Analysis
- Continuous Girder Analysis
- Retaining Wall Design

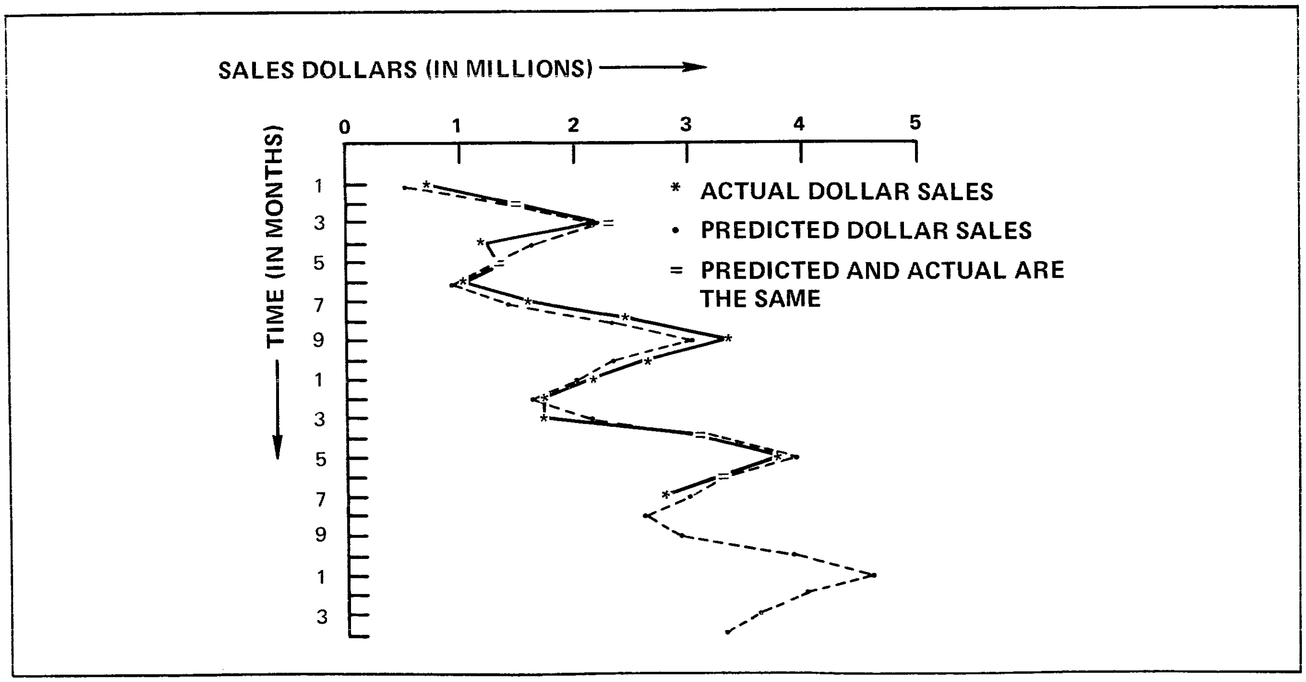


Figure 10. Time Series Forecasting and Analysis Graph

Biomedical Statistical Programs (BMD)

The Biomedical Statistical Programs (BMD) are a collection of statistical programs for analyzing large amounts of data. These programs give the research worker or statistician appropriate tools for effecting the type of analyses his data requires. Some analyses will require the use of several programs in succession; others will utilize only one program. The interrelationship of programs is assured through standardized formats and ordering of results. Research mathematicians and business analysts alike will find these programs invaluable in solving multivariant analyses, regression analyses, and time-series analyses.

MATHPAC

MATHPAC is a set of extensive and varied programs for mathematical/engineering applications. For example, engineering analysis of complex mathematical functions, such as the response surface pictured in Figure 11, is greatly aided by the use of MATHPAC subroutines. The Fortran-compatible subroutines are easily accessed by any user system in local batch, remote batch, or remote access mode. The freestanding routines are also utilized in local or remote batch or time-sharing processing. Fortran-compatible subroutines include special trigonometric functions, matrix algebra, random number generation, Lagrange interpolation, and Bessel functions. The freestanding routines include roots of a polynomial, eigenvalue/eigenvector solution, polynomial curve fitting, simultaneous equation solutions, and solving of ordinary differential equations.

Generalized Parts Explosion System (GEPEXS)

The Honeywell Generalized Parts Explosion System (GEPEXS) is an application system with initial user startup and application-expansion capabilities. Design of the system focuses on sound I-D-S techniques for simple implementation and adaptation to fit user application growth needs within his particular time and budget requirements.

Generalized system design makes GEPEXS especially applicable in determining schedules and material requirements for:

- Job shops or flow shops
- A single product or multiple products
- Subassemblies and raw materials

Generalized Inventory Management System (GEIMS)

Honeywell's Generalized Inventory Management System (GEIMS) is an application package that utilizes an integrated data base, minimizes implementation problems, and adapts to user requirements quickly and at a low cost.

This comprehensive inventory management system plans, schedules, and controls the amount of materials required to satisfy customer demands and to optimize inventory investment.

dataBASIC System

The dataBASIC system is a terminal-oriented data processing language that combines data base capabilities with BASIC-like procedural statements. The language is primarily designed for use by non-EDP personnel. The language permits the storage, maintenance, retrieval, and output of data on a content-addressable basis. The dataBASIC system is designed for operation in a general-purpose, time-sharing system environment. It uses a modified version of Integrated Data Store (I-D-S) for data file management. By using simple commands that are easy to learn, the user is able to manipulate data, make inquiries within the data base, and specify multiple conditions as criteria for record selection.

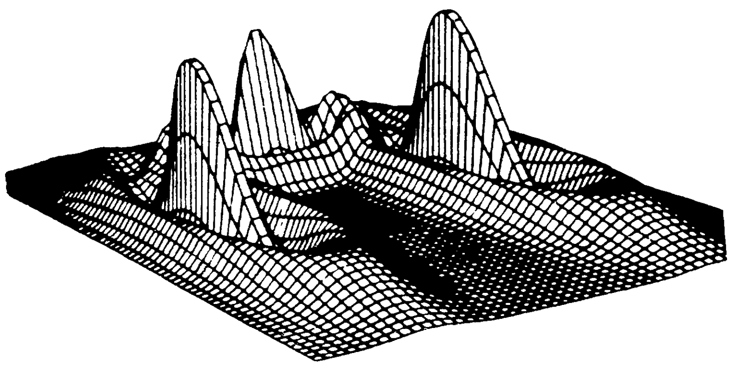


Figure 11. Sample Response Surface for MATHPAC Analysis

Section V
Memory Module

The memory module consists of a system controller and associated core storage. The system controller is the hub for all information transferred between the processors and input/output multiplexers and controls the core storage units. System controllers are independent of each other and can function simultaneously, thus providing overlapped memory accesses in multiple-controller systems.

SYSTEM CONTROLLER CHARACTERISTICS

The system controller is a passive system component providing the overall control of, and communication between, the active system modules; i.e., the processor, IOMs, and DATANET 355s. The basic functions performed by the system controller include:

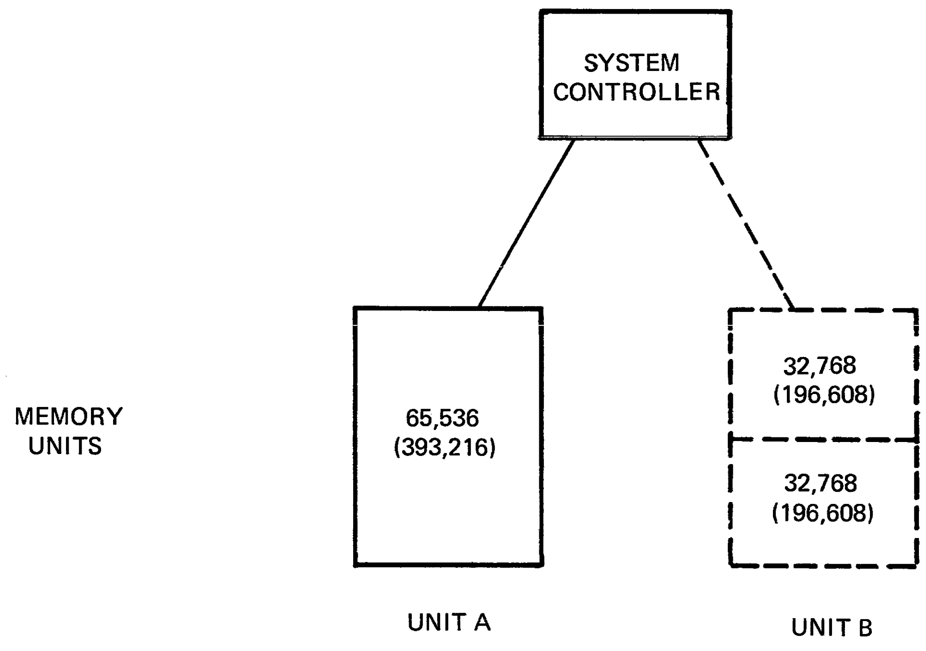
- Control of all communications in the system, including the switching of control signals, addresses, and data between active system modules.
- Control of program interrupts of the control processor by noncontrol processors, IOMs and DATANET 355s.
- Control of priorities among the various active system modules.
- Memory interleaving.
- Memory addressing.

The Series 6000 system controller includes the following elements:

- Up to eight memory ports for connection to the active modules. These ports are assigned a priority to permit the servicing of all demands in an orderly manner.
- A program-addressable memory port "lockout" mask that can be set by a control program to inhibit data transfers and interrupts.
- 32 program execute-interrupt cells. Any of the components interfacing with the system controller can set the interrupt cells as required to initiate appropriate control processor activity.
- A program-addressable execute-interrupt mask. The control program can override the built-in priority of the interrupt cells and exercise complete control of all system interrupts as required.
- A control processor designation switch. The system controller has an 8-position switch that corresponds to the eight ports to which the processors and other active modules are connected. This switch is used to direct interrupt information to the appropriate processor.
- A built-in elapsed time clock with accuracy of 1/512 of a millisecond (approximately two microseconds). The clock is 36 bits in length, allowing a 37.3 hour turnover.
- The Series 6000 system controllers and memory units are packaged separately. This results in separate test functions for the system controller and memory, thus providing rapid isolation of malfunctions.

Model 6030/6040 System Controller and Memory

The system controller for Models 6030 and 6040 controls memory sizes of 65,536 to 131,072 36-bit words as shown in Figure 12. Memory accesses retrieve or store two words (72 bits plus 2 parity bits). The cycle time for two words plus parity is 1.2 microseconds.



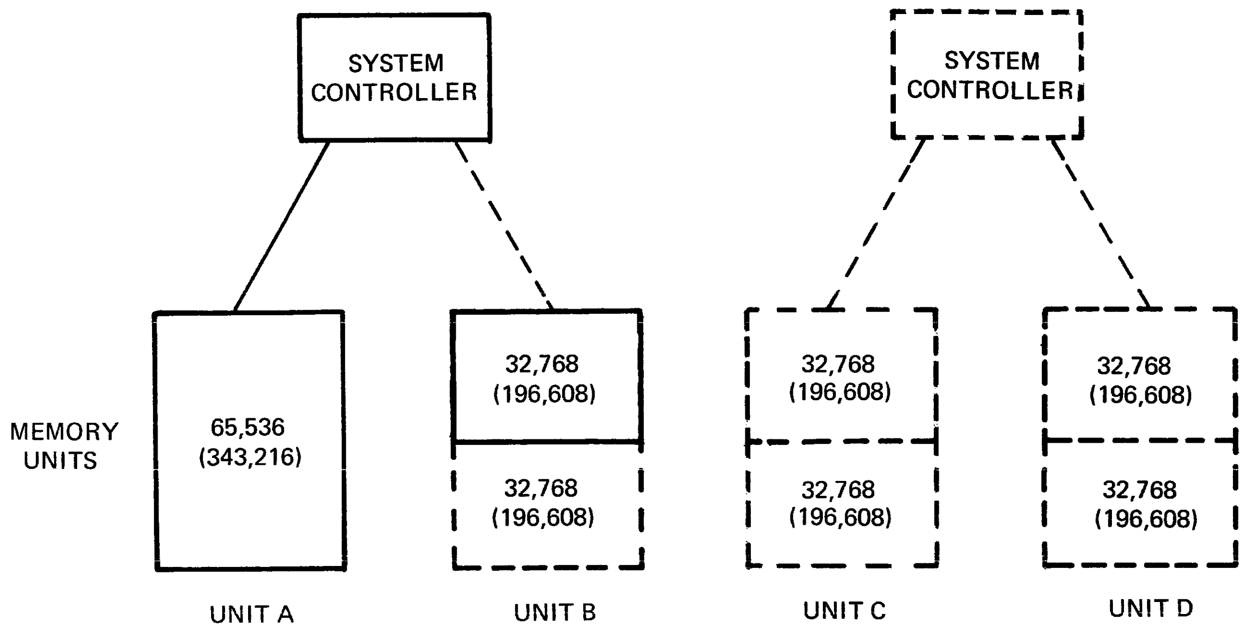
Total Memory	Unit A	Unit B
65,536 (393,216)	65,536 (393,216)	
98,304 (589,824)	65,536 (393,216)	32,768 (196,608)
131,072 (786,432)	65,536 (393,216	65,536 (393,216)

NOTE: The memory sizes are indicated in two forms:
1. In 36-bit words, indicated without parentheses.
2. In characters, indicated with parentheses.

Figure 12. Model 6030/6040 Memory Configuration

Model 6050/6060 System Controller and Memory

Models 6050 and 6060 can have one or two system controllers, depending on the memory size. One or two additional system controllers are available as an option. These models have a memory range from 98,304 to 262,144 words of core storage. Memory cycle time is 1.2 microseconds. The increment of memory growth is 32,768 words. As in all models of the Series 6000, memory accesses retrieve or store two words (72 bits plus two parity bits). Two- or four-way memory interleaving is availabile in Models 6050 and 6060.



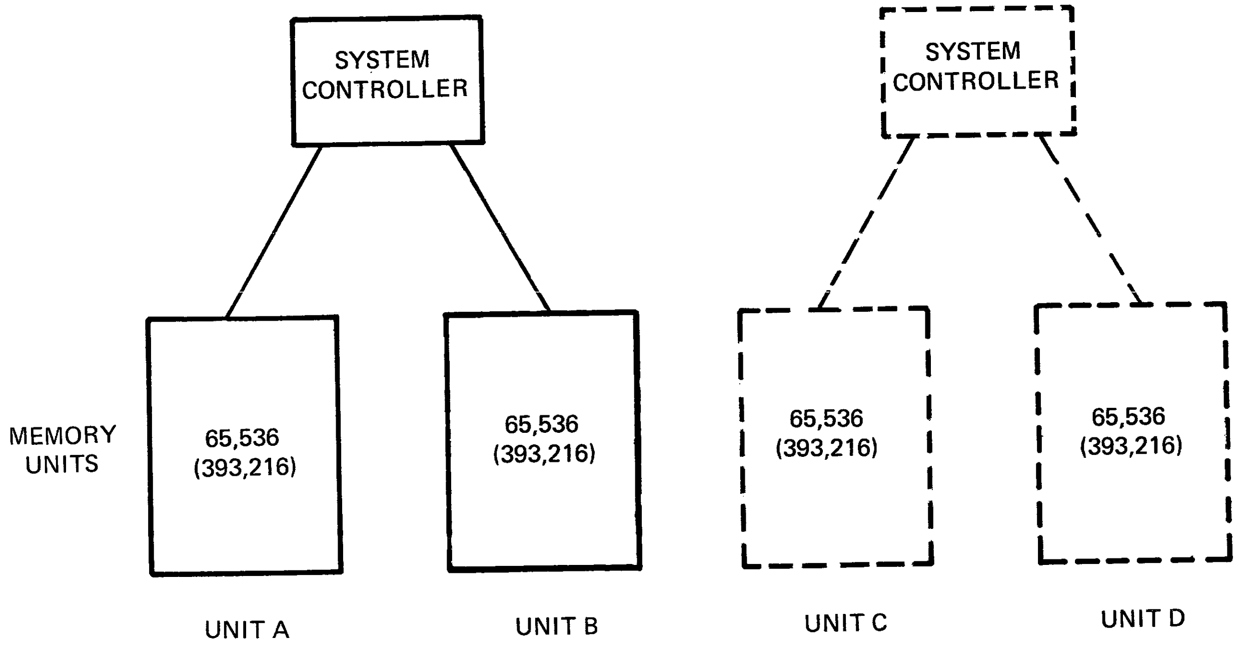
Total Memory	Unit A	Unit B	Unit C	Unit D
98,304 (589,824)	65,536 (393,216)	32,768 (196,608)		
131,072 (786,432)	65,536 (393,216)	65,536 (393,216)		
163,840 (983,040)	65,536 (393,216)	65,536 (393,216)	32,768 (196,608)	
196,608 (1,179,648)	65,536 (393,216)	65,536 (393,216)	65,536 (393,216)	
229,376 (1,376,256)	65,536 (393,216)	65,536 (393,216)	65,536 (393,216)	32,768 (196,608)
262,144 (1,572,864)	65,536 (393,216)	65,536 (393,216)	65,536 (393,216)	65,536 (393,216)

NOTE: The memory sizes are indicated in two forms:
1. In 36-bit words, indicated without parentheses.
2. In characters, indicated with parentheses.

Figure 13. Model 6050/6060 Memory Configuration

Model 6070/6080 System Controller and Memory

Models 6070 and 6080 can have one or two system controllers depending on the memory size. One or two additional system controllers are available as an option. These models have a memory range from 131,072 to 262,144 words of core storage. Memory cycle time is 500 nanoseconds. The increment of memory growth is 65,536 words. As in all models of the Series 6000, memory accesses retrieve or store two words (72 bits plus two parity bits). Two- or four-way memory interleaving is available in Models 6070 and 6080.



Total Memory	Unit A	Unit B	Unit C	Unit D
131,072 (786,432)	65,536 (393,216)	65,536 (393,216)		
196,608 (1,179,648)	65,536 (393,216)	65,536 (393,216)	65,536 (393,216)	
262,144 (1,572,864)	65,536 (393,216)	65,536 (393,216)	65,536 (393,216)	65,536 (393,216)

NOTE: The memory sizes are indicated in two forms:
1. In 36-bit words, indicated without parentheses.
2. In characters, indicated with parentheses.

Figure 14. Model 6070/6080 Memory Configuration

SYSTEM CONTROLLER PORTS

The ports on the system controller provide for connections to all active system modules, including processors, IOMs, and DATANET 355s. Simultaneous requests for memory accesses are serviced according to the order in which active system modules are connected to the system controller. This priority, established by the order of connections, can be modified by switches on the configuration panel. The switch capability prevents a single module from monopolizing all memory accesses.

MEMORY ADDRESSING

All active modules can directly address 262,144 words of storage. If a system contains less than 262,144 words of memory, operation with GECOS requires that there be no address gaps; i.e., that all memory addresses be contiguous.

MEMORY INTERLEAVING

In Models 6050, 6060, 6070, and 6080 the system controller provides for memory interleaving between two memory units (of equal size) connected to it. Those models with two system controllers provide for the additional capability of interleaving between the controllers, for four-way interleaving. Interleaving effectively increases the number of memory accesses available to active system modules and is employed for the following reasons:

- Higher processor execution rates are obtained, since the processor logic may have less delay than the memory cycle time.
- Processor interference in a multiprocessor system is reduced, since interleaving causes accesses to be more evenly distributed among system controllers.
- I/O interference is reduced for high-speed peripherals such as disk units.

Interleaving is transparent (except for timing) to all programs other than those specifically concerned with selecting a particular system controller for accessing interrupt cells or interrupt enable registers.

INTERRUPT ORIENTATION

In a modern multiprogramming or multiprocessing computer system, it is necessary to free both the hardware and software from the burden of checking other components of the system either for completion of, or requests for, service. To accomplish this, all Series 6000 active modules that have completed assigned tasks, or that require service, generate interrupts to the normal flow of instructions in a processor. In a multiprocessor system these interrupts can be generated by processors as well as by input/output devices.

Each system controller has its program interrupt cells connected in a priority sequence. Any interrupt request generated by an active module will set one of these interrupt cells, depending on which particular interrupt cell the interrupting device has been assigned to use. A "control" processor is assigned to respond to the interrupts.

Normally, upon the completion of each instruction pair in the processor, a check is made for the presence of an interrupt. If no interrupts are present, or if the interrupt inhibit function is active, instruction execution continues in the normal sequence. If one or more interrupts are present (and not inhibited) the system controller reports the identity of the highest priority cell that is set and then resets that interrupt cell. This causes the processor to take its next instruction from a preassigned location that is determined on the basis of the interrupt cell's identity. For each of the interrupt cells, two consecutive locations are provided in the memory, and the instructions from these locations are normally set up to store machine status and then transfer to the appropriate routine for servicing the interrupt. A 64-word (2 instructions for each of 32 cells) interrupt table, or interrupt vector, is thus required for each system controller in the system. When servicing of the interrupt is completed, GECOS dispatches to the highest-priority program in its queue.

The same processor must be the "control" processor for all of the interrupt cells; this is determined by a switch setting at the system controller.

The supervisor servicing a particular interrupt will normally preserve the instruction counter and indicators for the interrupted program. If necessary, the supervisor can load the interrupt enable register with a suitable combination of bits to prevent any undesired interrupts, e.g., an interrupt through the same interrupt cell. Servicing of the interrupt can then proceed without use of the interrupt inhibit bit. Thus supervisor can be protected against undesirable interrupts, but can be interrupted, in turn, by enabled, higher-priority interrupts.

Each input/output module will generate interrupts to indicate events such as:

- Successful completion of a requested I/O action.
- Unsuccessful initiation of a requested I/O action.
- Special interrupts.
- Error conditions.

Section VI
Processor

The processor has full program execution capability and conducts all actual computational processing within the information system. The processor, which communicates only with the system controllers, consists of a control unit, an operations unit and, on the EIS models, an extension unit. The operations unit contains the logic to execute arithmetic and logical operations. The control unit provides the interface between the operations unit and the system controller. It also performs instruction fetching, address preparation, memory protection, data fetching and storing, and overall timing. The extension unit provides additional registers for address modification, decimal arithmetic, and formatting. All three units operate with relative independence and maximum overlap to provide the highest rate of instruction execution in Models 6070 and 6080. This section contains discussions of the EIS processors, the processor registers, data formats, instruction format, address modification, and the instruction repertoire. Further details concerning the processor are discussed in the Programmers' Reference Manual.

EXTENDED INSTRUCTION SET (EIS)

The Series 6000 is well known for its configuration modularity and flexibility, which allow tailoring for a wide variety of work loads without the need for user program modification. This modularity and flexibility apply to the processor as well. The Extended Instruction Set (EIS) models of the Honeywell Series 6000 processors greatly increase the speed with which the processor can handle business-oriented applications. The EIS hardware (the extension unit) adds to the processor's standard repertoire of instructions, numerous new instructions that have powerful capabilities for processing bytes, BCD characters, packed decimal data, and bit strings. The EIS hardware also affords a second level of address indexing for all of the standard instructions as well as the EIS instructions. The EIS models afford a close correlation between hardware instructions and the functions performed by business-oriented languages.

This high degree of correlation between the hardware instructions and business-oriented language functions is the key to the power of the EIS hardware in relation to competitive offerings.

The EIS hardware accomplishes, with one instruction, functions that require several instructions in competitive machines considered highly oriented to business applications. This has a cumulative effect in system throughput; a smaller number of instructions are required, thus reducing execution time and program size. In turn, the multiprogramming depth (i.e., number of programs simultaneously in memory) is greater, thus making more efficient use of the system and increasing the throughput.

Salient Features of the EIS

Decimal Arithmetic

- Data types are packed decimal and direct ASCII (may be intermixed).
- Decimal arithmetic operands are 1 to 63 digits in length.
- Numeric data may be fixed point and/or floating point (intermixed fixed and floating-point data is allowed).
- Full set of decimal arithmetic instructions is provided (each is a single instruction with either two or three addresses) including add, subtract, multiply, and divide.
- All instructions provide a hardware rounding option.

Data Manipulation Capabilities

- There are four native data modes – ASCII, BCD, packed decimal, and bit string.
- Hardware data descriptors correspond closely to language data descriptors.

Data Movement

- Alphanumeric movement with character-fill in either direction.
- Numeric move with fill and/or rounding and scale change.
- Bit-string manipulations using all of the 16 different Boolean resultants.
- Radix conversion and transliteration instructions.

Data Comparison

- Alphanumeric comparison with fill.
- Numeric comparisons between fields of the same or different format and character type.
- Bit-string comparisons with fill.
- String scan for one or two characters.

Editing Capabilities

- Extremely powerful editing capabilities using micro operations are provided.
- Micro operations provide alphanumeric and numeric edited move instructions with the capability to edit character and numeric strings on a character-by-character or digit-by-digit basis, or in concatenated series of characters and digits.
- Micro operations are not altered by their execution; therefore, a sequence of micro operations can be set to describe a data field and then used repeatedly by the edit instructions.
- A *single* instruction can perform the most complicated edit function with great speed.

SYSTEM CONFIGURATION FLEXIBILITY

The processor includes standard features that directly contribute to the formation of multiprocessor system configuration flexibility or immediate reconfiguration.

The processor is equipped with as many as four ports for connection to system controllers and can directly address 262,144 words of memory. All instructions utilize a full 18-bit address field. Memory addresses are assigned consecutively, beginning with zero and continuing through the full available memory.

REGISTER DESCRIPTIONS

The internal processor registers that are accessible to the program are as follows:

Name	Mnemonic	Length
Accumulator register	AQ	72 bits
Eight index registers (n = 0,1,...., 7)	Xn	18 bits each
Exponent register	E	8 bits
Base address register	BAR	18 bits
Indicator register	IR	18 bits
Timer register	TR	27 bits
Instruction counter	IC	18 bits
Fault register	FR	36 bits
Mode register	MR	36 bits
Control unit history registers	CUHR	72 bits each
Operations unit history registers	OUHR	72 bits each

- The *accumulator register* (AQ) can be used:

  In floating-point operations as a mantissa register for both single and double precision.

  In fixed-point operations as an operand register for double precision.

  In fixed-point operations as operands for single precision where each AQ half serves independently of the other as an operand register for single precision; the halves become the A register (bits 0 through 35) and the Q register (bits 36 through 71).

  In address modification each half of the A register and of the Q register can be the source of an index; these halves then become AU (bits 0 through 17), AL (bits 18 through 35), QU (bits 36 through 53), and QL (bits 54 through 71).
- The *index registers* (Xn) can be used:

  In fixed-point operations as operand registers for half precision.

  In address modification as sources of index quantities.
- The *exponent register* (E) supplements the AQ register in floating-point operations as the exponent register.
- The *base address register* (BAR) is used in address translation and memory protection. It denotes the base address and the number of 1024-word blocks assigned to the program being executed. It is effective in the slave mode only.
- The *indicator register* (IR) is a generic term for all of the program-accessible indicators within the processor; the name is used where the set of indicators appears as a register, that is, as a source or destination of data. The format of the indicator register contents as they would be reflected in memory is shown in Figure 15.
- The *timer register* (TR) is decremented by one every two microseconds, and a timer runout fault occurs whenever its contents reach zero. If timer runout occurs in the master mode, the fault does not occur until the processor returns to the slave mode.
- The *instruction counter* (IC) contains the address of the next instruction to be executed.

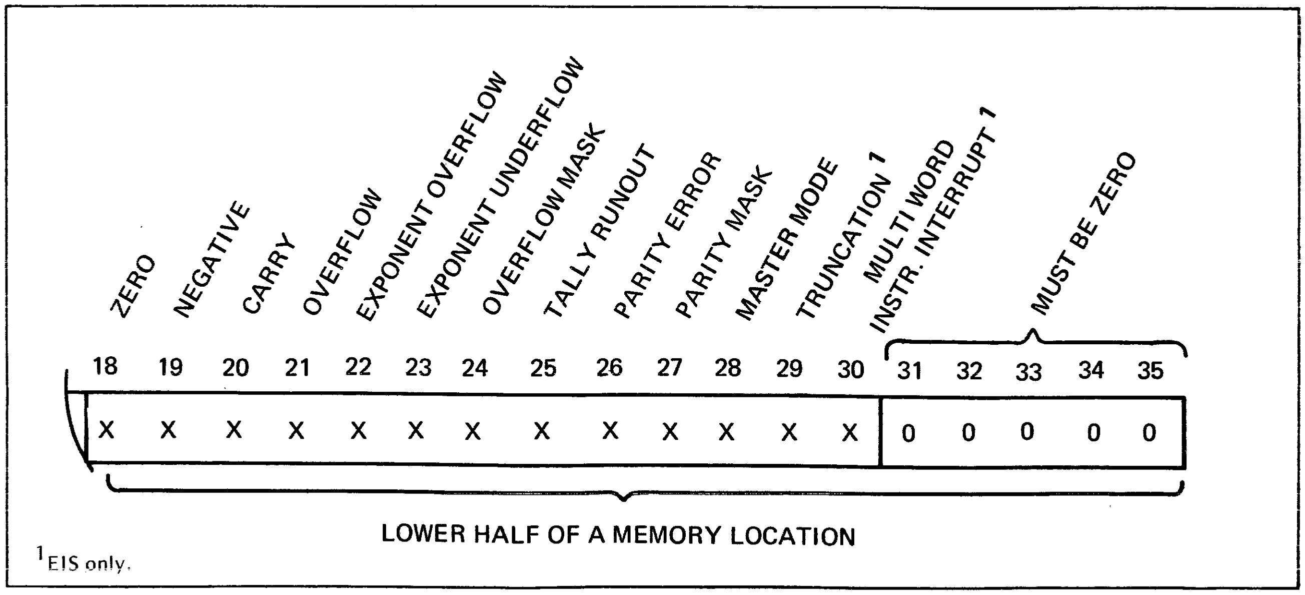


Figure 15. Indicator Register Format

- The *fault register* (FR) allows GECOS or TOLTS to detect and differentiate individual or multiple error indications.
- The *mode register* (MR) allows TOLTS to perform dynamic testing and marginal checking under program control.
- The *history registers* (CUHR and OUHR) are sixteen 72-bit registers for the control unit and sixteen 72-bit registers for the operations unit to store the last sixteen states of each unit for TOLTS use.

EIS REGISTER DESCRIPTIONS

The following additional processor registers are accessible to a program on an EIS processor:

Name	Mnemonic	Length
Address registers (n = 0,1,2,..., 7)	AR_n	24 bits each

- The address registers (AR_n) can be used for an additional level of address modification of single or multiword instructions followed by normal indexing and indirect modification found on the other Series 6000 models.

WORD FORMATS

Fixed-Point Data

Fixed-point operations are conducted with 36-bit, single-precision operands; 72-bit, double-precision operands; or 18-bit, half-word operands. The formats of the three types of operands are shown in Figure 16.

Fixed-point data is represented in twos complement form. The range of single-precision numbers is -2^{35} to $(2^{35}-1)$, the range of double-precision numbers is -2^{71} to $(2^{71}-1)$, and the range of half-word operands is -2^{17} to $(2^{17}-1)$.

Floating-Point Binary Data

Floating-point operations are conducted with either 36-bit, single-precision operands or 72-bit, double-precision operands. The formats of each are illustrated in Figure 17.

Floating-point numbers are represented by a mantissa of 28 or 64 bits and a binary exponent of 8 bits. Both the exponent and the mantissa are represented in twos complement form. The first bit of the mantissa indicates the sign of the quantity, and the exponent has a range of +127 to -128.

Alphanumeric Data

Six 6-bit or four 9-bit characters are contained in each data word. The designation of the characters within the word is as shown in Figure 18. Combinations of characters can be read or written to or from memory.

The EIS can address and manipulate data in word string, character string, or bit string modes.

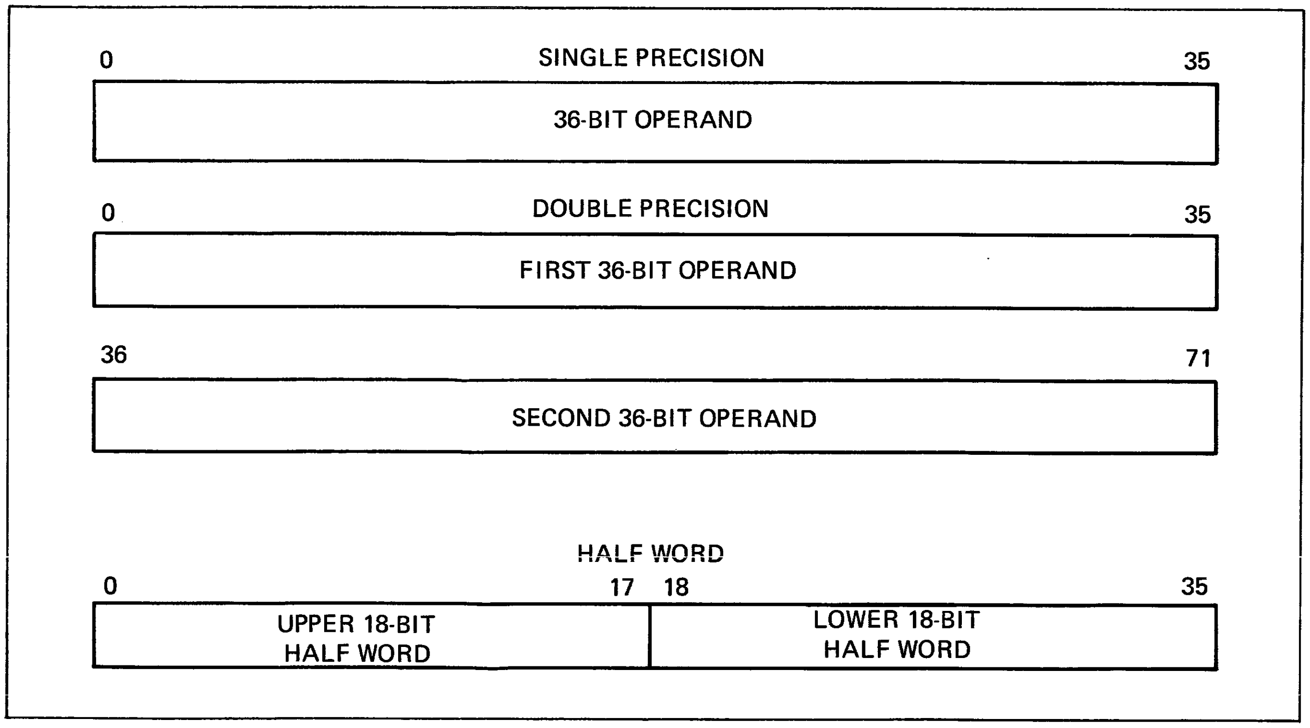


Figure 16. Fixed-Point Data Formats

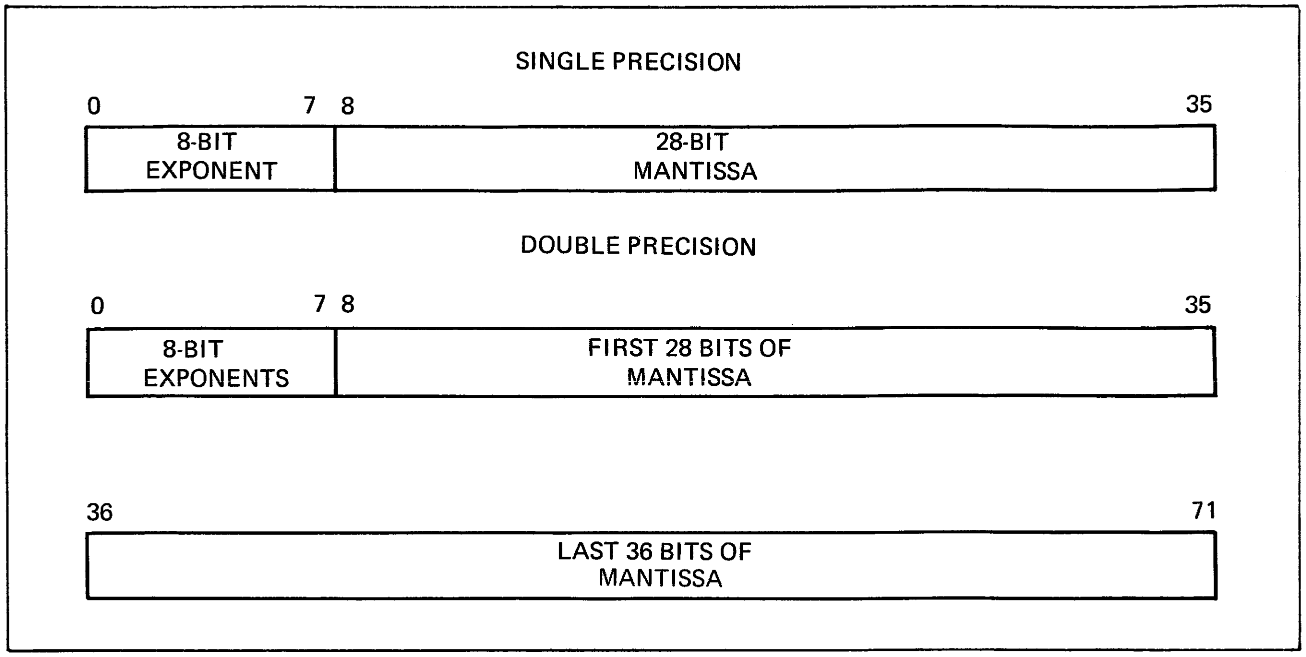


Figure 17. Floating-Point Data Formats

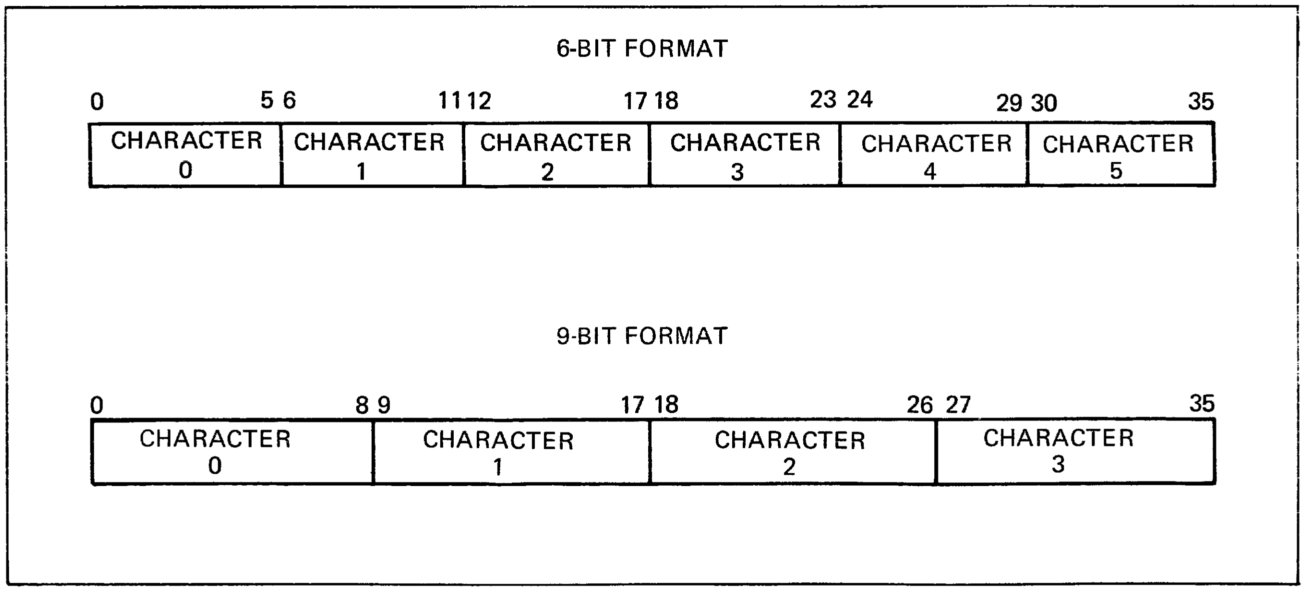


Figure 18. Six-Bit and Nine-Bit Alphanumeric Data Formats

Packed-Decimal Data

The EIS processor utilizes the ACSII 9-bit format to pack two packed-decimal characters in each ASCII character location. ASCII, in turn, uses the 9-bit character field as shown in Figure 19. When the EIS processor uses the bit string mode, the bit positions within a character are arranged as shown in Figure 20. Note that the "Z" in bit positions 0, 9, 18, and 27 represents the bit value 0 while other numbers in the fields represent the character positions.

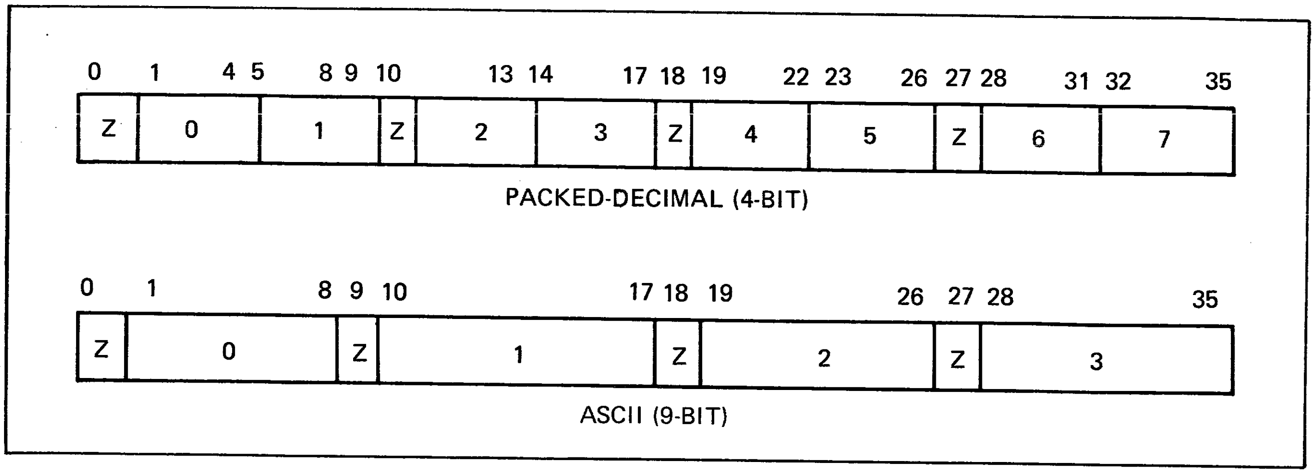


Figure 19. Decimal Formats for EIS

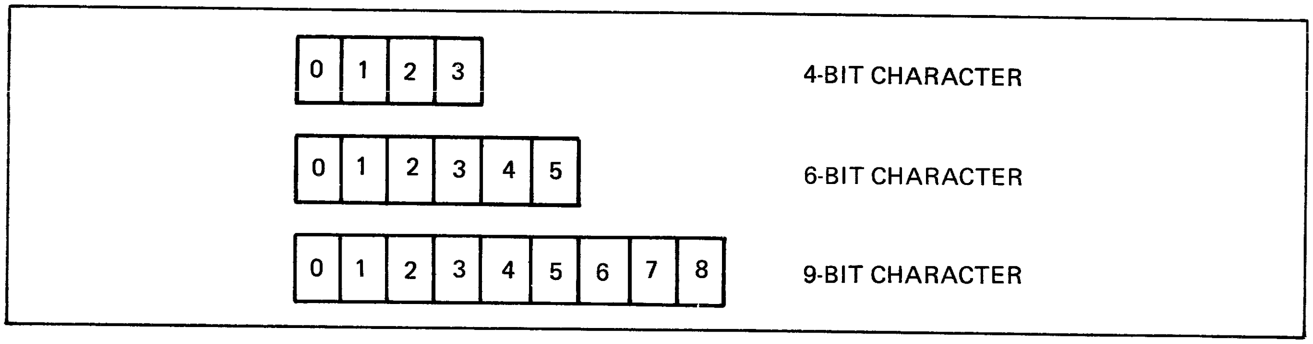


Figure 20. Bit Positions for EIS Bit String Mode

Floating-Point Decimal Data

Floating-point decimal numbers for EIS processors are represented as numeric digit strings in either 4-bit or 9-bit character formats.

The mantissa is stored as a leading signed decimal integer of *n* digits with the exponent stored as a trailing, 8-bit, twos complement number. The exponent represents a power of ten and indicates the number of places (character positions) the decimal point is to be moved left or right. Originally, the decimal point is assumed to be immediately to the right of the least significant digit of the number. Thus, the exponent range expressible in the 8-bit field is from 10^{-128} to 10^{+127}. The format for a floating-point decimal number is expressed in 9-bit characters such that the sign can start at *any* 9-bit character boundary.

In 4-bit character notation, there are four possibilities for floating-point decimal numbers. The number may start in either an odd or even character boundary and may contain an odd or even number of characters.

In the last example of Figure 21, the 8-bit exponent field, which now spans two character positions, is interpreted the same as in 9-bit character mode. The other two formats are formed with *n* even; this effectively exchanges the two exponent representations in the formats shown in Figure 21. Floating-point zero is represented as $+0 \times 10^{+127}$. The sign is represented as shown in Figure 22.

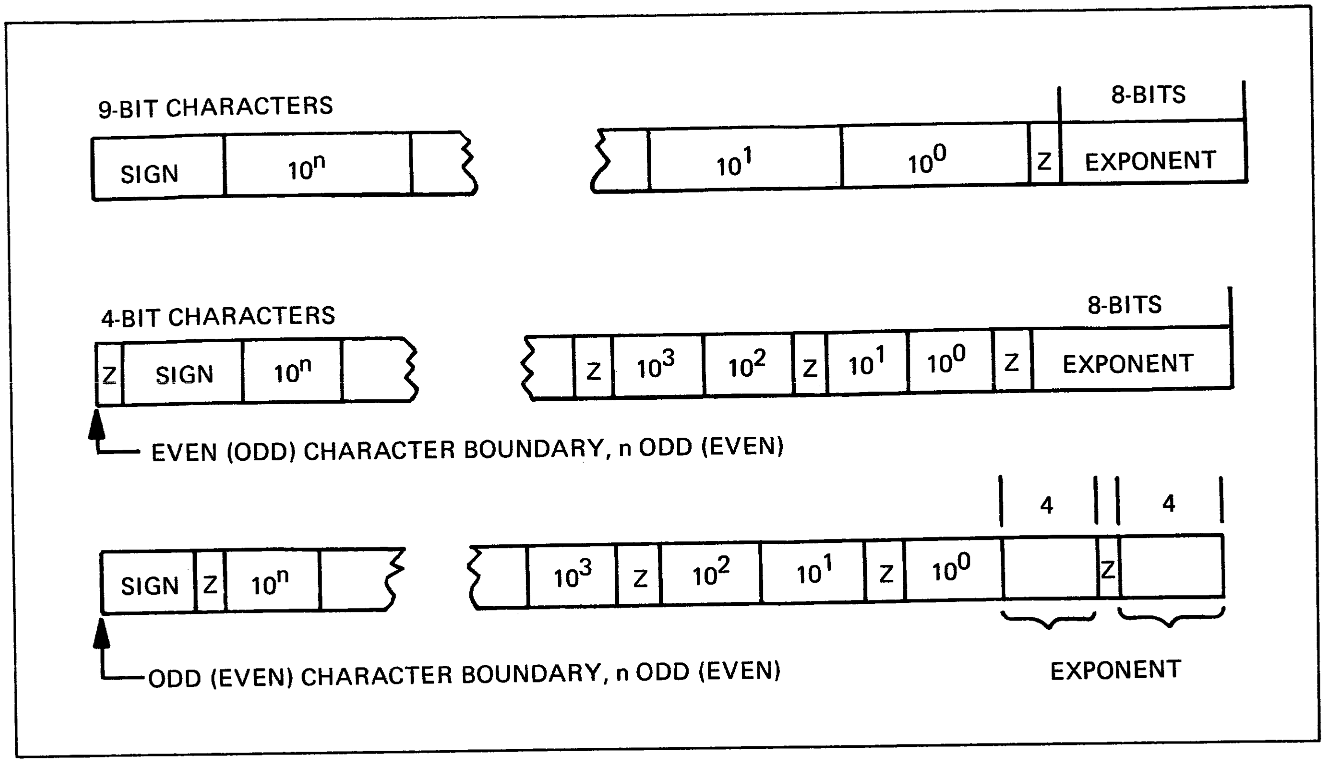


Figure 21. Floating-Point Decimal Formats

	PLUS	MINUS
4-Bit	13_8	15_8
9-Bit	53_8	55_8

Figure 22. Decimal Sign Position Formats

Instruction Format

The instruction repertoire of the processor includes more than 185 basic single-address instructions. With the exception of three Repeat instructions, the character-handling instructions, and EIS instructions, the instructions are formatted as shown in Figure 23.

The operation (op) part of the instruction specifies the operation to be performed and the registers that are involved.

The address (y) field of the instruction word specifies the storage location to obtain an operand or to place the result of the specified operation. Some of the exceptions to this general rule include:

- Shift instruction, where the address field designates the number of bit positions to shift.
- Program sequence transfers, where the address field designates the location of the next instruction to be executed.
- A group of instructions where the address field specifies the suboperations.

The address modifier (t_m) portion of the tag field specifies how the address contained in the instruction word is to be modified to form the effective address of the operand.

The designator (t_d) portion of the tag field specifies the type of register modification to be used.

The interrupt inhibit (i) portion of the instruction word, when set on, inhibits program interrupts in the processor by either fault or program execute interrupts from the memory.

NOTE: Extended instruction format (multiword) follows the discussion on address modification.

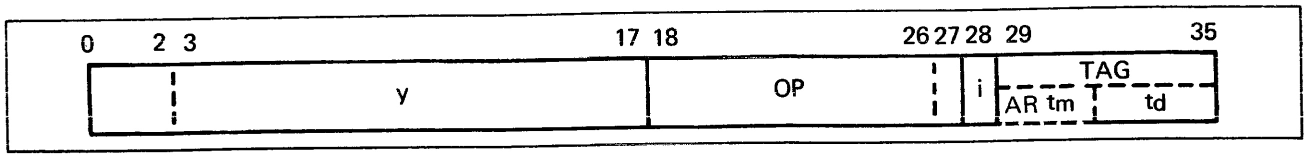


Figure 23. General Instruction Format

ADDRESS MODIFICATION

Modification Types

Address modification is performed in four basic ways, using the identifiers R, RI, IR, and IT on all Series 6000 processors. In addition, EIS processors may use address register modification and operand descriptors as explained later in this section. The modification types R, RI, IR, or IT are specified by unique binary codes placed in the t_m field (bits 30 and 31 in Figure 23) of the instruction word. The registers used to modify R, RI, or IR addresses are indicated in the t_d field of an instruction or indirect word.

- **Register modification (R)** – Modify the address by adding to it the contents of the indicated register, producing the effective address of the operand. Indexing registers are X0-X7, AU, AL, QU, QL, and IC. The direct operand modifier symbols DU and DL can be used to treat the address directly as the operand.
- **Register then indirect (RI)** – First perform the indicated register modification in the last indirect word encountered. (Some special conditions for modification may be specified in the indirect word.
- **Indirect then register (IR)** – First obtain the indirect word from the original address; then conduct the modification specified in the indirect word. Upon completion of the indirect addressing, perform the indicated register modification in the last indirect word encountered. (Some special conditions for modification may be encountered in the indirect words; these involve R, RI, and IT.)
- **Indirect then tally (IT)** – Obtain the indirect word using original address; then use the new address field of the indirect word to get the effective operand address. Next, follow one of ten possible variations, indicated in the t_d field of the instruction or indirect word that specified IT. Three of the variations (SC,SCR, and CI) are used for handling characters.

Register Designator – Each of the three modification types R, RI, and IR includes an indexing step which is further specified by the register designator t_d. In most cases, t_d really specifies the register from which the index is obtained. However, t_d may also specify that the effective address Y is to be used directly as the operand. The 16 registers are: None, index registers (X_0, X_1,...,X_7), the upper or lower half of the accumulator or quotient register, the instruction counter, and the effective address itself as the upper and lower half operand.

Under R, RI, IR modification, indexing the operand address of an instruction or indirect word adds no time to instruction execution.

IT Modification – The 11 possible variations of the IT modification are listed and described below. When used, each variation mnemonic appears as a unique 4-bit configuration in the t_d field of its instruction.

Mnemonic		Name
(t_d)	= I	Indirect (unmodified)
	= ID	Increment address, decrement tally
	= DI	Decrement address, increment tally
	= SC	Sequence character
	= CI	Character from indirect
	= AD	Add delta
	= F	Fault
	= IDC	Increment address, decrement tally, and continue
	= DIC	Decrement address, increment tally, and continue
	= SD	Subtract delta
	= SCR	Sequence character reverse

Character Operations

The character handling capability shown below is available on all Series 6000 systems. Operations involving characters are performed using instructions that specify the IT type of address modification in the t_m field. The I of IT obtains an indirect word that, in turn, references a location containing the character or characters to be used. The indirect word holds tallying and character-identifying information while the referenced location holds the character data. The t_d field of the original instruction must contain the sequence character (SC), sequence character reverse (SCR), or character from indirect (CI) designator variations.

Each time the original instruction is used under the SC or SCR variation, the tally and character position information in the indirect word are automatically changed by one, (1) to prepare for operating on the next sequential character and, (2) for terminating operations when the tally reaches zero.

For 6-bit character operations (Figure 24) in which the operand is taken from memory, the effective operand from memory is presented as a single word with the specified character justified to character position 5; positions 0-4 are presented as zero. For operations in which the result is placed in memory, character 5 of the operand replaces the specified character in memory location y; the remaining characters in memory location y are not changed.

For 9-bit character operations (Figure 24) in which the operand is taken from memory, the effective operand from memory is presented as a single word with the specified character justified to character position 3; positions 0-2 are presented as zero. For operations in which the result is placed in memory, character 3 of the operand replaces the specified character in memory location y; the remaining characters in memory location y are not changed.

EIS Address Modification

The EIS processor has all of the address modification capabilities previously discussed and in addition has a "second level indexing" using the additional address registers whose format is shown in Figure 25. The address registers permit character and bit addressing.

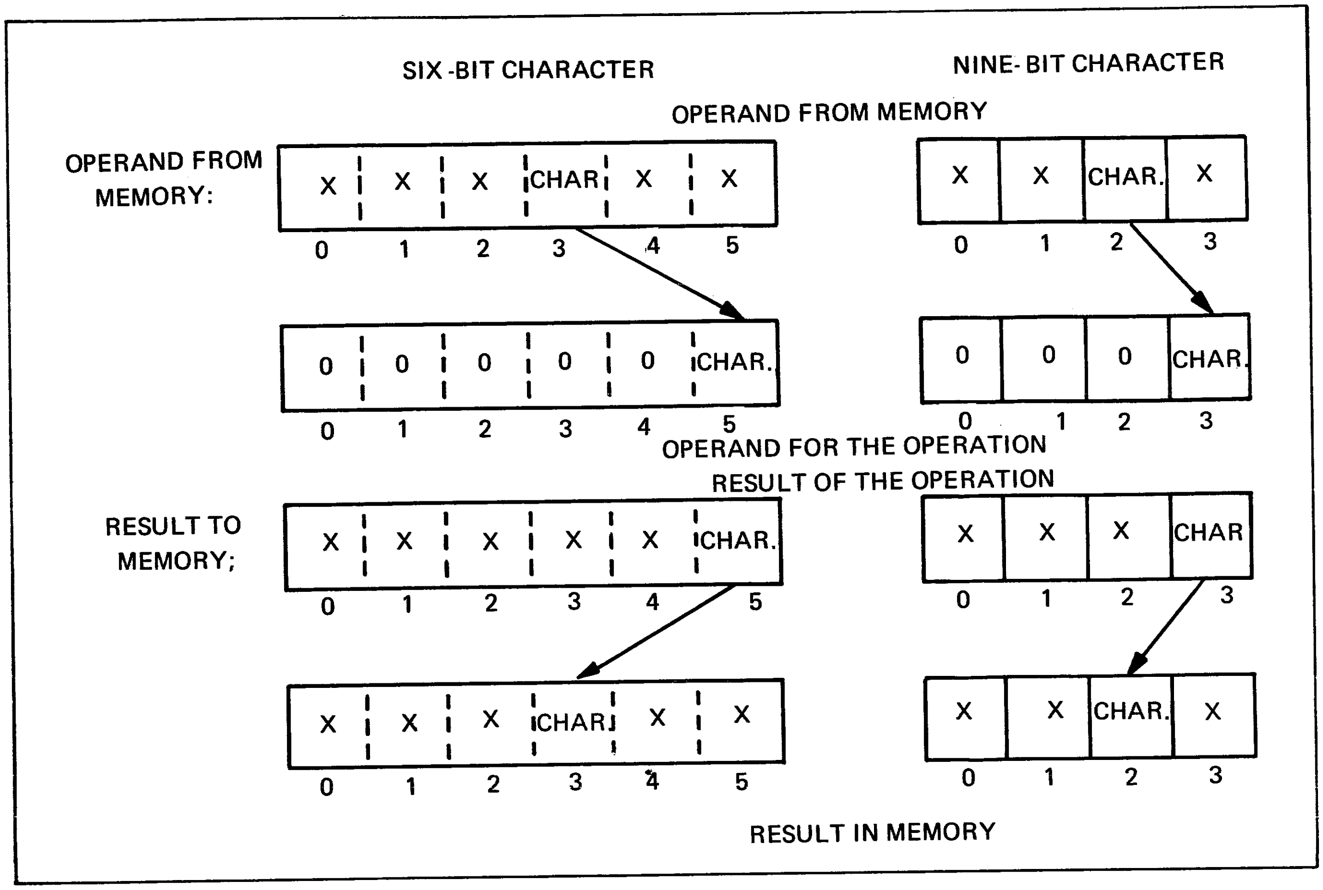


Figure 24. Typical Character Operations

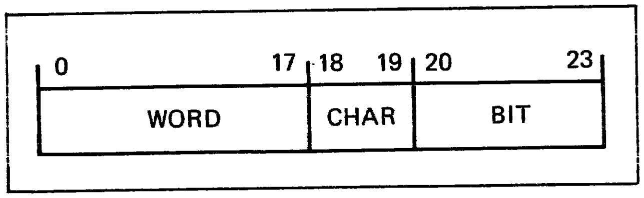


Figure 25. Address Register Format

When an address register is used in address preparation, its usage is specified in the instruction word. Some EIS processor instructions require more than one word; these are explained later. All single-word instructions with or without EIS have the same general instruction format. (See Figure 23). Bit 29 (AR) indicates whether an address register is to be used. If bit 29 is 1, the first three bits (0-2) of the y field designate which address register is to be used.

The procedure (illustrated in Figure 26) for single-word instructions (shown in Figure 23) using address registers is:

1. Bits 0-2 of y designate the address register to be used.
2. Bit 3 of y is used as the fill bit for bit positions 0-2, thus forming a twos complement signed number.
3. The resultant twos complement number is added to the value of the address register contents (bits 0-17), ignoring bits 18-23. (The address register is unchanged.)
4. Address modification continues as specified by t_m and t_d fields.

NOTE: Step 2 gives a range of plus 16,384 to minus 16,384 words.

INSTRUCTION REPERTOIRE

Most of the instructions available in the instruction repertoire are familiar to experienced programmers of large-scale computers. However, additional instructions have been provided on all models to give the programmer extended capability for character handling, decision-making, and advanced programming techniques involving list processing. On EIS models many powerful instructions, as well as the second-level indexing and descriptor words (to be discussed later), closely correlate to higher-level business languages (such as COBOL) to allow fast, efficient execution of business programs.

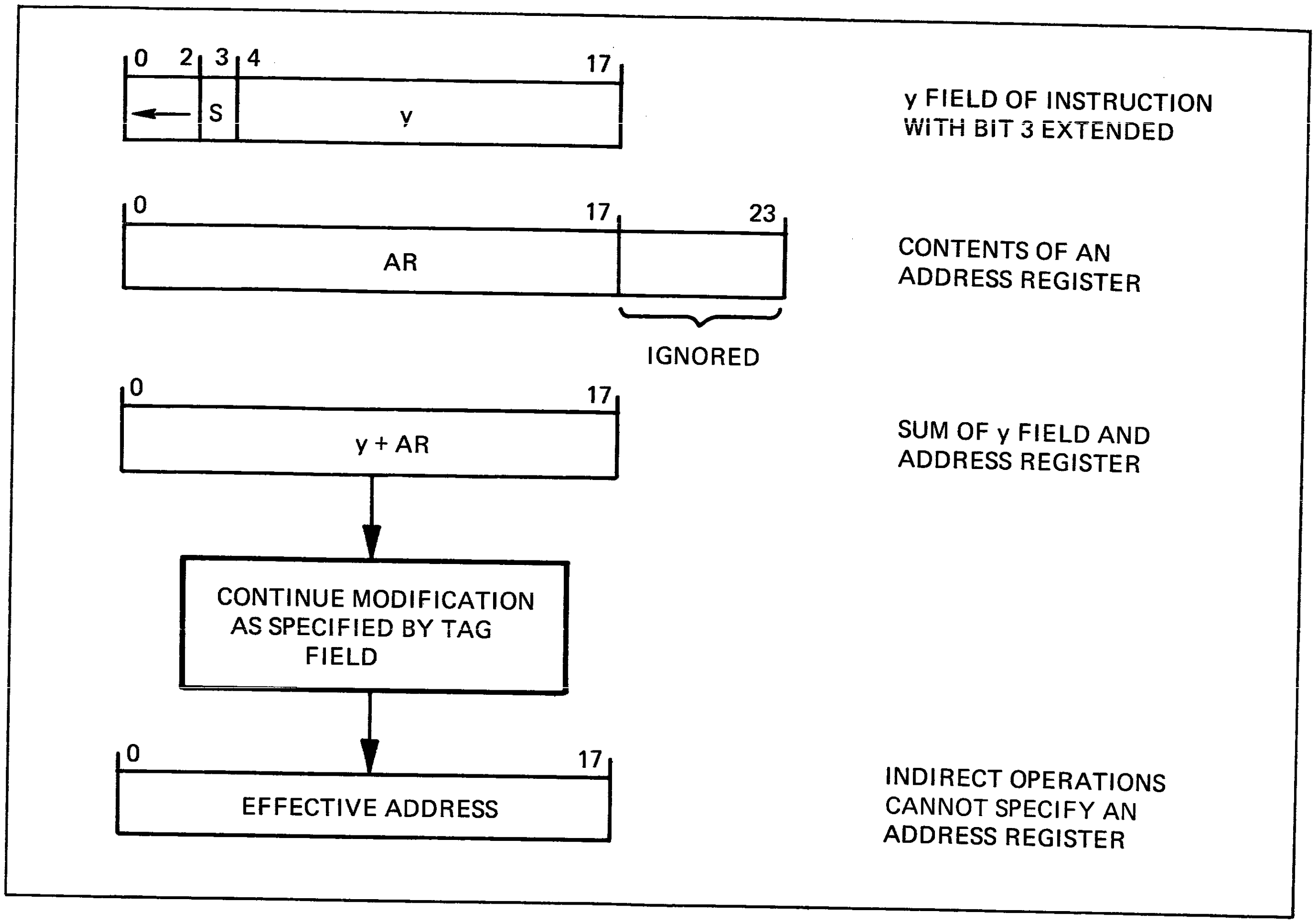

0 2 3 4 17
S
y
y FIELD OF INSTRUCTION WITH BIT 3 EXTENDED
0 17 23
AR
CONTENTS OF AN ADDRESS REGISTER
IGNORED
0 17
y + AR
SUM OF y FIELD AND ADDRESS REGISTER
CONTINUE MODIFICATION AS SPECIFIED BY TAG FIELD
0 17
EFFECTIVE ADDRESS
INDIRECT OPERATIONS CANNOT SPECIFY AN ADDRESS REGISTER

Figure 26. Address Modification of a Single-Word Instruction with the Use of an Address Register

On all models the basic instructions have provisions for multiple variations by permitting the programmer to specify not only the type of address modification desired, but also the source registers associated with particular operation codes. For example, the operation field for a Transfer and Save Instruction Counter in Index instruction specifies the index in the operation field, leaving full address modification capability free for destination calculation.

The processor performs efficient operations on 6-, 9-, 18-, 36-, and 72-bit operands.

The following paragraphs briefly describe salient features of the major instruction types. A complete listing of the full instruction repertoire and execution times is included in Appendices A and B.

Data Movement

Character handling and manipulation is facilitated by the "indirect and tally" (IT) indexing option and by instructions for directly loading and storing selected characters of the accumulator or quotient register.

Instructions are also included for directly loading the index registers from either memory of the A and Q registers, directly storing any register into memory, and loading registers with the complement of the memory location specified.

The Effective Address to Register (EAX) instructions permit the effective address of such an instruction to be placed in any of the index registers, in the A register, or in the Q register. Thus, any effective address referenced frequently in a program can be stored in a register and used without loss of processing time in repeatedly redeveloping the effective address. Furthermore, the EAX instructions provide the programmer with the ability to transfer data among any of the index registers, the A register, and the Q register.

Fixed-Point Arithmetic (Binary)

Instructions for both fractional and integer multiplication and division afford the programmer freedom from scaling the results of such operations. Fractional multiplications are performed with the multiplicand in the A register; the result appears in bit positions 0 through 70 of the AQ register, automatically scaled with the binary point to the left of position 0. Integer multiplications are performed with the multiplicand in the Q register; the result appears in bit positions 1 through 71 of the AQ register, automatically scaled with the binary point to the right of position 71.

Instruction	Registers Used	Principal Functions
Compare Magnitude	A	Compares absolute values
Compare	A,Q,AQ,X0-X7	Compares algebraic values or characters
Comparative AND	A,Q,AQ,X0-X7	Tests for all zeroes in word fields
Comparative NOT	A,Q,AQ,X0-X7	Tests for all ones in word fields
Compare Masked	A,Q	Searches for identical, selectable word fields
Compare with Limits	A,Q	Searches for a value within selectable limits

Figure 27. Fixed-Point Comparison

Fractional divisions use the full range of the AQ register for the dividend; the quotient appears in the A register with the remainder in the Q register. The binary point is automatically scaled to the left of position 0. Integer divisions have the integer dividend in the Q register, with the binary point to the right of position 35. After division, the quotient is in the Q register with the binary point automatically placed to the right of position 35; the remainder is in the A register.

Normally, integer operations of divide and multiply occur in the Q register, and fractional operations of divide and multiply occur in the A register. This convention permits easy programming of fixed-point arithmetic operations.

Two arithmetic and three logic instructions used frequently in program coding are provided for combining the contents of memory locations directly with the contents of registers and storing the results in the same locations, without recourse to separate store instructions. In all cases, the programmer can use the 18-bit indexing registers, X0 through X7, and the 36-bit A and Q registers. In effect, the Add and Subtract to Storage instructions make arithmetic accumulators of all available memory locations. In all cases, the register contents are undisturbed.

Boolean Operations

The logical operations AND, exclusive OR, and inclusive OR are permitted between storage and the Xn registers, A and Q registers, and the AQ register.

Comparison

Compare operations do not alter the contents of storage or the specified register, but merely set or clear the appropriate indicator as the result dictates. The compare instructions enable the programmer to make many types of program decisions. (The fixed-point compare instructions are shown in Figure 27.)

Floating-point compare instructions are included for single- and double-precision operations on absolute values and algebraic values. All compare instructions are repeatable using the RPT, RPD, or RPI instructions described below. (See "Execute Instructions.")

Floating-Point Arithmetic (Binary)

Floating-point operations can be performed on both single- and double-precision data words; complete sets of data movement, arithmetic, and control instructions are provided for use in both types of operations. Unless otherwise specified by the programmer, the mantissas of all floating-point operation results are automatically normalized by the hardware. In additions and subtractions, addends and subtrahends are automatically aligned.

Operations on floating-point numbers are performed using an extended register composed of a 72-bit AQ register, which holds the mantissa, and a separate 8-bit exponent register; operations on the exponent and mantissa are performed by two separate adders. The existence of separate exponent and mantissa registers and adders enables the programmer to intermix efficiently single- and double-precision instructions.

The floating-point instruction repertoire includes two especially convenient divide instructions: Floating Divide Inverted (FDI) and Double-Precision Floating Divide Inverted (DFDI). These instructions cause the contents of the memory location to be divided by the contents of the AQ register – the reciprocal of other divide instructions in the repertoire. Thus, regardless of whether the contents of the AQ register must be a dividend or a divisor, the programmer can always perform a division without recourse to wasteful data movement operations.

Floating Negate, Normalize, Add to Exponent, and Single- and Double-Precision Compare instructions further facilitate effective programming.

Transfer of Control

The complement of program transfer instructions permits the storage of the instruction counter in indexing registers X0 through X7 (at programmer option), an unconditional transfer, and conditional transfers. Conditional transfers on zero, plus, and carry have corollary transfers: nonzero, minus, and no carry. Transfers on overflows and underflows are to maskable fault routines. If the normal fault routine is masked, transfer is at programmer option.

Execute Instructions

The Execute Single and Execute Double (XEC and XED) instructions allow the programmer to execute remote instructions singularly or in pairs. Because the instruction counter is not disturbed by the execution of remote instructions, the program will continue sequentially after executing the XEC or XED referenced instructions, providing the referenced instructions do not alter the instruction counter. If a referenced instruction affects the instruction counter, a program transfer occurs.

Several special instructions are provided to expand programmer options and to reduce coding work through utilization of hardware features.

Three repeat instructions in the repertoire provide unusual programming advantages: Repeat (RPT), Repeat Double (RPD), and Repeat Link (RPL). The RPT and RPD instructions permit execution of the next one or two instructions a selected number of times according to program requirements; they are especially useful for operating upon sequential lists in memory. For example, if RPT is used with any of several compare instructions to search a list, termination of the repeats will occur when a "hit" is made according to programmer option as established by conditions in the RPT instruction. The "hit" causes transfer to the next sequential instruction.

High-speed movement of large volumes of data can be accomplished by repeated execution of Load AQ, Store AQ instructions, using the RPD command. Multiple executions of these instructions require memory access only for data to be moved.

The RPL instruction is similar in its execution to the RPT and RPD; it facilitates the processing of threaded lists scattered through memory.

Code Conversion Instructions

The Binary to Binary Coded Decimal (BCD) instruction converts the magnitude of a 36-bit or smaller binary number to its decimal equivalent in BCD form. The conversion is made automatically, one decimal digit per instruction execution, using previously-stored conversion constants. The BCD form of the converted number is readily available for further operations.

The Gray to Binary (GTB) instruction converts a 36-bit word containing data in the Gray code (for example, coded analog information from an analog-to-digital input device) to its binary equivalent in only one execution of the instruction. This instruction enhances the use of the information system in real-time applications.

Gating Instructions

To provide GECOS with program gating for multiprocessor configurations, three slave mode instructions provide for clearing the memory cell referenced to zero after the contents are transferred to the processor. These are LDAC, LDQC, and SZNC.

Master Mode Instructions

The Connect instruction is the only input/output instruction in the information system repertoire. The processor, having set up the I/O commands in the system memory, issues a Connect instruction to the input/output multiplexer, which then assumes I/O responsibility.

All master mode operations, including the Connect instruction, are reserved for GECOS and other special-purpose operations that are written as extensions to GECOS.

EIS PROCESSOR INSTRUCTIONS

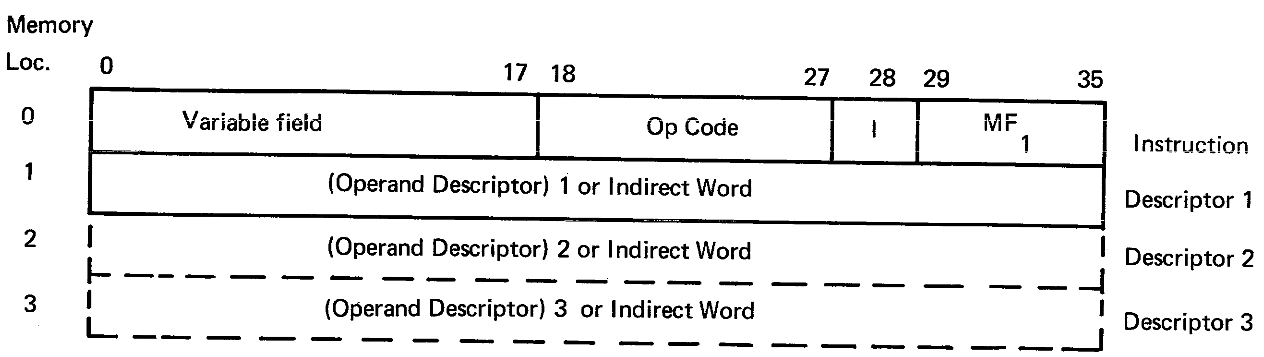
Special EIS instructions not only provide powerful arithmetic capabilities but also provide "on-the-fly" code translation to permit data of different format codes to be used as operands of an instruction. For example, an ASCII-scaled number can be multiplied directly by a packed-decimal floating number and the rounded result stored as a different scaled packed-decimal number using only one EIS instruction. Also during execution of an edited move instruction a series of up to 63 of the 17 different micro operations may be used to edit a string of up to 63 characters from any format into any format, using only one EIS instruction and a series of 6-bit micro operations.

The EIS processors execute 30 basic multiword instructions and have 24 additional single-word instructions. The multiword instructions have 1 to 3 operand descriptors following the first word, which has the same general appearance as single word instructions. Most of the additional single-word instructions are associated with loading, storing, or modifying the EIS address registers.

The two edited move instructions provide micro operations, which are summarized later in this section.

EIS Multiword Instruction Format

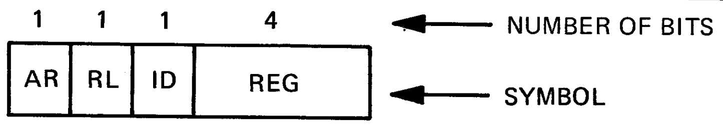
The alphanumeric, numeric, and bit string EIS instructions use extended (multiword) instruction format (Figure 28), modification field (MF) format (Figure 29), and operand descriptor word format (Figures 30 and 31). The register (REG) codes are approximately the same as non-EIS R-type modification. In addition, for indirect string length specification the length field codes are similar to the REG field. Exceptions occur for the use of the y field as direct operands (DU and DL), the instruction counter (IC), and the lower accumulator or lower quotient (AL, QL); DU, DL, and IC cause faults, while AL and QL use more than the usual 18 bits to provide counts of 20 to 24 bits, depending on the type of field string. (See the Programmers Reference Manual for further details.)



Memory
Loc.
0
17 18
27
28 29
35
0
Variable field
Op Code
I
MF_1
Instruction
1
(Operand Descriptor) 1 or Indirect Word
Descriptor 1
2
(Operand Descriptor) 2 or Indirect Word
Descriptor 2
3
(Operand Descriptor) 3 or Indirect Word
Descriptor 3

Variable Field	–	contains additional information concerning the operation to be performed, depending on the particular instruction.
Op Code	–	the 10-bit operation code field.
I	–	the interrupt inhibit bit.
MF_1	–	Modification Field 1 describes address modification that is to be performed for descriptor 1. When descriptor 2 and 3 are present, there will be a corresponding MF 2 and 3 within the variable field to describe the address modification to be performed on these operands.

Figure 28. Extended Instruction Format



AR – Address Register Specifier

0 - no address register used.

1 - bits 0-2 of the operand descriptor address field y specify the address register to be used in computing the effective address for the operand.

RL – Register or Length

0 - operand length is specified in the length field of the operand descriptor.

1 - Length of operand is contained in a register that is specified by a code in the length field (bits 32-35) of the operand descriptor in the format of REG.

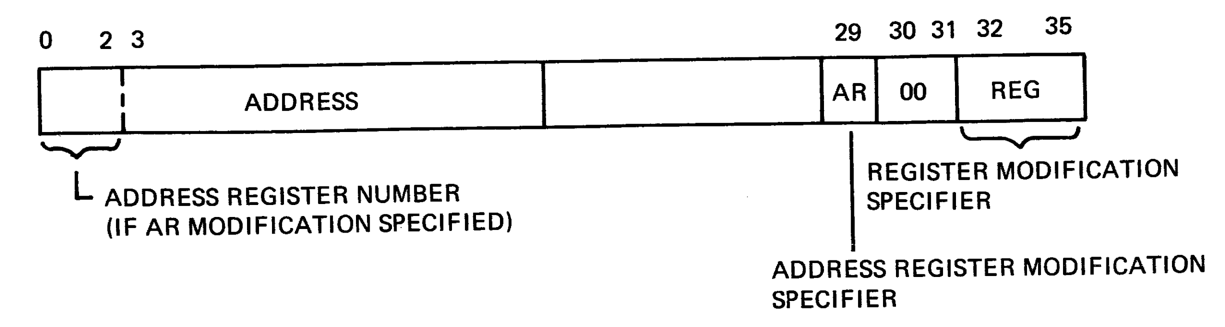
ID – Indirect Operand Descriptor

0 - the operand descriptor follows the instruction word in its sequential store location.

1 - the operand descriptor location contains an indirect word that points to the operand descriptor. Only one level of indirection is allowed.

REG – Address modification register selection for R-type modification of the operand descriptor address field.

Figure 29. Modification Field Format


0
2 3
29
30 31
32
35
ADDRESS
AR
00
REG
ADDRESS REGISTER NUMBER (IF AR MODIFICATION SPECIFIED)
REGISTER MODIFICATION SPECIFIER
ADDRESS REGISTER MODIFICATION SPECIFIER

THE AR AND REG FIELDS ARE IDENTICAL IN FUNCTION WITH THE CORRESPONDING MODIFICATION FIELDS IN THE INSTRUCTION.

Figure 30. Operand Descriptor

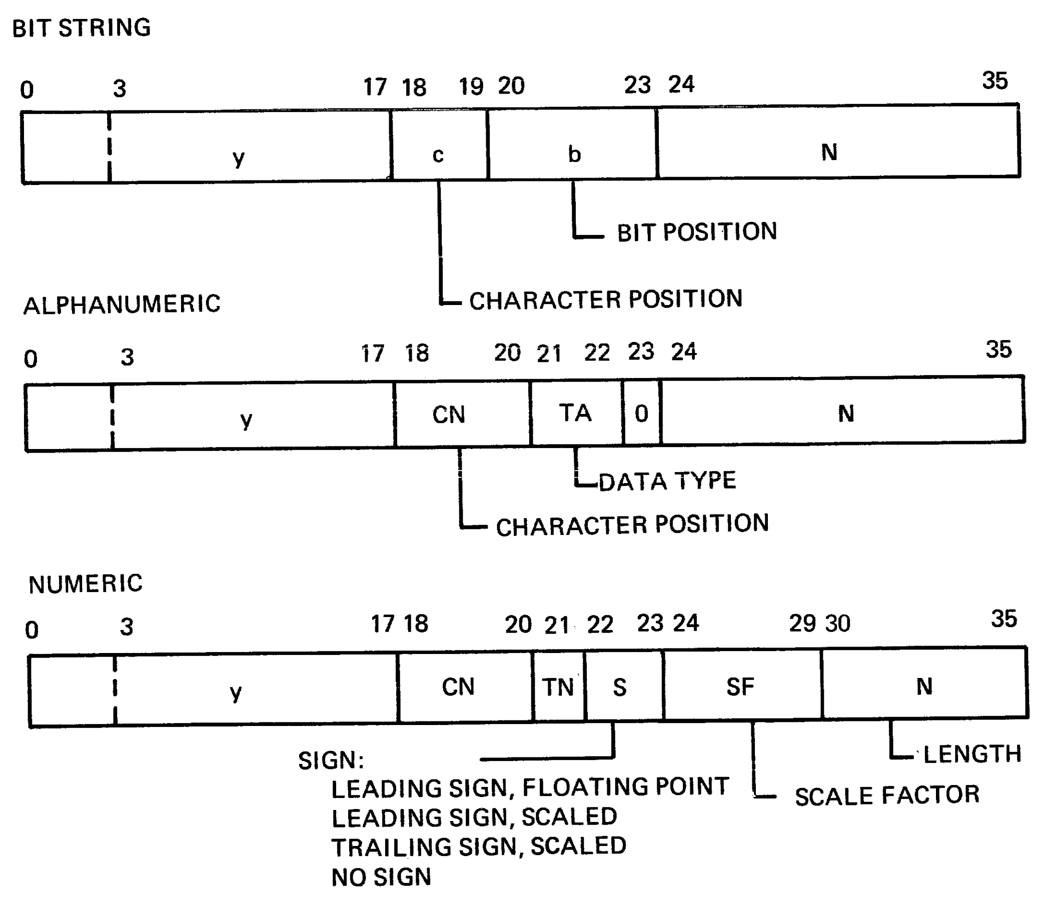


Figure 31. Operand Descriptors by Type

y = Original data word address.
18 bits (0-17) if address register not specified.
15 bits (3-17) if address register specified, with bit 3 extended; i.e., if bit 3 is zero, bits 0-2 are also zero; if bit 3 is one, bits 0-2 are ones also.

c = original character position within a word of 9-bit characters.

Code	Char.
00	0
01	1
10	2
11	3

b = original bit position within a 9-bit character

Code	Bit	Code	Bit
0000	0	0101	5
0001	1	0110	6
0010	2	0111	7
0011	3	1000	8
0100	4		

All other combinations of these 4 bits are illegal codes and will cause an Illegal Procedure Fault (IPR).

N = either the number of characters or bits in the data string or a 4-bit code (32-35) that specifies a register that contains the number of characters or bits.

CN = original character number within the data word referenced by the original data word address. Code for the CN depends on the data type as shown below.

	Legal Codes	Char. No.	Illegal Codes
ASCII (9-bit)	000	0	001
	010	1	011
	100	2	101
	110	3	111
Hollerith (6-bit)	000	0	110
	001	1	111
	010	2	
	011	3	
	100	4	
	101	5	

	Legal Codes	Char. No.
Packed-Decimal (4-bit)	000	0
	001	1
	010	2
	011	3
	100	4
	101	5
	110	6
	111	7

Figure 31. Operand Descriptors by Type (Cont.)

TA = a code that defines which type alphanumeric characters are in data.

Code	Data Type
00	ASCII
01	Hollerith
10	Packed-Decimal
11	Illegal - Causes IPR fault

TN = a code that defines which type numeric characters are specified.

Code	Data Type
0	ASCII
1	Packed-Decimal

S = Sign and Decimal Type

Code	Description
00	Leading sign, floating point
01	Leading sign, scaled
10	Trailing sign, scaled
11	No sign, scaled

SF = Scaling Factor – A twos complement binary number that indicates the scale position for a scaled decimal number. Scale of zero indicates the scale point is immediately to the right of the least significant digit (the number is an integer). This gives a range of scale factor from -32 to +31.

Figure 31. Operand Descriptors by Type (Cont.)

EIS Multiword Instruction Repertoire

There are 30 basic multiword instructions; they are classified into four general categories: alphanumeric instructions, numeric instructions, bit string instructions, and conversion instructions. A general description of each category follows. A complete list of instructions and timings may be found in Appendix A for all instructions except EIS; execution timings for EIS models are listed in Appendix B.

Alphanumeric Instructions are concerned with moving, transliteration, complete editing, and comparing. The operands for these instructions can be any combination of the alphanumeric types (9-bit ASCII, 6-bit BCD, or 4-bit packed-decimal), and are translated "on-the-fly" to permit the different types of character strings to be manipulated in the same instruction.

Numeric Instructions not only include decimal arithmetic but also moving, comparing, and complete editing. The decimal arithmetic hardware has unusual and valuable combinations of characteristics. The first characteristic makes decimal add, subtract, multiply, and divide available. Second, the instructions can be 2- or 3-operand instructions. Third, the operands can be 9-bit ASCII or packed-decimal, or both. Fourth, the numbers can be floating-point with leading sign, or can be scaled fixed-point with trailing sign, leading sign, or no sign. Again, as with the alphanumeric instructions, these various characteristics are handled on-the-fly by a single instruction.

Bit String Instructions consist of five basic instructions. One compares two strings. The other four basic instructions extend to provide all 16 Boolean operations for each of the four instructions to combine strings and set indicators.

Conversion Instructions provide for decimal/binary conversion. These use bit-string format for binary and scaled-decimal for the decimal operand.

EIS Single-Word Instruction Repertoire

There are 24 basic single-word instructions for EIS processors. These include instructions for loading and storing of address registers, instructions for altering the contents of address registers, special instructions to use address registers to manipulate the address portion of numeric and alphanumeric descriptors, instructions to save and restore interrupt registers, and instructions for conditional transfer based on indicators. (A complete listing and description can be found in the Programmers' Reference Manual.)

EIS Edited Move Micro Operations

Both alphanumeric and numeric edited move instructions utilize micro operations (MOPS) to perform versatile editing functions in an efficient manner. The sequence of micro steps to be executed is contained in storage and is referenced by the second operand descriptor of the multiword edited move instructions. Some of the micro operators require special characters for insertion in the string of characters being manipulated. These special characters are contained in the Edit Insertion Table.

Micro Operation Sequence – The operand descriptor for micro operations points to a string of 9-bit characters that specify the micro operation to be performed during an edited move. Each of the 9-bit characters defines a micro operation. The character format is shown in Figure 32.

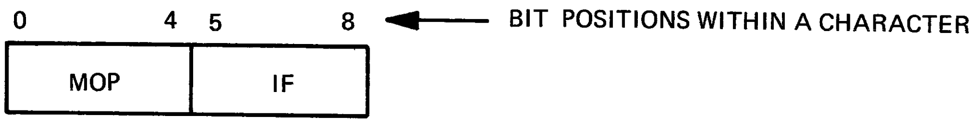
The micro operations will be terminated when the MOP string length is exhausted or when either the source or receive string lengths become exhausted. An attempt to call a MOP from an exhausted MOP string causes an IPR fault.

Edit Insertion Table – When executing an edit instruction the processor uses a register of eight 9-bit characters to hold insertion information. This register, called the Edit Insertion Table, is automatically preloaded at the start of each edit instruction with the ASCII characters, respectively: blank, asterisk, plus sign, minus sign, dollar sign, comma, period, and zero-corresponding to 1 through 8 positions in the table. All or any of the entries can be changed during the edit instruction execution by means of the load table entry (LTE) or change table (CHT) micro operations.

Edit Flags – Four edit flags are provided for use with MOPS:

- *End Suppression* (ES) flag is initially OFF and is set ON by a micro operation when zero suppression ends.
- *Sign* (SN) flag is initially set OFF if the sending field has an alphanumeric descriptor. If the sending field has a numeric descriptor, the sign is initially read from the sending string from the character digit position defined by the sign and decimal type field(s). SN is set OFF if positive, ON if negative.
- *Zero* (Z) flag is initially set ON. It is set OFF whenever a source digit is read that is not zero.
- *Blank-When-Zero* (BZ) flag is initially OFF and is set ON by either the ENF or SES micro operation. If at the completion of a move, both the Z and BZ are ON, the receiving field will be forced filled with the character held in the insertion table entry 1.

A complete list of micro operations is included in Appendix D.



MOP – Micro Operation

A code which specifies the micro operation to be performed.

IF – Information Field

Indicates one of the following, depending upon the MOP:

1) Number of source digits to be manipulated (1-16).
2) Number of the particular entry in the edit insertion table to be used.
3) An interpretation of the "blank-when-zero" operation.

Figure 32. Micro Operation Format

Section VII
Input/Output Multiplexer (IOM)

In Series 6000 Information Systems, the input/output multiplexer (IOM) is the coordinator of all input/output operations between the complement of peripheral subsystems and system controllers. The input/output multiplexer operates essentially as a hard-wired program device controlled by, and sharing memory with, a processor. Data transfers between a peripheral device and memory are accomplished by the IOM while the processor runs the jobs. Peripheral devices are controlled by processor-prepared control words stored in the memory.

Significant features of the input/output multiplexer include:

- Complete memory protection for all input/output multiplexer data transfers.
- Total awareness of the number, types, and states of up to 24 I/O subsystems per IOM.
- Hardware-software integration resulting from control programs designed to take full advantage of equipment features.
- Ability to interrupt processor operations.
- Scatter-gather of noncontiguously stored I/O program data.
- A special I/O processor for peripheral channel management.
- Maximum data transfer capacity in excess of 6 million characters per second.
- Maximum single-channel data transfer capacity of 1.3 million characters per second.
- Simultaneous operation of peripheral subsystems in a wide range of equipment configurations.
- Communication with multiple independent system controllers.
- Scratchpad storage for control words on some models.
- Six special channels to provide for specific system functions including new test and diagnostic aids.

IOM DESCRIPTION

The IOM consists of the IOM central and a variable number of channels. The IOM central controls access to storage for each of the channels and is capable of performing one storage access cycle at a time through the appropriate system controller. The time for this service is substantially less than the time required for a requesting channel to accumulate or disperse the data. As a result, the IOM central is time-shared by a number of channels operating concurrently.

The eight types of IOM channels are classified as data or special channels as listed below:

Data Channels

- Common Peripheral Channel
- Peripheral Subsystem Interface Channel

Special Channels

- Bootload Channel
- Connect Channel
- System Fault Channel
- Wraparound Channel
- Snapshot Channel
- Scratchpad Access Channel

DATA CHANNELS

Of primary interest to the user are the data channels. Data channels are responsible for information transfer to all external peripheral devices. This responsibility includes the control of instruction, data, and status flow between the IOM and the peripheral control unit.

The channel control words of the data channels of larger models are held in the IOM central scratchpad storage. The scratchpad is a high speed store accessible by the IOM channels and the processor through the scratchpad access channel. The scratchpad storage provides higher-speed servicing of data transfers through the data channels, and reduces the number of memory accesses required for control word retrieval and update.

The IOM provides information transfer to external peripherals through the two types of data channels – common peripheral channels and peripheral subsystem interface channels.

Common Peripheral Channels

Common peripheral channels (CPC) control the flow of instructions, data, and status between the IOM central and a peripheral control unit. The mode of operation used by all common peripheral channels is indirect. The IOM central generates storage addresses based on the control of words stored in scratchpad storage. These channels transfer data in excess of 650,000 characters per second.

Peripheral Subsystem Interface Channels

The peripheral subsystem interface (PSI) is a new interface used between the IOM and advanced peripheral subsystems, in place of the common peripheral channel (CPC). Peripherals available for Series 600 use the CPC; thus, the IOM could have both CPC and PSI channels on the same system.

The PSI is significant because it allows devices having very high transfer rates to be attached and used to their fullest extent. These channels transfer data up to 1.3 million characters per second.

SPECIAL CHANNELS

The six special channels are a standard part of the IOM and provide specific functions under control of the operating system, test and diagnostic software, or external switches.

Bootload Channel

The bootload channel consists of a nondestructive, 10-word storage area implemented as an IOM channel. When activated, it forces a list of control words and a limited number of interrupt vectors and control words to be stored in core storage and then executed by a processor so as to place the system in operation.

Connect Channel

The connect channel controls the distribution of instructions that initiate the operation of any addressable channel capable of receiving instructions.

System Fault Channel

The system fault channel is responsible for reporting the occurrence of system fault conditions to the software. System faults are abnormal conditions resulting from the failure of hardware or system software. Abnormal conditions in customer programs are reported by the affected data channel and are classified as user faults rather than as system faults.

Wraparound Channel

The wraparound channel is a special channel intended for use by test and diagnostic software. By means of instructions issued by the software, the wraparound channel causes the IOM central to perform any specific type of service on behalf of any data channel number designated in the instruction. The wraparound channel includes a data register and a status register which can be loaded and stored by means of appropriate services, so that most data paths and control sequences in the IOM central can be checked under program control.

Snapshot Channel

The snapshot channel is a specific channel for use by test and diagnostic software. It permits the software to sample certain signals in the IOM central on the occurrence of events selected by the software.

Scratchpad Access Channel

The scratchpad access channel is a special channel used by the operating system and test and diagnostic software. It permits the software to read from or write to the scratchpad.

DATA COMMUNICATIONS INTERFACE

Any common peripheral data channel can be used as a data communications interface to a DATANET 30 or DATANET 305. The channel selected is known to GECOS and is used with all job I/O activities associated with the communications processor. Outgoing data from a job communicating through a remote terminal is handled by GECOS and the IOM in the same manner as I/O transfers to any other peripheral device connected through the input/output multiplexer. When the communications processor has incoming data for the job, the processor causes an automatic processor interrupt via the IOM. The interrupt causes GECOS to initiate an I/O data transfer from the communications processor to the associated job.

SCATTER-GATHER CAPABILITY

Associated with each input/output channel and stored in main memory may be scatter-gather lists (data control lists) that are part of each job. The words making up these lists are referred to as data control words (DCWs). During write operations, data can be gathered from these blocks of memory locations and transferred to any peripheral device; analogously, data can be scattered during read operations. After completion of all data transfers associated with a scatter-gather list, control is returned via the program priority interrupt facility and processor to GECOS.

IOM MEMORY PROTECTION

The data control list in each job is composed of data control words that contain, among other data information used by the input/output multiplexer to achieve I/O memory protection on all I/O transfers.

For each block of data in a job, there is a DCW containing the relative starting address and the word count of the block. At the start of every block of data, the IOM adds the lower address limit (submitted by GECOS) to the relative starting address to complete the absolute starting address. The word count is added to the absolute starting address and compared with the upper address limit (also submitted by GECOS). Thus, the lower and upper absolute address limits for each data block are checked; memory protection is guaranteed. If any address limit is out of tolerance, the IOM transfers control to a fault routine.

IOM PROGRAM INTERRUPTS

The input/output multiplexer can cause program interrupts of processor operations under the following conditions:

- Completion of a previously initiated I/O activity.
- Occurrence of special conditions (completion of magnetic tape rewind, return to ready status after manual attention to a printer, etc.).
- Termination of an activity because of abnormal or fault conditions requiring attention of software.

The IOM accomplishes the program interrupt by setting a predetermined program interrupt cell in the system controller. The system controller then causes a program interrupt in a processor so that appropriate action can be taken by the software.

Section VIII
Remote Input/Output Operations

Data communication systems, essential to many existing and future computer applications, are integral parts of the Series 6000 Information System. GECOS permits multiple remote terminals and computers at widely-separated locations to communicate with the system. One-way or two-way communication over common and private carrier lines, coaxial cables, or via microwave transmissions can be channeled through a selection of DATANET front-end communications processors.

REMOTE PROCESSING CAPABILITIES

Remote processing capabilities are provided in many modes of the Series 6000 Multidimensional System:

- Remote batch
- Remote access
- Time sharing
- Transaction processing

Through these dimensions or modes of operation, the remote terminal user has access to the full capabilities of the system.

GECOS takes advantage of features within the information system to enable the user (in various data communications applications) to achieve:

- Effective remote data processing – The multiprogramming environment can include a large number of users in both local and remote locations with diverse processing needs.
- Radical reductions in turnaround time – Total turnaround time, the period between the submission of a job for processing and the receipt of the desired output, is a function of several factors:
  - Input time
  - Time spent in awaiting processing
  - Actual processing time
  - Time spent in awaiting output
  - Actual output time

Through the use of multiple local and remote input/output devices, simultaneous processing and input/output operations, and automatic job interrupts, the nonproductive waiting time and output time can be reduced to a level commensurate with the high-speed processing capabilities of the information system.

Standard terminals accommodated by GECOS include:

- Honeywell Series 100 remote batch computers
- Honeywell VIP 765/775 visual display terminals
- Teletype Models 33, 35, and 37
- GE TermiNet 300
- IBM 2741

The three communications processors that provide the front-end processing of Series 6000 Information Systems are:

- DATANET 355 Communications Processor
- DATANET 305 Communications Processor
- DATANET 30 Communications Processor

These processors are described below.

DATANET 355 PROCESSOR

The DATANET 355 is a high-performance, stored-program communications processor designed to match large-volume communications needs of the Series 6000 Multidimensional Information Systems. The DATANET 355 features total integrated circuit logic construction and a memory size of 16,384 or 32,768 words (18 bits) with a cycle time of one microsecond. The DATANET 355 accommodates data of variable word lengths – 6, 9, 18, or 36 bits. All data word lengths are individually addressable to allow highly efficient processing of tabular data. Ninety-eight instructions in an 18-bit format are provided, with one single-address instruction per word. Three index registers and multilevel indirect addressing, with indexing at all levels, give an addressable storage capability of up to 32,768 words.

The input/output is designed to facilitate efficient real-time, concurrent servicing of multiple terminals and peripheral devices. Up to 16 adapters can be provided to accommodate a total data transfer rate of up to 500,000 words per second (with 6, 9, 18, or 36 bits per word). Sixteen levels of priority interrupt, with 16 sublevels per level and corresponding interrupt masks, are provided.

The system organization of the DATANET 355 follows the pattern of the Series 6000. The DATANET 355 is a storage-oriented computer with its own independent memory, processor, and input/output modules. These three basic DATANET 355 modules are independently timed and operate asynchronously with each other. The processor and the input/output controller (which are active units) process data at their own rates and request cycles from the storage module (a passive unit) as the need arises. Only when the processor executes certain input/output instructions must the processor and the input/output controller of the DATANET 355 communicate with each other.

Communications Subsystem

The DATANET 355 communications subsystem, as a module of the Series 6000 Information System, is capable of simultaneously handling up to 200 teletypewriter users (110 to 300 bps), or 32 remote batch users (voice-grade or broadband), or 32 CRT subsystems (voice-grade), or an appropriate mix of these three classes. (See Figure 33.)

The DATANET 355 communications subsystem consists of a DATANET 355 processor with 16,384 or 32,768 words of memory and the following input/output adapters:

- DATANET 355 Intercomputer Adapter (ICA) with up to four ICA ports to interface with the Series 6000 system controllers
- Up to three High-Speed Line Adapter units (HSLA)
- Up to six Low-Speed Line Adapter units (LSLA)
- A console adapter for connection of a teletypewriter console
- An adapter link to Series 6000 mass storage

High-Speed Line Adapter (HSLA)

The HSLA is a multiline communication controller with up to 32 concurrently operating lines. The following four types of channels are available on the HSLA:

- Broadband (19,200 to 50,000 bps)
- General purpose (75 to 9600 bps)
- Dual synchronous (2000 to 9600 bps)
- Dual asynchronous (110 to 1800 bps)

These channels are modular in design and can be configured in any combination not to exceed 16 total per HSLA (32 lines). Because of its flexibility, the HSLA and its channel offering can interface with every type of remote terminal supported on the Series 6000 systems.

Low-Speed Line Adapter (LSLA)

The LSLA provides the primary facility for connecting low-speed terminals to the DATANET 355. Up to 52 terminals at 110 bps, or 26 terminals at 134.5/150 bps, or 17 terminals at 300 bps, or a combination of these can be connected to a single LSLA. The LSLA operates with low-speed terminals in either full- or half-duplex mode for asynchronous data transfer. The LSLA operates on the principle of time-division multiplexing, developing a message frame composed of a number of 8-bit characters, called time-slots, each time-slot containing one complete character associated with a particular terminal.

System Organization

The DATANET 355 functions as a module of the Series 6000 Information System and is completely programmed to provide the front-end data communications processing functions for the Series 6000. Terminal capabilities include Honeywell Series 100 remote batch computers; Honeywell VIP 765/775 video display terminals; GE TermiNet 300; Teletype Models 33, 35, 37; IBM 2741; and other compatible devices.

DATANET 30 PROCESSOR

The DATANET 30 is stored-program data communications processor that provides front-end processing functions for the Series 6000 Information Systems.

Features

- 18-bit word, binary operation
- Magnetic core memory with a 7-microsecond cycle time available in a 16,384-word capacity
- Stored-program operation, with indirect addressing and multiple indexing
- Up to 60 input/output channels, with patch-plug adaptation to character and word length and to transmitting and receiving speeds

System Configurations

Two software packages are available for the DATANET 30 to provide front-end processing functions for Series 6000 systems:

1. Mixed-speed program – services any one of the following at any time:
   - 31 teletypewriters (110 to 150 bps)
   - 10 Series 100 remote batch computers (voice-grade)
   - 4 Series 100 remote batch computers (broadband)
   - 4 765/775 VIP terminals (voice-grade)
   - A combination of the above devices
2. Low-speed program – services up to 60 teletypewriter devices (110 to 150 bps).

DATANET 305 PROCESSOR

The DATANET 305 is a stored-program data communications processor designed for Series 6000 users having a small-to-moderate communications work load. The advanced multidimensional concepts of GECOS are made available to growth-oriented users, providing them with an economical initial step function that permits easy conversion to the DATANET 355 – when high-volume communication requirements exist.

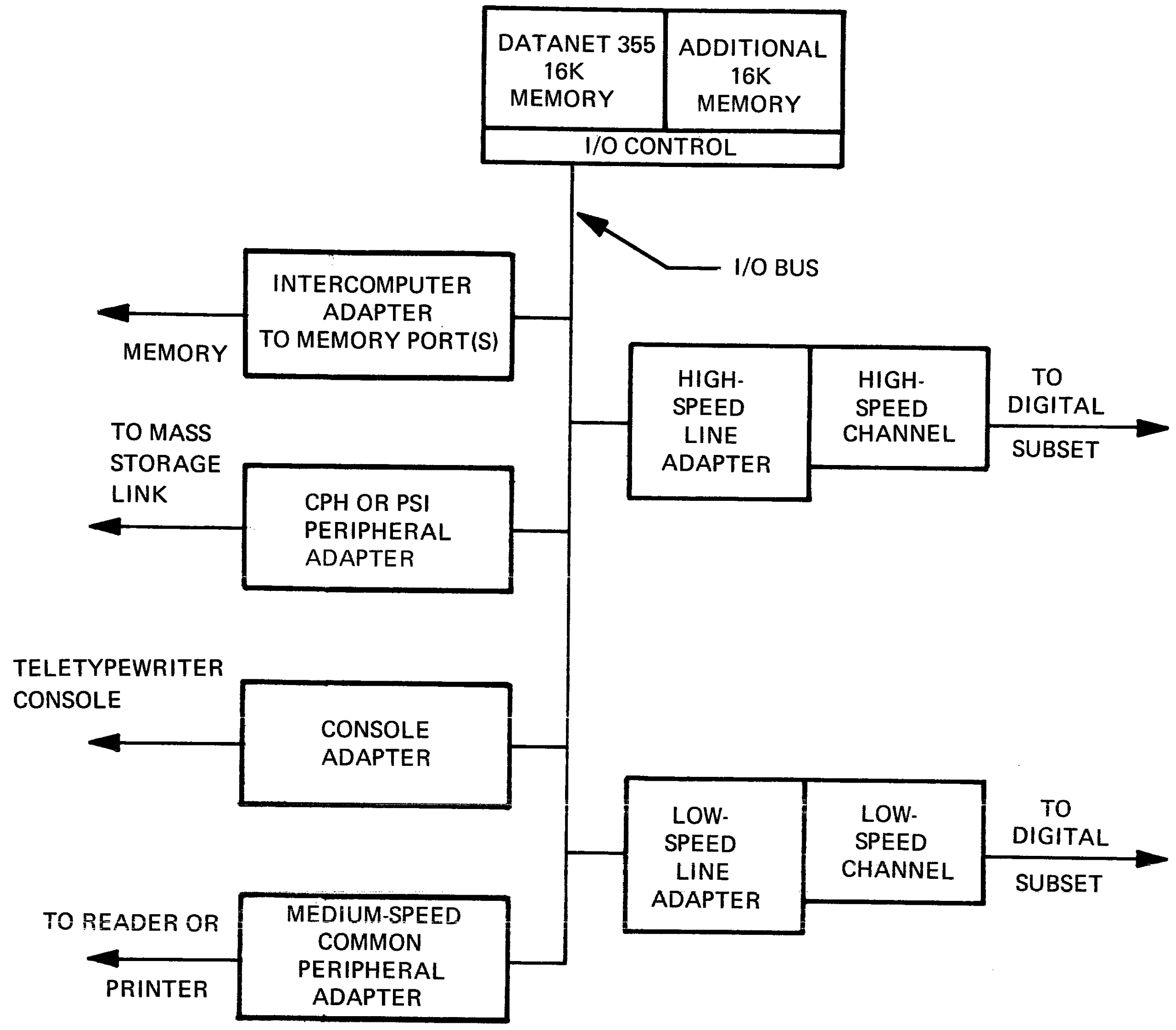
System Configurations

Three fixed system configurations are offered:

1. The DCP301 configuration services up to 12 low-speed asynchronous lines. The following line speeds and terminal types may be selected:

Model 33 and 35 teletypewriters	– 110 bps
Model 37 teletypewriters	– 150 bps
TermiNet 300	– 300 bps

2. The DCP302 configuration, for remote batch users, services two voice-grade, synchronous, half-duplex lines at speeds of 2000, 2400, or 4800 bps. The DCP302 supports the standard remote computer interface.
3. The DCP303 configuration services 12 low-speed, asynchronous lines (maximum of 8 at 300 bps) and two voice-grade circuits, as described under DCP301 and DCP302.



MAXIMUM CONFIGURATION:

16	CONNECTIONS TO THE I/O BUS
32K	MEMORY
3	HIGH-SPEED LINE ADAPTER UNITS
32	HIGH-SPEED LINE INTERFACE UNITS PER ADAPTER
6	LOW-SPEED LINE ADAPTER UNITS
52	LOW-SPEED LINE INTERFACE UNITS PER ADAPTER
1	INTERCOMPUTER ADAPTER

Figure 33. DATANET 355 Communications Subsystem

Section IX
Peripheral Subsystems

All peripheral subsystems communicate with Series 6000 Information Systems through an input/output multiplexer (IOM). There are two standard interfaces available on the IOM for the connection of peripheral controllers – the common peripheral interface (CIP) transfers data in excess of 650 thousand characters per second, while the peripheral subsystem interface (PSI) is capable of transferring data in excess of 1.3 million characters per second. These interfaces support a variety of peripheral device controls, such as magnetic disk and tape controls, card reader, printer and punch controls, and data communications processors.

GECOS issues commands to the peripherals through the IOM. Peripheral devices can also request action by transmitting special interrupts to GECOS.

Representative peripheral subsystems that can be utilized with Series 6000 Information Systems are described in this section.

CONTROL CONSOLE

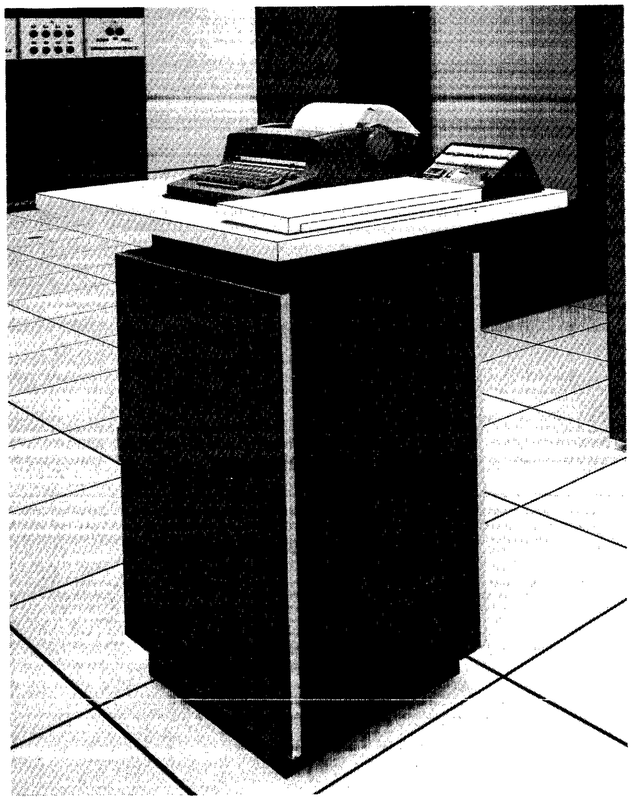


Figure 34. Control Console

The primary function of a control console is to provide for direct communication between the operator and GECOS. The master console is a freestanding unit of suitable operator working height that connects to a common peripheral interface channel of the IOM; the console is controlled just as a peripheral subsystem. Significant processor and system controller functions are displayed on the system status display panel, keeping the operator constantly informed of running status. The status display panel contains the following indicators and controls:

SYSTEM READY light
OPERATOR ATTEN light and switch (attention)
DIS light (processor waiting)
INSTRUCTIONS EXECUTED/SECOND indicator
CONSOLE READY light
MASTER MODE light
SLAVE MODE light
EMERGENCY POWER OFF switch

The console includes an input/output typewriter that accepts input data via keyboard entry and transmits the data to the input/output multiplexer. Output messages from the IOM are printed at 15 characters per second (nominal). An operating routine within GECOS provides responses to operator requests through the typewriter.

DSS180 REMOVABLE DISK STORAGE SUBSYSTEM

The Type DSS180 Removable Disk Storage Subsystem provides intermediate-size mass storage for Series 6000 systems. The DSS180 is a highly flexible subsystem with an expansion capacity of up to 18 drives in increments of single disk drives. Optional dual simultaneous channels and additional nonsimultaneous channels provide the DSS180 subsystem with unlimited combinations of configurations whether required for shared access systems, for redundancy requirements to provide high data availability, or for required throughput in the central system.

Designed with the latest technology, the DSU180 Disk Pack Drive contains a "voice coil" actuator to provide higher reliability on a fast average access time of 34 milliseconds. This fast positioning time, together with an average latency time of 12.5 milliseconds, allows a record to be accessed in an average of 46.5 milliseconds. Total capacity of each drive is 27.5 million characters (formatted) with a total on-line capacity of 500 million characters.

Features of the DSS180 include:

- High data availability with dual crossbarred channels and optional switched channels.
- Large data base capacity of up to 18 disk drives.
- Record "seek" optimization with up to 16 disk drives performing off-line seeks while two additional drives are simultaneously transferring data to the central system.
- Communications processor sharing access to the data base through the channeling flexibility built into the DSS180.
- A transfer rate of 416,000 six-bit characters per second.

Positioning Times of the DSS180 subsystem include:

Minimum:	10 milliseconds
Average:	34 milliseconds
Maximum:	60 milliseconds
Latency:	12.5 milliseconds (average)

Disk Pack – The Honeywell Sentinel DCT170 disk pack is physically identical to the 2316 pack. It consists of eleven 14-inch disks mounted on a common spindle. Twenty surfaces are available for data recording.

Disk and Record Layout – Data is grouped in a total of 72,000 continuously addressable sectors of 384 characters each. A total of 360 sectors (138,240 characters) are accessible in each actuator position or cylinder.

Bits per character	6
Characters per sector	384
Sectors per track	18
Characters per track	6,912
Tracks per cylinder	20
Characters per cylinder	138, 240
Cylinders per disk pack	200 plus 3 spares
Characters per disk pack	27,648,000

Control – The control handles up to 18 on-line disk pack drives. Command decoding, data checking features, and data control are provided in the control hardware.

Disk File Electronics – The disk file electronics is a separate cabinet containing multiaccess device switches, power supply, and data recovery logic.

Dual-Channel Option – Dual channels permit two processing systems (one communications) to share the same data base, thus providing higher throughput and data availability.

Simultaneity – Each channel is capable of simultaneous data transfers to different spindles while overlapping seeks on all other spindles.

Block Count – Hardware block count provides file protection where multiple files share the same cylinder.

Subsystem Configurations – The DSS180 basic configuration consists of control, disk file electronics, and three disk pack drives. (See Figure 35.) The basic configuration can be expanded up to 247 million 6-bit characters by adding up to six additional disk pack drives, a total of nine drives.

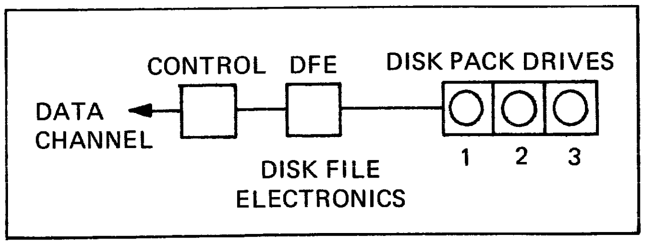


Figure 35. DSS180 Basic Configuration

The DSS180 maximum configuration (Figure 36) consists of control, dual channels (optional), and two disk file electronics units providing capacity capability of up to 497 million characters (18 disk pack drives).

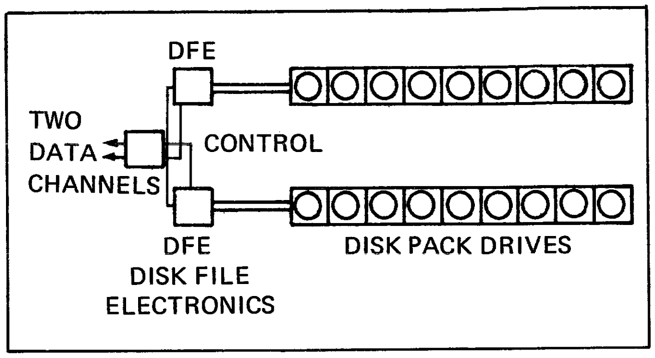


Figure 36. DSS180 Maximum Configuration

DSS190 REMOVABLE DISK STORAGE SUBSYSTEM

The Type DSS190 Removable Disk Storage Subsystem provides the fast access and large capacity storage needed to perform remote access, transaction processing, and time sharing in large Series 6000 data base management systems. This high-performance, large-capacity disk subsystem utilizes advanced concepts that greatly improve response time and systems throughput.

Features – The DSS190 includes an advanced disk control design that provides the subsystem with the following features of increased performance and reliability:

- A command set that is compatible with other Series 6000 direct-access devices.
- Error detection and correction for increased reliability.
- Error recovery – the control performs a series of retry commands without system intervention, which allows recovery from marginal facility errors.
- Error statistics are recorded to provide increased serviceability.
- Rotational position sensing permits optimization of channel time by releasing the channel except during actual record data transfer time.
- Block multiplexer feature of the subsystem and IOM provides the capability to execute up to eight command sequences simultaneously by overlapping channel data transfers.
- Large capacity of 16 drives for a maximum of 2.13 billion characters per control.
- Dual simultaneous channels allow shared access to the data base, increased throughput, and redundancy.
- A transfer rate of 1,074,000 six-bit characters per second.

Dual-Channel Option – Dual channels permit two processing systems (one communications) to share the same data base, thus providing higher throughput and data availability.

Positioning Times of the DSS190 subsystem include:

Minimum:	10 milliseconds
Average:	30 milliseconds
Maximum:	55 milliseconds
Latency:	8.3 milliseconds (average)

Disk Packs are Honeywell M4050 disk packs consisting of twelve 14-inch disks mounted on a common spindle. A total of 19 surfaces are available for data recording.

Disk and Record Layout – The track format is variable length in any multiple of 384-character sectors.

Bits per character	6
Characters per sector	Multiples of 384
Characters per track (maximum)	17,370
Tracks per cylinder	19
Characters per cylinder	330,000
Cylinders per disk pack	404 plus 7 spares
Characters per disk pack	133,320,000

Control – The disk control handles up to 16 on-line disk spindles. Command decoding, data checking features and data control are provided in the control hardware.

Simultaneity – Channels are capable of simultaneous data transfers to different spindles while overlapping seeks on all other spindles.

Block Multiplexer – The Type DSS190 uses multiple logical channels, command stacking, seek overlap, and sector search to provide maximum throughput and response time.

Block Count – Hardware block count ensures protection between files where multiple files share the same cylinder.

File Protection – Write protection at the device level, format protection, and access control of Read, Update, and Write are provided at the control level.

Data Integrity – Data integrity is ensured through extensive error detection, automatic retry, automatic retry with adjustments, and error correction within the subsystem.

Subsystem Configurations – Each removable disk pack stores 133 million 6-bit characters. On-line storage per disk subsystem ranges from a minimum capability of 266 million characters (two operating spindles) up to 2.13 billion characters (16 operating spindles). (See Figures 37 and 38 for the minimum and maximum configurations.)

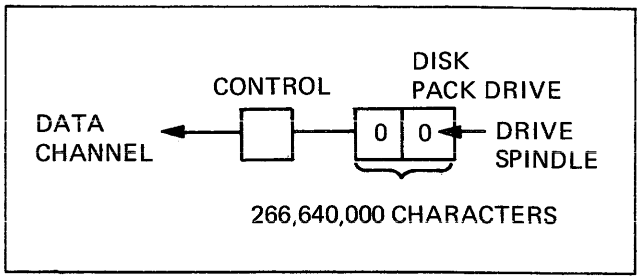


Figure 37. DSS190 Minimum Configuration

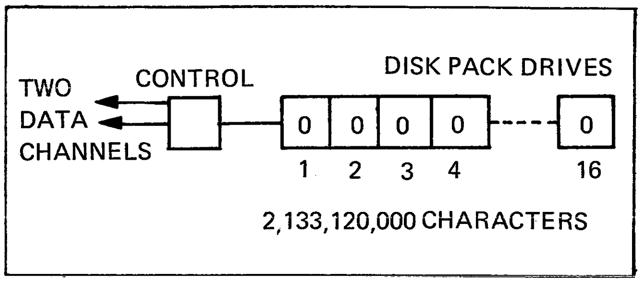


Figure 38. DSS190 Maximum Configuration

DSS270 DISK STORAGE SUBSYSTEM

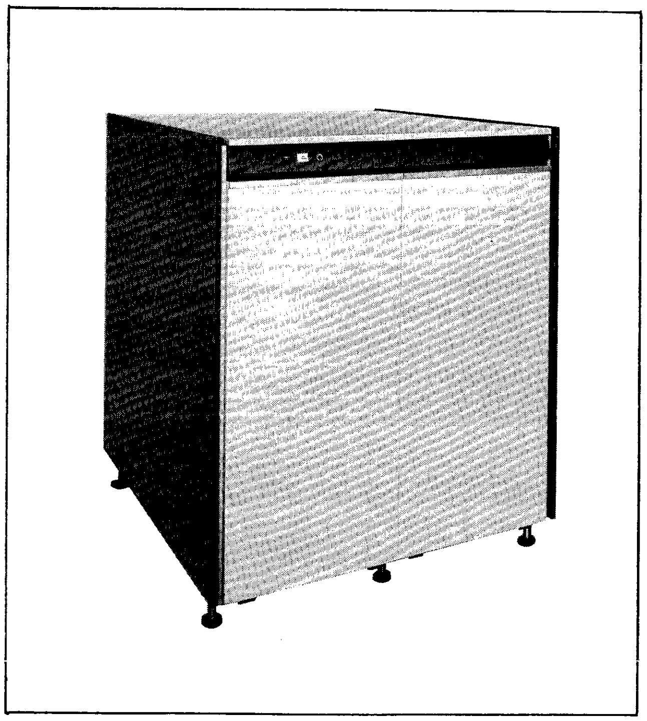


Figure 39. DSS270 Disk Storage Subsystem

The Type DSS270 Disk Storage Subsystem provides rapid, random access retrieval of stored data and serves as an extension of the processor core memory. Communication with the information system is through a common peripheral interface (CPI) channel of the IOM.

Storage Capacity – DSU270 Disk File capacity is 15.3 million 6-bit characters. Up to 20 disk files can be connected to one control to provide over 307 million characters of storage per subsystem.

Transfer Rate – Maximum transfer rate on each channel is 333,000 six-bit characters per second.

Access Time – The read/write mechanism on the DSU270 is of fixed head-per-track design. Average record access time is 26 milliseconds; maximum is 50.3 milliseconds.

Disk and Record Layout – Data is grouped in a total of 40,000 continuously addressable sectors of 384 characters each per disk file. Thus, a maximum subsystem can address up to 800,000 data sectors.

Control – The Type DSS270 control handles up to four Type DFE270 Disk File Electronics Units. Each disk file electronics unit controls up to five disk files, providing a total subsystem capacity of 20 DSU270s. The control communicates with the processor via one CPI channel, or two CPI channels when the additional data channel option is installed.

Dual-Channel Simultaneity – An option is provided in the control for an Additional Data Channel (Type ADC270). With this option, independent and simultaneous data transfer and command communication is allowed between the control and IOM, with the restriction that data transfer simultaneity cannot occur within the same disk file electronics unit.

Dual-Channel Interference Protection – To prevent access interference in dual-channel configurations, a disk file electronics unit is automatically reserved for a particular channel after that channel has established communication with a successful select command.

Sector Count Feature – A hardware sector count feature provides file protection by allowing the software to limit the number of continuous data sectors that can be read or written with a single data transfer command. The sector count limit can vary from 1 to 4,096.

Checking – Nine checks are performed on control information and data in the subsystem. These checks consist of data parity and timing, and control information parity and validity. Additionally, nine hardware conditions are continually monitored.

Error Detection – Incorporated in the hardware is a data error detection feature that enables the subsystem to identify any error condition on reading a record.

Subsystem Configuration – A minimum subsystem configuration is one DSC270 Control, one DFE270 Disk File Electronics Unit, and one DSU270 Disk File. The illustration below represents a maximum subsystem configuration, with over 307 million characters of data and the dual-channel option.

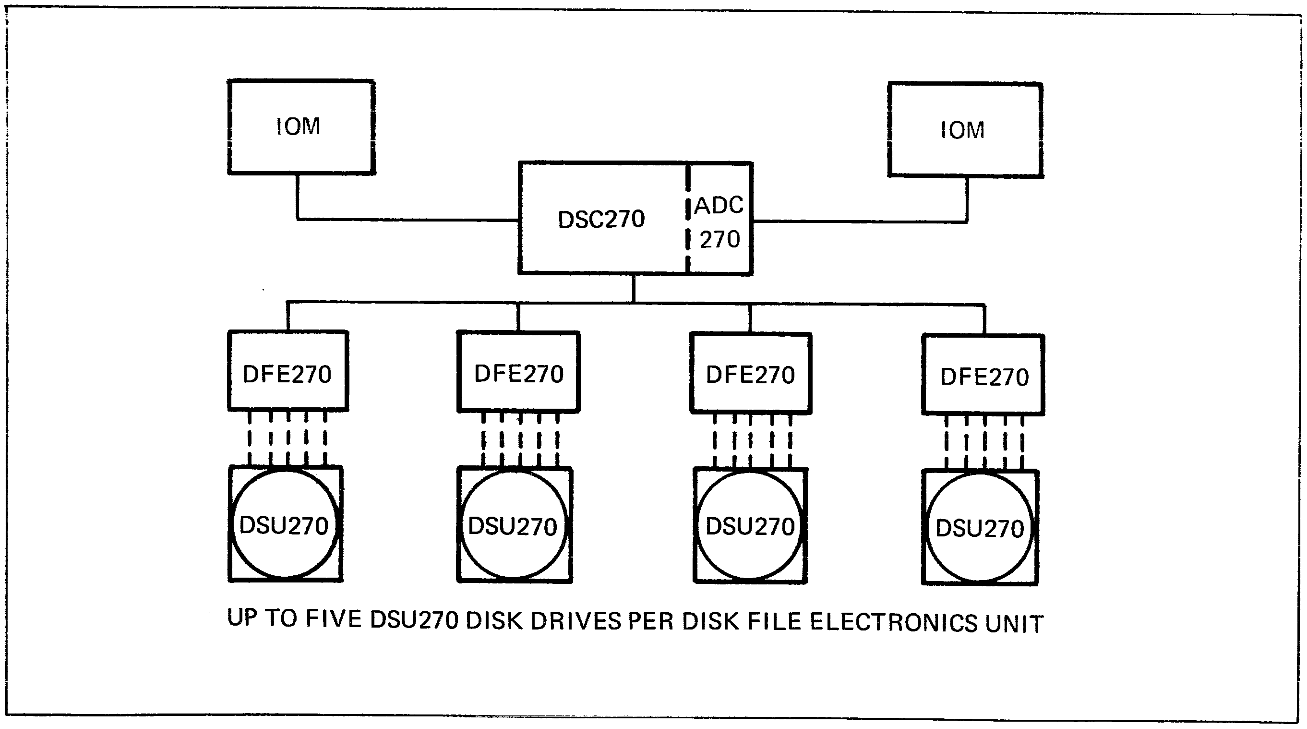


Figure 40. DSS270 Maximum Configuration

DSS167 REMOVABLE DISK STORAGE SUBSYSTEM

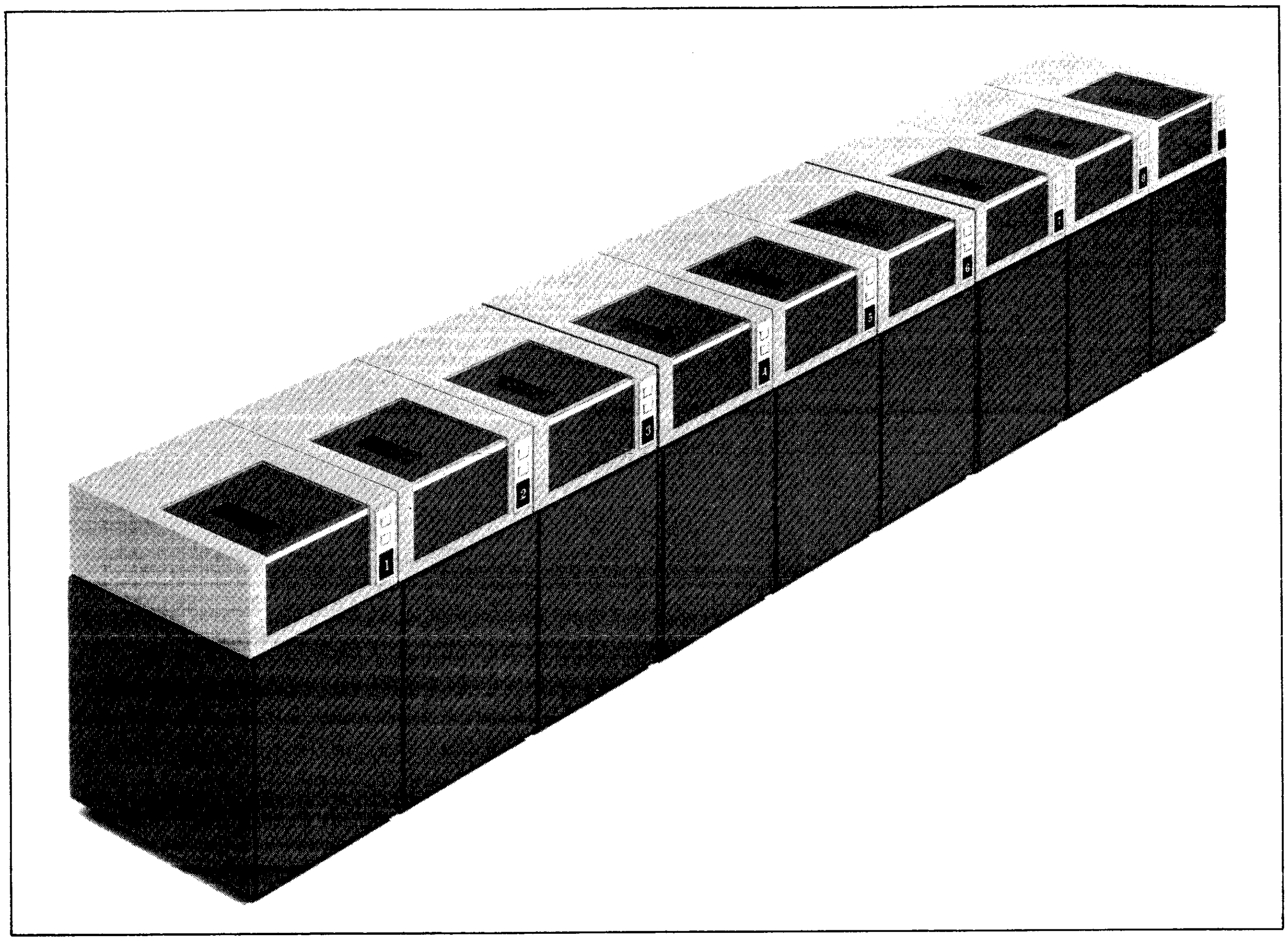


Figure 41. DSS167 Disk Storage Subsystem

The Type DSS167 Removable Disk Storage Subsystem provides a random access storage medium for medium-to-large-size files contained on removable disk packs. Certain disk packs can also be specified as permanent (nonremovable) and serve as conventional systems storage. Communication with the information system is through a common peripheral interface (CPI) channel of the IOM.

Storage Capacity – The DSS167 subsystem is available with six or nine disk pack drives and will control up to eight drives on-line, with disk packs mounted on each of the drives. In a nine-drive subsystem, one of the drives will act as a spare. Each disk pack contains 15 million 6-bit characters for a total on-line storage capacity of 90 million characters for a six-drive subsystem or 120 million characters for a nine-drive subsystem.

Transfer Rate – Each removable disk pack contains 11 disks mounted on a vertical spindle, and rotates at a speed of 2400 revolutions per minute. The nominal transfer rate is 208 KC per second.

Access Mechanism – The access mechanism has 20 horizontal access arms mounted on a single vertical assembly. One read/write head is mounted on each arm and is positioned to read or write on the upper or lower surface of each of the 11 disks exclusive of the upper surface of the top disk and the lower surface of the bottom disk. The entire assembly moves horizontally such that the read/write heads have access to the entire recording area. The access time factors are:

Track-to-track seek time	25.0 milliseconds
Maximum seek time	135.0 milliseconds
Average seek time	75.0 milliseconds
Average latency time	12.5 milliseconds
Average access time	87.5 milliseconds

Disk and Record Layout – Each disk surface contains 203 tracks of information, three of which are used as spares. Each track contains 10 sectors of 384 characters each, or 3840 characters per track. Since all of the 20 read/write heads operate in the same vertical plane, 20 tracks of data are available without movement of the access mechanism.

Data File Protection – A hardware sector count feature provides file protection by allowing the software to limit the number of continuous data sectors that can be read or written with a single data transfer command. The sector count limit can vary from 1 to 200.

Checking – The following checks are performed on the control information and data in the subsystem:

- Write Inhibit
- File Positioning
- Transfer Timing
- Transmission Parity
- Invalid Seek Address
- Header Verification Failure
- Check Character
- Data Compare Alert
- Block Count Limit
- Defective Track Detected
- Invalid Operation Code
- Invalid Device Code

Error Detection – Upon recognition by the software of an unrecoverable writing error, the track in error will be marked as defective and an alternate (or spare) track will be assigned in lieu of the original track. Subsequent accesses to the defective track will cause the software to locate the alternate track before the requested read or write operation is performed.

System Configuration – The DSS167 is available in two configurations, as illustrated below.

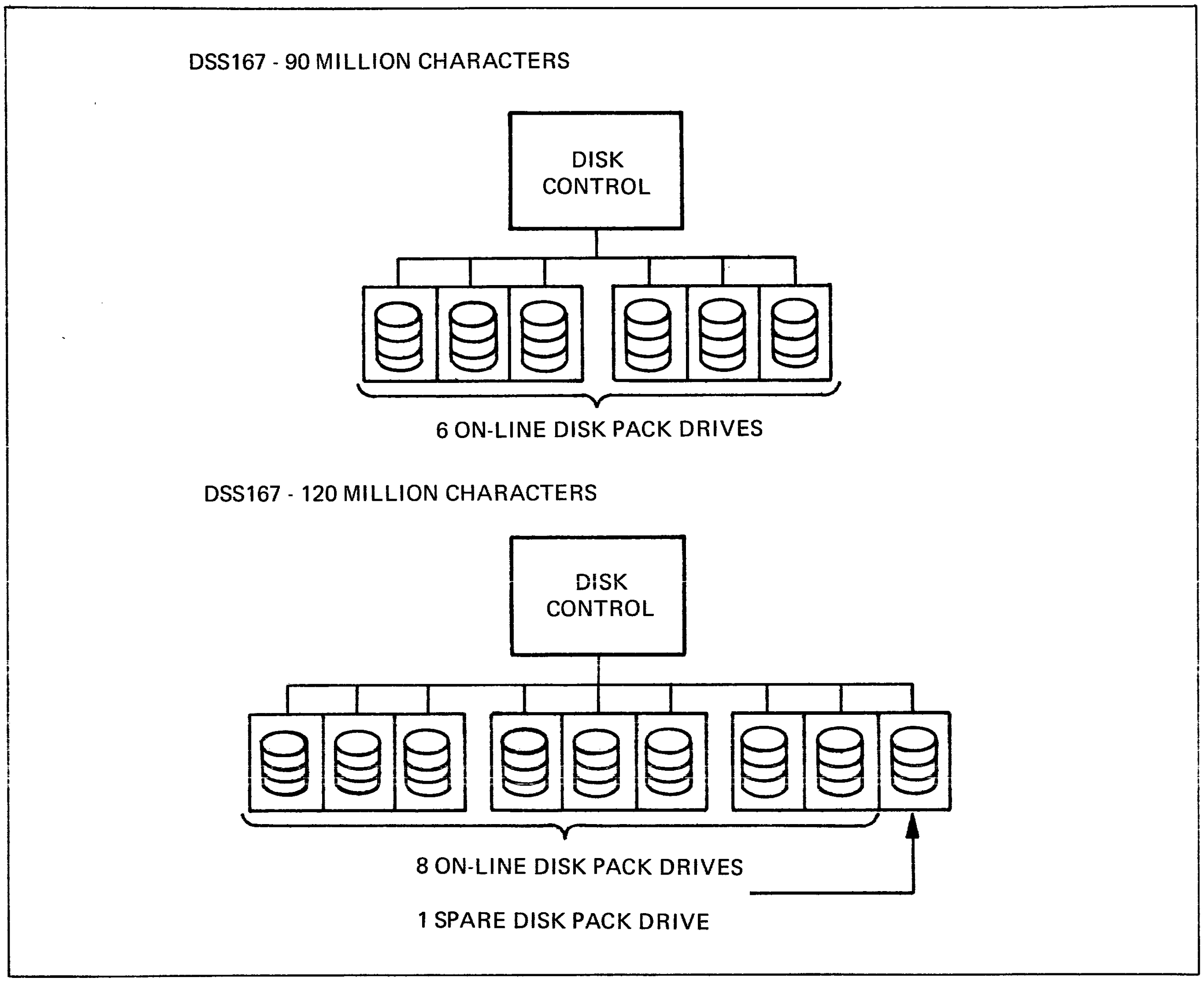

DSS167 - 90 MILLION CHARACTERS
DISK CONTROL
6 ON-LINE DISK PACK DRIVES
DSS167 - 120 MILLION CHARACTERS
DISK CONTROL
8 ON-LINE DISK PACK DRIVES
1 SPARE DISK PACK DRIVE

Figure 42. DSS167 Subsystem Configurations

DSS170 REMOVABLE DISK STORGE SUBSYSTEM
The Type DSS170 Removable Disk Storage Subsystem provides a random access storage medium for large removable files as well as large, permanent, on-line random access storage.

Storage Capacity – The DSS170 subsystem consists of eight disk pack drives plus one spare drive, or nine in total. Each drive controls one disk pack containing 27.5 million 6-bit characters. The disk control can address any eight of the drives at a given time, for a total on-line storage capacity of 220 million characters formatted in 384-character sectors.

Transfer Rate – Each disk pack contains 11 disks mounted on a vertical spindle, and rotates at a speed of 2400 revolutions per minute. The nominal transfer rate is 416 KC per second, through a single channel.

Access Mechanism – The access mechanism has 20 horizontal access arms mounted on a single vertical assembly. One read/write head is mounted on each arm and is positioned to read and write on the upper or lower surface of each of the 11 disks, exclusive of the upper surface of the top disk and the lower surface of the bottom disk. The entire assembly moves horizontally such that the read/write heads have access to the entire recording area. The access time factors are:

Track-to-track seek time	25.0 milliseconds
Maximum seek time	135.0 milliseconds
Average seek	60.0 milliseconds
Average latency time	12.5 milliseconds
Average access time	72.5 milliseconds

Disk and Record Layout – Each disk surface contains 200 tracks of information. Each track consists of 18 sectors of 384 characters each, or 6912 characters per track. Since each disk pack contains 20 recording surfaces and all of the 20 read/write heads operate in the same vertical plane, 20 tracks of data or 138,240 characters are accessible without movement of the access mechanism.

Data File Protection – A hardware sector count feature provides file protection by allowing the software to limit the number of continuous data sectors that can be read or written with a single data transfer command. The sector count limit can vary from 1 to 360.

Error Detection – Each disk pack has a minimum of 1.5 percent spare storage distributed equally across all recording surfaces. Assignment of alternate tracks upon detection of a defective track is the same as described for the DSS167 subsystem.

Checking – The following checks are performed on control information and data in the subsystem:

Write Inhibit	Check Character
File Positioning	Data Compare Alert
Transfer Timing	Block Count Limit
Transmission Parity	Defective Track Detected
Invalid Seek Address	Invalid Operation Code
Header Verification Failure	Invalid Device Code

Subsystem Configuration – The subsystem configuration of a DSS170 includes eight active disk pack drives and one spare. The configuration is illustrated below.

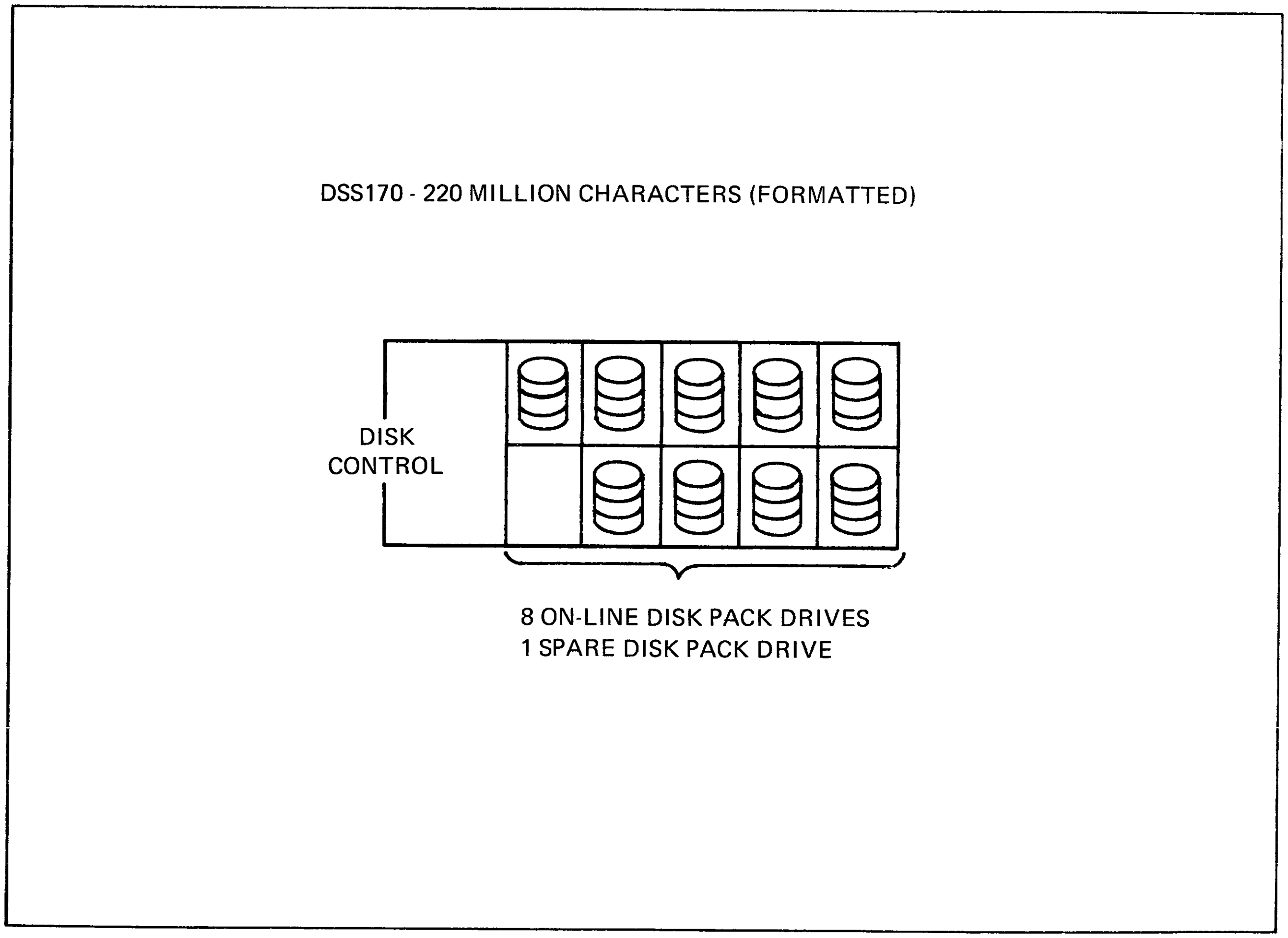

DSS170 - 220 MILLION CHARACTERS (FORMATTED)
DISK CONTROL
8 ON-LINE DISK PACK DRIVES
1 SPARE DISK PACK DRIVE

Figure 43. DSS170 Subsystem Configuration

MAGNETIC TAPE SUBSYSTEMS

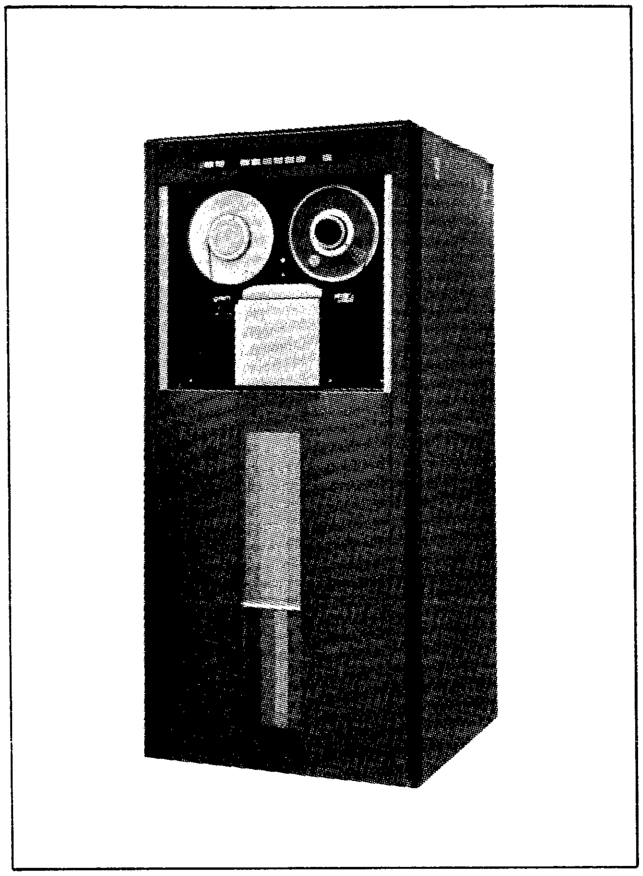


Figure 44. Magnetic Tape Unit

Data Medium is half-inch wide, magnetic-oxide plastic tape, up to 2400 feet long.

Data Formats are binary (standard) and special decimal.

Checking features include:

Transfer Timing	Missing Character
Blank Tape Read	Longitudinal Parity
Transmission Parity	Bit Detected During Erase
Lateral Parity	

Average Rewind Speed is 300 ips (500 ips on MTH502/505).

Features of the magnetic tape subsystems include:

- Either single-channel or dual-channel control of tape units.
- Either 2-density (MTH200, MTH201, MTH372, MTH402, MTH404, MTH492, MTH502, and MTH505) or 3-density (MTH300, MTH301, MTH373, MTH405, and MTH493).
- Program or operator control of recording density.
- Special decimal mode provides compatibility with non-Honeywell tape systems.
- Dual-gap read-write heads for read-after-write checking.
- Tape control buffering during data transfers.
- File protection through use of a write-permit ring.

Modularity and Flexibility – A full range of tape handlers with various speeds, densities, and transfer rates is available with Series 6000 magnetic tape subsystems (see Table 2).

TABLE 2. MAGNETIC TAPE UNIT CHARACTERISTICS

Type Number	Tape Tracks	Tape Speed (ips)	Density (bpi)	Transfer Rates (thousands of 6-bit characters per second)
MTH200	7	37.5	200/556	7.5/21
MTH300	7	37.5	200/556/800	7.5/21/30
MTH201	7	75	200/556	15/42
MTH301	7	75	200/556/800	15/42/60
MTH372	7	150	200/556	30/83
MTH373	7	150	200/556/800	30/83/120
MTH404	9	75	200/556	20/56
MTH405	9	75	200/556/800	20/56/80
MTH492	9	150	200/556	40/111
MTH493	9	150	200/556/800	40/111/160
MTH502	9	75	200/556/800/1600	20/55/80/160
MTH505	9	125	200/556/800/1600	33/92/133/266

All of the magnetic tape controls, except the controls for MTH502 and MTH505 handlers, connect to the IOM via common peripheral interface channels. The controls for MTH502 and MTH505 handlers connect to the IOM via peripheral subsystem interface channels, permitting higher transfer rates.

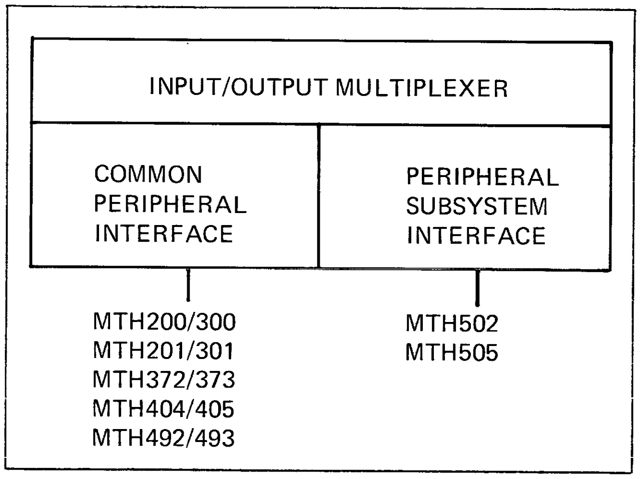


Figure 45. Magnetic Tape Interface

A single-channel magnetic tape subsystem connected to an IOM permits reading or writing of any one of up to eight magnetic tape units connected to that control. Reading or writing proceeds simultaneously with other peripheral operations on other peripheral channels and with processor operations.

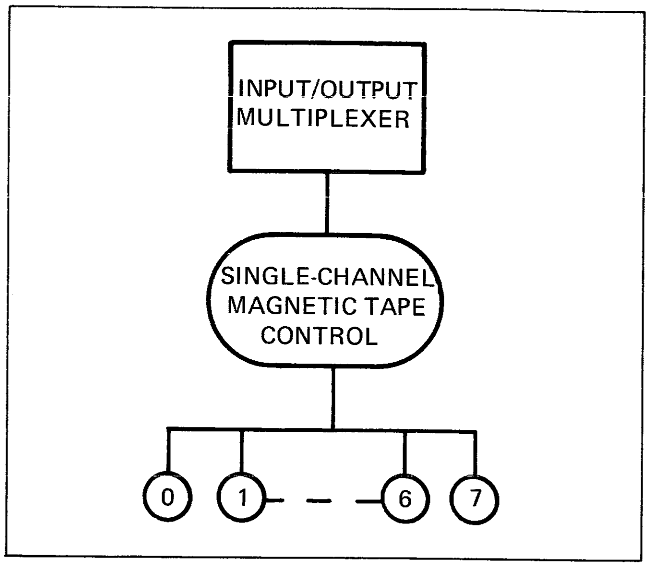


Figure 46. Single-Channel Magnetic Tape Subsystem

Dual-channel control of a magnetic tape subsystem provides for automatic overlapping of read and write operations on any of the associated tape units. As shown below, a subsystem including a dual-channel control and two or more tape units permits accessing any tape unit through either channel and tape control. Thus, if one channel is busy with an assigned tape unit, access can be gained to any other tape unit on the subsystem through the other channel.

As shown by the dotted line, a magnetic tape subsystem featuring dual-channel control can be connected between two IOMs.

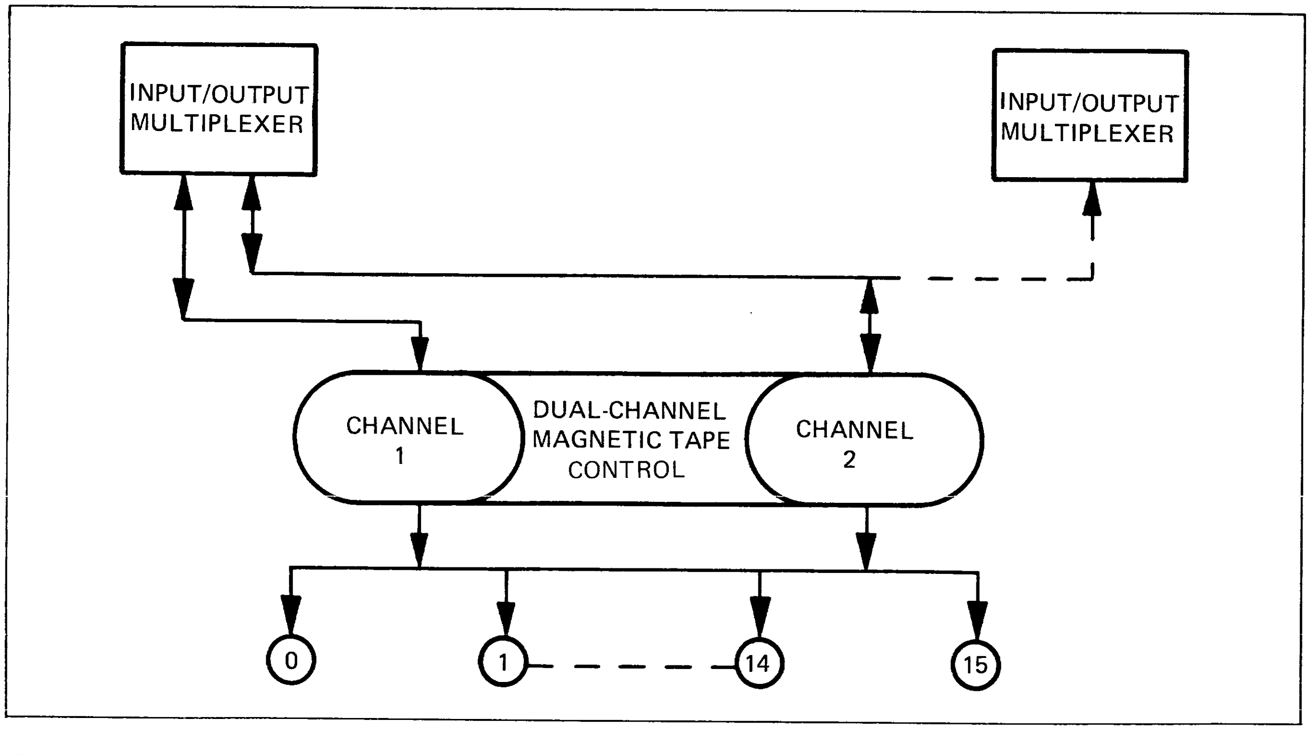


Figure 47. Dual-Channel Magnetic Tape Subsystem

PRT300 PRINTER AND CONTROL

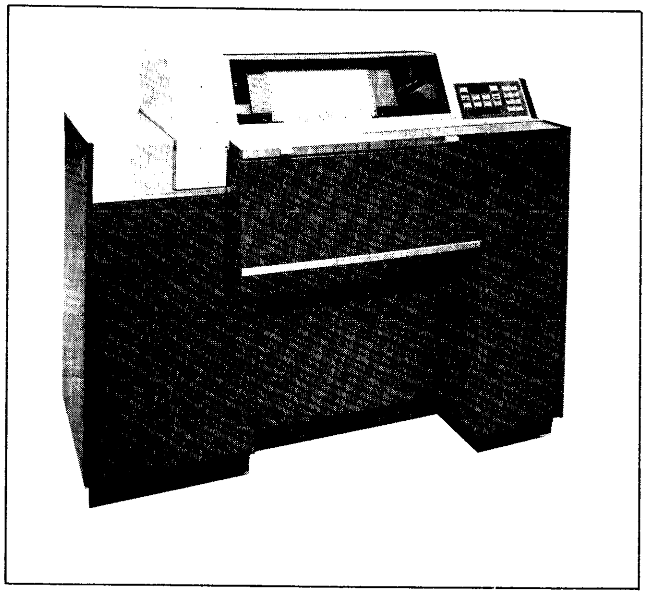


Figure 48. PRT300 Printer

Data Medium consists of continuous forms, 3 to 21 inches wide and up to 22 inches long. The Type PRT300 prints an original and up to five copies.

Speeds include 1150 LPM with a 48-character set, 1050 LPM with a 63-character set, and up to 2500 LPM burst rate (16 character set only).

Data Formats – Train cartridges can be interchanged by the operator. The format includes 10 characters per horizontal inch, 136 characters per line, and a selection of six or eight lines per vertical inch.

Character Sets – Various character sets are available with 16 to 63 different printable characters per train cartridge. The train holds 288 character positions.

Checking – Parity checks are performed on characters in the print buffer, on train image characters in the buffer, and on the address to the printer.

Features of the PRT300 Printer include:

- Edit and nonedit print modes.
- Vertical format control by print command; paper can be skipped to any of 15 coded positions on the paper tape loop.
- Horizontal formatting under program control – can cause 0 to 120 (in multiples of eight) blank positions to occur in print line.
- Connects to a common peripheral interface channel of the input/output multiplexer.

PRT201 PRINTER AND CONTROL

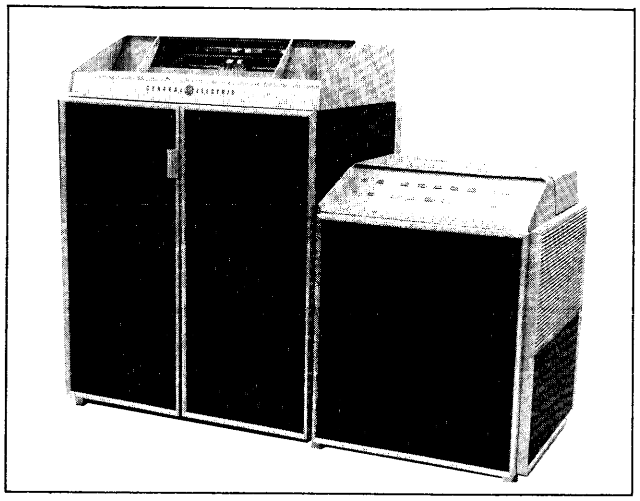


Figure 49. PRT201 Printer

Data Medium consists of continuous forms, 3 to 19 inches wide and up to 22 inches long. The Type PRT201 prints an original and up to four carbon copies.

Speeds include 1200 LPM with 46 most-used (adjacent) characters, or 938 LPM with all 64 standard characters.

Data Formats include 10 characters per horizontal inch, 136 characters per line, and a selection of six or eight lines per vertical inch.

Operational Modes – The edit mode deletes edit symbols from the print line and allows programmed blank insertions and slewing. The nonedit mode prints special characters in the print line data for debug memory dumps.

Checking – Parity checks are performed on data characters and on vertical format unit tape punch configurations.

Features of the PRT201 Printer include:

- Photoelectric sensing of the vertical format unit tape for reliability.
- Vertical format unit prevents runaway slewing.
- Separate vertical format unit mechanism for each mode of vertical line density.
- Programmed control of slewing by vertical format unit tape or countdown, includes top-of-page slew.

CRZ201 CARD READER AND CONTROL

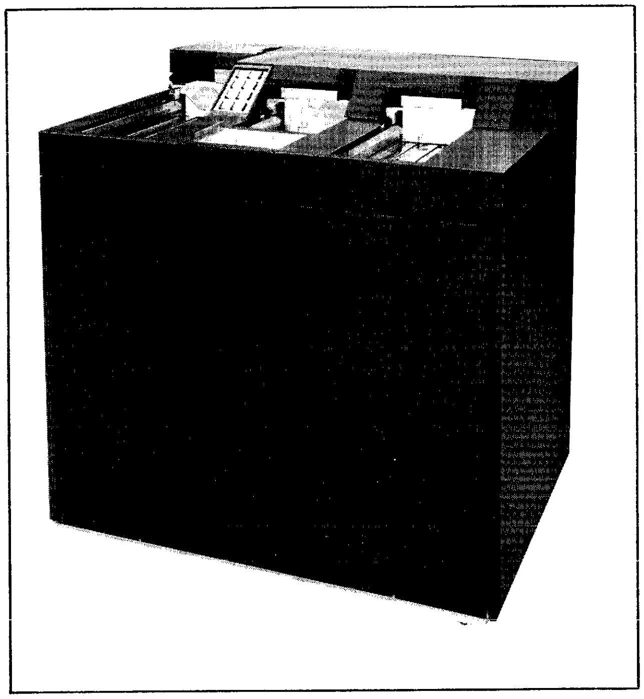


Figure 50. CRZ201 Card Reader

Data Medium consists of standard 80-column cards with round or square corners.

Speeds are 900 cards per minute (80-column) or 1200 cards per minute (51-column) under program-controlled feed.

Data Formats include standard Hollerith card code, binary, and intermixed.

Checking – The CRZ201 performs checks for the following conditions:

Card Feed	Character Validity (decimal mode)
Card Synchronization	Hopper Empty
	Stacker Full
Read Head Alert	
Card Jam	

Features of the Type CRZ201 Card Reader include:

- 2000-card hopper and stacker capacity.
- Continued operation while loading or removing cards.
- Card reading simultaneous with other peripheral and processor operations.
- Reads intermixed Hollerith and binary cards (special-character controlled).

CPZ201 CARD PUNCH AND CONTROL

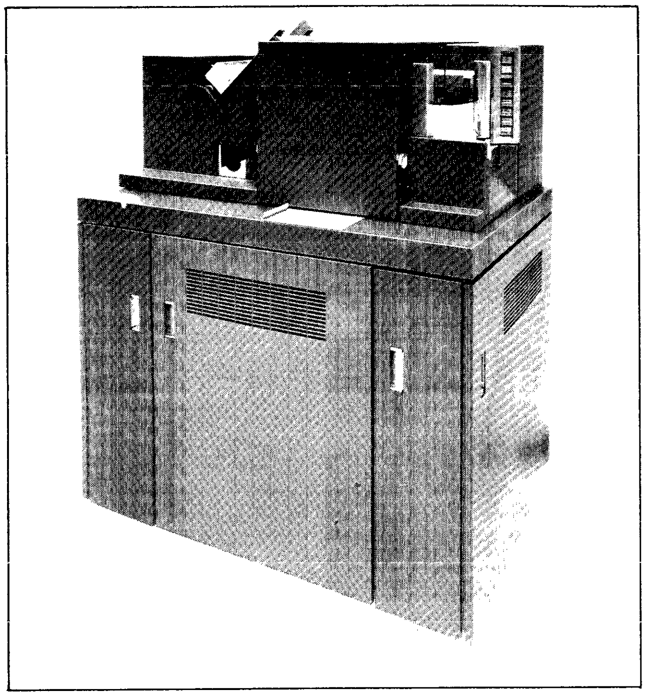


Figure 51. CPZ201 Card Punch

Data Medium consists of standard 80-column cards with round or square corners.

Speed of the Type CPZ201 is 300 cards per minute.

Data Formats include standard Hollerith card code, edited Hollerith, and 12-row binary.

Checking – The CPZ201 performs checks for the following conditions:

Card Feed	Stacker Full
Card Synchronization	Chad Box Absent
Parity	Chad Box Full
Card Jam	Read-After-Punch
Hopper Empty	

Features of the CPZ201 Card Punch include:

- 1200-card stacker and 3000-card hopper capacity.
- Continued operation while loading or removing cards.
- Card punching simultaneous with other peripheral and processor operations.

Appendix A
Instruction Execution Times

CALCULATION OF INSTRUCTION EXECUTION TIMES

The instruction execution times are based on fetching of instructions in pairs from storage and, in the case of overlap-type instructions[1], on overlap between the operation execution of the overlap-type instruction and the fetching and address modification of the next instruction. (See "Execution Timing Variation," page 68.)

Certain conditions delay the fetching of instruction pairs, and certain operations disrupt instruction overlapping. Under these conditions the following time adjustments should be made (all times are in microseconds).

	MODEL		
CONDITION	6030/6040	6050/6060	6070/6080
1. If an instruction alters a register, and the next instruction begins its address modification procedure with an R or RI type of modification that uses this same register . . .	+0	+0.8	+1.0
2. If an instruction from an even storage location alters the next instruction . . .	+1.8	+1.8	+1.2
3. If a transfer of control instruction is located at an odd storage location . . .	+1.0	+1.0	+0.8
4. If a transfer of control transfers to an instruction located at an odd storage location . . .	+1.1	+0.9	+0.6
5. If a store-type[2] instruction located in an odd storage location is preceded by a nonstore-type instruction . . .			
Single-Precision Store	+0.2	+0.6	+0.3
Read-Alter-Rewrite	+0	+0	+0
Double-Precision Store	+0.2	+0.6	+0.3
6. If an overlap-type instruction is followed either by the store-type instruction from an odd storage location or by a transfer-of-control instruction (depending on the particular instruction sequence) . . .	Indefinite amount of time		
7. The instruction execution times of shift and floating-point operations are listed as "average" times based on a number of shift steps. A single shift step may effect a shift by one, four, or sixteen positions. The times shown are the arithmetic average of shift sizes from 1 to 36 bits.			
Actual times for these instructions may vary by up to . . .	±2.0	±0.3	±0.3
8. Address modifications do not require any time adjustments except in the following cases:			
RI-type, for the indirect cycle	+2.0	+2.0	+1.3
IR-type, for the indirect cycle	+2.0	+2.0	+1.3
IT-type, for the indirect cycle with restoring of the indirect word...			
IT Even	+4.0	+4.0	+2.6
IT Odd	+4.0	+4.0	+2.6
IT-type, for the indirect cycle with nonrestoring of the indirect word...			
(CI and I)	+2.0	+2.0	+1.3
Index designator DU or DL except when used with a first modification of the R or RI type and the preceding instruction being an overlap-type instruction...	-1.3	-0.7	+0

[1]Overlap-type instructions include multiplications, divisions, shifts, floating-point operation, loads, and stores (except in Models 6030 and 6040, which have no overlap).

[2]Store-type instructions = store, floating store, add and subtract stored, AND, OR, and EXCLUSIVE OR to storage, etc.

TABLE 3. INSTRUCTION EXECUTION TIMES[1]

INSTRUCTION		MODEL 6030/6040	MODEL 6050[2]/6060	MODEL 6070/6080
Data Movement				
Load				
LDA	Load A	3.0	1.8	0.7
LDQ	Load Q	3.0	1.8	0.7
LDAQ	Load AQ	3.0	1.8	0.7
LDXn	Load Xn from Upper	3.0	1.8	0.7
LXLn	Load Xn from Lower	3.0	1.8	0.7
LREG	Load Registers	10.4	5.4	2.3
LCA	Load Complement A	3.0	1 8	0.7
LCQ	Load Complement Q	3.0	1.8	0.7
LCAQ	Load Complement AQ	3.0	1.8	0.7
LCXn	Load Complement Xn	3.0	1.8	0.7
EAA	Effective Address to A	1.7	1.1	0.7
EAQ	Effective Address to Q	1.7	1.1	0.7
EAXn	Effective Address to Xn	1.7	1.1	0.7
LDI	Load Indicator Register	2.0	2.6	1.5
LDAC	Load A and Clear	3.0	2.2	1.4
LDQC	Load Q and Clear	3.0	2.2	1.4
Store				
STA	Store A	2.7	1.9	1.0
STQ	Store Q	2.7	1.9	1.0
STAQ	Store AQ	2.7	1.9	1.0
STXn	Store Xn into Upper	2.7	1.9	1.0
SCLn	Store Xn into Lower	2.7	1.9	1.0
SREG	Store Registers	7.5	5.6	2.5
STCA	Store Character of A (6-Bit)	2.7	1.9	1.0
STCQ	Store Character of Q (6-Bit)	2.7	1.9	1.0
STBA	Store Character of A (9-Bit)	2.7	1.9	1.0
STBQ	Store Character of Q (9-Bit)	2.7	1.9	1.0
STI	Store Indicator Register	2.8	2.7	1.7
STT	Store Timer Register	2.6	2.7	1.7
SBAR	Store Base Address Register	2.8	2.7	1.7
STZ	Store Zero	2.7	1.9	1.0
STC1	Store Instruction Counter plus 1	2.8	2.7	1.7
STC2	Store Instructor Counter plus 2	2.8	2.7	1.7
Shift (see condition 7, page 63)				
ARS	A Right Shift	3.8	1.3	1.1
QRS	Q Right Shift	3.8	1.3	1.1
LRS	Long Right Shift	3.8	1.3	1.1
ALS	A Left Shift	3.8	1.3	1.1
QLS	Q Left Shift	3.8	1.3	1.1
LLS	Long Left Shift	3.8	1.3	1.1
ARL	A Right Logic	3.8	1.3	1.1
QRL	Q Right Logic	3.8	1.3	1.1
LRL	Long Right Logic	3.8	1.3	1.1
ALR	A Left Rotate	3.8	1.3	1.1
QLR	Q Left Rotate	3.8	1.3	1.1
LLR	Long Left Rotate	3.8	1.3	1.1

TABLE 3. INSTRUCTION EXECUTION TIMES (Cont.)

INSTRUCTION		MODEL 6030/6040	6050/6060	6070/6080
Fixed-Point Arithmetic				
Addition				
ADA	Add to A	3.0	1.8	0.7
ADQ	Add to Q	3.0	1.8	0.7
ADAQ	Add to AQ	3.0	1.8	0.7
ADXn	Add to Xn	3.0	1.8	0.7
ASA	Add Stored to A	4.3	3.6	1.8
ASQ	Add Stored to Q	4.3	3.6	1.8
ASXn	Add Stored to Xn	4.3	3.6	1.8
ADLA	Add Logic to A	3.0	1.8	0.7
ADLQ	Add Logic to Q	3.0	1.8	0.7
ADLAQ	Add Logic to AQ	3.0	1.8	0.7
ADLXn	Add Logic to Xn	3.0	1.8	0.7
AWCA	Add with Carry to A	3.0	1.8	0.7
AWCQ	Add with Carry to Q	3.0	1.8	0.7
ADL	Add Low to AQ	3.0	1.8	0.7
AOS	Add One to Storage	4.3	3.6	1.8
Subtraction				
SBA	Subtract from A	3.0	1.8	0.7
SBQ	Subtract from Q	3.0	1.8	0.7
SBAQ	Subtract from AQ	3.0	1.8	0.7
SBXn	Subtract from Xn	3.0	1.8	0.7
SSA	Subtract Stored from A	4.3	3.6	1.8
SSQ	Subtract Stored from Q	4.3	3.6	1.8
SSXn	Subtract Stored from Xn	4.3	3.6	1.8
SBLA	Subtract Logic from A	3.0	1.8	0.7
SBLQ	Subtract Logic from Q	3.0	1.8	0.7
SBLAQ	Subtract Logic from AQ	3.0	1.8	0.7
SBLXn	Subtract Logic from Xn	3.0	1.8	0.7
SWCA	Subtract with Carry from A	3.0	1.8	0.7
SWCQ	Subtract with Carry from Q	3.0	1.8	0.7
Multiplication				
MPY	Multiply Integer	13.5	3.6	3.6
MPF	Multiply Fraction	13.5	3.6	3.6
Division				
DIV	Divide Integer	25.7	7.3	7.3
DVF	Divide Fraction	25.7	7.3	7.3
Negate				
NEG	Negate A	1.7	1.1	0.7
NEGL	Negate Long	1.7	1.1	0.7
Boolean Operations				
AND				
ANA	AND to A	3.0	1.8	0.7
ANQ	AND to Q	3.0	1.8	0.7
ANAQ	AND to AQ	3.0	1.8	0.7
ANXn	AND to Storage Xn	3.0	1.8	0.7
ANSA	AND to Storage A	4.3	3.6	1.8
ANSQ	AND to Storage Q	4.3	3.6	1.8
ANSXn	AND to Storage Xn	4.3	3.6	1.8

TABLE 3. INSTRUCTION EXECUTION TIMES (Cont.)

INSTRUCTION		MODEL 6030/6040	6050/6060	6070/6080
OR				
ORA	OR to A	3.0	1.8	0.7
ORQ	OR to Q	3.0	1.8	0.7
ORAQ	OR to AQ	3.0	1.8	0.7
ORXn	OR to Xn	3.0	1.8	0.7
ORSA	OR to Storage A	4.3	3.6	1.8
ORSQ	OR to Storage Q	4.3	3.6	1.8
ORSXn	OR to Storage Xn	4.3	3.6	1.8
Exclusive OR				
ERA	Exclusive OR to A	3.0	1.8	0.7
ERQ	Exclusive OR to Q	3.0	1.8	0.7
ERAQ	Exclusive OR to AQ	3.0	1.8	0.7
ERXn	Exclusive OR to Xn	3.0	1.8	0.7
ERSA	Exclusive OR to Storage A	4.3	3.6	1.8
ERSQ	Exclusive OR to Storage Q	4.3	3.6	1.8
ERSXn	Exclusive OR to Storage Xn	4.3	3.6	1.8
Comparison				
Compare				
CMPA	Compare with A	3.0	1.8	0.7
CMPQ	Compare with Q	3.0	1.8	0.7
CMPAQ	Compare with AQ	3.0	1.8	0.7
CMPXn	Compare with Xn	3.0	1.8	0.7
CWL	Compare with Limits	3.0	1.8	0.7
CMG	Compare Magnitude	3.0	1.8	0.7
SZN	Set Zero and Negative Indicators from Storage	3.0	1.8	0.7
CMK	Compare Masked	3.3	1.8	0.8
SZNC	Set Zero and Negative Indicators from Storage and Clear	3.0	2.2	1.4
Comparative AND				
CANA	Comparative AND with A	3.0	1.8	0.7
CANQ	Comparative AND with Q	3.0	1.8	0.7
CANAQ	Comparative AND with AQ	3.0	1.8	0.7
CANXn	Comparative AND with Xn	3.0	1.8	0.7
Comparative NOT				
CNAA	Comparative NOT with A	3.0	1.8	0.7
CNAQ	Comparative NOT with Q	3.0	1.8	0.7
CNAAQ	Comparative NOT with AQ	3.0	1.8	0.7
CNAXn	Comparative NOT with Xn	3.0	1.8	0.7
Floating-Point Arithmetic				
Load				
FLD	Floating Load	3.0	1.8	0.7
DFLD	Double-Precision Floating Load	3.0	1.8	0.7
LDE	Load Exponent Register	3.0	1.8	0.7
Store				
FST	Floating Store	2.7	1.9	1.0
DFST	Double-Precision Floating Store	2.7	1.9	1.0
STE	Store Exponent Register	2.7	1.9	1.0
FSTR[3]	Floating Store Rounded	3.6	1.9	1.3
DFSTR	Double-Precision Floating Round Stored	3.6	1.9	1.3

TABLE 3. INSTRUCTION EXECUTION TIMES (Cont.)

INSTRUCTION		MODEL 6030/6040	MODEL 6050/6060	MODEL 6070/6080
Addition				
FAD	Floating Add	6.8	2.2	1.7
UFA	Unnormalized Floating Add	6.8	2.2	1.7
DFAD	Double-Precision Floating Add	6.8	2.2	1.7
DUFA	Double-Precision Unnormalized Floating Add	6.8	2.2	1.7
ADE	Add to Exponent Register	3.0	1.8	0.7
Subtraction				
FSB	Floating Subtract	6.7	2.2	1.7
UFS	Unnormalized Floating Subtract	6.7	2.2	1.7
DFSB	Double-Precision Floating Subtract	6.7	2.2	1.7
DUFS	Double-Precision Unnormalized Floating Subtract	6 7	2.2	1.7
Multiplication				
FMP	Floating Multiply	11.8	3.2	3.1
UFM	Unnormalized Floating Multiply	11.2	3.2	2.9
DFMP	Double-Precision Floating Multiply	22.2	6.2	6.2
DUFM	Double-Precision Unnormalized Floating Multiply	21.6	6.0	6.0
Division				
FDV	Floating Divide	26.3	7.5	7.5
FDI	Floating Divide Inverted	25.8	7.3	7.3
DFDV	Double-Precision Floating Divide	42.5	12.3	12.2
DFDI	Double-Precision Floating Divide Inverted	41.9	12.1	12.1
Negate, Normalize, Round				
FNEG	Floating Negate	2.4	1.1	0.8
FNO	Floating Normalize	2.4	1.1	0.8
FRD	Floating Round	2.4	1.1	0.8
DFRD	Double-Precision Floating Round	2.4	1.1	0.8
Compare				
FCMP	Floating Compare	4.9	1.8	1.1
FCMG	Floating Compare Magnitude	4.9	1.8	1.1
DFCMP	Double-Precision Floating Compare	4.3	1.8	1.0
DFCMG	Double-Precision Floating Compare Magnitude	4.3	1.8	1.0
FSZN	Floating Set Zero and Negative Indicators from Storage	3.0	1.8	0.7
	Transfer of Control			
Unconditional Transfer				
TRA	Transfer Unconditionally	1.8	1.8	1.2
TSXn	Transfer and Set Xn	1.8	1.8	1.2
TSS	Transfer After Setting Slave	1.8	1.8	1.2
RET	Return	3.8	3.8	2.5
Conditional Transfer (Transfer Taken)				
TZE	Transfer on Zero	1.8	1.8	1.2
TNZ	Transfer on Not Zero	1.8	1.8	1.2
TMI	Transfer on Minus	1.8	1.8	1.2
TPL	Transfer on Plus	1.8	1.8	1.2
TRC	Transfer on Carry	1.8	1.8	1.2
TNC	Transfer on No Carry	1.8	1.8	1.2
TOV	Transfer on Overflow	1.8	1.8	1.2
TEO	Transfer on Exponent Overflow	1.8	1.8	1.2
TEU	Transfer on Exponent Underflow	1.8	1.8	1.2
TTF	Transfer on Tally-Runout Indicator Off	1.8	1.8	1.2
Condition Not Met (Transfer Not Taken)		1.2	1.2	0.8

TABLE 3. INSTRUCTION EXECUTION TIMES (Cont.)

INSTRUCTION		MODEL 6030/6040	6050/6060	6070/6080
	Miscellaneous Operations			
NOP	No Operation	1.2	1.0	0.7
BCD	Binary to Binary-Coded Decimal	7.2	2.5	1.8
GTB	Gray to Binary	22.6	6.6	6.6
XEC	Execute	2.4	2.4	1.8
XED	Execute Double	2.6	2.6	1.9
MME	Master Mode Entry	2.6	2.6	1.9
DRL	Derail	2.6	2.6	1.9
RPT	Repeat	1.2	1.0	0.7
RPD	Repeat Double	1.2	1.0	0.7
RPL	Repeat Link	1.2	1.0	0.7
	Master Mode Operations			
Master Mode				
DIS[4]	Delay Until Interrupt Signal	Note 5	Note 5	Note 5
LBAR[4]	Load Base Address Register	2.8	2.6	1.5
LDT[4]	Load Timer Register	2.8	2.8	1.5
SMIC[4]	Set Memory Controller Interrupt Cells	3.2	1.9	1.7
LCPR	Load Central Processor Registers	2.8	2.6	1.5
SCPR	Store Central Processor Registers	2.8	2.7	1.7
RSCR	Read System Controller Registers	2.8	1.8	1.7
SSCR	Set System Controller Registers	2.9	1.9	1.0
Master Mode and Control Processor				
RMCM[4]	Read Memory Controller Mask Registers	2.9	1.8	0.7
SMCM[4]	Set Memory Controller Mask Registers	2.7	1.9	1.0
CIOC[4]	Connect I/O Channel	2.7	1.9	1.0

EXECUTION TIMING VARIATION

The data shown in Table 3 is based upon actual Series 6000 performance. A long sequence of instructions with the same operation code was executed and then averaged to get the execution time for each instruction shown in the table. The times shown, therefore, are neither the fastest nor the slowest speeds possible; they are a mid-range value resulting from the tests that were made.

In large, modular systems, such as the Series 6000, instruction execution times are affected by many factors, such as:

Configuration (type and number of modules)
Cable lengths
Physical layout
Variations in operand bit configurations
Level of field change orders

The systems used to develop the data shown used 10 foot, internal cables between processors and system controllers. The "H" and "HH" configurations, recommended by Honeywell for 1- and 2-processor systems, also use 10 foot, internal cables in this same manner.

[1]All times are in microseconds.

[2]The timings for Model 6050 are for the noninterleaved configuration. An improvement of up to 15 percent can be achieved if interleaving is employed.

[3]When normalization does not take place, subtract 0.3 microsecond from the listed time.

[4]Causes command fault if executed in slave mode.

[5]Interrupted by timer runout fault unless inhibit bit set.

Appendix B

Preliminary Timing Formulas for EIS Processors

CALCULATION OF EIS INSTRUCTION TIMES

The time necessary to execute an instruction depends upon many factors. In higher-performance machines such as Models 6070 and 6080 the timings may show considerable fluctuation depending upon whether certain instructions are located in even or odd locations, whether indirect addressing is utilized, cable length, component tolerances, memory interference, and, in the case of the EIS instructions, depending on the operand length and its position relative to word boundaries. Thus, timing formulas are necessarily approximate.

The following preliminary timing formulas are simplified to allow processor performance to be estimated with a minimum effort. Timings resulting from these formulas give approximate best-case predictions.

Timings for the 6080 assume interleaved memories. Timings for the 6060 are developed with and without interleaving. Timings for the 6040 assume neither interleaving nor instruction overlap.

Corrections for indirect addressing, sensitivity to even or odd memory locations, or previous operations that affect needed registers must be added to the computed values, just as for a basic (non-EIS) operation.

A word of caution in relation to manual "compilation" of COBOL programs using EIS: the COBOL compiler using EIS for execution retains most of the optimizing capabilities implemented in the current compiler. This means that data fields are word-aligned wherever possible. In such cases, non-EIS sequences are sometimes employed to improve efficiency in places where an EIS instruction could also be used. For example, a long, word-aligned BCD data movement would probably be implemented using "REPEAT DOUBLE, LOAD, STORE" sequence (RPD, LDAQ, STAQ) instead of the obvious MLR instruction. However, the use of EIS instructions in every case will only result in a more conservative estimate of processor performance.

NOTES ON FORMAT

Most character handling instructions include an explanatory relationship of the following form:

$$O_1 + O_2 \rightarrow O_3$$

This information, when present, is located to the right of the instruction definition statement. The example above should be interpreted as follows: Operand 1 is added to Operand 2 to yield Operand 3. Subscripts on various coefficients are related to these numbers. Thus, $NDWB_2$ is the number of double-word boundaries crossed by Operand 2.

DEFINITIONS

N_i	=	Number of characters in Operand O_i, where i = 1, 2, or 3.
$NDWB_i$	=	Number of double-word boundaries crossed by O_i, where i = 1, 2, or 3.
N	=	Maximum of $[N_1, N_2 \ldots\ldots]$
N_Q	=	Number of quotient digits generated.
$NPDW_i$	=	Number of Packed Double Words (16 digits/PDW) in Operand O_i (The nominal minimum value of NPDW = 1).
NPDW	=	Maximum of $[NPDW_1, NPDW_2]$
NMOP	=	Number of micro operations executed.
SF_1	=	1 if operand alignment is required; = 0 otherwise
SF_2	=	1 if result scaling is required; = 0 otherwise.
R	=	1 if rounding is specified; = 0 otherwise.
Q	=	Internal number of 4 double-word (32 ASCII characters) blocks in N.
H	=	Internal number of 4 double-word (48 BCD characters) blocks in N.

TABLE 4. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6080 PROCESSOR (INTERLEAVED MEMORIES)

ALPHANUMERIC INSTRUCTIONS

MLR (Move Alphanumeric Left to Right)

$$T = \begin{Bmatrix} 3.15 \\ 4.10 \end{Bmatrix} + \begin{Bmatrix} .21 \\ .09 \end{Bmatrix} N_2 + \begin{Bmatrix} 0 \\ .5 \end{Bmatrix} Q, \begin{Bmatrix} N \leq 9 \\ N > 9 \end{Bmatrix} \text{ (ASCII)}$$

$$T = \begin{Bmatrix} 3.25 \\ 4.20 \end{Bmatrix} + \begin{Bmatrix} .4 \\ .2 \end{Bmatrix} N_2 + \begin{Bmatrix} 0 \\ .5 \end{Bmatrix} H, \begin{Bmatrix} N \leq 9 \\ N > 9 \end{Bmatrix} \text{ (BCD)}$$

MRL (Move Alphanumeric Right to Left)

$$T = \begin{Bmatrix} 3.40 \\ 4.50 \end{Bmatrix} + \begin{Bmatrix} .22 \\ .09 \end{Bmatrix} N_2 + \begin{Bmatrix} 0 \\ .5 \end{Bmatrix} Q, \begin{Bmatrix} N \leq 9 \\ N > 9 \end{Bmatrix} \text{ (ASCII)}$$

$$T = \begin{Bmatrix} 3.70 \\ 4.80 \end{Bmatrix} + \begin{Bmatrix} .4 \\ .2 \end{Bmatrix} N_2 + \begin{Bmatrix} 0 \\ .5 \end{Bmatrix} H, \begin{Bmatrix} N \leq 9 \\ N > 9 \end{Bmatrix} \text{ (BCD)}$$

MVT (Move Alphanumeric with Transliteration)

$$T = 3.84 + .700\,N_1$$

TABLE 4. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6080 PROCESSOR (INTERLEAVED MEMORIES) (Cont.)

MVE (Move Alphanumeric Edited)
$T = 3.38 + .4\,(NDWB_1 + NDWB_2 + NDWB_3) + .42\,(NMOP + N_2)$

CMPC (Compare Alphanumeric Character String)
$T = 3.06 + .4\,(NDWB_1 + NDWB_2) + .3\,N$

SCD (Scan Character Double)
$T = 3.46 + .4\,(NDWB_1) + .3\,N$

SCDR (Scan Character Double Reverse)
$T = 3.66 + .4\,(NDWB_1) + .3\,N$

SCM (Scan Character with Mask)
$T = 3.46 + .4\,(NDWB_1) + .3\,N$

SCMR (Scan Character with Mask Reverse)
$T = 3.66 + .4\,(NDWB_1) + .3\,N$

TCT (Test Character and Translate)
$T = 3.46 + .4\,(NDWB_1) + .7\,N$

TCTR (Test Character and Translate Reverse)
$T = 3.66 + .4\,(NDWB_1) + .7\,N$

ARITHMETIC INSTRUCTIONS

MVN (Move Numeric)
$T = 3.08 + .4\,(NDWB_1 + NDWB_2) + 1.2\,(NPDW) + .4(SF_2)$

CMPN (Compare Numeric)
$T = 3.06 + .4\,(NDWB_1 + NDWB_2) + 1.2\ NPDW\ +2.5SF_1$

MVNE (Numeric Edited Move)
$T = 3.38 + .4\,(NDWB_1 + NDWB_2 + NDWB_M) + .42\,(NMOP + N_2)$

AD2D (Add Decimal - 2 operands)
$T = 2.98 + .4\,(NDWB_1 + 2NDWB_2) + 1.6(NPDW) +2.5(SF_1) + .4(SF_2) + .4(R)$

AD3D (Add Decimal - 3 operands)
$T = 3.38 + .4\,(NDWB_1 + NDWB_2 + NDWB_3) + 1.6(NPDW) +2.5\,(SF_1) + .4(SF_2) + .4(R)$

SB2D (Subtract Decimal - 2 operands)
$T = 2.98 + .4\,(NDWB_1 + 2NDWB_2) + 1.6(NPDW) +2.5\,(SF_1) + .4(SF_2) + .4(R)$

SB3D (Subtract Decimal - 3 operands)
$T = 3.38 + .4\,(NDWB_1 + NDWB_2 + NDWB_3) + 1.6(NPDW) +2.5\,(SF_1) + .4(SF_2) + .4(R)$

MP2D (Multiply Decimal - 2 operands)
$T = 4.98 + .4\,(NDWB_1 + 2NDWB_2) + 1.0N_2 + .4(R) + .4(SF_2)$

MP3D (Multiply Decimal - 3 operands)
$T = 5.38 + .4\,(NDWB_1 + NDWB_2 + NDWB_3) + 1.0\,N_2 + .4(R) + .4(SF_2)$

DV2D (Divide Decimal - 2 operands)
$T = 6.98 + .4\,(NDWB_1 + 2NDWB_2) + 1.76\,(N_Q)\,(NPDW) + .4(R) + .4(SF_2)$

DV3D (Divide Decimal - 3 operands)
$T = 7.38 + .4\,(NDWB_1 + NDWB_2 + NDWB_3) + 1.76\,(N_Q)\,(NPDW) + .4(R) + .4(SF_2)$

DVDR (Divide Decimal with Remainder)
$T = 7.78 + .4\,(NDWB_1 + 2NDWB_2 + NDWB_3) + 1.76\,(N_Q)\,(NPDW) + .4(SF_2)$

TABLE 4. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6080 PROCESSOR (INTERLEAVED MEMORIES) (Cont.)

BIT STRING INSTRUCTIONS

CSL (Combine Bit Strings Left)
$T = 3.38 + .4 (NDWB_1 + NDWB_2) + .4 N$

CSR (Combine Bit Strings Right)
$T = 3.58 + .4 (NDWB_1 + NDWB_2) + .4 N$

SZTL (Set Zero and Truncation Indicators with Bit Strings Left)
$T = 3.06 + .4 (NDWB_1 + NDWB_2) + .4 N$

SZTL (Set Zero and Truncation Indicators with Bit Strings Right)
$T = 3.26 + .4 (NDWB_1 + NDWB_2) + .4 N$

CMPB (Compare Bit Strings)
$T = 3.06 + .4 (NDWB_1 + NDWB_2) + .4 N$

Conversion Instructions

DTB (Decimal-to-Binary Conversion)
$T = 3.93 + .4 B$

BTD ((Binary-to-Decimal Conversion)
$T = 4.98 + .4 B$

B = number of binary digits produced or converted.

Address Register Instructions

LARn	(Load Address Register n)	= 1.66
LAREG	(Load Address Registers)	= 4.93
SARn	(Store Address Register n)	= 1.66
SAREG	(Store Address Registers)	= 4.93
AWD	(Add Word Displacement to ARn)	= .7
A9BD	(Add 9-Bit Character Displacement to ARn)	= .7
A6BD	(Add 6-Bit Character Displacement to ARn)	= 4.15
A4BD	(Add 4-Bit Character Displacement to ARn)	= .7
ABD	(Add Bit Displacement to ARn)	= 4.15
SWD	(Subtract Word Displacement from ARn)	= .7
S9BD	(Subtract 9-Bit Character Displacement from ARn)	= .7
S6BD	(Subtract 6-Bit Character Displacement from ARn)	= 4.15
S4BD	(Subtract 4-Bit Character Displacement from ARn)	= .7
SBD	(Subtract Bit Displacement from ARn)	= 4.15
AARn	(Alphanumeric Descriptor to ARn)	= 1.66
NARn	(Numeric Descriptor to ARn)	= 1.66
ARAn	(ARn to Alphanumeric Descriptor)	= 3.32
ARn	(ARn to Numeric Descriptor)	= 3.32

Other Instructions

SPL	(Store Pointers and Lengths)	= 4.93
LPL	(Load Pointers and Lengths)	= 4.93
TRTN	(Transfer on Truncation Indicator ON)	= 1.2
TRTF	(Transfer on Truncation Indicator OFF)	= 1.2
TTN	(Transfer on Tally Runout Indicator ON)	= 1.2
TPNZ	(Transfer on Plus and Non-Zero)	= 1.2
	Conditional Transfer Not Taken	= .8

TABLE 5. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6060 PROCESSOR (INTERLEAVED MEMORIES)

Alphanumeric Instructions

MLR (Move Alphanumeric Left to Right)

$T = \begin{Bmatrix} 5.3 \\ 6.8 \end{Bmatrix} + \begin{Bmatrix} .4 \\ .2 \end{Bmatrix} N_2, \begin{Bmatrix} N \leq 9 \\ N > 9 \end{Bmatrix}$ (ASCII)

$T = \begin{Bmatrix} 5.5 \\ 7.0 \end{Bmatrix} + \begin{Bmatrix} .5 \\ .4 \end{Bmatrix} N_2, \begin{Bmatrix} N \leq 12 \\ N > 12 \end{Bmatrix}$ (BCD)

MRL (Move Alphanumeric Right to Left)

$T = \begin{Bmatrix} 5.6 \\ 7.1 \end{Bmatrix} + \begin{Bmatrix} .4 \\ .2 \end{Bmatrix} N_2, \begin{Bmatrix} N \leq 9 \\ N > 9 \end{Bmatrix}$ (ASCII)

$T = \begin{Bmatrix} 5.8 \\ 7.3 \end{Bmatrix} + \begin{Bmatrix} .5 \\ .4 \end{Bmatrix} N_2, \begin{Bmatrix} N \leq 12 \\ N > 12 \end{Bmatrix}$ (BCD)

MVT (Move Alphanumeric with Transliteration)

$T = 5.5 + 1.2\ N_1$

MVE (Move Alphanumeric Edited)

$T = 5.03 + .65\ (NDWB_1 + NDWB_2 + NDWB_3) + .42\ (NMOP + N_2)$

CMPC (Compare Alphanumeric Character String)

$T = 4.81 + .65\ (NDWB_1 + NDWB_2) + .3\ N$

SCD (Scan Character Double)

$T = 5.21 + .65\ (NDWB_1) + .3\ N$

SCDR (Scan Character Double Reverse)

$T = 5.41 + .65\ (NDWB_1) + .3\ N$

SCM (Scan Character with Mask)

$T = 5.21 + .65\ (NDWB_1) + .3\ N$

SCMR (Scan Character with Mask Reverse)

$T = 5.41 + .65\ (NDWB_1) + .3\ N$

TCT (Test Character and Translate)

$T = 5.21 + .65\ (NDWB_1) + 1.2\ N$

TCTR (Test Character and Translate Reverse)

$T = 5.41 + .65\ (NDWB_1) + 1.2\ N$

Arithmetic Instructions

MVN (Move Numeric)

$T = 4.63 + .65\ (NDWB_1 + NDWB_2) + 1.2\ (NPDW) + .4(SF_2)$

CMPN (Compare Numeric)

$T = 4.51 + .65\ (NDWB_1 + NDWB_2) + 1.2\ NPDW + 2.5\ SF_1$

MVNE (Numeric Edited Move)

$T = 5.03 + .65\ (NDWB_1 + NDWB_2 + NDWB_M) + .42\ (NMOP + N_2)$

AD2D (Add Decimal - 2 operands)

$T = 4.63 + .65\ (NDWB_1 + 2NDWB_2) + 1.6\ (NPDW) + 2.5(SF_1) + .4(SF_2) + .4(R)$

AD3D (Add Decimal - 3 operands)

$T = 5.03 + .65\ (NDWB_1 + NDWB_2 + NDWB_3) + 1.6\ (NPDW) + 2.5\ (SF_1) + .4(SF_2) + .4(R)$

SB2D (Subtract Decimal - 2 operands)

$T = 4.63 + .65\ (NDWB_1 + 2NDWB_2) + 1.6\ (NPDW) + 2.5(SF_1) + .4(SF_2) + .4(R)$

SB3D (Subtract Decimal - 3 operands)

$T = 5.03 + .65\ (NDWB_1 + NDWB_2 + NDWB_3) + 1.6\ (NPDW) + 2.5(SF_1) + .4(SF_2) + .4(R)$

MP2D (Multiply Decimal - 2 operands)

$T = 6.88 + .65\ (NDWB_1 + 2NDWB_2) + 1.0N_2 + .4(R) + .4(SF_2)$

TABLE 5. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6060 PROCESSOR (INTERLEAVED MEMORIES) (Cont.)

MP3D (Multiply Decimal - 3 operands)

$T = 7.28 + .65\ (NDWB_1 + NDWB_2 + NDWB_3) + 1.0N_2 + .4(R) + .4(SF_2)$

DV2D (Divide Decimal - 2 operands)

$T = 8.88 + .65\ (NDWB_1 + 2NDWB_2) + 1.76\ (N_Q)\ (NPDW) + .4(R) + .4(SF_2)$

DV3D (Divide Decimal - 3 operands)

$T = 9.28 + .65\ (NDWB_1 + NDWB_2 + NDWB_3) + 1.76\ (N_Q)\ (NPDW) + .4(R) + .4(SF_2)$

DVDR (Divide Decimal with Remainder)

$T = 9.68 + .65\ (NDWB_1 + 2NDWB_2 + NDWB_3) + 1.76\ (N_Q)\ (NPDW) + .4(SF_2)$

Bit String Instructions

CSL (Combine Bit Strings Left)

$T = 5.23 + .65\ (NDWB_1 + NDWB_2) + .4\ N$

CSR (Combine Bit Strings Right)

$T = 5.43 + .65\ (NDWB_1 + NDWB_2) + .4\ N$

SZTL (Set Zero and Truncation Indicators with Bit Strings Left)

$T = 4.76 + .65\ (NDWB_1 + NDWB_2) + .4\ N$

SZTR (Set Zero and Truncation Indicators with Bit Strings Right)

$T = 4.96 + .65\ (NDWB_1 + NDWB_2) + .4\ N$

CMPB (Compare Bit Strings)

$T = 4.76 + .65\ (NDWB_1 + NDWB_2) + .4\ N$

Conversion Instructions

DTB (Decimal-to-Binary Conversion)

$T = 5.38 + .4\ B$

BTD (Binary-to-Decimal Conversion)

$T = 6.38 + .4\ B$

B = number of binary digits produced or converted.

Address Register Instructions

LARn	(Load Address Register n)	= 2.36
LAREG	(Load Address Registers)	= 5.63
SARn	(Store Address Register n)	= 2.36
SAREG	(Store Address Registers)	= 5.63
AWD	(Add Word Displacement to ARn)	= 1.0
A9BD	(Add 9-Bit Character Displacement to ARn)	= 1.0
A6BD	(Add 6-Bit Character Displacement to ARn)	= 4.45
A4BD	(Add 4-Bit Character Displacement to ARn)	= 1.0
ABD	(Add Bit Displacement to ARn)	= 4.45
SWD	(Subtract Word Displacement from ARn)	= 1.0
S9BD	(Subtract 9-Bit Character Displacement from ARn)	= 1.0
S6BD	(Subtract 6-Bit Character Displacement from ARn)	= 4.45
S4BD	(Subtract 4-Bit Character Displacement from ARn)	= 1.0

TABLE 5. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6060 PROCESSOR (INTERLEAVED MEMORIES) (Cont.)

SBD	(Subtract Bit Displacement from ARn)	= 4.45
AARn	(Alphanumeric Descriptor to ARn)	= 2.36
NARn	(Numeric Descriptor to ARn)	= 2.36
ARAn	(ARn to Alphanumeric Descriptor)	= 4.72
ARn	(ARn to Numeric Descriptor)	= 4.72

Other Instructions

SPL	(Store Pointers and Lengths)	= 5.63
LPL	(Load Pointers and Lengths)	= 5.63
TRTN	(Transfer on Truncation Indicator ON)	= 1.8
TRTF	(Transfer on Truncation Indicator OFF)	= 1.8
TTN	(Transfer on Tally Runout Indicator ON)	= 1.8
TPNZ	(Transfer on Plus and Non-Zero)	= 1.8
	Conditional Transfer Not Taken	= 1.2

TABLE 6. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6060 PROCESSOR (NO INTERLEAVING)

Alphanumeric Instructions

MLR (Move Alphanumeric Left to Right)

$$T = \left\{\begin{matrix} 5.35 \\ 7.1 \end{matrix}\right\} + \left\{\begin{matrix} .45 \\ .30 \end{matrix}\right\} N_2 \quad \text{(ASCII)} \quad \left\{\begin{matrix} N \leqslant 9 \\ N > 9 \end{matrix}\right.$$

$$T = \left\{\begin{matrix} 4.9 \\ 6.8 \end{matrix}\right\} + \left\{\begin{matrix} .85 \\ .7 \end{matrix}\right\} N_2 \quad \text{(BCD)} \quad \left\{\begin{matrix} N \leqslant 12 \\ N > 12 \end{matrix}\right.$$

MRL (Move Alphanumeric Right to Left)

$$T = \left\{\begin{matrix} 5.55 \\ 7.3 \end{matrix}\right\} + \left\{\begin{matrix} .45 \\ .30 \end{matrix}\right\} N_2 \quad \text{(ASCII)} \quad \left\{\begin{matrix} N \leqslant 9 \\ N > 9 \end{matrix}\right.$$

$$T = \left\{\begin{matrix} 5.1 \\ 7.0 \end{matrix}\right\} + \left\{\begin{matrix} .85 \\ .7 \end{matrix}\right\} N_2 \quad \text{(BCD)} \quad \left\{\begin{matrix} N \leqslant 12 \\ N > 12 \end{matrix}\right.$$

MVT (Move Alphanumeric with Transliteration)

$$T = 7.8 + 1.5\ N_2$$

MVE (Move Alphanumeric Edited)

$$T = 6.6 + 1.2\ (NDWB_1 + NDWB_2 + NDWB_3) + .42\ (NMOP + N_2)$$

CMPC (Compare Alphanumeric Character String)

$$T = 4.6 + 1.2\ (NDWB_1 + NDWB_2) + .3\ N$$

SCD (Scan Character Double)

$$T = 5.8 + 1.2\ (NDWB_1) + .3\ N$$

SCDR (Scan Character Double Reverse)

$$T = 6.0 + 1.2\ (NDWB_1) + .3\ N$$

SCM (Scan Character with Mask)

$$T = 5.8 + 1.2\ (NDWB_1) + .3\ N$$

SCMR (Scan Character with Mask Reverse)

$$T = 6.0 + 1.2\ (NDWB_1) + .3\ N$$

TABLE 6. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6060 PROCESSOR (NO INTERLEAVING) (Cont.)

TCT (Test Character and Translate)

$T = 5.8 + 1.2\ (NDWB_1) + 1.5\ N$

TCTR (Test Character and Translate Reverse)

$T = 6.0 + 1.2\ (NDWB_1) + 1.5\ N$

Arithmetic Instructions

MVN (Move Numeric)

$T = 5.4 + 1.2\ (NDWB_1 + NDWB_2) + 1.2(NPDW) + .4(SF_2)$

CMPN (Compare Numeric)

$T = 4.2 + 1.2\ (NDWB_1 + NDWB_2) + 1.2\ NPDW + 2.5\ SF_1$

MVNE (Numeric Edited Move)

$T = 6.6 + 1.2\ (NDWB_1 + NDWB_2 + NDWB_M) + .42\ (NMOP + N_2)$

AD2D (Add Decimal - 2 operands)

$T = 5.4 + 1.2\ (NDWB_1 + 2NDWB_2) + 1.6(NPDW) + 2.5(SF_1) + .4(SF_2) + .4(R)$

AD3D (Add Decimal - 3 operands)

$T = 6.6 + 1.2\ (NDWB_1 + NDWB_2 + NDWB_3) + 1.6\ (NPDW) + 2.5(SF_1) + .4(SF_2) + .4(R)$

SB2D (Subtract Decimal - 2 operands)

$T = 5.4 + 1.2\ (NDWB_1 + 2NDWB_2) + 1.6(NPDW) + 2.5(SF_1) + .4(SF_2) + .4(R)$

SB3D (Subtract Decimal - 3 operands)

$T = 6.6 + 1.2\ (NDWB_1 + NDWB_2 + NDWB_3) + 1.6\ (NPDW) + 2.5(SF_1) + .4(SF_2) + .4(R)$

MP2D (Multiply Decimal - 2 operands)

$T = 7.4 + 1.2\ (NDWB_1 + 2NDWB_2) + 1.0N_2 + .4(R) + .4(SF_2)$

MP3D (Multiply Decimal - 3 operands)

$T = 8.6 + 1.2\ (NDWB_1 + NDWB_2 + NDWB_3) + 1.0\ N_2 + .4(R) + .4(SF_2)$

DV2D (Divide Decimal - 2 operands)

$T = 9.4 + 1.2\ (NDWB_1 + 2NDWB_2) + 1.76\ (N_Q)\ (NPDW) + .4(R) + .4(SF_2)$

DV3D (Divide Decimal - 3 operands)

$T = 10.6 + 1.2\ (NDWB_1 + NDWB_2 + NDWB_3) + 1.76\ (N_Q)\ (NPDW) + .4(R) + .4(SF_2)$

DVDR (Divide Decimal with Remainder)

$T = 11.8 + 1.2\ (NDWB_1 + 2NDWB_2 + NDWB_3) + 1.76\ (N_Q)\ (NPDW) + .4(SF_2)$

Bit String Instructions

CSL (Combine Bit Strings Left)

$T = 6.6 + 1.2\ (NDWB_1 + NDWB_2) + .4\ N$

CSR (Combine Bit Strings Right)

$T = 6.8 + 1.2\ (NDWB_1 + NDWB_2) + .4\ N$

SZTL (Set Zero and Truncation Indicators with Bit Strings Left)

$T = 4.2 + 1.2\ (NDWB_1 + NDWB_2) + .4\ N$

SZTR (Set Zero and Truncation Indicators with Bit Strings Right)

$T = 4.4 + 1.2\ (NDWB_1 + NDWB_2) + .4\ N$

CMPB (Compare Bit Strings)

$T = 4.2 + 1.2\ (NDWB_1 + NDWB_2) + .4\ N$

TABLE 6. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6060 PROCESSOR (NO INTERLEAVING) (Cont.)

Conversion Instructions

DTB (Decimal-to-Binary Conversion)

$T = 5.4 + .4\,B$

BTD (Binary-to-Decimal Conversion)

$T = 6.6 + .4\,B$

B = number of binary digits produced or converted.

Address Register Instructions

LARn	(Load Address Register n)	= 2.76
LAREG	(Load Address Registers)	= 6.03
SARn	(Store Address Register n)	= 2.76
SAREG	(Store Address Registers)	= 6.03
AWD	(Add Word Displacement to ARn)	= 1.0
A9BD	(Add 9-Bit Character Displacement to ARn)	= 1.0
A6BD	(Add 6-Bit Character Displacement to ARn)	= 4.45
A4BD	(Add 4-Bit Character Displacement to ARn)	= 1.0
ABD	(Add Bit Displacement to ARn)	= 4.45
SWD	(Subtract Word Displacement from ARn)	= 1.0
S9BD	(Subtract 9-Bit Character Displacement from ARn)	= 1.0
S6BD	(Subtract 6-Bit Character Displacement from ARn)	= 4.45
S4BD	(Subtract 4-Bit Character Displacement from ARn)	= 1.0
SBD	(Subtract Bit Displacement from ARn)	= 4.45
AARn	(Alphanumeric Descriptor to ARn)	= 2.76
NARn	(Numeric Descriptor to ARn)	= 2.76
ARAn	(ARn to Alphanumeric Descriptor)	= 5.52
ARn	(ARn to Numeric Descriptor)	= 5.52

Other Instructions

SPL	(Store Pointers and Lengths)	= 6.03
LPL	(Load Pointers and Lengths)	= 6.03
TRTN	(Transfer on Truncation Indicator ON)	= 1.8
TRTF	(Transfer on Truncation Indicator OFF)	= 1.8
TTN	(Transfer on Tally Runout Indicator ON)	= 1.8
TPNZ	(Transfer on Plus and Non-Zero)	= 1.8
	Conditional Transfer Not Taken	= 1.2

TABLE 7. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6040 PROCESSOR

Alphanumeric Instructions

MLR (Move Alphanumeric Left to Right)

$$T = \begin{Bmatrix} 7.41 \\ 9.73 \end{Bmatrix} + \begin{Bmatrix} .64 \\ .43 \end{Bmatrix} N_2 \quad \text{(ASCII)} \quad \begin{Bmatrix} N \leqslant 9 \\ N > 9 \end{Bmatrix}$$

$$T = \begin{Bmatrix} 6.77 \\ 9.30 \end{Bmatrix} + \begin{Bmatrix} 1.04 \\ .83 \end{Bmatrix} N_2 \quad \text{(BCD)} \quad \begin{Bmatrix} N \leqslant 12 \\ N > 12 \end{Bmatrix}$$

TABLE 7. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6040 PROCESSOR (Cont.)

MRL (Move Alphanumeric Right to Left)

$T = \begin{Bmatrix} 7.61 \\ 9.93 \end{Bmatrix} + \begin{Bmatrix} .64 \\ .43 \end{Bmatrix} N_2$ (ASCII) $\begin{Bmatrix} N \leqslant 9 \\ N > 9 \end{Bmatrix}$

$T = \begin{Bmatrix} 6.97 \\ 9.50 \end{Bmatrix} + \begin{Bmatrix} 1.04 \\ .83 \end{Bmatrix} N_2$ (BCD) $\begin{Bmatrix} N \leqslant 12 \\ N > 12 \end{Bmatrix}$

MVT (Move Alphanumeric with Transliteration)

$T = 11.05 + .1.9 \; N_2$

MVE (Move Alphanumeric Edited)

$T = 9.35 + 1.7 \; (NDWB_1 + NDWB_2 + NDWB_3) + .42 \; (NMOP + N_2)$

CMPC (Compare Alphanumeric Character String)

$T = 6.25 + 1.7 \; (NDWB_1 + NDWB_2) + .3 \; N$

SCD (Scan Character Double)

$T = 7.95 + 1.7 \; (NDWB_1) + .3 \; N$

SCDR (Scan Character Double Reverse)

$T = 8.15 + 1.7 \; (NDWB_1) + .3 \; N$

SCM (Scan Character with Mask)

$T = 7.95 + 1.7 \; (NDWB_1) + .3 \; N$

SCMR (Scan Character with Mask Reverse)

$T = 8.15 + 1.7 \; (NDWB_1) + .3 \; N$

TCT (Test Character and Translate)

$T = 7.95 + 1.7 \; (NDWB_1) + 1.7 \; N$

TCTR (Test Character and Translate Reverse)

$T = 8.15 + 1.7 \; (NDWB_1) + 1.7 \; N$

Arithmetic Instructions

MVN (Move Numeric)

$T = 7.65 + 1.7 \; (NDWB_1 + NDWB_2) + 1.2 \; (NPDW) + .4(SF_2)$

CMPN (Compare Numeric)

$T = 5.95 + 1.7 \; (NDWB_1 + NDWB_2) + 1.2 \; NPDW + 2.5 \; SF_1$

MVNE (Numeric Edited Move)

$T = 9.35 + 1.7 \; (NDWB_1 + NDWB_2 + NDWB_M) + .42 \; (NMOP + N_2)$

AD2D (Add Decimal - 2 operands)

$T = 7.65 + 1.7 \; (NDWB_1 + 2NDWB_2) + 1.6 \; (NPDW) + 2.5 \; (SF_1) + .4(SF_2) + .4(R)$

AD3D (Add Decimal - 3 operands)

$T = 9.35 + 1.7 \; (NDWB_1 + NDWB_2 + NDWB_3) + 1.6 \; (NPDW) + 2.5(SF_1) + .4(SF_2) + .4(R)$

SB2D (Subtract Decimal - 2 operands)

$T = 7.65 + 1.7 \; (NDWB_1 + 2NDWB_2) + 1.6 \; (NPDW) + 2.5(SF_1) + .4(SF_2) + .4(R)$

SB3D (Subtract Decimal - 3 operands)

$T = 9.35 + 1.7 \; (NDWB_1 + NDWB_2 + NDWB_3) + 1.6 \; (NPDW) + 2.5(SF_1) + .4(SF_2) + .4(R)$

MP2D (Multiply Decimal - 2 operands)

$T = 9.65 + 1.7 \; (NDWB_1 + 2NDWB_2) + 1.0N_2 + .4(R) + .4(SF_2)$

MP3D (Multiply Decimal - 3 operands)

$T = 11.35 + 1.7 \; (NDWB_1 + NDWB_2) + NDWB_3) + 1.0 \; N_2 + .4(R) + .4(SF_2)$

DV2D (Divide Decimal - 2 operands)

$T = 11.65 + 1.7 \; (NDWB_1 + 2NDWB_2) + 1.76 \; (N_Q) \; (NPDW) + .4(R) + .4(SF_2)$

TABLE 7. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6040 PROCESSOR (Cont.)

DV3D (Divide Decimal - 3 operands)

$T = 13.35 + 1.7 (NDWB_1 + NDWB_2 + NDWB_3) + 1.76 (N_Q) (NPDW) + .4(R) + .4(SF_2)$

DVDR (Divide Decimal with Remainder)

$T = 15.05 + 1.7 (NDWB_1 + 2NDWB_2 + NDWB_3) + 1.76 (N_Q) (NPDW) + .4(SF_2)$

Bit String Instructions

CSL (Combine Bit Strings Left)

$T = 9.35 + 1.7 (NDWB_1 + NDWB_2) + .4 N$

CSR (Combine Bit Strings Right)

$T = 9.55 + 1.7 (NDWB_1 + NDWB_2) + .4 N$

SZTL (Set Zero and Truncation Indicators with Bit Strings Left)

$T = 5.95 + 1.7 (NDWB_1 + NDWB_2) + .4 N$

SZTR (Set Zero and Truncation Indicators with Bit Strings Right)

$T = 6.15 + 1.7 (NDWB_1 + NDWB_2) + .4 N$

CMPB (Compare Bit Strings)

$T = 5.95 + 1.7 (NDWB_1 + NDWB_2) + .4 N$

Conversion Instructions

DTB (Decimal-to-Binary Conversion)

$T = 7.65 + .4 B$

BTD (Binary-to-Decimal Conversion)

$T = 8.85 + .4 B$

B = number of binary digits produced or converted.

Address Register Instructions

LARn	(Load Address Register n)	= 2.76
LAREG	(Load Address Registers)	= 11.04
SARn	(Store Address Register n)	= 2.76
SAREG	(Store Address Registers)	= 11.04
AWD	(Add Word Displacement to ARn)	= 1.2
A9BD	(Add 9-Bit Character Displacement to ARn)	= 1.2
A6BD	(Add 6-Bit Character Displacement to ARn)	= 4.65
A4BD	(Add 4-Bit Character Displacement to ARn)	= 1.2
ABD	(Add Bit Displacement to ARn)	= 4.65
SWD	(Subtract Word Displacement from ARn)	= 1.2
S9BD	(Subtract 9-Bit Character Displacement from ARn)	= 1.2
S6BD	(Subtract 6-Bit Character Displacement from ARn)	= 4.65
S4BD	(Subtract 4-Bit Character Displacement from ARn)	= 1.2
SBD	(Subtract Bit Displacement from ARn)	= 4.65
AARn	(Alphanumeric Descriptor to ARn)	= 2.76
NARn	(Numeric Descriptor to ARn)	= 2.76
ARAn	(ARn to Alphanumeric Descritpor	= 5.52
ARn	(ARn to Numeric Descriptor)	= 5.52

TABLE 7. PRELIMINARY INSTRUCTION TIMING FORMULAS FOR MODEL 6040 PROCESSOR (Cont.)

Other Instructions

SPL	(Store Pointers and Lengths)	= 11.04
LPL	(Load Pointers and Lengths)	= 11.04
TRTN	(Transfers on Truncation Indicator ON)	= 1.8
TRTF	(Transfer on Truncation Indicator OFF)	= 1.8
TTN	(Transfer on Tally Runout Indicator ON)	= 1.8
TPNZ	(Transfer on Plus and Non-Zero	= 1.8
	Conditional Transfer Not Taken	= 1.2

Appendix C
Complete List of EIS Instructions

ALPHANUMERIC INSTRUCTIONS

MLR	Move Alphanumeric Left to Right
MRL	Move Alphanumeric Right to Left
MVT	Move Alphanumeric with Translation
MVE	Move Alphanumeric Edited
CMPC	Compare Alphanumeric Character String
SCD	Scan Character Double
SCDR	Scan Character Double in Reverse
TCT	Test Character and Translate
TCTR	Test Character and Translate in Reverse
SCM	Scan with Mask
SCMR	Scan with Mask in Reverse

NUMERIC INSTRUCTIONS

MVN	Move Numeric
CMPN	Compare Numeric
MVNE	Move Numeric Edited
AD3D	Add Using 3 Decimal Operands
AD2D	Add Using 2 Decimal Operands
SB3D	Subtract Using 3 Decimal Operands
SB2D	Subtract Using 2 Decimal Operands
MP3D	Multiply Using 3 Decimal Operands
MP2D	Multiply Using 2 Decimal Operands
DV3D	Divide Using 3 Decimal Operands
DV2D	Divide Using 2 Decimal Operands
DVDR	Decimal Divide with Remainder

BIT STRING INSTRUCTIONS

CSL	Combine Bit Strings Left
CSR	Combine Bit Strings Right
SZTL	Set Zero and Truncation Indicators with Bit Strings Left
SZTR	Set Zero and Truncation Indicators with Bit Strings Right
CMPB	Compare Bit Strings

CONVERSION INSTRUCTIONS

DTB	Decimal-to-Binary Convert
BTD	Binary-to-Decimal Convert

ADDRESS REGISTER INSTRUCTIONS

Load Address Registers

LARn	Load Address Register n
LAREG	Load Address Registers

Store Address Registers

SARn	Store Address Register n
SAREG	Store Address Registers

Alter Contents of Address Registers

AWD	Add Word Displacement to Specified AR
A9BD	Add 9-Bit Character Displacement to Specified AR
A6BD	Add 6-Bit Character Displacement to Specified AR
A4BD	Add 4-Bit Character Displacement to Specified AR
ABD	Add Bit Displacement to Specified AR
SWD	Subtract Word Displacement from Specified AR
S9BD	Subtract 9-Bit Character Displacement from Specified AR
S6BD	Subtract 6-Bit Character Displacement from Specified AR
S4BD	Subtract 4-Bit Character Displacement from Specified AR
SBD	Subtract Bit Displacement from Specified AR

Special Address Register Instructions

AARn	Alphanumeric Descriptor to ARn
NARn	Numeric Descriptor to ARn
ARAn	ARn to Alphanumeric Descriptor
ARNn	ARn to Numeric Descriptor

INSTRUCTIONS FOR SAVING AND RESTORING INTERRUPT REGISTERS

SPL	Store Pointers and Lengths
LPL	Load Pointers and Lengths

TRUNCATION INDICATOR TEST INSTRUCTIONS

TRTN	Transfer on Truncation Indicator ON
TRTF	Transfer on Truncation Indicator OFF

OTHER TRANSFER INSTRUCTIONS

TTN	Transfer on Tally Runout Indicator ON
TPNZ	Transfer on Plus and Non-Zero

Appendix D
Complete List of Micro Operations

Mnemonic	Micro Operation
MVZB	Move with Zero Suppression and Blank Replacement
MVZA	Move with Zero Suppression and Asterisk Replacement
MVLS	Move with Floating Sign (+, ⁻) Insertion.
MFLC	Move with Floating Currency Symbol ($) Insertion
ENF	End Floating Suppression
SES	Set "End Suppression" Flag
INBS	Insert Blank on Suppress
INSA	Insert Asterisk on Suppress
INSN	Insert on Negative
INSP	Insert on Positive
INSM	Insert Multiple Characters
LTE	Load Edit Insertion Table Entry
CHT	Change Edit Insertion Table
IGN	Ignore Source Character
MVC	Move Source Character
MSES	Move and Set Sign Indicator
MORS	Move And Or Sign to Source Character

Appendix E
Series 6000 Six-bit BCD Character Set

Standard Character Set	Internal Machine Code	Octal Code	Hollerith Card Code	Standard Character Set	Internal Machine Code	Octal Code	Hollerith Card Code
0	00 0000	00	0	↑	10 0000	40	11-0
1	00 0001	01	1	J	10 0001	41	11-1
2	00 0010	02	2	K	10 0010	42	11-2
3	00 0011	03	3	L	10 0011	43	11-3
4	00 0100	04	4	M	10 0100	44	11-4
5	00 0101	05	5	N	10 0101	45	11-5
6	00 0110	06	6	O	10 0110	46	11-6
7	00 0111	07	7	P	10 0111	47	11-7
8	00 1000	10	8	Q	10 1000	50	11-8
9	00 1001	11	9	R	10 1001	51	11-9
[	00 1010	12	2-8	—	10 1010	52	11
#	00 1011	13	3-8	$	10 1011	53	11-3-8
@	00 1100	14	4-8	*	10 1100	54	11-4-8
:	00 1101	15	5-8	)	10 1101	55	11-5-8
>	00 1110	16	6-8	;	10 1110	56	11-6-8
?	00 1111	17	7-8	'	10 1111	57	11-7-8
ƀ	01 0000	20	(blank)	+	11 0000	60	12-0
A	01 0001	21	12-1	/	11 0001	61	0-1
B	01 0010	22	12-2	S	11 0010	62	0-2
C	01 0011	23	12-3	T	11 0011	63	0-3
D	01 0100	24	12-4	U	11 0100	64	0-4
E	01 0101	25	12-5	V	11 0101	65	0-5
F	01 0110	26	12-6	W	11 0110	66	0-6
G	01 0111	27	12-7	X	11 0111	67	0-7
H	01 1000	30	12-8	Y	11 1000	70	0-8
I	01 1001	31	12-9	Z	11 1001	71	0-9
&	01 1010	32	12	←	11 1010	72	0-2-8
.	01 1011	33	12-3-8	,	11 1011	73	0-3-8
]	01 1100	34	12-4-8	%	11 1100	74	0-4-8
(	10 1101	35	12-5-8	=	11 1101	75	0-5-8
<	01 1110	36	12-6-8	"	11 1110	76	0-6-8
!	01 1111	37	12-7-8	!	11 1111	77	0-7-8

Appendix F
American Standard Code for Information Interchange (ASCII)

Bits b7 b6 b5					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b4	b3	b2	b1	Col. / Row	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SS	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	¦
1	1	0	1	13	CR	GS	-	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

The Other Computer Company:

Honeywell

HONEYWELL INFORMATION SYSTEMS

3620
4271
Printed in U.S.A.

In the U.S.A.: 200 Smith Street, MS 061, Waltham, Massachusetts 02154
In Canada: 2025 Sheppard Avenue East, Willowdale, Ontario

1701

DA48, Rev. 0