

FEB 19 1971

SYSTEMS 86 Software



SYSTEMS 86... Total integration of hardware and software.



The real-time computer systems you use in the 1970's must offer a high degree of multiprogramming efficiency. Task scheduling and control must be versatile. Background throughput must be high. And task switching must be accomplished with minimum software overhead. There are many more requirements, but perhaps they can all be summed up in the need for superior price/performance.

The requirements of the 1970's can no longer be satisfied by taking the traditional approach of first designing the computer and then adapting the software to make do with the facilities already engineered into the hardware. Rather, hardware and software must be designed concurrently. They must be totally integrated for rapid and economical adaptation to a dynamic mix of environments.

But, how do you get a computer system that meets all these requirements?

SYSTEMS 86 is the answer.

With the SYSTEMS 86 Real-Time Computer System we've departed from the concept of separate hardware and software design. Working hand in hand with engineers, SYSTEMS programmers have

applied their full efforts toward total integration of software and hardware to give users of SYSTEMS 86 superior price/performance.

By making numerous hardware-software design tradeoffs, SYSTEMS programmers and engineers have made SYSTEMS 86 a truly balanced system. To provide maximum efficiency, for example, many functions traditionally delegated to the hardware are turned over to the software.

Similarly, the SYSTEMS 86 hardware includes many features that relieve much of the burden from the software.

Out of this integration of hardware and software have evolved the programming systems needed to deal with your application. Because the programming systems are designed in the same light as the SYSTEMS 86 hardware, they handle your total job faster, more thoroughly, and more economically.

SYSTEMS 86 programming includes:

- **Batch Processing System** provides complete facilities for performing any mixture of program assemblies, compilations, and executions.



It also allows users to apply the full capabilities of SYSTEMS 86 to their real-time applications.

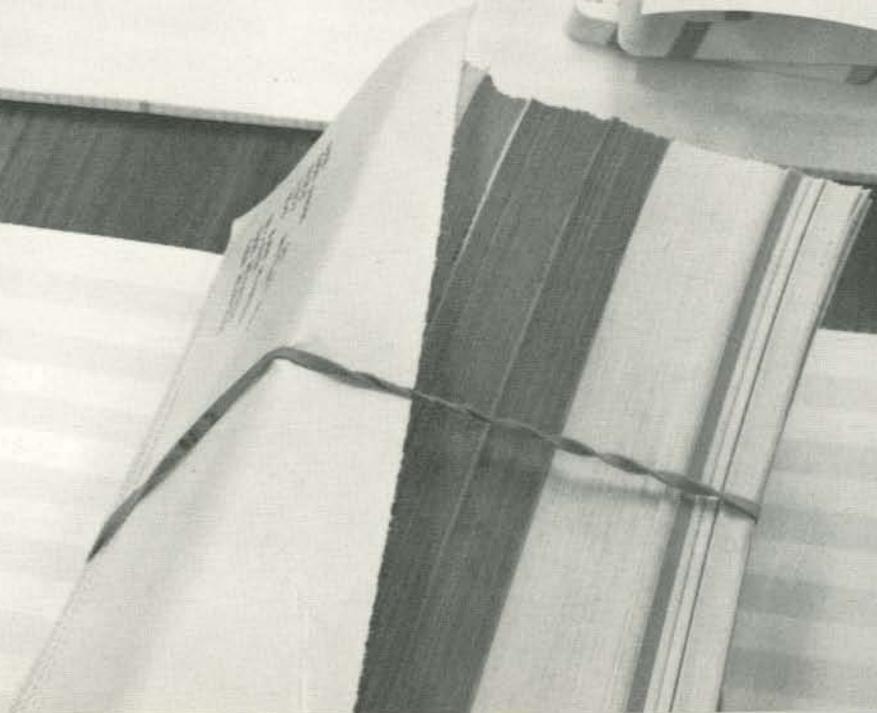
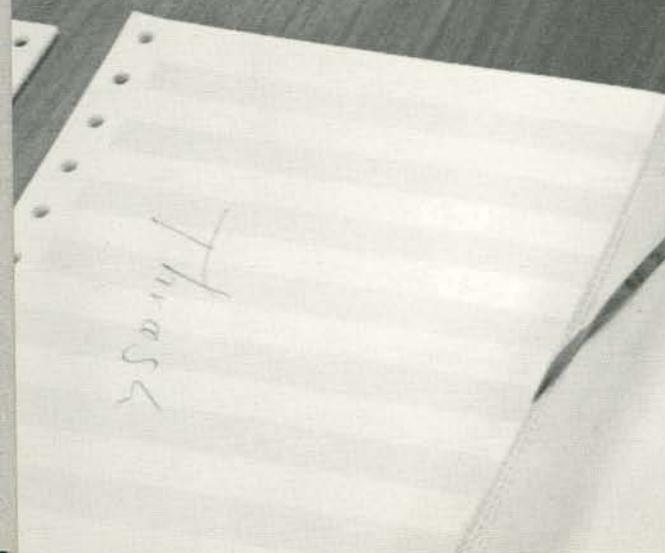
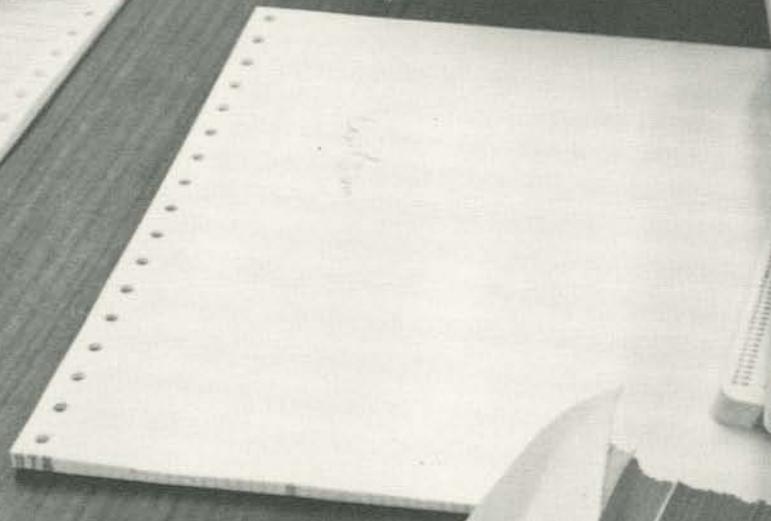
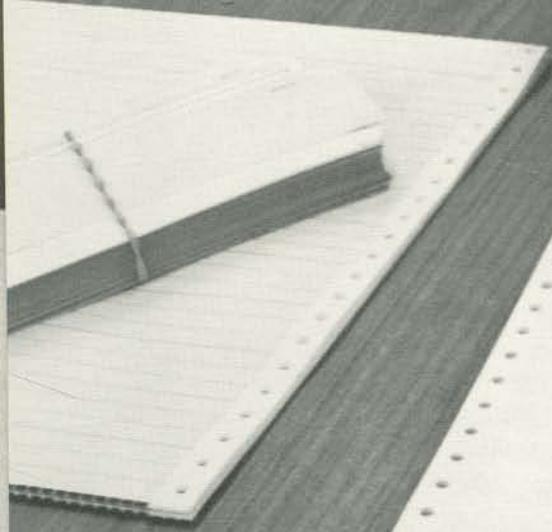
- **Real-Time Monitor** provides the ideal multi-programming environment for real-time applications, batch processing, and general-purpose scientific applications.
- **Extended FORTRAN IV** gives users of SYSTEMS 86 an entirely new level of capability for compiling and writing programs in the procedure-oriented language of FORTRAN.
- **Symbolic Assembler** allows users to code and assemble their real-time application programs with an easy-to-use symbolic language that provides the full flexibility of machine language.
- **Macro Assembler** provides all the capabilities

of the Symbolic Assembler with additional capabilities for nesting macros, executing recursive macro calls, and passing parameters on to nested macros.

- **BASIC**, which is currently under development, provides capabilities for multi-user terminal-oriented communications with SYSTEMS 86.

So, you see, SYSTEMS 86 programming systems cover every aspect of the problem — whether it be real-time data acquisition and control in a multi-programming environment, data processing, scientific computations, statistical studies, multi-user, terminal-oriented programming, or any combination of these.

In addition SYSTEMS 86 includes complete hardware diagnostic routines that simplify troubleshooting and reduce down-time to minutes. Plans are also underway for a COBOL compiler to allow users to take advantage of the SYSTEMS 86 outstanding potential for solving business-oriented problems.



Batch Processing System

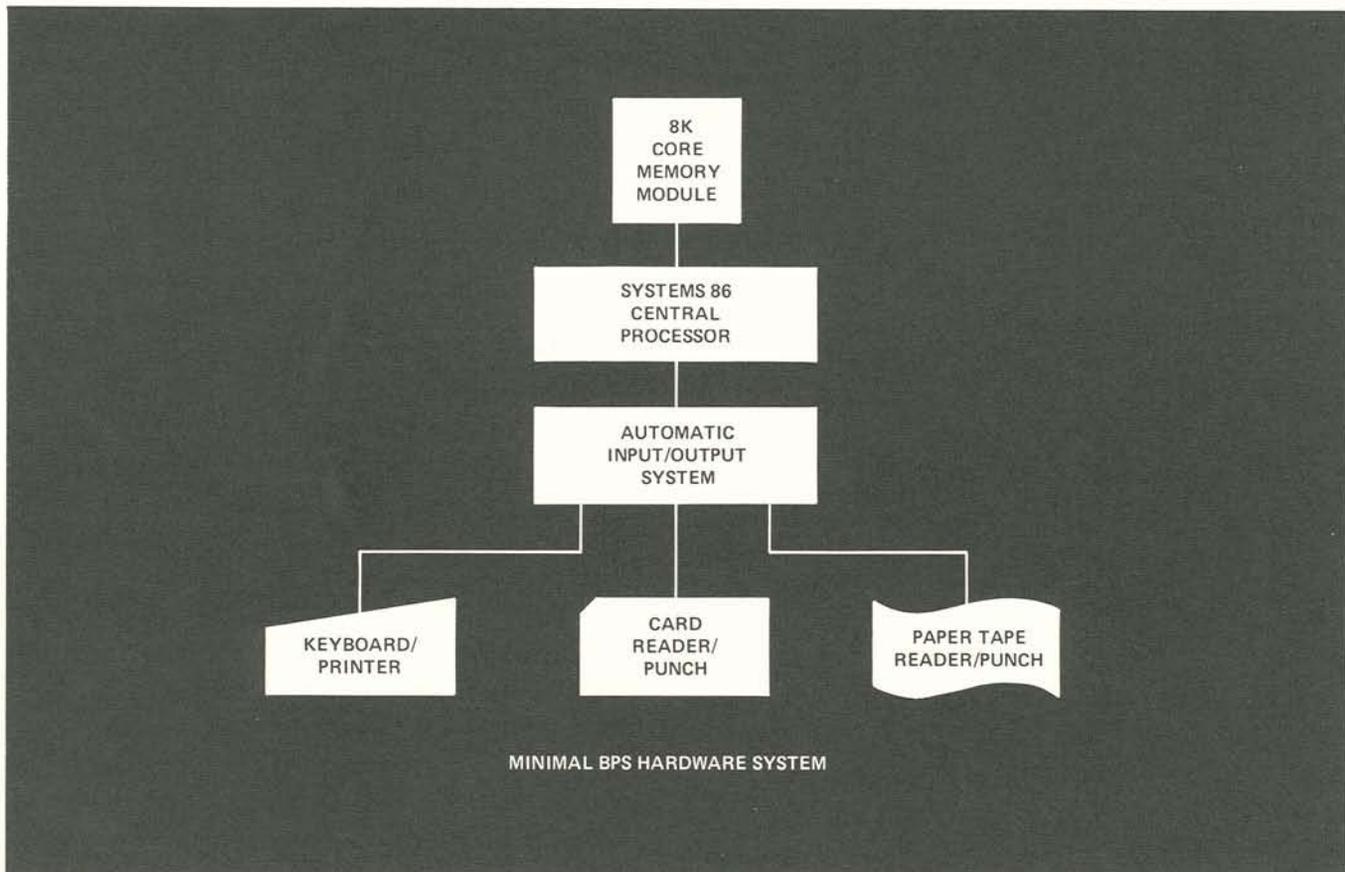
The Batch Processing System provides users of SYSTEMS 86 with complete facilities for performing any mixture of program assemblies, compilations, and executions in a job-stack environment.

All elements of the Batch Processing System are designed for optimum utilization of storage facilities and peripherals. Thus, the entire system can operate in a SYSTEMS 86 hardware configuration containing only one or two 8K-word memory modules, a paper tape system, and a card reader. The structure of the Batch Processing System, however, allows users to improve system performance by adding memory modules, peripherals, direct memory access channels, and other optional equipment to the hardware system. Magnetic tape drives and disc files, for example, can be added to hold processors and utility programs.

When operating under control of the Batch Processing System, SYSTEMS 86 isn't limited to handling data processing jobs. On the contrary, users can apply the full capabilities of SYSTEMS 86 to their real-time applications.

Real-time programs can be connected to virtually any interrupt level not being used by the Batch Processing System. These include external interrupts, the real-time clock interrupt, and peripheral service interrupts. Users assign system facilities to a real-time process on a dedicated basis. In addition to the interrupt levels, these facilities include disc and core residency, I/O channels, and peripheral devices. These assignments remain in force until the users or the process terminates the program.

Users with real-time applications can take advantage of the exceptional linkage and communications



facilities of the Batch Processing System and code their real-time programs exclusively in Extended FORTRAN IV.

The capabilities of the Batch Processing System are summarized as follows:

- Extensive job control language for directing all system activities
- Input/Output through calls to resident routines
- Simultaneous input/output and processing
- Processing of all standard traps and interrupts
- Program assembly with SYSTEMS Macro Assembler and Symbolic Assembler
- FORTRAN IV compilation and execution
- Program debugging aids including dump, snapshot, and trace
- Program loading including linkage of external routines and COMMON blocks
- Utility processors for editing source and object files and performing media conversion
- Program overlays
- Operator intervention facilities available at all times
- System generation with installation and configuration options

The BPS consists of modular software elements that are integrated with the SYSTEMS 86 hardware to provide processing capabilities tailored to each

user's needs. The major elements included in the Batch Processing System are as follows:

- Job Control Processor
- Control Monitor
- Input/Output Service System
- Link Loader
- System Processors

JOB CONTROL PROCESSOR

The Job Control Processor initiates and oversees the execution of all System Processors and user's programs. It performs such operations as manipulating files, redefining program origins, displaying (dumping) memory content, altering logical file assignments, transferring CPU control to programs, and implementing communications with the operator.

Users direct the activities of the Job Control Processor by arranging a sequence of job control statements in the System Control File. The job control statements form a comprehensive, yet easy-to-use language that simplifies computer utilization—even when dealing with the most complex assortment of program assemblies, compilations, and executions.

In processing the job control statements, the Job Control Processor provides the operator with a printed log of all statements that directly influence control of the system.

Although the Job Control Processor is a resident portion of the Batch Processing System, in most cases it is only required prior to the time that the Link Loader is entered. During the time when the Job Control Processor is not being used, programs can obtain additional memory by overlaying the Job Control Processor.



CONTROL MONITOR

By executing the call monitor instruction (CALM) the user's programs can direct the Control Monitor to perform a variety of system services. These services include loading overlay segments and subprograms, dynamically assigning logical files to devices, dynamically equating one logical file to another, initiating input/output, terminating programs, and communicating with the operator.

The Control Monitor also provides service routines for processing various traps and interrupts. For example, when the Control Monitor encounters a trap resulting from a program error, it activates a routine that aborts the program. It also sends a printed message to the operator stating the reason for aborting the program and the memory address where the program was aborted.

Another interrupt service routine included in the

Control Monitor enables the user to suspend, abort, and restart programs. The user calls the routine by depressing the Console Interrupt button on the console. He then uses the teletypewriter to specify the required operation.

INPUT/OUTPUT SERVICE SYSTEM

The input/output facilities for the Batch Processing System simplify programming while providing maximum system efficiency by overlapping input/output and processing. Device assignment is automatic, and input/output is performed through calls to resident routines.

To transfer data to or from a peripheral device, the program simply executes a CALM instruction directing the Control Monitor to activate the Input/Output Service System (IOSS). IOSS automatically initiates the resident I/O handler routine for the designated peripheral device. Under interrupt control, the I/O handler directs the entire data transfer operation and performs all necessary code translations.

At the user's discretion, IOSS can return control of the CPU to the calling program immediately after initiating the input/output operation. Thus input/output operations can proceed concurrent with computational jobs and other input/output operations.

LINK LOADER

Through the services provided by the Link Loader, users can load object modules from any logical file into core memory. The Link Loader allocates common storage areas and establishes program linkages. Diagnostics are provided with the Link Loader to assist the user in detecting errors.

Programs can directly call the Link Loader to load subprograms from the system library. The Link Loader makes all linkages to referenced entry points and common blocks in both the calling program and its subprograms.

Because the Link Loader is required only while loading core, the Batch Processing System permits users to obtain maximum use of memory by overlaying the Link Loader with blank COMMON. When the user's program has run to completion or is aborted, the Link Loader is automatically reloaded into its original memory location.

SYSTEM PROCESSORS

The following processors are included in the Batch Processing System to provide facilities for creating, coding, debugging, and modifying the entire array of user's programs: Extended FORTRAN IV, Macro Assembler, Symbolic Assembler, Media Conversion Program, Source Update, Library Generator, Debug Program, and System Generation Program.

SYSTEMS Extended FORTRAN IV, the Macro Assembler, and the Symbolic Assembler run under both the Batch Processing System and the Real-Time Monitor. The extensive capabilities that they offer to both real-time and scientifically oriented users are fully described in latter portions of this brochure.

Media Conversion Program. The Media Conversion Program enables users to easily perform operations ranging from the simple procedure of duplicating punched cards to complex editing and merging of data recorded on various types of input media. The key to the versatility of the Media Conversion Program is that it departs from the rigid structure commonly found in such utility programs. In place of using a fixed set of functions, the Media Conversion Program consists of a set of modular routines that the user can arrange through the use of FORTRAN-like control statements into the sequence that best fits his application.

Source Update. With the Source Update Program, users can correct and modify programs written in source language. By employing just four concise control statements, users can direct the Source Update Program to delete selected lines from source programs, replace deleted lines with new

lines, and insert virtually any number of new lines between any two lines within a source program. During a single program execution, the Source Update Program can modify several source language programs. It can handle source input files in either compressed or non-compressed format, and it can output in either format at the user's discretion.

Library Generator. The Library Generator operates on object modules that are code-compatible with the outputs of the assemblers and the FORTRAN IV Compiler. Its ability to perform a wide range of functions makes it an ideal tool for modifying and manipulating all types of assembled and compiled library modules. Capabilities of the Library Generator include:

- Selectively merging object modules recorded on several types of I/O media
- Copying object modules from any I/O medium onto any other I/O medium
- Selectively deleting object modules from a file
- Resequencing a file of object modules
- Listing the content of any file of object modules



- Creating a system library file
- Creating a directory of library files stored on the system discs

Debug Program. The Debug Program lets the user check out his programs and get them in an operational condition in the shortest possible time. A set of fourteen control statements gives the user complete control over the entire debugging process. Operations performed by the Debug Program include:

- Tracing any program segment or even an entire program
- Performing snapshot displays of major registers and core dumps of any group of memory locations
- Modifying the content of memory and the general purpose registers
- Filling selected portions of memory with any constant value
- Dynamically assigning files and devices to files
- Pausing services at any location
- Transferring and returning control to programs
- Loading programs dynamically

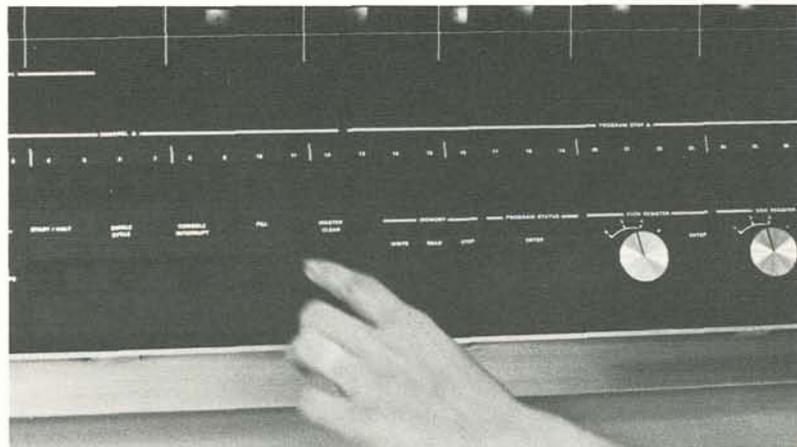
Users can control debugging on-line from the system teletypewriter while using the line printer to record such large volumes of information as trace outputs and core dumps. Because of the continuous processing capabilities of the Debug Program, the card reader can be used as an alternate control device.

System Generation Program. The System Generation Program (SYSGEN) tailors the Batch Processing System to precisely match each user's SYSTEMS 86 hardware configuration and processing requirements.

Using a set of SYSGEN control statements, the user simply describes his system and its operating environment. SYSGEN, in turn, configures the Batch Processing System to fit this description. It selects the required modules from a file containing all available software modules and generates all necessary tables and resident portions of the Batch Processing System. SYSGEN then outputs the generated system to the storage device selected by the user.

Users with requirements for a special purpose executive routine can use SYSGEN to create a Batch Processing System containing only the minimum elements needed for loading. This effectively produces a "stand-alone" loader with external label linkage and COMMON storage handling capabilities. After the user's program has been loaded, the core memory required for the Batch Processing System can be reclaimed and used for storing data.

Once SYSGEN completes its job, the user places the Batch Processing System into operation simply by depressing the Clear and Start switches on the console. When these switches are depressed, the system bootstraps the Batch Processing System from bulk storage to core memory. It then starts processing the job control statements.





Real-Time Monitor

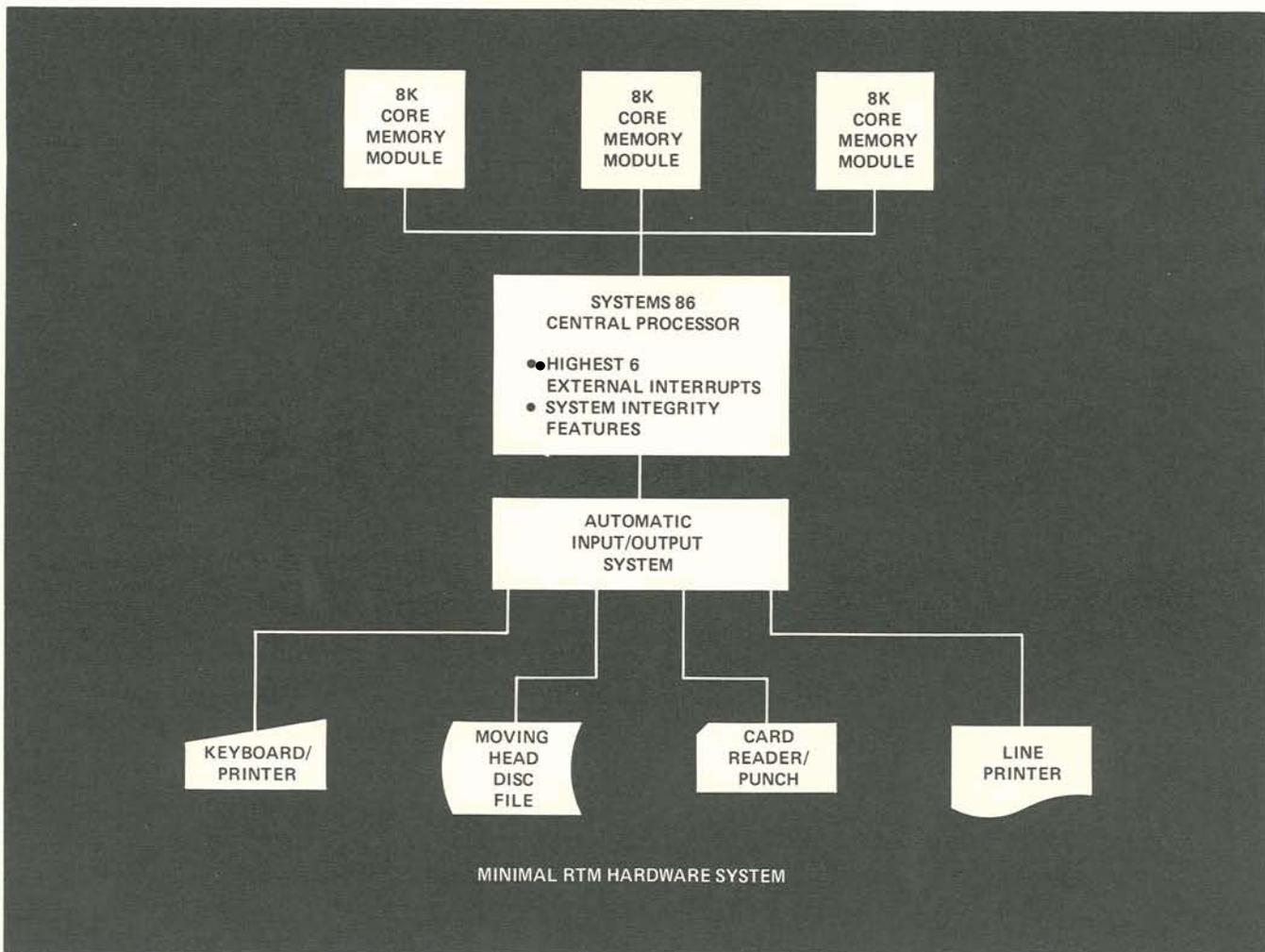
During the 1960's, Real-Time Monitors established a concept of foreground and background processing. Some computers during that time were capable of handling only a single foreground program; while others that could handle multiple foreground tasks were limited to a single nonreal-time background program. The SYSTEMS 86 Real-Time Monitor is designed for user needs of the 1970's. It is a totally task oriented system that enables users to multi-program up to sixty-four tasks in either the foreground or background — and in any combination of real-time and nonreal-time programs.

The task orientation of the Real-Time Monitor makes SYSTEMS 86 ideally suited for real-time ap-

plications, batch processing, and general-purpose scientific applications.

The Real-Time Monitor also gives SYSTEMS 86 strong remote terminal capabilities through either the conversational mode using BASIC or the remote job entry mode which permits the standard entry of jobs into the batch stream from remote terminals.

Users of SYSTEMS 86 can take advantage of the overlay and "roll-out" capabilities of the Real-Time Monitor to gain maximum use of available core. Programs requiring more operating core memory than is available can be organized into segments that sequentially overlay portions of memory. The Real-Time Monitor accepts the overlay structure of the

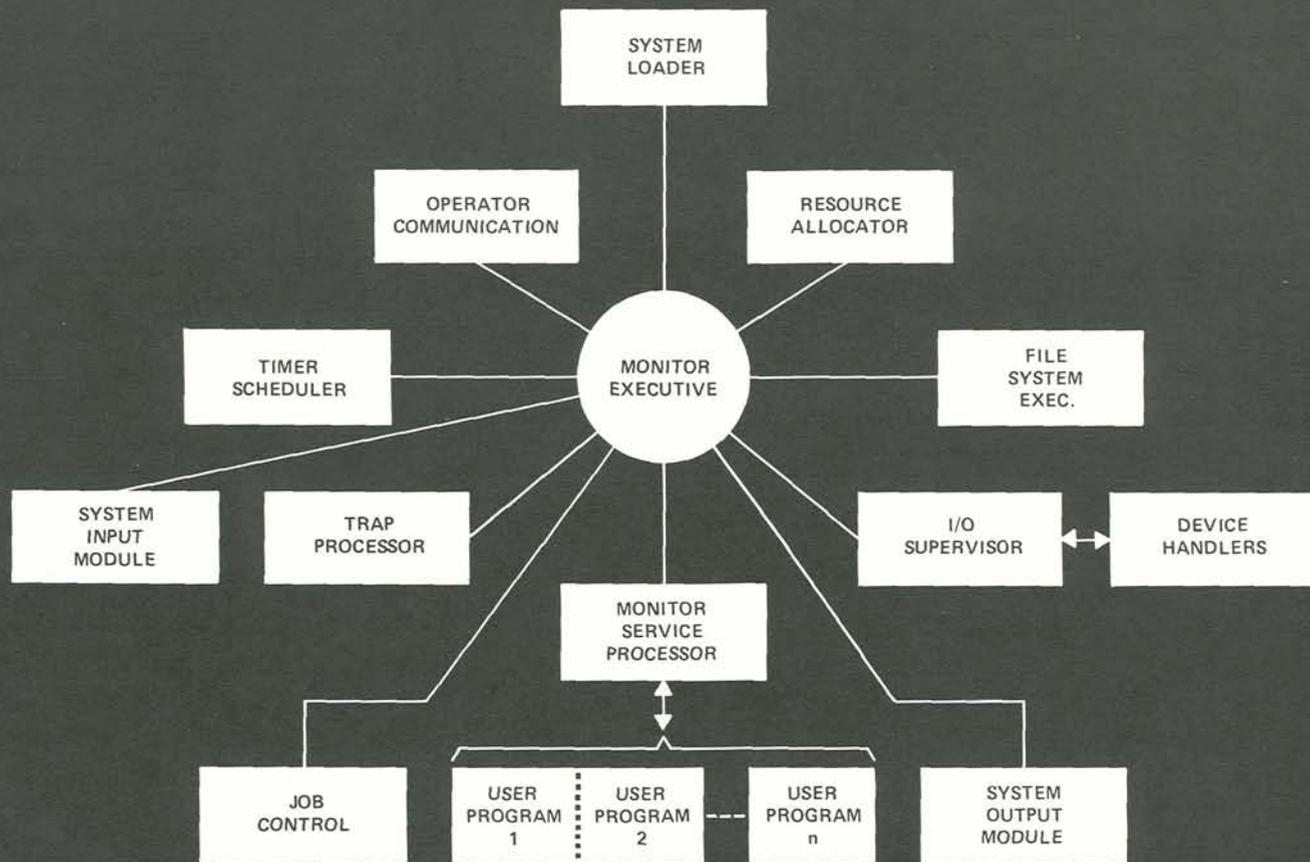


user's programs and ensures correct transfer from the disc to memory. In cases where the core requirements of the foreground programs and program segments exceed the foreground core area, the Real-Time Monitor provides additional foreground core by rolling out the background memory content onto a disc storage device.

The Real-Time Monitor activates programs by interrupt, by requests from other programs, by commands from the operator, by job control directives, and by commands from the time scheduler. Through these varied methods of program activation, the Real-Time Monitor gives users exceptional flexibility to adapt the system operating modes to changing real-time conditions.

The Real-Time Monitor can operate in a SYSTEMS 86 hardware configuration containing only three 8K-word memory modules, a 2 mega-byte disc file, a line printer, and a card reader/punch. Although the Real-Time Monitor will run with just one disc, multiple discs will improve performance considerably — especially for applications requiring a large amount of permanent file space. Users requiring permanent file save and restore capabilities must also include magnetic tape drives in their hardware configuration. By adding core memory, users can reduce software overhead by decreasing the frequency of program roll-out.

In addition to the following major capabilities and features, the Real-Time Monitor offers all the capa-



REAL-TIME MONITOR SYSTEM MODULES

bilities listed for the Batch Processing System:

- Multiprogramming environment supporting concurrent execution of 64 programs
- Interrupt and trap processing for all levels
- Timer scheduling of foreground programs
- System security via dynamic memory protection feature
- Servicing of all standard peripheral devices
- Reentrant monitor services available to both foreground and background programs
- On-line cataloging of foreground and background programs
- Dynamic assignment of foreground and background memory sizes
- File management
- Automatic roll-out of user programs
- On-line system control from the console
- Debugging with dump, snapshot, and trace
- Batch job processing through a job control language

The basic modular elements of the Real-Time Monitor are graphically illustrated here. In addition, the Real-Time Monitor offers an extensive complement of nonresident system processors including Extended FORTRAN IV, a Symbolic Assembler, a Macro Assembler, a File Manager, a Link Editor, a System Editor, and a Debug Program.

PRIORITY INTERRUPT SYSTEM

Under the structure of the Real-Time Monitor, SYSTEMS 86 provides a multilevel priority interrupt system that the user can adjust according to his needs. The interrupt system includes both hardware and software priority levels.

Hardware priority levels include transfer and service interrupts, traps, and external interrupts. The Real-Time Monitor provides trap servicing and processing routines for the standard traps and the 16 device controller channels. The structure of the Real-Time Monitor also permits the user to add modular service routines to meet the unique requirements of his application. Transfer interrupts are serviced entirely by the Automatic Input Output System and therefore operate without software support.

To meet the immediate response requirements of time-critical situations, users can connect real-time programs to any of the external hardware interrupt levels. These interrupts minimize system response time by placing the total burden of resolving priority, task switching, and processing initiation on the SYSTEMS 86 hardware.

In addition to the hardware interrupts, the Real-Time Monitor provides 64 software priority levels for controlling the user's foreground programs, system programs, and background programs. Users can assign multiple programs to any priority level and thus achieve a high level of multiprogramming versatility.

The software priority levels are used at allocation time for peripheral and core allocation, at I/O time for device control, and at dispatch time for CPU control.

MONITOR EXECUTIVE

The Monitor Executive is the nucleus of the Real-Time Monitor. It schedules CPU control for all programs running under the 64 software priority levels. When an interrupt occurs indicating a change in system environment (e.g., completion of a disc

transfer), the Monitor Executive allocates CPU time to the highest priority program requesting service.

In making the transition from one program to another, the Monitor Executive performs such "housekeeping" operations as register push-down, register pop-up, setting the dedicated system index registers, saving the program status word, and setting the required memory protect registers.

CORE ALLOCATION

The Real-Time Monitor is completely dynamic in its allocation of core memory. Thus the user need not define a fixed partitioned area of core for each program. Instead, at system generation time, the user establishes the maximum amount of core that can be allocated to background programs. All remaining core not dedicated to other functions becomes the foreground operating area.

Many programs can reside in the foreground area of core. But in cases where a foreground program requires more memory than is available, the Real-Time Monitor rolls out the background core content to the disc and reassigns the background core to the foreground program. When the foreground program no longer requires the borrowed portion of core, the Real-Time Monitor rolls in the back-

ground core content to its original location in memory.

FILE MANAGEMENT

File management is extremely critical in the operating environment of SYSTEMS 86 because the Real-Time Monitor exploits the files in many ways. Permanent files are created for user programs, user data, and system data. Temporary files provide system scratch storage, user scratch storage, and system output data storage for devices like the system printer and card punch.

The file management system for the Real-Time Monitor consists of the resident File System Executive and the nonresident File Manager. Together they supervise all file space in core, on the fixed head discs, and on the movable head discs.

Operating as a reentrant, foreground service, the File System Executive allocates and deallocates temporary file space, retrieves permanent file space definition, and updates permanent file space definition. The File Manager, on the other hand, operates in the background to maintain permanent files. Under control of a set of directives, it creates, deletes, saves, and restores files.

The file management system gives users the fastest possible access to any permanent file. The key to this fast access is the use of a unique mapping scheme. Each permanent file has a directory entry consisting of the space definition for the file. Each directory entry, in turn, is referenced directly by a mapping algorithm that computes an address unique to the file name.

Unlike permanent files, temporary files have no directory entries. But their space is allocated dynamically. As a result, the user can readily expand his allocated file space as his file requirements increase.

CATALOGING

By exercising the facilities of the System Cataloger, users of SYSTEMS 86 can build custom real-time



and batch processing capabilities on line. During cataloging, the system stores programs on the disc in a system loadable format. The program can be called by the system at any time. Through permanent file save and restore functions, the cataloged program is preserved. Thus the user need not re-enter each cataloged program each time he initializes the system.

The user can catalog a program as a main program segment and overlay segments. Thus, a large program can be organized into several smaller sections. The smaller sections can then be sequentially overlaid in core during execution.

Programs that can be cataloged include real-time tasks and batch programs. Parameters such as the program's priority level, whether it is a foreground or background program, and its privileged/non-privileged mode of operation can be cataloged along with the program.

By specifying whether tasks are privileged or non-privileged, users can dynamically control system security. Tasks designated to run in the privileged mode are free to execute any instruction in the SYSTEMS 86 instruction repertoire. They also have read/write access to all memory locations. Programs designated to run in the non-privileged mode, on the other hand, have limited access to memory and are prevented from executing any of the privileged instructions.

TIME SCHEDULING

Users of SYSTEMS 86 can take advantage of the Real-Time Monitor's Time Scheduler to initiate servicing of real-time foreground processes on a periodic basis. Events that can be time scheduled include program activation, program resumption, flag setting, and interrupt activation. Operating in the foreground, the Time Scheduler is activated at periodic intervals by the interrupt servicing routine for the Real-Time Clock. During system generation, the user assigns the priority level and timing period that precisely coincides with his application requirements.

MONITOR SERVICES

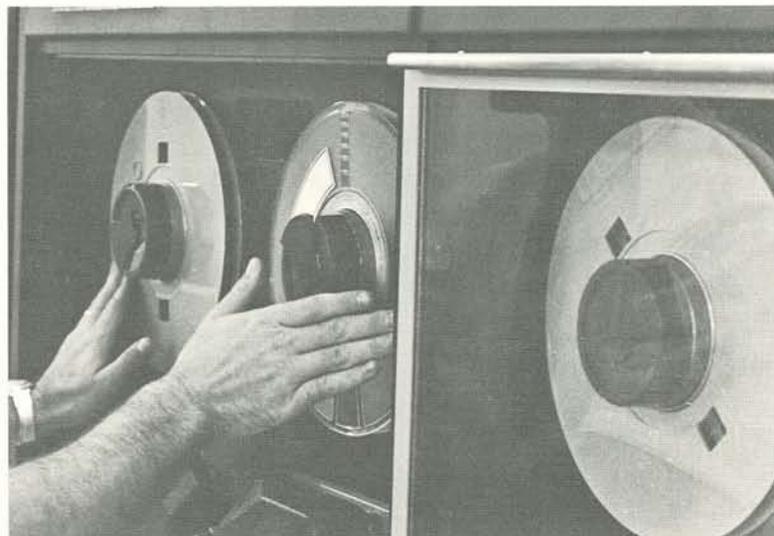
The Real-Time Monitor offers an extensive array of monitor service routines designed to perform many frequently required operations with maximum efficiency. Using the CALM instruction, both foreground and background programs can call these routines. They perform such operations as controlling program activation, requesting timed events, connecting programs and device handlers to interrupt levels, dynamically resetting the priority level of any program, loading segments and initiating their execution, interrogating various system parameters, and controlling various I/O functions.

All of the monitor service routines are reentrant. Thus, each service routine is always available to the currently active program.

More than thirty monitor service routines are provided as standard modular components of the Real-Time Monitor. The "open-ended" design of the Real-Time Monitor, however, gives each user complete freedom to add whatever service routines are required to tailor the Real-Time Monitor to his specific application.

INPUT/OUTPUT OPERATIONS

The Input/Output Supervisor (IOS) provides complete and easy-to-use I/O services that relieve the



programmer of many detailed chores. While keeping software overhead to an absolute minimum, IOS receives and processes all I/O requests from both user and system programs. It performs all logical error checking and parameter validation. IOS also logically processes termination of all I/O operations and assigns I/O control to the appropriate device handler. The device handler, in turn, executes the I/O data exchange, processes service interrupts, and performs device testing.

COMMUNICATIONS FACILITIES

It is well recognized that communications play an important part in the ability of the system to respond to dynamically changing environments. The Real-Time Monitor, therefore, offers complete facilities for conducting communications between individual users, between internal system elements, and between the system and the user.

Users communicate with one another through the temporary and permanent files and via job status flags which can be set and interrogated by monitor service routines.

Internal system elements communicate through temporary files, system queues, and the systems communications region. The system communications region occupies approximately 2K-words of

lower memory. It contains information common to all system modules and the system processors.

Communications between the user and the system are implemented through a selection of monitor services. In addition, the Real-Time Monitor contains an Operator's Communication Module which provides an on-line communications interface between the computer operator and the monitor. Using the services of this module, the computer operator can perform many operations, including activating and aborting programs.

BATCH PROCESSING

Through the statements included in the job control language, users direct all batch processing operations performed by the Real-Time Monitor. These statements define the programs to be executed within each job and describe each program's operating environment.

To simplify and accelerate the user's programming task, SYSTEMS programming specialists have designed the job control language to be exceptionally convenient and easy to use. Thus, users can take full advantage of the Real-Time Monitor's batch processing capabilities without acquiring a detailed knowledge of the system's characteristics.

Batch jobs enter the system through a local input device, such as a card reader or paper tape reader. Facilities are also available for entering jobs into the batch stream from remote data terminals. Under control of the System Input Module, the batch jobs are buffered on the disc. When a complete job is available on the disc, the Job Control Processor takes over under the supervision of the job control language statements. Peripherals are assigned, core is allocated, and finally the System Loader loads the program into memory.

The Executive dispatches CPU control to the program according to the program's priority level. Once the program gains control of the CPU, it remains active until either a higher priority program requests CPU facilities or until the program itself voluntarily



releases control.

When the program terminates, termination procedures automatically remove the peripheral assignments, deallocate core, purge the I/O buffers, and initiate system output.

DEBUGGING CAPABILITIES

Operating in the background, the Debug processor aids the user in verifying and debugging his application programs.

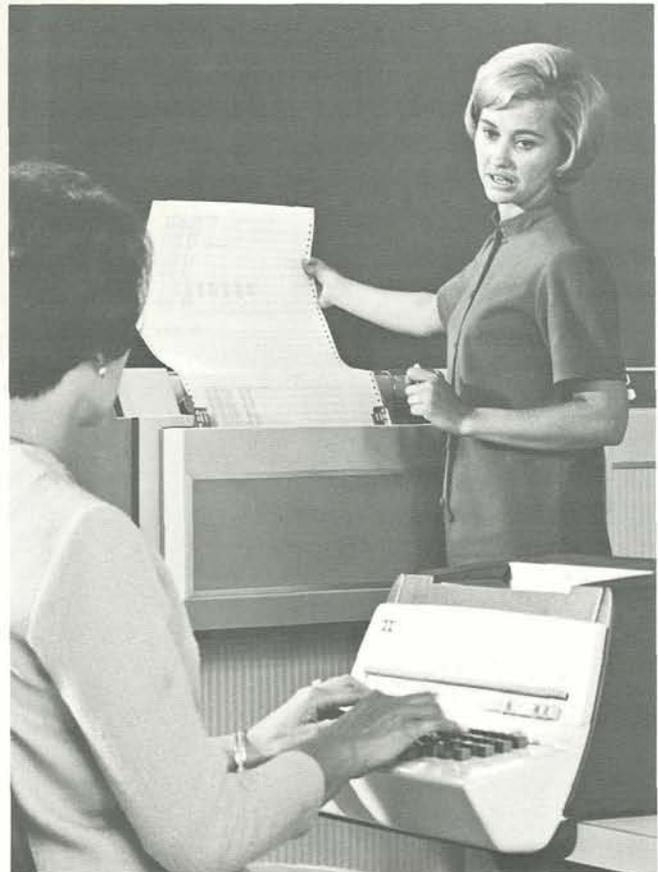
In the process of debugging, users can perform such operations as tracing programs, making snapshot displays of registers and memory, modifying the content of registers and memory, dynamically assigning files, and loading programs.

Users can control debugging on-line from the system teletypewriter while using the line printer to record such large volumes of information as trace outputs and core dumps.

SYSTEM GENERATION

The System Generation Program (SYSGEN) allows the user to generate a Real-Time Monitor tailor-made to his installation and his requirements. Using a set of directives, the user simply describes his system and its operating environment. SYSGEN in turn configures the Real-Time Monitor to fit this description. The directives are logically classified into the following categories:

- **Hardware Configuration Control** specifies such hardware parameters as core size and device controller channel assignments.
- **System Master Directory/Permanent File Control** controls initialization of a portion of the System Master Directory and the restoration of permanent file content into specified mass storage devices.
- **System Master Directory/System Module Control** controls initialization of the system entries



in the System Master Directory and the loading of the System Master File into the primary storage device.

- **Software Configuration Control** specifies the software modules to be used and the files that provide system storage.
- **System Parameter Control** specifies various system parameters for controlling table lengths, queue lengths, priority levels, etc.
- **System Module Load Control** facilitates debugging by allowing system modules loaded directly from the SYSGEN input device to override modules with the same name that were loaded from the System Master File.

Extended FORTRAN IV

Extended FORTRAN IV gives users of SYSTEMS 86 an entirely new level of capability for compiling and writing programs in the procedure-oriented language of FORTRAN.

Because Extended FORTRAN IV is procedure oriented rather than machine oriented, programmers can forget about the intricacies of the system and concentrate on creating programs to solve their application problems. Extended FORTRAN IV makes the programmer's job easier. Thus, it reduces the time and cost involved in preparing application programs.

Extended FORTRAN IV can operate in a SYSTEMS 86 hardware configuration with storage facilities consisting of only 16K of core supplemented by a disc file. It performs with equal efficiency under both the Real-Time Monitor and the Batch Processing System. The compiler is self-initializing. In three passes, it transforms source text into a highly optimized binary object program that's ready for execution by SYSTEMS 86. Because each pass of the compiler is a separately loadable program segment, core residency requirements are minimized and optimization is maximized.

The design of Extended FORTRAN IV is based on the specifications of the American National Standards Institute.* But to provide users of SYSTEMS 86 with a version of FORTRAN IV that's precisely tailored to their needs, we've added a number of extensions to the prescribed standards. These extensions dramatically enhance the operational characteristics of the system.

As a result, Extended FORTRAN IV is easier to use. The processor size is reduced while its efficiency is increased. In addition, programming time and program run time are kept to an absolute minimum.

Perhaps the most significant advantage of Extended FORTRAN IV is that it optimizes overall system performance by capitalizing on the unprecedented capabilities of the SYSTEMS 86 hardware.

The more powerful extensions to the standard version of FORTRAN IV which are included in

SYSTEMS Extended FORTRAN IV are:

- **In-line symbolic coding** allows the user the flexibility to include the full set of assembly-language source statements within the main body of his program.
- **Mixed-mode arithmetic expressions** provide the user with the capability of mixing various types of variables within the same arithmetic expression.
- **Array extensions** provide the user with the freedom of using any arithmetic expression as an array subscript.
- **Real-time capabilities** include facilities for interrupt utilization, as well as scheduling and testing input/output device assignments and status.
- **Asynchronous I/O** allows concurrent use of central-processor time while data is being transferred to and from memory.
- **Multiple entry and return** reduces core requirements by allowing the generation of one subroutine with various entry and return points.
- **ENCODE and DECODE features** provide a means of memory-to-memory data conversion in user-supplied internal buffer areas.
- **Bit and character types** provide for unique identification of bit and/or byte variables for maximum utilization of memory and the SYSTEMS 86 instruction set.

The code optimization capabilities of Extended FORTRAN IV significantly reduce the size of the

*Specifications published in X3.9 - 1966 by the American National Standards Institute.

compiled program. As a result, programs operate in less core and are executed faster.

Two passes of the compiler are used to ensure a high level of object code optimization. During the first pass, the compiler optimizes each statement in the source program on a line-by-line basis. During the second pass, it performs cross-statement opti-

mization between active labels. This eliminates redundant expressions and enables the program to make maximum use of system resources.

Extended FORTRAN IV offers the flexibility required to operate in changing environments. Data storage areas, for example, are designed to expand dynamically, thereby permitting maximum use of

$$\begin{array}{l}
 a_1 - \text{Min}(a_1, a_2) \\
 \cos^{-1}(a) \\
 \log_e(a) \\
 \sum_{i=1}^{100} A_i \\
 a_1 + a_2 \sqrt{-1} \\
 C_k = \sum_{j=1}^{15} A_{kj} B_j, \quad k=1, 2, \dots, 15 \\
 \log_{10}(a) \\
 \sqrt{a} \\
 \text{Sign of } a_2 \\
 \text{times } |a_1|
 \end{array}$$

all available core space. In place of determining the size of available core through a preset constant assembled into the complex, Extended FORTRAN IV uses a parameter available at the time of compilation.

A significant advantage of Extended FORTRAN IV is that it will produce compiler-generated, in-line coding for a specified subset of the intrinsic functions. The subset consists of the following routines:

ABS	DIM	CMPLX
IABS	IDIM	DCMPLX
DABS	REAL	CONJG
SIGN	AIMAG	DCONJC
ISIGN	DREAL	DBLE
DSIGN	DIMAG	SNGL

Extended FORTRAN IV offers comprehensive source language diagnostics that give programmers maximum assistance in eliminating all errors from their programs. Errors within statements are classified as either terminal or nonterminal so that the programmer can readily determine if additional errors exist within a statement. Compilation of the program always continues to completion to provide maximum diagnostic information.

As with all SYSTEMS 86 software, Extended FORTRAN IV comes complete with reference and maintenance documentation that fully describes the use, operation, and maintenance of the system.

Users of Extended FORTRAN IV can also exploit the capabilities of the extensive mathematical subroutine library. The Math Library greatly enhances the usability of SYSTEMS 86 software. In addition to supporting Extended FORTRAN IV, it supports assembly language programs and is, itself, coded in assembly language to provide maximum efficiency and speed. The Math Library, includes the full set of over 80 subroutines for single- and double-precision, fixed- and floating-point, as well as complex and double-precision complex calculations. A special package for matrix manipulation is also provided.

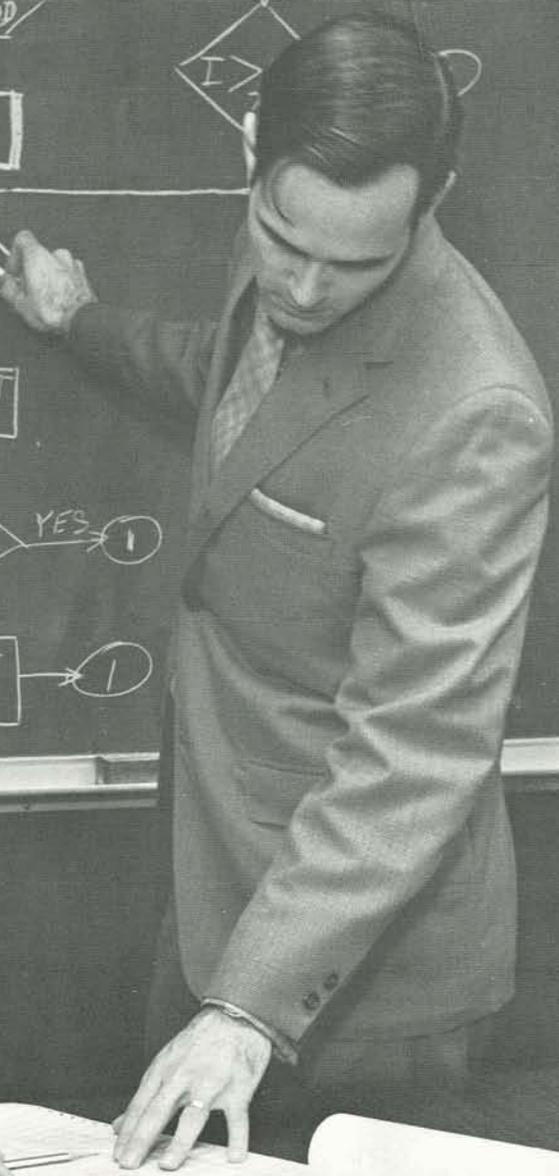
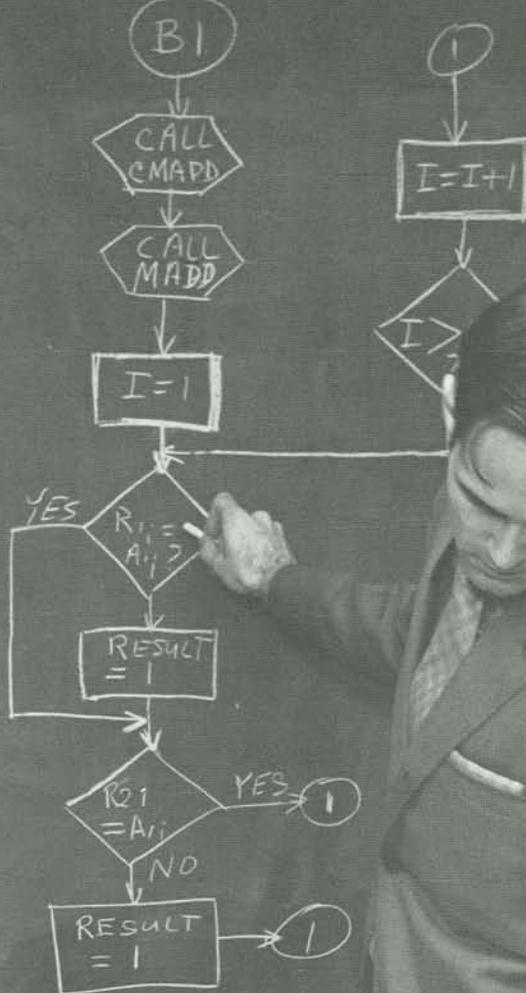


DIVIDE NUMBER BY 10
SAVING QUOTIENT AS
NEW NUMBER

CONVERT REMAINDER
TO ASCII DIGIT

PLACE DIGIT INTO
RESULT REGISTERS

CONBAD 1
SSU 1 THIS
PAS



Assemblers

Users of SYSTEMS 86 can code and assemble their real-time application programs quickly, easily, and efficiently using either the Symbolic Assembler or the powerful Macro Assembler.

Both assemblers give users of SYSTEMS 86 the full flexibility of machine-language programming, but without the inherent disadvantages of machine language. In place of using the unwieldy 32-bit binary language of the computer, users represent memory addresses, actual values, and machine components such as registers with easy-to-use symbols. All machine instructions are represented by mnemonic codes.

Because the language of the assemblers closely adheres to the actual machine operations, it is ideal for programming time-critical processes and other such applications that require the system to respond to a precise interpretation of the user's program.

The Macro Assembler is a powerful processor which permits users to nest macros, execute recursive macro calls, and pass parameters on to nested macros. Even though the Macro Assembler is a two-pass processor, it maintains an extremely fast internal assembly speed of 6000 source statements per minute.

The macro instruction capability allows users to combine into a single macro instruction all machine instructions needed to handle any specific routine procedure. To reuse the macro for subsequent runs, the user simply issues a macro instruction call that includes the associated parameters for the macro. The Macro Assembler, in turn, generates the "in-line" code for the specified macro instruction.

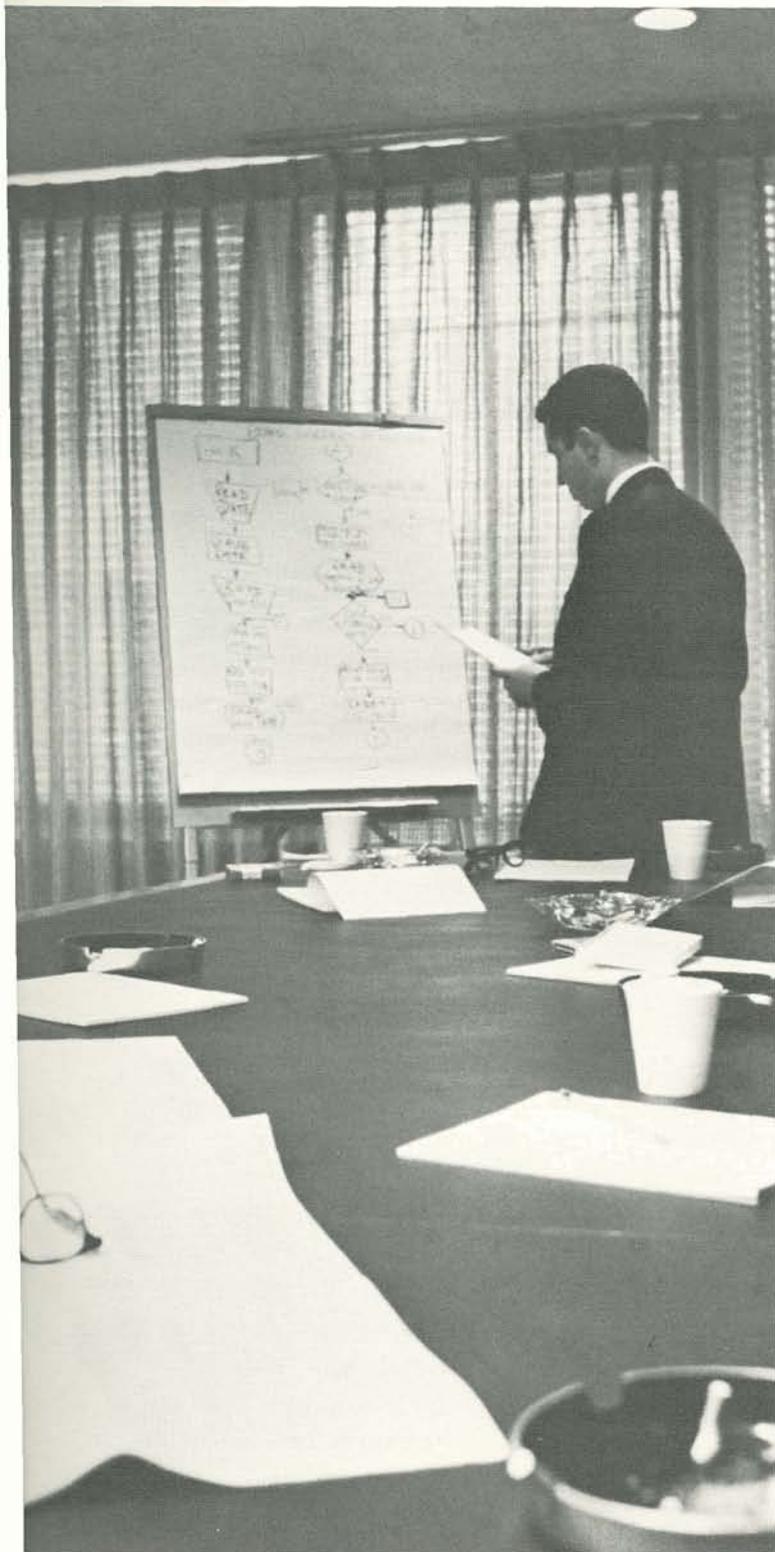
Because the Macro Assembler allows users to initiate specific processes without calling a subroutine or recording each section of the process, programming is easier, errors are reduced, and programs are executed faster.

Some of the other significant features offered by the Macro Assembler are summarized as follows:

- **Macro Library** gives all source language programs access to a common collection of macro prototypes. Once the user enters a macro prototype into the library, other source programs can use it just by including the corresponding macro call statement.
- **Extensive Macro Directives** allow users to easily define macro body statements and conditional assembly statements.
- **Generated Statement Variations** allow conditional assembly directives to vary the number and format of generated statements within a macro.
- **Internal Label Generation** provides a unique label whenever a dummy parameter (i.e., symbolic character string) is not assigned an actual value by the macro call statement.
- **Concatenation** allows users to combine dummy parameters with character strings and other dummy parameters within the label field, the operation field, and the operand field of any macro body statement.

The Symbolic Assembler is a subset of the Macro Assembler. It is designed specifically for users who have minimum core requirements and do not require macro capability. The internal assembly speed of the Symbolic Assembler exceeds 10,000 source statements per minute.

Both assemblers run under either the Batch Processing System or the Real-Time Monitor and are truly I/O independent. This enables the user to assign the input or output to any available medium. One unique feature of both assemblers is their ability to define FORTRAN COMMON blocks. This makes it easy to transfer data among FORTRAN programs and symbolically assembled subroutines.

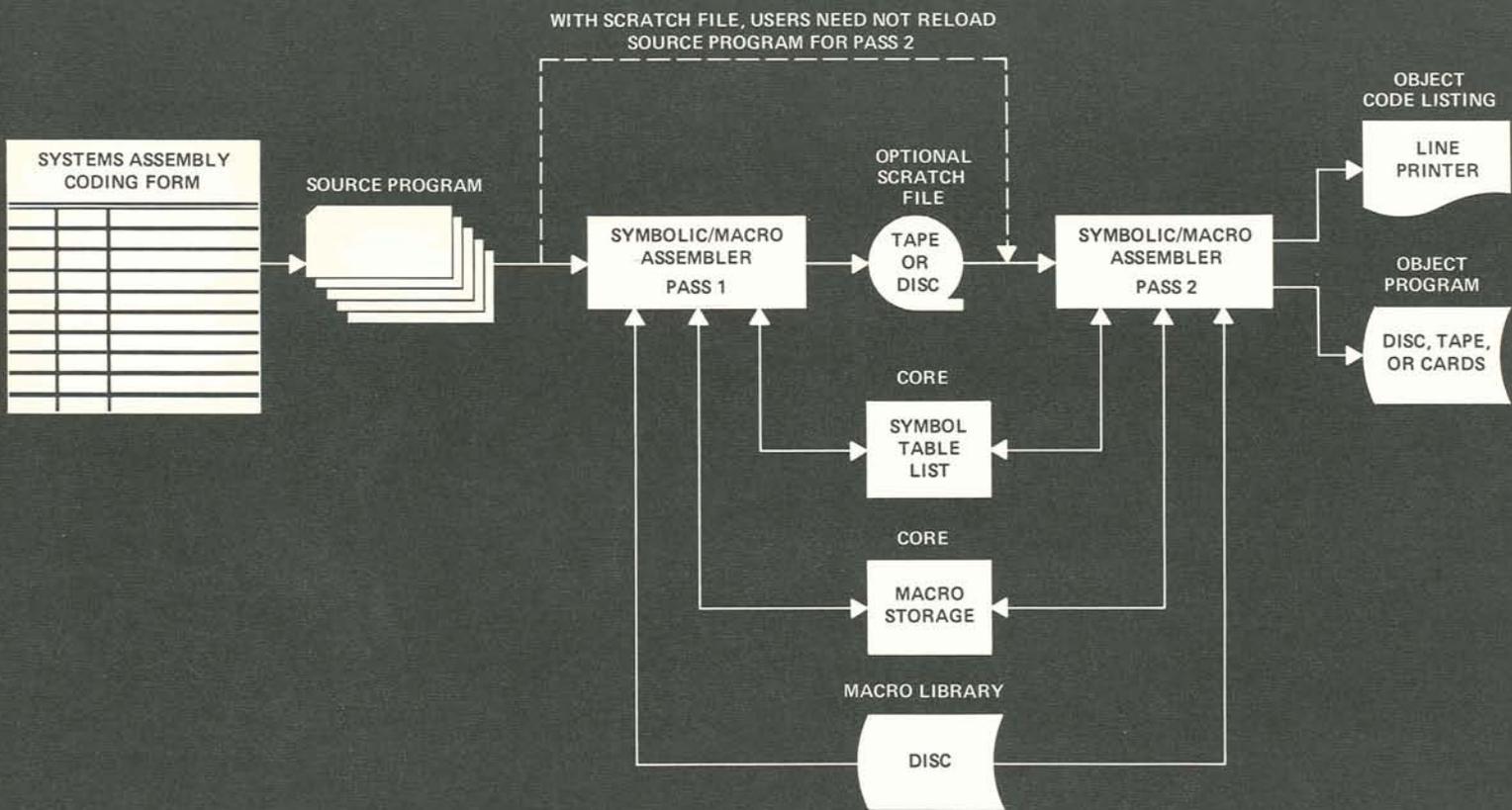


With either assembler, users can arbitrarily group programs into sections. Individual sections, for example, can be written independently for such elements as data, subroutines, and even the main program. At the user's discretion, program sections can be designated as COMMON. Programs and program sections can be assigned absolute addresses or they can be relocatable.

Relocatable sections can be dynamically reassigned to any suitable area of memory. For added efficiency, users can place program sections in a library where they can be readily accessed by other programs. Through the linking facilities of the assemblers, users can define symbols in one assembly and refer to them through other independently assembled programs.

The assemblers permit users to easily modify their programs because all address changes are made in symbolic form. When symbolic address changes are made, the assemblers automatically reassign memory addresses and provide the user with an updated listing of all program modifications and new memory addresses.

During program assembly, the assemblers analyze source programs for both actual and potential errors. When the assemblers detect such errors, they place flags in the object listing at all points where errors exist. By including a suitable listing device in the system, users can also obtain a printout of all major errors such as illegal operators.



DATA FLOW FOR
SYSTEMS 86 SYMBOLIC ASSEMBLER AND MACRO ASSEMBLER

SYSTEMS



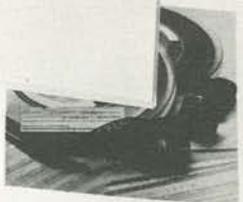
Reference Manual
SYSTEMS 86 Computer

SYSTEMS



Programming Reference Manual
Assembler/Macro-Assembler

SYSTEMS



Programming Reference Manual
Batch Processing System

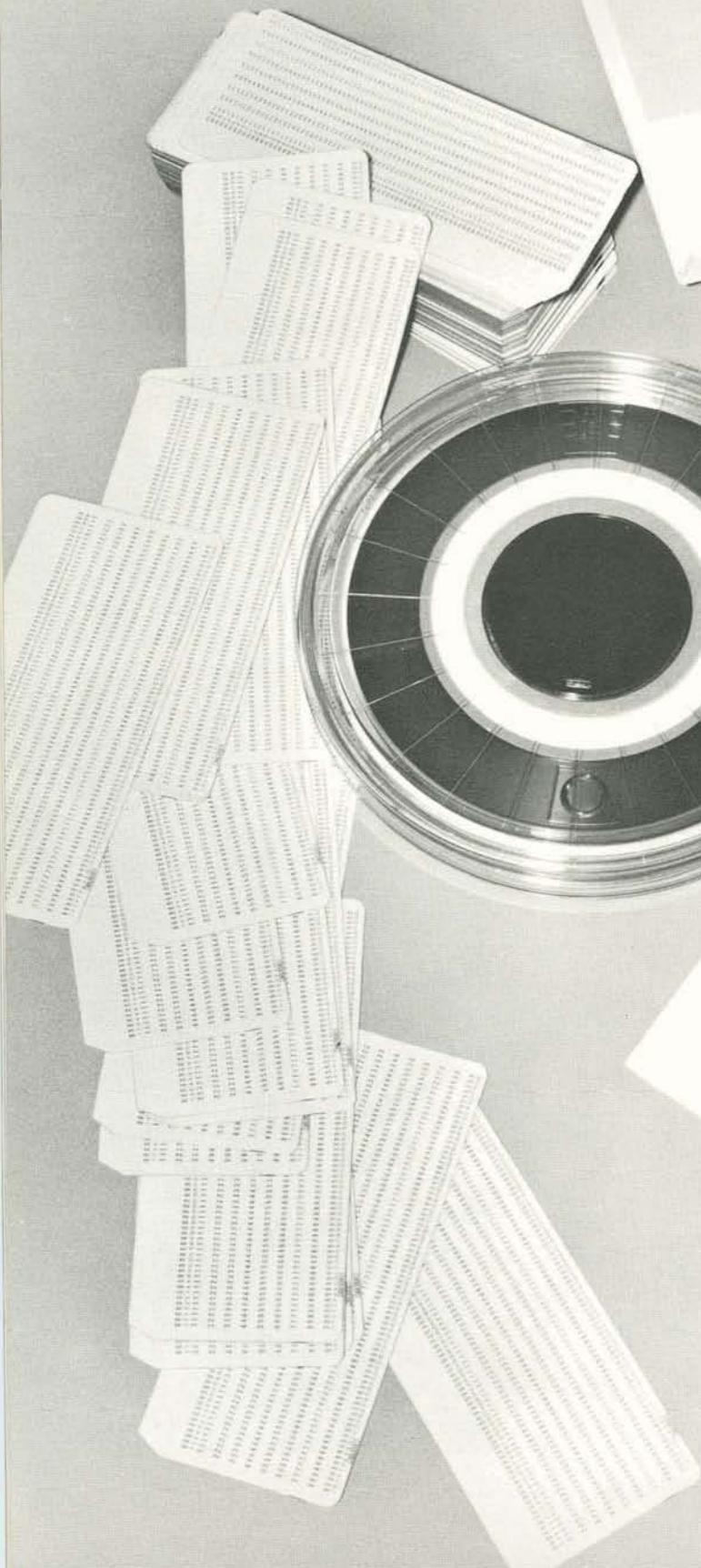
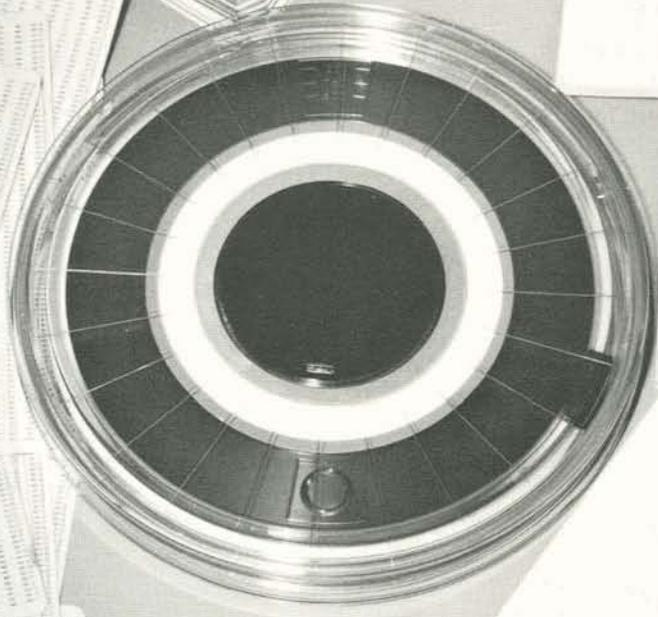
SYSTEMS



Programming Reference Manual
SYSTEMS FORTRAN IV

Programming Reference Manual
BASIC

Programming Reference Manual
Diagnostics



Other SYSTEMS 86 Software

Because we want our customers to get every bit of performance their SYSTEMS 86 is capable of delivering, we also offer a BASIC compiler and a complete array of hardware diagnostics. COBOL is planned for the future.

In addition, our experience in designing custom systems enables us to personalize software systems to meet your individual needs. You specify the requirements and we'll tailor a software package to do your job efficiently and economically.

BASIC

BASIC, which is currently under development, is a multi-user, terminal-oriented programming system that can operate under either the Batch Operating System or the Real-Time Monitor. It permits users to communicate with SYSTEMS 86 using simple, concise statements. Because of its simplicity it is intended primarily for use by non-programmers.

The user enters BASIC programs into SYSTEMS 86 simply by typing them at the teletypewriter or similar remote-terminal device. BASIC programs consist of a group of numbered statements that resemble ordinary mathematical notation. The computer responds in the same type of easy-to-understand statements.

BASIC for SYSTEMS 86 also provides for command editing, program listing compilation, execution, program storage, program retrieval, matrix computations, and string manipulation.

COBOL

SYSTEMS 86 is just as adept at solving your business-oriented problems as it is at handling your simulation, test, and control tasks. We therefore

plan to provide a COBOL compiler. Full specifications and operating information will be released in the future.

DIAGNOSTICS

To keep system down-time to an absolute minimum, SYSTEMS 86 software includes diagnostic routines that allow users to rapidly isolate malfunctions in all major hardware elements and peripherals.

For complete checkout of the mainframe, the diagnostic runs a two part testing sequence. During the first part, it subjects SYSTEMS 86 to a rigorous series of tests that progressively become more demanding. The second part of the mainframe diagnostic thoroughly tests the arithmetic capabilities of the system.

Diagnostics for the peripheral devices exercise the equipment with the same degree of thoroughness as provided by the mainframe diagnostic. Each diagnostic ascertains the ability of the associated peripheral device to respond to various input/output patterns and formats. The user, in turn, receives a report of all errors.

SOFTWARE TO HANDLE THE TOTAL PROBLEM

SYSTEMS 86 software covers every aspect of the problem — whether it be real-time data acquisition and control in a multiprogramming environment, data processing, scientific computations, statistical studies, multi-user, terminal-oriented programming, or any combination of these. And, because the software is totally integrated with the SYSTEMS 86 hardware, it handles your complete workload faster, more thoroughly, and more economically.

