Chapter 36

D825—a multiple-computer system for command and control[1]

James P. Anderson / Samuel A. Hoffman
Joseph Shifman / Robert J. Williams

Introduction

The D825 Modular Data Processing System is the result of a Burroughs study, initiated several years ago, of the data processing requirements for command and control systems. The D825 has been developed for operation in the military environment. The initial system, constructed for the Naval Research Laboratory with the designation AN/GYK-3(V), has been completed and tested. This paper reviews the design criteria analysis and design rationale that led to the system structure of the D825. The implementation and operation of the system are also described. Of particular interest is the role that developed for an operating system program in coordinating the system components.

Functional requirements of command and control data processing

By "command and control system" is meant a system having the capacity to monitor and direct all aspects of the operation of a large man and machine complex. Until now, the term has been applied exclusively to certain military complexes, but could as well be applied to a fully integrated air traffic control system or even to the operation of a large industrial complex. Operation of command and control systems is characterized by an enormous quantity of diverse but interrelated tasks—generally arising in real time—which are best performed by automatic data-processing equipment, and are most effectively controlled in a fully integrated central data processing facility. The data processing functions alluded to are those typical of data processing, plus special functions associated with servicing displays, responding to manual insertion (through consoles) of data, and dealing with communications facilities. The design implications of these functions will be considered here.

Availability criteria. The primary requirement of the data-processing facility, above all else, is availability. This requirement, essentially a function of hardware reliability and maintainability, is, to the user, simply the percentage of available, on-line, operation time during a given time period. Every system designer must trade off the costs of designing for reliability against those incurred by unavailability, but in no other application are the costs of unavailability so high as those presented in command and control. Not only is the requirement for hardware reliability greater than that of commercial systems, but downtime for the complete system for preventive maintenance cannot be permitted. Depending upon the application, some greater or lesser portion of the complete system must *always* be available for primary system functions, and *all* of the system must be available *most* of the time.

The data processing facility may also be called upon, except at the most critical times, to take part in exercising and evaluating the operation of some parts of the system, or, in fact, in actual simulation of system functions. During such exercises and simulations, the system must maintain some (although perhaps partially and temporarily degraded) real-life and real-time capability, and must be able to return quickly to full operation. An implication here, of profound significance in system design, is, again, the requirement that *most* of the system be *always* available; there must be no system elements (unsupported by alternates) performing functions so critical that failure at these points could compromise the primary system functions.

Adaptability criteria. Another requirement, equally difficult to achieve, is that the computer system must be able to analyze the demands being made upon it at any given time, and determine from this analysis the attention and emphasis that should be given to the individual tasks of the problem mix presented. The working configuration of the system must be completely adaptable so as to accommodate the diverse problem mixes, and, moreover, must respond quickly to important changes, such as might be indicated by external alarms or the results of internal computations (exceeding of certain thresholds, for example), or to changes in the hardware configuration resulting from the failure of a system component or from its intentional removal from the system. The system

[1] *AFIPS Proc. FJCC,* vol. 22, pp. 86–96, 1962.

must have the ability to be dynamically and automatically restructured to a working configuration that is responsive to the problem-mix environment.

Expansibility criteria. The requirement of expansibility is not unique to command and control, but is a desirable feature in any application of data processing equipment. However, the need for expansibility is more acute in command and control because of the dependence of much of the efficacy of the system upon an ability to meet the changing requirements brought on by the very rapidly changing technology of warfare. Further, it must be possible to incorporate new functions in such a way that little or no transitional downtime results in any hardware area.

Expansion should be possible without incurring the costs of providing more capability than is needed at the time. This ability of the system to grow to meet demands should apply not only to the conventionally expansible areas of memory and I/O but to computational devices, as well.

Programming criteria. Expansion of the data-processing facility should require no reprogramming of old functions, and programs for new functions should be easily incorporated into the overall system. To achieve this capability, programs must be written in a manner which is independent of system configuration or problem mix, and should even be interchangeable between sites performing like tasks in different geographic locales. Finally, because of the large volume of routines that must be written for a command and control system, it should be possible for many different people, in different locations and of different areas of responsibility, to write portions of programs, and for the programs to be subsequently linked together by a suitable operating system.

Concomitant with the latter requirement and with that of configuration-independent programs is the desirability of orienting system design and operation toward the use of a high-level procedure-oriented language. The language should have the features of the usual algorithmic languages for scientific computations, but should also include provisions for maintaining large files of data sets which may, in fact, be ill-structured. It is also desirable that the language reflect the special nature of the application; this is especially true when the language is used to direct the storage and retrieval of data.

Design rationale for the data-processing facility

The three requirements of availability, adaptability, and expansibility were the motivating considerations in developing the D825 design. In arriving at the final systems design, several existing and proposed schemes for the organization of data processing systems were evaluated in light of the requirements listed above. Many of the same conclusions regarding these and other schemes in the use of computers in command and control were reached independently in a more recent study conducted for the Department of Defense by the Institute for Defense Analysis [Kroger et al., 1961].

The single-computer system. The most obvious system scheme, and the least acceptable for command and control, is the single-computer system. This scheme fails to meet the availability requirement simply because the failure of any part—computer, memory, or I/O control—disables the entire system. Such a system was not given serious consideration.

Replicated single-computer systems. A system organization that had been well known at the time these considerations were active involves the duplication (or triplication, etc.) of single-computer systems to obtain availability and greater processing rates. This approach appears initially attractive, inasmuch as programs for the application may be split among two or more independent single-computer systems, using as many such systems as needed to perform all of the required computation. Even the availability requirement seems satisfied, since a redundant system may be kept in idle reserve as backup for the main function.

On closer examination, however, it was perceived that such a system had many disadvantages for command and control applications. Besides requiring considerable human effort to coordinate the operation of the systems, and considerable waste of available machine time, the replicated single computers were found to be ineffective because of the highly interrelated way in which data and programs are frequently used in command and control applications. Further, the steps necessary to have the redundant or backup system take over the main function, should the need arise, would prove too cumbersome, particularly in a time-critical application where constant monitoring of events is required.

Partially shared memory schemes. It was seen that if the replicated computer scheme were to be modified by the use of partially shared memory, some important new capabilities would arise. A partially shared memory can take several forms, but provides principally for some shared storage and some storage privately allotted to individual computers. The shared storage may be of any kind—tapes, discs, or core—but frequently is core. Such a system, by providing a direct path of communication between computers, goes a long way toward satisfying the requirements listed above.

The one advantage to be found in having some memory private to each computer is that of data protection. This advantage vanishes when it is necessary to exchange data between computers, for if a computer failure were to occur, the contents of the private memory of that computer would be lost to the system. Furthermore, many tasks in the command and control application require access to the same data. If, for example, it would be desirable to permit some privately stored data to be made available to the fully shared memory or to some other private memory, considerable time would be lost in transferring the data. It is also clear that a certain amount of utilization efficiency is lost, since some private memory may be unused, while another computer may require more memory than is directly available, and may be forced to transfer other blocks of data back to bulk storage to make way for the necessary storage. It might be added in passing that if private I/O complements are considered, the same questions of decreased overall availability and decreased efficiency arise.

Master/slave schemes. Another aspect of the partially shared memory system is that of control. A number of such systems employ a *master/slave* scheme to achieve control, a technique wherein one computer, designated the master computer, coordinates the work done by the others. The master computer might be of a different character than the others, as in the PILOT system, developed by the National Bureau of Standards [Leiner et al., 1957], or it may be of the same basic design, differing only in its prescribed role, as in the Thompson Ramo Wooldridge TRW400 (AN/FSQ-27) [Porter, 1960]. Such a scheme does recognize the importance, for multicomputer systems, of the problem of coordinating the processing effort; the master computer is an effective means of accomplishing the coordination. However, there are several difficulties in such a design. The loss of the master computer would down the whole system, and the command and control availability requirement could not, consequently, be met. If this weakness is countered by providing the ability for the master control function to be automatically switched to another processor, there still remains an inherent inefficiency. If, for example, the workload of the master computer becomes very large, the master becomes a system bottleneck resulting in inefficient use of all other system elements; and, on the other hand, if the workload fails to keep the master busy, a waste of computing power results. The conclusion is then reached that a master should be established only when needed; this is what has been done in the design of the D825.

The totally modular scheme. As a result of these analyses, certain implications became clear. The availability requirement dictated a decentralization of the computing function—that is, a multiplicity of computing units. However, the nature of the problem required that data be freely communicable among these several computers. It was decided, therefore, that the memory system would be completely shared by all processors. And, from the point of view of availability and efficiency, it was also seen to be undesirable to associate I/O with a particular computer; the I/O control was, therefore, also decoupled from the computers.

Furthermore, a system with several computers, totally shared memory, and decoupled I/O seemed a perfect structure for satisfying the adaptability requirements of command and control. Such a structure resulted in a flexibility of control which was a fine match for the dynamic, highly variable, processing requirements to be encountered.

The major problem remaining to realize the computational potential represented by such a system was, of course, that of coordinating the many system elements to behave, at any given time, like a system specifically designed to handle the set of tasks with which it was faced at that time. Because of the limitations of previously available equipment, an operating system program had always been identified with the equipment running the program. However, in the proposed design, the entire memory was to be directly accessible to all computer modules, and the operating system could, therefore, be decoupled from any specific computer. The operation of the system could be coordinated by having any processor in the complement run the operating system only as the need arose. It became clear that the master computer had actually become a program stored in totally shared memory, a transformation which was also seen to offer enhanced programming flexibility.

Up to this point, the need for identical computer modules had not been established. The equality of responsibility among computing units, which allowed each computer to perform as the master when running the operating system, led finally to the design specification of identical computer modules. These were freely interconnected to a set of identical memory modules and a set of identical I/O control modules, the latter, in turn, freely interconnected to a highly variable and diverse I/O device complement. It was clear that the complete modularity of system elements was an effective solution to the problem of expansibility, inasmuch as expansion could be accomplished simply by adding modules identical to those in the existing complement. It was also clear that important advantages and economies resulting from the manufacture, maintenance, and spare parts provisioning for identical modules also accrue to such a system. Perhaps the most important result of a totally modular organization is that redun-

dancy of the required complement of any module type, for greater reliability, is easily achieved by incorporating as little as one additional module of that type in the system. Furthermore, the additional module of each type need not be idle; the system may be looked upon as operating with active spares.

Thus, a design structure based upon complete modularity was set. Two items remained to weld the various functional modules into a coordinated system—a device to electronically interconnect the modules, and an operating system program with the effect of a master computer, to coordinate the activities of the modules into fully integrated system operation.

In the D825, these two tasks are carried out by the *switching interlock* and the *Automatic Operating and Scheduling Program* (AOSP), respectively. Figure 1 shows how the various functional modules are interconnected via the interlock in a matrix-like fashion.

System implementation

Most important in the design implementation of the D825 were studies toward practical realization of the switching interlock and the AOSP. The computer, memory, and I/O control modules permitted more conventional solutions, but were each to incorporate some unusual features, while many of the I/O devices were selected from existing equipment. With the exception of the latter, all of theses elements are discussed here briefly. (A summary of D825 characteristics and specifications is included at the end of the paper.)

Switching interlock. Having determined that only a completely shared memory system would be adequate, it was necessary to find some way to permit access to any memory by any processor, and, in fact, to permit sharing of a memory module by two or more processors or I/O control modules.

A function distributed physically through all of the modules of a D825 system, but which has been designated in aggregate the switching interlock, effects electronically each of the many brief interconnections by which all information is transferred among computer, memory, and I/O control modules. In addition to the electronic switching function, the switching interlock has the ability to detect and resolve conflicts such as occur when two or more computer modules attempt access to the same memory module.

The switching interlock consists functionally of a crosspoint switch matrix which effects the actual switching of bus interconnections, and a bus allocator which resolves all time conflicts resulting from simultaneous requests for access to the same bus or system module. Conflicting requests are queued up according to the priority assigned to the requestors. Priorities are preemptive in that the appearance of a higher priority request will cause service of that request before service of a lower priority request already in the queue. Analyses of queueing probabilities have shown that queues longer than one are extremely unlikely.

The priority scheduling function is performed by the bus allocator, essentially a set of logical matrices. The conflict matrix detects the presence of conflicts in requests for interconnection. The priority matrix resolves the priority of each request. The logical product of the states of the conflict and priority matrices determines the state of the queue matrix, which in turn governs the setting of the crosspoint switch, unless the requested module is busy.

The AOSP: an operating system program. The AOSP is an operating system program stored in totally shared memory and therefore available to any computer. The program is run only as needed to exert control over the system. The AOSP includes its own executive routine, an operating system for an operating system, as it were, calling out additional routines, as required. The configuration of the AOSP thus permits variation from application to application, both in sequence and quantity of available routines and in disposition of AOSP storage.

The AOSP operates effectively on two levels, one for system control, the other for task processing.

The system control function embodies all that is necessary to call system programs and associated data from some location in the I/O complement, and to ready the programs for execution by finding and allocating space in memory, and initiating the processing. Most of the system control function (as well as the task processing function) consists of elaborate bookkeeping for: programs being run, programs that are active (that is, occupy memory space), I/O commands being executed, other I/O commands waiting, external data blocks to be received and decoded, and activation of the appropriate programs to handle such external data. It would be inappropriate here to discuss the myriad details of the AOSP; some idea of its scope, however, can be obtained from the following list of some of its major functions:

1 Configuration determination
2 Memory allocation
3 Scheduling
4 Program readying and end-of-job cleanup
5 Reporting and logging

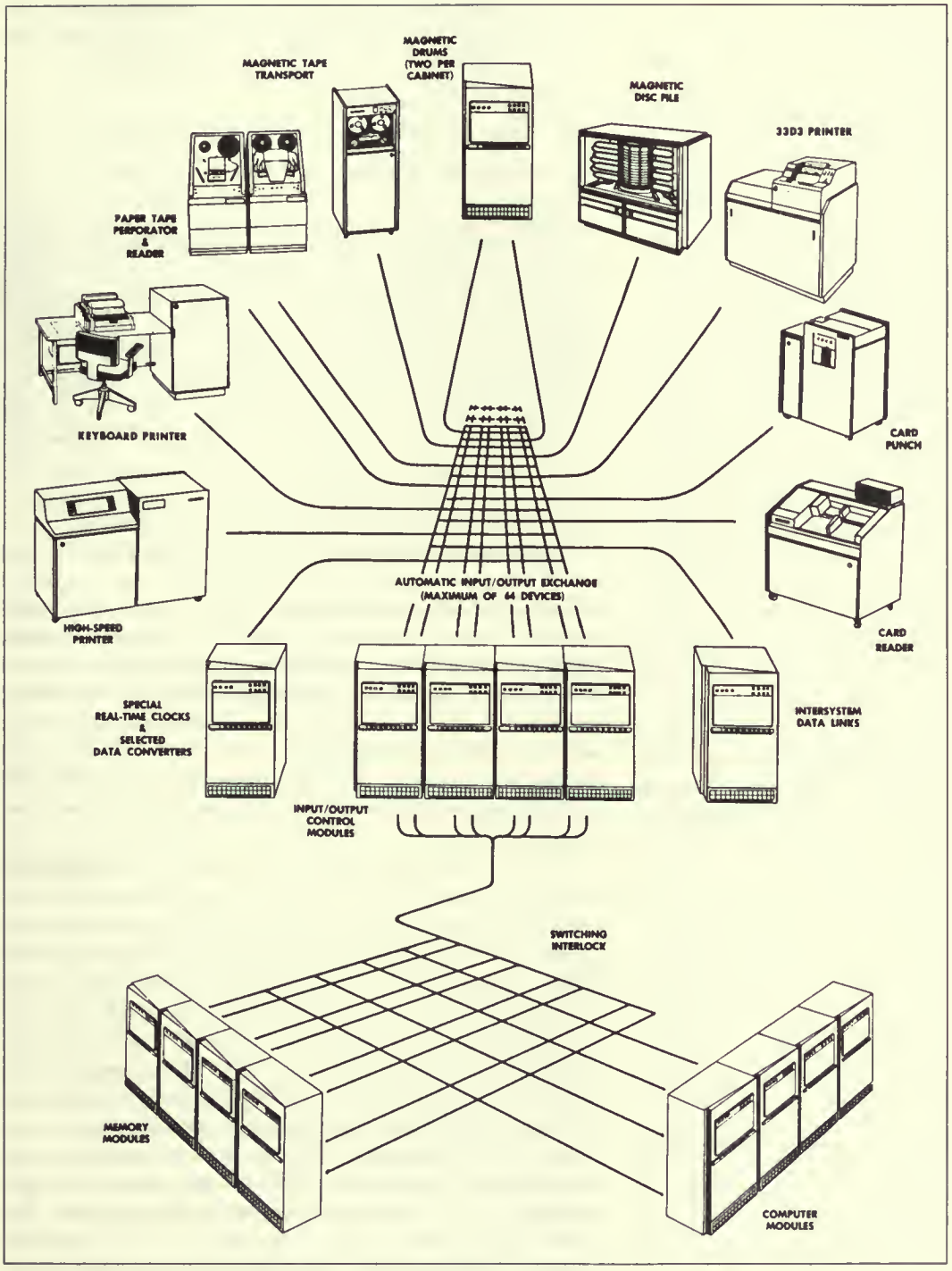

Fig. 1. System organization, Burroughs D825 modular data processing system.

6 Diagnostics and confidence checking

7 External interrupt processing

The task processing function of the AOSP is to execute all program I/O requests in order to centralize scheduling problems and to protect the system from the possibility of data destruction by ill-structured or conflicting programs.

AOSP response to interrupts. The AOSP function depends heavily upon the comprehensive set of interrupts incorporated in the D825. All interrupt conditions are transmitted to all computer modules in the system, and each computer module can respond to all interrupt conditions. However, to make it possible to distribute the responsibility for various interrupt conditions, both system and local, each computer module has an interrupt mask register that controls the setting of individual bits of the interrupt register. The occurrence of any interrupt causes one of the system computer modules to leave the program it has been running and branch to the suitable AOSP entry, entering a *control mode* as it branches. The control mode differs from the normal mode of operation in that it locks out the response to some low-priority interrupts (although recording them) and enables the execution of some additional instructions reserved for AOSP use (such as setting an interrupt mask register or memory protection registers, or transmitting an I/O instruction to an I/O control module).

In responding to an interrupt, the AOSP transfers control to the appropriate routine handling the condition designated by the interrupt. When the interrupt condition has been satisfied, control is returned to the original object program. Interrupts caused by normal operating conditions include:

1 16 different types of external requests

2 Completion of an I/O operation

3 Real-time clock overflow

4 Array data absent

5 Computer-to-computer interrupts

6 Control mode entry (normal mode halt)

Interrupts related to abnormalities of either program or equipment include:

1 Attempt by program to write out of bounds

2 Arithmetic overflow

3 Illegal instruction

4 Inability to access memory, or an internal parity error; parity error on an I/O operation causes termination of that operation with suitable indication to the AOSP

5 Primary power failure

6 Automatic restart after primary power failure

7 I/O termination other than normal completion

While the reasons for including most of the interrupts listed above are evident, a word of comment on some of them is in order.

The array-data-absent interrupt is initiated when a reference is made to data that is not present in the memory. Since all array references such as A[k] are made relative to the base (location of the first element) of the array, it is necessary to obtain this address and to index it by the value k. When the base of array A is fetched, hardware sensing of a presence bit either allows the operation to continue, or initiates the array-data-absent interrupt. In this way, keeping track of data in use by interacting programs can be simplified, as may the storage allocation problem.

The primary power failure interrupt is highest priority, and *always* pre-emptive. This interrupt causes all computer and I/O control modules to terminate operations, and to store all volatile information either in memory modules or in magnetic thin-film registers. (The latter are integral elements of computer modules.) This interrupt protects the system from transient power failure, and is initiated when the primary power source voltage drops below a predetermined limit.

The automatic restart after primary power failure interrupt is provided so that the previous state of the system can be reconstructed.

A description of how an external interrupt is handled might clarify the general interrupt procedure. Upon the presence of an external interrupt, the computer which has been assigned responsibility to handle such interrupts automatically stores the contents of those registers (such as the program counter) necessary to subsequently reconstitute its state, enters the control mode, and goes to a standard (hardware-determined) location where a branch to the external request routine is located. This routine has the responsibility of determining which external request line requires servicing, and, after consulting a table of external devices (teletype buffers, console keyboards, displays, etc.) associated with the interrupt lines, the computer constructs and transmits an input instruction to the requesting device for an initial message. The computer then makes an entry in the table of the I/O complete program (the program that handles I/O complete interrupts) to activate the appropriate responding routine when the message is

read in. A check is then made for the occurrence of additional external requests. Finally, the computer restores the saved register contents and returns in normal mode to the interrupted program.

AOSP control of I/O activity. As mentioned above, control of all I/O activity is also within the province of the AOSP. Records are kept on the condition and availability of each I/O device. The locations of all files within the computer system, whether on magnetic tape, drum, disc file, card, or represented as external inputs, are also recorded. A request for input by file name is evaluated, and, if the device associated with this name is readily available, the action is initiated. If for any reason the request must be deferred, it is placed in a program queue to await conditions which permit its initiation. Typical conditions which would cause deferral of an I/O operation include:

1 No available I/O control module or channel.

2 The device in which the file is located is presently in use.

3 The file does not exist in the system.

In the latter case, typically, a message would be typed out on the supervisory printer, asking for the missing file.

The I/O complete interrupt signals the completion of each I/O operation. Along with this interrupt, an I/O result descriptor is deposited in an AOSP table. The status relayed in this descriptor indicates whether or not the operation was successful. If not successful, what went wrong (such as a parity error, or tape break, card jams, etc.) is indicated so that the AOSP may initiate the appropriate action. If the operation was successful, any waiting I/O operations which can now proceed are initiated.

AOSP control of program scheduling. Scheduling in the D825 relies upon a job table maintained by the AOSP. Each entry is identified with a name, priority, precedence requirements, and equipment requirements. Priority may be dynamic, depending upon time, external requests, other programs, or a function of many variable conditions. Each time the AOSP is called upon to select a program to be run, whether as a result of the completion of a program or of some other interrupt condition, the job table is evaluated. In a real-time system, situations occur wherein there is no system program to be run, and machine time is available for other uses. This time could be used for auxiliary functions, such as confidence routines.

The AOSP provides the capability for program segmentation at the discretion of the programmer. Control macros embedded in the program code inform the AOSP that parallel processing with two or more computers is possible at a given point. In addition, the programmer must specify where the branches indicated in this manner will join following the parallel processing.

Computer module. The computer modules of the D825 system are identical, general-purpose, arithmetic and control units. In determining the internal structure of the computer modules, two considerations were uppermost. First, all programs and data had to be arbitrarily relocatable to simplify the storage allocation function of the AOSP; secondly, programs would not be modified during execution. The latter consideration was necessary to minimize the amount of work required to pre-empt a program, since all that would have to be saved to reinstate the interrupted program at a later time would be the data for that program and the register contents of the computer module running the program at the time it was dumped.

The D825 computer modules employ a variable-length instruction format made up of quarter-word syllables. Zero-, one-, two-, or three-address syllables, as required, can be associated with each basic command syllable. An implicitly addressed accumulator stack is used in conjunction with the arithmetic unit. Indexing of all addresses in a command is provided, as well as arbitrarily deep indirect addressing for data.

Each computer module includes a 128-position thin-film memory used for the stack, and also for many of the registers of the machine, such as the program base register, data base register, the index registers, limit registers, and the like.

The instruction complement of the D825 includes the usual fixed-point, floating-point, logical, and partial-field commands found in any reasonably large scientific data processor.

Memory module. The memory modules consist of independent units storing 4096 words, each of 48 bits. Each unit has an individual power supply and all of the necessary electronics to control the reading, writing, and transmission of data. The size of the memory modules was established as a compromise between a module size small enough to minimize conflicts wherein two or more computer or I/O modules attempt access to the same memory module, and a size large enough to keep the cost of duplicated power supplies and addressing logic within bounds. It might be noted that for a larger modular processor system, these trade-offs might indicate that memory modules of 8192 words would be more suitable. Modules larger than this—of 16,384 or 32,768 words, for example—would make construction of relatively small equipment complements meeting the requirements set forth above quite

difficult. The cost of smaller units of memory is offset by the lessening of catastrophe in the event of failure of a module.

I/O control module. The I/O control module executes I/O operations defined and initiated by computer module action. In keeping with the system objectives, I/O control modules are not assigned to any particular computer module, but rather are treated in much the same way as memory modules, with automatic resolution of conflicting attempted accesses via the switching interlock function. Once an I/O operation is initiated, it proceeds independently until completion.

I/O action is initiated by the execution of a transmit I/O instruction in one of the computer modules, which delivers an I/O descriptor word from the addressed memory location to an inactive I/O control module. The I/O descriptor is an instruction to the I/O control module that selects the device, determines the direction of data flow, the address of the first word, and the number of words to be transferred.

Interposed between the I/O control modules and the physical external devices is another crossbar switch designated the I/O exchange. This automatic exchange, similar in function to the switching interlock, permits two-way data flow between any I/O control module and any I/O device in the system. It further enhances the flexibility of the system by providing as many possible external data transfer paths as there are I/O control modules.

Equipment complements. A D825 system can be assembled (or expanded) by selection of appropriate modules in any combination of: one to four computer modules, one to 16 memory modules, one to ten I/O control modules, one or two I/O exchanges, and one to 64 I/O devices per I/O exchange in any combination selected from: operating (or system status) consoles, magnetic tape transports, magnetic drums, magnetic disc files, card punches and readers, paper tape perforators and readers, supervisory printers, high-speed line printers, selected data converters, special real-time clocks, and intersystem data links.

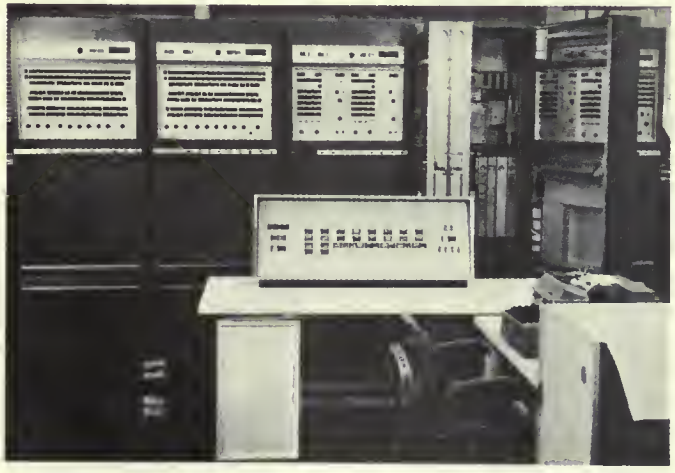

Fig. 2. Typical D825 equipment array.

Table 1 Specifications, D825 modular data processing system

Computer module:	4, maximum complement
Computer module, type:	Digital, binary, parallel, solid-state
Word length:	48 bits including sign (8 characters, 6 bits each) plus parity
Index registers: (in each computer module)	15
Magnetic thin-film registers: (in each computer module)	128 words, 16 bits per word, 0.33-μsec read/write cycle time
Real-time clock: (in each computer module)	10 msec resolution
Binary add:	1.67 μsec (average)
Binary multiply:	36.0 μsec (average)
Floating-point add:	7.0 μsec (average)
Floating-point multiply:	34.0 μsec (average)
Logical AND:	0.33 μsec
Memory type:	Homogeneous, modular, random-access, linear-select, ferrite-core
Memory capacity:	65,536 words (16 modules maximum, 4096 words each)
I/O exchanges per system:	1 or 2
I/O control modules:	10 per exchange, maximum
I/O devices:	64 per exchange, maximum
Access to I/O devices:	All I/O devices available to every I/O control module in exchange
Transfer rate per I/O exchange:	2,000,000 characters per second
I/O device complement:	All standard I/O types, including 67 kc magnetic tapes, magnetic drums and discs, card and paper tape punches and readers, character and line printers, communications and display equipment

Figure 2 is a photograph of some of the hardware of a completed D825 system. The equipment complement of this system includes two computer modules, four memory modules (two per cabinet), two I/O control modules (two per cabinet), one status display console, two magnetic tape units, two magnetic drums,

a card reader, a card punch, a supervisory printer, and an electrostatic line printer.

D825 characteristics are summarized in Table 1.

Summary and conclusion

It is the belief of the authors that modular systems (in the sense discussed above) are a natural solution to the problem of obtaining greater computational capacity—more natural than simply to build larger and faster machines. More specifically, the organizational structure of the D825 has been shown to be a suitable basis for the data processing facility for command and control. Although the investigation leading toward this structure proceeded as an attack upon a number of diverse problems, it has become evident that the requirements peculiar to this area of application are, in effect, aspects of a single characteristic, which might be called *structural freedom*. Furthermore, it is now clear that the most unique characteristic of the structure realized—integrated operation of freely intercommunicating, totally modular elements—provides the means for achieving structural freedom.

For example, one requirement is that some specified minimum of data processing capability be always available, or that, under any conditions of system degradation due to failure or maintenance, the equipment remaining on line be sufficient to perform primary system functions. In the D825, module failure results in a reduction of the on-line equipment configuration but permits normal operation to continue, perhaps at a reduced rate. The individual modules are designed to be highly reliable and maintainable, but system availability is not derived solely from this source, as is necessarily the case with more conventional systems. The modular configuration permits operation, in effect, with active spares, eliminating the need for total redundancy.

A second requirement is that the working configuration of the system at a given moment be instantly reconstructable to new forms more suited to a dynamically and unpredictably changing work load. In the D825, all communication routes are public, all modules are functionally decoupled, all assignments are scheduled dynamically, and assignment patterns are totally fluid. The system of interrupts and priorities controlled by the AOSP and the switching interlock permits instant adaptation to any work load, without destruction of interrupted programs.

The requirement for expansibility calls simply for adaptation on a greater time scale. Since all D825 modules are functionally decoupled, modules of any types may be added to the system simply by plugging into the switching interlock or the I/O exchange. Expansion in all functional areas may be pursued far beyond that possible with conventional systems.

It is clear, however, that the D825 system would have fallen far short of the goals set for it if only the hardware had been considered. The AOSP is as much a part of the D825 system structure as is the actual hardware. The concept of a "floating" AOSP as the force that molds the constituent modules of an equipment complement into a system is an important notion having an effect beyond the implementation of the D825. One interesting by-product of the design effort for the D825 has, in fact, been a change of perspective; it has become abundantly clear that computers do not run programs, but that programs control computers.

References

AndeJ62; KrogM61; LeinA57; PortR60; ThomR63