Development of the LOGICON 2 + 2 system

by ALBERT L. DEAN, JR.

LOGICON, Inc.
San Diego, California

INTRODUCTION

LOGICON is a systems and software engineering firm that has been heavily involved during the past nine years in developing special systems and software to be used primarily for military and space flight applications. In March of 1969, LOGICON began to investigate the feasibility of using a time-sharing system in its operations. Because a large portion of LOGICON personnel were involved in software development, it was felt that a time-sharing system used as a program development and documentation environment could be of significant aid to these activities.

An examination of other LOGICON operations revealed that many activities such as contract administration, project planning and control, personnel management, financial planning, and marketing could also make effective use of a time-sharing system. Not surprising was the fact that many of LOGICON's business activities are very similar to activities of firms of the same size but operating entirely different businesses. A key capability required by nearly all of these activities was a need to interact with files and data bases while performing concurrent computational operations.

A survey of time-sharing systems available indicated that none of these systems really provided interactive file and data-base management capabilities. This result seemed to indicate the need for such a product; therefore, LOGICON embarked upon the development of a time-sharing system called the LOGICON 2+2 System.

The 2+2 System has been developed specifically to provide very powerful computational, file, and data-base management capabilities on an interactive basis. The 2+2 System is unique—not only because of its file and data-base management capabilities, but also because it was developed for the most part with "off the shelf" fixed-wire hardware, microprogrammed hardware and software designs and/or components. These components have been organized to form an effective environment in which processes may be run in a multiplexed fashion for many simultaneous users. Each user has access to a virtual machine which executes one or more processes in his behalf free from undesired interaction with other users.

This paper describes an analysis of the LOGICON 2+2 System mission, the system requirements derived from this analysis, the development methods employed in creating the system, and the operational and organizational characteristics of the resulting system.

SYSTEM MISSION ANALYSIS

The mission analysis for the LOGICON 2+2 System began with a delineation of system objectives. Four major objectives were defined for the 2+2 System. The 2+2 System must:

- provide capabilities that will allow it to be used as a software development facility by LOGICON in its various system development activities.
- be capable of supporting a variety of applications in the general business environment, such that it can be used by LOGICON or other business firms as an in-house general purpose system or as a vehicle for supplying time-shared services to outside users.
- be capable of supporting a wide class of users, from the novice user to expert programmer, over a performance spectrum ranging from interactive to batch processing.
- be organized in a way that will allow it to be easily modified or extended so that it can be used to form a variety of special purpose systems that may be useful as products.

Given the above objectives, a mission analysis was developed for the 2+2 System. This mission analysis was broken into three major categories: (1) marketing, (2) application, and (3) user requirements.

Marketing requirements

The marketing requirements were developed according to type of customer. Three types of potential customers were identified for the 2+2 System. These are:

- Customers who need one or more general purpose computer systems with interactive shared-access capabilities to meet in-house information processing requirements.
- Customers who are or are planning to provide and market interactive remote access computer services in one or more application areas.
- Customers who need, either for themselves or as contractors to other organizations, a general purpose system that can be easily specialized to form the primary component of a dedicated application system.

An analysis of the potential market represented by these customers produced two major results.

First, a number of potential 2+2 System customers were currently subscribing or were considering subscribing to a time sharing service to meet their interactive processing needs. If the 2+2 System was to replace these services through in-house installation or to replace the equipment providing the service, it must meet two basic requirements: (1) the 2+2 System had to be able to demonstrate a significant cost/performance advantage over existing services and systems, and (2) the facilities provided by the 2+2 System must be compatible with the facilities offered by existing systems (at least on a subset basis), while providing increased capabilities.

Second, five major areas of application were identified: (1) scientific computation, (2) general business data processing, (3) education, (4) data bank operation, and (5) specialty applications such as health and construction. The area of scientific computation, while the largest market, is currently becoming saturated, and as a result—very competitive. The general business data processing market is largely unserved and seems to have even greater potential than scientific computation, particularly in the sectors involving small manufacturing and large wholesaling. The education, data bank operation, and specialty application markets all have good potential but require significant development. As a result, it was decided that the 2+2 System must primarily provide those capabilities required to serve the scientific computation and general business data processing markets, and where possible, provide those facilities needed by the education, data bank, and specialty markets.

Application requirements

A study was carried out to identify the capabilities required to support the preceding areas of application. The results are summarized as follows:

Scientific computation covers the activities of desk calculation, statistical computation, engineering design, simulation, and general mathematics. To support scientific computation, the 2+2 System need not provide extensive data-base management capabilities. The software requirements for this application area are almost exclusively connected with the development of high level problem-oriented languages and their processors. Fast computational processors and high-speed batch, graphic display, and printing keyboard terminals constitute the major hardware requirements for this application.

General business data processing covers the traditional activities of accounting, payroll, inventory control, production control, and internal financial and sales analysis. This application area requires that its software provide extensive file and data-base management capabilities coupled with a set of application procedures for handling: payroll and labor distribution; order entry, shipping, billing, and sales statistics; purchase order preparation and accounts payable; inventory control, production control, and cost accounting.

The hardware required for this application consist of:

- keyboard printing terminals with print speeds from 10 to 50 characters per second for order entry, billing, and small report preparation.
- graphic display terminals for data entry and inquiry.
- remote batch terminals for high-speed data input and output.
- mass storage devices with extensive capacity for files and their associated data bases.

Data bank operation covers the application of an operator-established and maintained data base that is referenced by his customers on a "read only" basis to secure some item of information such as credit ratings, checking account balances, stock quotations, etc. The software required for this application also involves extensive file and data management capabilities. The hardware requirements are equivalent to those of general business data processing, except that special terminals are probably required.

Education covers the application areas of computer aided instruction, test scoring, student counseling, and reference retrieval. The software requirements for this area include extensive data-base management functions and simple interactive program preparation facilities to allow the user to easily construct course presentations, aptitude tests, and reference inquiries. The hardware required is again similar to that needed for general business data processing, with the addition of specialized terminals for instruction.

Specialty applications include areas typified by a highly specialized orientation to work functions unique to a particular occupation. The software requirements for the construction industry include the need for PERT, cost forecasting, and accounting packages that utilize a data base for required parameters. In the medical field, software packages are required to accommodate patient management, inventories of drugs, personnel scheduling, patient billing, clinical history maintenance, insurance coverage, and patient/bed scheduling. Hardware requirements can be met with current equipment, with the exception that special terminals will probably be needed.

User requirements

User requirements for the **2+2** System were developed according to user types, functions, and system loading.

Types of users

The number and types of computer users are expanding at a high rate. To provide facilities that will accommodate all of these users, it is necessary to categorize them into a few major categories.

Novice. A novice user produces no programs; he simply supplies parameters to programs produced by others. Included in this category are clerks, secretaries, foremen, librarians, teachers, and managers. They will need no knowledge of the application, and in fact—they will resist learning to use a system if it significantly detracts or absorbs time from their primary work assignments.

Technician. A technician is a professional in some field other than programming, such as engineering, finance, marketing, manufacturing, etc. The term technician is used in the dictionary sense ("a specialist in the technical details of an occupation") rather than in the more limited sense. He prepares programs to solve his own problems or those of a novice. He is not aware of the hardware features or capabilities. In general, he uses only a small sub-set of the hardware and software features of the system. He is aware of only a "virtual machine" much like the current BASIC systems; with additional facilities for report composition, text editing, and data-base management.

Programmer. A programmer produces programs to serve the computational or data processing needs of an organization. He programs largely in standard procedural languages such as COBOL, FORTRAN, ALGOL, PL/1 and RPG. He will, on occasion, use special system development languages. He uses the software supplied by the system manufacturer, which includes job control facilities, sort, and general utility programs. The fundamental system software commands represent his interface with the system.

User Functions

In order to further identify the **2+2** System requirements, a description was prepared the way each type of user would use the system. The following represents a summary of this analysis.

Novice. The novice user interfaces with the system on an interactive level that permits him to select and control the operation of the system as his commands are being executed. The kinds of functions utilized by the novice user include data entry, inquiry, display, report generation, and function invocation; and in some cases, file and data-base definition.

Technician. The technician works at a programming level that will provide the functions necessary to write programs that then become new operations at the interactive level. The kinds of functions used by the technician include file and data-base definition, data entry, procedure and process definition, editing, and simple program debugging.

Programmer. The programmer works at a defining level that will allow the creation of new commands and programming facilities for use at the novice and technician levels. To accommodate these activities, the system must provide the facilities needed to: define hierarchies of named files; assign access attributes to allow controlled sharing of files; enter programs, data, and text into files; translate source programs to form executable code; test the resulting object programs and insert these programs into the system to form new commands and programming facilities.

User loading

To provide some basic guidelines for developing the **2+2** System, it was essential to estimate the system load produced by the various users. The system loading, as given below, is based upon data from current systems and includes the load from all interaction types.

TABLE I—User Attended Terminal Load per Terminal

	Application Type		
Load Component	Computational Time Sharing	Remote Data Processing	Inquiry
Processing instructions per second	3,500	3,500	250
Data-base accesses per second	.14	.226	.25
Instructions per data-base access	25,000	15,500	1,000
Disc type data-base capacity (megabytes)	1	1	1
Archival type data-base capacity (megabytes)	10	10	10

User-attended terminal load. The load that a user applies to a computer system is heavily influenced by the logistics and scheduling of jobs to and from the computer system. This load is essentially independent from the speed of the user's terminal. The user adjusts his output requirements so that the time he spends waiting for output is tolerable. In current systems with 10 cps teletypes, about 35 to 40 percent of the user's total terminal time is spent in printing for all purposes.

Loads for three different application types are given in Table I. The processor load value in instructions per second, per terminal, is essentially independent of computer speed and compiler software efficiency. It can be shown that this processing load is primarily determined by the scheduling parameters used by the system. For systems having the same response time, those with larger quanta will serve fewer users and do more processing per user than those with smaller quanta. The value of 3,500 instructions per second, per terminal (Table I), is typical of systems designed to do user attended terminal work.

The data-base access load is expressed in terms of processor instructions executed per each access, and it is independent of any scheduling influences. The data-base load expressed in accesses per second follows the processor load instructions per second, as the scheduling method is changed.

Batch and remote batch load. If the user is not at a terminal waiting for results while his job is being run, that job is called a batch job. The chief difference between batch work and user attended terminal work is that the user does not receive his results immediately and he may not submit another job until after he has received and studied his results. The man-machine interaction cycle is much longer than with a user attended terminal, and the rate at which users submit successive jobs is much less dependent upon the length of that user's job.

User jobs are done in a batch mode when they are too long to wait for in a user attended terminal mode or when they require too much output for an individual user terminal and a faster, shared, batch terminal is required.

Table II shows some typical characteristics of a batch or remote batch system load.

SYSTEM REQUIREMENTS

Using the preceding analysis of the LOGICON 2+2 System mission, requirements for the basic organization of the system and its hardware and software components were developed.

Hardware requirements

The **2+2** hardware system was conceived to consist of four major components (1) an application subsystem, (2) an extended memory subsystem, (3) a communications—I/O subsystem, and (4) a control subsystem.

Application subsystem

The application subsystem was defined to consist of an arithmetic processor connected to a high-speed memory. The requirements developed for the applica-

TABLE II—Batch and Remote Batch Load Characteristics

Characteristic	Quantity
Mean executed processor instructions for one program execution (activity)	35 million
Mean lines-of output for one program execution (on-line high-speed printer)	2,000
Mean executed instructions per data-base access	15,500
Mean executed instructions per line of output (on-line high-speed printer)	17,500
Remote batch-restricted output, executed instructions per line of output	35,000

tion subsystem were:

- that it be able to sustain an instruction rate of 3500–7000 instructions per second, per active user.
- the processor should be microprogrammed to allow specialization as needed for a given application area.
- it must be able to support fragmentation of programs and data, through either paging or segmentation, and further provide access control over these program fragments.
- the processor should provide extensive multilevel interrupt capabilities.

Extended memory subsystem

The extended memory subsystem was defined to consist of a drum storage unit and a controller connected directly to the application subsystem's memory. The requirements derived for the extended memory subsystem were:

- the drum storage unit must have a capacity equal to at least 10 times the amount of virtual memory allowed for each user.
- when the latency and transmission rate for a block of data is considered, the "average" access time for a word in the extended memory subsystem should be no greater than 10 times the cycle-time for application memory.

Communications—I/O subsystem

The communications—I/O was defined to consist of a processor with communications and peripheral adapters connected directly both to the high-speed memory of the application subsystem and to its own high-speed working memory. Attached to the communications—I/O subsystem are a variety of peripherals and communication devices. The requirements developed for the communications—I/O subsystem were:

- it must be capable of supporting multiple communication lines, either duplex or half-duplex, with bandwidths ranging from 110 to 300 baud for the low-speed lines and 2,000 to 19,200 baud for the high-speed lines.
- all device connections for the low-speed devices on the communications—I/O subsystem (including the low-speed peripherals) must conform to EIA RS 232 interface standards.
- it must support ASCII codes.
- the processor should be micro-programmed and have sufficient speed to allow line management and device control to be accomplished by the processor, versus special device adapters.
- it must be capable of storing .5 million bytes of information on-line, per user of the system.
- when the seek time and transmission rate are considered, the "average" access time for a word in the disk storage units should be no greater than 10 times the access-time of the extended memory subsystem.
- at least some portion of the disc storage capacity should exist on removable media.

Control subsystem

The control subsystem was defined to consist of a processor connected to the communication—I/O memory and the application memory. The requirements derived for the control subsystem were:

- it must be connected to the other subsystems via interrupt and acknowledge lines to form a control network within the system.
- it must be microprogrammed to allow specialization as required for its control functions.

Software requirements

Three major requirements were defined for the 2+2 software. The software must:

- be based upon a proven design.
- when integrated with the hardware, provide users with a uniform environment free of hardware idiosyncrasies.
- provide a graduated set of capabilities in response to different types of users.

With these requirements in mind, the 2+2 software was conceived to consist of two major systems; the operating system and the application system.

The operating system

Requirements developed for the operating system specified that it must:

- be organized in a modular fashion to facilitate modification or extension as required.

- allow for a distributed type of operation where a user can select, use, and pay only for those modules required by his job.

The operating system was defined to consist of four major components: (1) the file manager, (2) the I/O manager, (3) the process manager, and (4) the data-base manager.

The file manager must provide the capabilities required to:

- form hierarchies of named files.
- allow access attributes to be assigned to these files, and enforce these access definitions.
- enter or extract information in or from these files.
- enable file backup and recovery in the event of a failure.

The I/O manager must provide the facilities required to:

- control the communications and peripheral devices of the system.
- support two modes of data access; random and sequential.
- provide the functions of read, rewrite, and append; where rewrite refers to the ability to write over existing data, and append refers to writing on an "unwritten" portion of data.
- supply format control over both physical and logical streams of data; where physical format is dictated by the device, and logical refers to a user supplied structuring.

The process manager must provide the capabilities required to:

- create processes consisting of procedures and data.
- allocate resources required to execute these processes.
- schedule the execution of processes, both system and user.
- monitor the execution of processes.
- maintain an accounting of the usage of the system resources by processes.

The data-base manager must supply the capabilities required to:

- define multi-level data structures and access attributes.
- associate defined data with one of four access methods; sequential, indexed-sequential, linked list, or multi-list as defined for a designated file.

The applications system

Requirements defined for the application system were that it must:

- provide a uniform interface to the users of the 2+2; i.e., interactive commands and batch control statements for equivalent functions are the same.
- provide "total" functional capability in each of its subsystems so that all the facilities required by a user for the solution of his problems are available to him with the subsystem.
- provide a graduated set of capabilities that match the need of the various types of users; novice, technician, programmer, operator, and "owner."

The application system was defined to consist of three major subsystems or classes of subsystems; an executive, language subsystems, and application library.

The executive subsystem must provide the facilities needed to:

- establish an interface between the user and the system in the form of a command language and command language interpreter.
- interpret user commands and invoke the appropriate system functions.
- allow for the modification and extension of the command language.

The language subsystems were divided into three categories, corresponding to type of user: (1) novice, (2) technician, and (3) programmer.

The language subsystems supplied for the novice user must:

- operate on an interactive command level basis.
- provide the functions required to do data entry, inquiry, display, and updating of data bases.
- provide the functions to carry out simple desk computations using formatted forms and reports.

The language subsystems supplied for the technician must:

- provide the functions needed to accomplish file and data-base definition, data entry, procedure and process definition, editing, and simple program debugging.

The language subsystems supplied for the programmer must allow him to:

- define hierarchies of named files.
- assign access attributes to allow controlled sharing of files.
- enter information into these files consisting of programs, data and text.
- translate source programs to form files of executable code.
- select, edit, merge, and link files to form processes that are ready for execution.
- load linked processes and invoke execution.
- monitor and retrieve the results of an executed process.
- select, edit, and format source language, flowchart, and text files to form the documentation for a new command or system.

The application library must provide a series of scientific, engineering, and general business packages. In particular for business—these packages should be concerned with payroll, inventory control, accounting, production control, and internal financial and sales analysis.

DEVELOPMENT METHODS

The development methods employed for the LOGICON 2+2 System merit discussion for several reasons. *First,* while a number of time-sharing system descriptions have been published, only fragmentary information has been presented concerning the methods used to develop these systems. *Second,* the methods used to develop the 2+2 System represent a systematic approach to the construction of a time-sharing system.

The discussion of the 2+2 System development is divided into two sections. The first section describes the general development scheme, while the second section describes the development tools utilized to produce the system.

General development scheme

The general scheme used for development of the 2+2 System can be called an "outside-in to inside-out" approach. This process is illustrated by Figure 1.

The notion of outside-in to inside-out development is analogous to the evaluation of an expression, in that one starts at the outside and proceeds inward identifying the primary operands and operators and their ordering. Once the primary operands are identified, the process is reversed and the constituents of an expression are combined, according to the applicable operator rules, to produce the value defined by the expression. Similarly, the development of the 2+2 System proceeded from the "outside-in to inside-out."

Outside-in

The outside-in path is divided into two phases; specification and design. During the specification phase, a system requirements specification was produced that identified the types of service to be supported by the system, the classes of users to be served, the functions to be supplied to these users, and the possible major components required to provide these functions. Using the system requirements specification as a guide, the functional, organizational, and operational characteristics of the 2+2 System were defined. From these definitions, a system architecture specification was developed.

Given the system architecture specification, effort was then directed toward producing detailed designs for subsystems and their major components as called out in the system architecture specification. It should be made clear, if not already so, that one does not proceed directly through the transformation of requirements—to the architecture—to the detailed design—without some iteration and intuitive knowledge of the feasibility of the resulting object.

Inside-out

At this point, the detailed designs for each subsystem and major component were transformed into executable

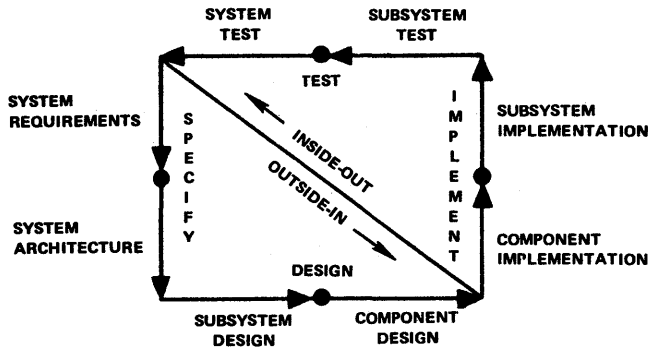

Figure 1—Outside-in to inside-out approach

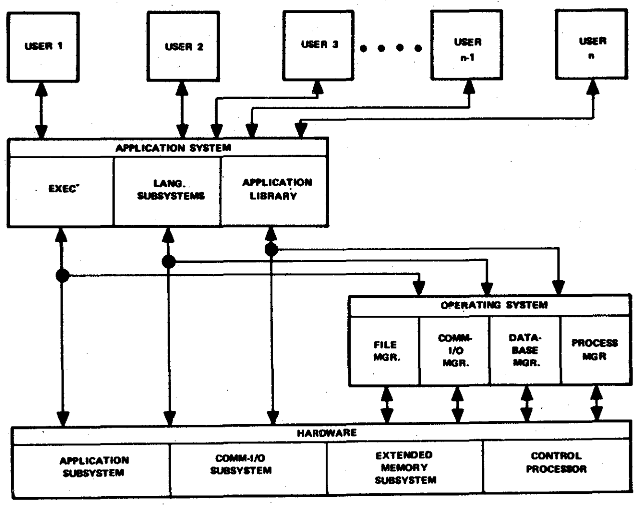

Figure 2—2+2 system logical structure

hardware/software logic. This transformation was accomplished by translating the design for each module of a component or subsystem with the implementation language appropriate to that module; i.e., wire-wrap lists, microprogram assembly language, or system assembly language.

As each module was implemented, an effort was made to ascertain that a module met the functional and operational requirements defined for it. This was accomplished by building a skeleton for the next higher structure in which the component was to reside; and then, supplying the appropriate input parameters and calls to the module, verifying the correctness of the output parameters and returns, and measuring critical execution time and frequency of use. This process was repeated at successive levels of integration until a complete subsystem was developed. The subsystems were then tested, and upon successful test completion, integrated into the system. As each subsystem was integrated into the system, an iterative testing procedure was applied. This procedure basically started at the innermost modules, insuring their continued integrity, and proceeded outward until the newly integrated modules or subsystems were checked.

Development tools

The tools utilized to develop the 2+2 System may be grouped into two categories; support system facilities and target system facilities.

The *support system facilities* were established on a time-sharing service system on which LOGICON purchased time. The facilities provided by the support system consisted of a simulator for the target machine; an assembler, to produce code for the target machine or its simulator; a program debugging package running under the simulator, for aiding in the checkout of system software; and the standard file and editing facilities of the time-sharing system on which the support system resides.

The *target system facilities* consisted of a simple run-time executive with a simple dump and trace capability for loading, executing, and monitoring the execution of software modules developed for the 2+2 System. In addition, a simple file system was developed to allow modules to be placed on the 2+2 System mass storage. Finally, a linking loader was also placed in the development executive for aiding in the generation of system programs.

SYSTEM DESCRIPTION

The LOGICON 2+2 System is composed of six major components: (1) the application subsystem, (2) the extended memory subsystem, (3) the communication—I/O subsystem, (4) the control subsystem, (5) the operating system, and (6) the application system. The logical organization of the LOGICON 2+2 System is illustrated in Figure 2.

These six components have been integrated to form an environment containing the facilities necessary to concurrently process a wide range of user applications. This concurrent processing is accomplished by multiplexing the system's facilities against the requirements of the system's users. In addition, these facilities have been regularized to eliminate hardware inconsistencies. As a result, each user is provided with a virtual machine containing the facilities required for his application, free from undesired interaction with other users and hardware idiosyncrasies.

The fundamental unit of user activity in the system is a process. A process is a related sequence of operations executable on the processor(s) comprising the virtual machine. These operations are executed in one of two modes; user and system. User mode operations are executed directly while system mode operations are indirectly executed. System mode operations are hardware instructions or extensions to these hardware instructions implemented as software procedures, which require protected and controlled execution in order to assure effective and uniform usage.

To utilize the 2+2 System, a user sends a connect signal to system via his terminal device. The system responds by establishing a process called the executive for the user. The executive first validates the user's access to the system and then operates as an interface to the user, interpreting the users commands and invoking system functions in response to these commands.

Through commands, the user is able to create and cause the execution of one or more processes on behalf of his application. These processes may be system supplied as in the case of the language subsystems or user supplied applications or combinations of both.

In the remaining portion of this paper, the components comprising the 2+2 System are described in more detail.

The 2+2 hardware

The 2+2 hardware is composed of four major subsystems: (1) the application subsystem, (2) the communications—I/O subsystem, (3) the extended memory subsystem, and (4) the control subsystem. The physical organization of the 2+2 System is illustrated in Figure 3.

Application subsystem

The application subsystem has been constructed using a microprogrammed processor augmented with a Virtual Address Translator (designed by LOGICON) and a 900 nano-second, 16-bit word core memory ranging from 32 to 64 thousand words. The instruction set for the application processor was developed by LOGICON specifically for the 2+2 System using the microprogramming capability of the processor.

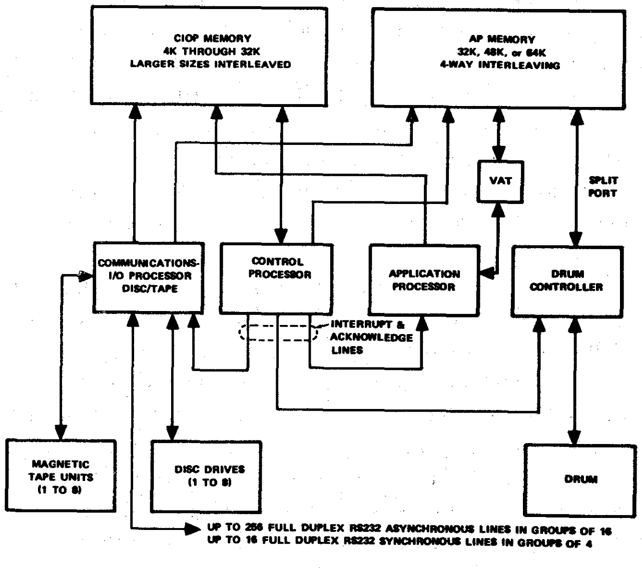
CIOP MEMORY
4K THROUGH 32K
LARGER SIZES INTERLEAVED
AP MEMORY
32K, 48K, or 64K
4-WAY INTERLEAVING
VAT
SPLIT PORT
COMMUNICATIONS-I/O PROCESSOR DISC/TAPE
CONTROL PROCESSOR
APPLICATION PROCESSOR
DRUM CONTROLLER
INTERRUPT & ACKNOWLEDGE LINES
MAGNETIC TAPE UNITS (1 TO 8)
DISC DRIVES (1 TO 8)
DRUM
UP TO 256 FULL DUPLEX RS232 ASYNCHRONOUS LINES IN GROUPS OF 16
UP TO 16 FULL DUPLEX RS232 SYNCHRONOUS LINES IN GROUPS OF 4

Figure 3—2+2 physical organization

The Virtual Address Translator, or VAT, allows virtual addresses consisting of a page number and offset to be translated into physical addresses. The core memory provides for four-way interlace and page protection facilities of read, write, and execute. The instruction execution rate of the application subsystem is approximately .5 million instructions per second, which equates to a capability of supporting between 64 to 128 users—depending upon core memory size.

The communication—I/O subsystem

The communications—I/O subsystem was also constructed with a microprogrammed processor. This processor is connected to the application subsystem's memory and has its own high-speed working memory of 900 nano-seconds; a 16-bit word core memory, ranging in size from 8 to 32 thousand words; a set of asynchronous line adapters for 16 to 128 lines, in groups of 16; with bandwidths ranging from 110 to 300 baud; a set of synchronous line adapters for 4 to 16 lines, in groups of 4, with bandwidths of 2,000 to 19,000 baud; and peripheral adapters for the disk and magnetic tape storage devices. Normally, one synchronous line is used to attach a remote batch terminal consisting of a 300 cpm card reader and a 400 lpm line printer.

The disk storage units have removable media, 28 million-byte storage capacity, and an average access time of 45 milliseconds. Up to eight disk units can be attached to the mass storage controller.

Tape storage consists of from one to eight tape drives working 25 ips; 800 bps with 7 or 9 channel recording. Based upon the use of eight disk storage units, the 2+2 System can support 348 users having minimal storage requirements and up to 64 users requiring heavy storage needs. The "access-time" to disk storage is less than 10 times the access-time to the extended memory subsystem.

The extended memory subsystem

The extended memory subsystem for the 2+2 System was constructed using a drum having a 1 million word capacity, expandable to 4 million words, with a 1 million word-per-second transfer rate and an average access time of 8.5 milliseconds. The extended memory subsystem is connected to the application subsystem memory. The drum controller was designed and developed by LOGICON to provide the required interface.

The control subsystem

The control subsystem was constructed using a microprogrammed processor. The instruction set for the control processor is almost identical to that of the application processor.

The control processor performs all system scheduling and I/O management tasks. It therefore controls the information flow between the application processor memory, extended memory, and mass storage. To accomplish these control functions, it is connected to the communications—I/O processor, the application processor, and the drum controller with interrupt and acknowledge lines. The control processor is connected to both the communications—I/O subsystem's and the applications subsystems's memory.

The 2+2 *operating system*

The 2+2 operating system is composed of four major subsystems: (1) the file subsystem, (2) the I/O subsystem, (3) the control subsystem, and (4) the data-base subsystem.

The file manager

Files in the 2+2 System are organized in a hierarchical fashion or tree structure of finite length (five levels) with a root directory origin. The structure contains pointers or entries for all nodes at the next lower level. These nodes may be directories themselves or point to some next level. The terminal nodes for the structure are files.

The file manager provides the operations needed to allow directories of files and files to be created, linked, modified, opened, closed, and deleted. The file manager consists of four major components: (1) the directory control module, (2) the access control module, (3) the storage mapping module, and (4) the backup and recovery module. The *directory control module* provides those functions required to create directories, create files and as a result create an entry in the appropriate directory, delete directory entries thereby deleting a file, and deleting directories. The *access control module* provides the capabilities needed to specify and enforce the use and access allowed for shared files. The access attributes provided are read, write, append, and execute. The *storage mapping module* maintains maps of all storage devices and the allocation of the space represented by these devices. The *backup and recovery module* provides the functions required to periodically dump files and to reestablish files in the event of a mishap.

The I/O manager

The I/O manager provides operations such as the reading and writing of sequential and random files and enabling the execution of physical input-output operations expressed in terms of logical parameters.

The I/O manager is organized into three major components: (1) the device control modules, (2) the device strategy modules, and (3) the I/O control module. The *device control modules* are particularized to manage a specific type of device and to control its operation, both in normal and error mode. The *device strategy modules* are responsible for the logical management of the device in terms of the data stream being transmitted to or from the device. The device strategy modules provide the function needed to access data in a random or sequential fashion, to format data, and for the logical operations of read, write, or append. The I/O *control module* is responsible for scheduling and managing the flow of data streams from one device to another, from one process to another, etc.

The process manager

The process manager provides the functions necessary to create, activate, block, and terminate processes, and also allocate resources and schedule their usage. In addition, it provides a mechanism for interprocess and interuser communication. It allows a process to create an "event," to be notified when an "event" occurs, and to delete "events." The process manager is composed of four major modules: (1) the scheduler, (2) the resource allocator, (3) process control, and (4) the accounting module.

The *scheduler module* is responsible for ordering the execution of processes in the 2+2 System. The scheduler accomplishes this ordering by maintaining a queue or process execution requests that are selected through a multi-level priority algorithm. The *resource allocator* provides the functions required to allocate and manage the resources of the 2+2 System. The *process control* is responsible for process creation, loading, execution, unloading, and destruction. During execution, the process control acts as the interface between the system and a user's process. The *accounting module* is responsible for metering, pricing, and billing for the facilities of the system used by its users.

The data-base manager

The data-base manager is responsible for the management of records in a data base and consists of three major modules: (1) the record access control module, (2) the record content module, and (3) the record format module. The record access control module is concerned with two types of access control. The first type of access control is involved with the logical operations of extracting or inserting a record in a data base through one of four access methods; sequential, indexed sequential, linked list, and multi-list. The second type of access control is involved with security conventions defined for a record, and insuring that these conventions are not violated.

The record content module is responsible for associating and maintaining lists of the elements that comprise a record. The record format module is responsible for providing the facilities for mapping contents of a record from one format to another for input-output operations. The record format module provides the facilities for associating a data base with the descriptions of the formats to be applied to that data base.

The **2+2** *application system*

The **2+2** application system is composed of three major components: (1) the executive, (2) the language subsystems, and (3) the application library.

The executive

The executive consists of three major elements: (1) a command language, (2) the command interpreter, and (3) the command library. The command language is the vehicle by which the user and the system communicate with each other. Through the use of the *command language*, the user is able to issue instructions to the system, directing it to perform some desired operation or set of operations. The *command language interpreter*, upon receipt of a command, scans the command and its arguments, calls the needed command subroutines from the command library, links these procedures, and transfers control to the linked procedures for execution. The *command library* contains a series of subroutines that enable the operation of the commands defined for the **2+2** System. Users of the **2+2** System have the capability to augment this command library through the use of commands supplied for that purpose by the system.

The language subsystems

The application system consists of a number of subsystems, each of which support a given language. In general, each language subsystem consists of a subsystem executive, command processor, language processor, and routine processor. There are three levels of language subsystems. Each level equates to the type of system user involved; novice, technician, or programmer.

For the novice user, a desk computation and data-base entry/inquiry command system has been developed called DESK-DATA, which combines the properties of a desk calculator and a simple data-base management system. The DESK-DATA language allows the novice user to establish data base, enter data into or make inquiries against the data bases, generate reports from these data bases whether on hard copy or a graphic display device, and define and perform simple computations on the data bases.

For the technician, three language subsystems are supplied: (1) an extended form of BASIC that provides extensive file and data-base management capabilities, (2) APL, also augmented with file and data-base facilities, and (3) a powerful text editing language.

For the programmer, FORTRAN IV, a subset of PL/1, and assembly language are provided.

The application library

The application library is composed of a large and growing collection of programs designed to perform explicit functions on behalf of its users. The application system is perhaps the most important component of the **2+2** System in the sense that it is directly concerned with the user's requirements.

SUMMARY

The **2+2** System is in the final stages of development and it is now clear that it will meet the functional and operational requirements set out for it. The success of this effort was in large part due to the project team involved. Special acknowledgment is given to R. E. Wolfe and John Mallory for the microcode design and development; Frank J. Rosbach, Allen Ginzburg, and Jan C. Bartlett for the design and development of the **2+2** Operating System; Glen S. Jerpseth, Robert A. Thompson, Paul K. Richards, Leland S. Purrier, and James A. Craig for design and development of the **2+2** Application system; and to George P. Futas and Donald F. Lacy for their design and development of the

Extended Memory subsystem. Special thanks are given to Frank C. Robbins for his aid in editing this paper.

REFERENCES

1 A BENSOUSSAN C T CLINGEN R C DALEY
The multics virtual memory
Second ACM Symposium on Operation System Principles Princeton New Jersey October 1969

2 R C DALEY J B DENNIS
Virtual memory processes and sharing in multics
First ACM Symposium on Operating System Principles Gatlinberg Tennessee October 1967

3 B W LAPSON
Scheduling and protection in interactive multi-processor systems
Project Genie Doc P-11 University of California Berkeley February 1967

4 B W LAMPSON
Time-sharing system reference manual
Project Genie Doc R-21 University of California Berkeley August 1966

5 B W LAMPSON W W LICHTENBERGER M W PIRTLE
A user machine in a time-sharing system
Proceedings IEEE December 1966

6 J H SALTZER
Traffic control in a multiplexed computer system
MAC-TR-30 (thesis) MIT Cambridge Massachusetts July 1966

7 M J SPIER E I ORGANICK
The multics interprocess communication facility
Second ACM Symposium on Operating System Principles Princeton New Jersey October 1969