Information and Management Sciences

AUERBACH®

A NASA FAMILY OF

MINICOMPUTER SYSTEMS

APPENDIX A TO

TECHNICAL REPORT
1958-100-TR-004

By

M. P. DEREGT
JOHN E. DULFER

Submitted to:

NASA Headquarters

Under

Contract No. NASW-2285

June 19, 1972

AUERBACH Associates, Inc.
1501 Wilson Boulevard
Arlington, Virginia
22209

TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

FOREWORD

This appendix has been written during the course of a contract with NASA Headquarters to assess teleprocessing and computer technology for the 1975-1985 time frame, and to identify and describe areas of investigation leading to more effective and efficient use of NASA's computer and teleprocessing facilities. After a number of these topics had been investigated, NASA selected standardization of minicomputer components and two other topics for inclusion in the final report. That document was conceived as a summary of the previous results of the contract and three appended development plans, one for each of the selected topics. The second topic is the study of approaches to developing standard specifications for the upcoming, very large mass storage systems. Because of the mutual influences of large data structures and processing hardware, this study includes the design characteristics of data management systems and special storage processors. The third topic is management, organization and procedures for efficient development of correct software for large and complex applications. The emphasis in this area is on advances in methodology and technology, stressing the practical engineering properties of software design and fabrication, and the theoretical aspects of computer science.

All of these topics proved to be elusive and difficult to define at first. During the writing of the present appendix, the emphasis shifted from planning the development to elaborating on the concept. This fact has given rise to certain inconsistencies of an editorial nature. They have been allowed to remain in this appendix because of the press of time, the demands of the other two topics and because they do not seriously impede understanding of what is proposed herein. The reader is asked to overlook them in favor of the substance of this appendix. For a more complete understanding of its context the reader is referred to its parent document and two companion appendices entitled:

- Mass Storage Hardware and Software Standards (provisional title), Appendix B
- A Program for the Systematic Evolution of a NASA Software Technology, Appendix C.

SECTION 1. SUMMARY

1.1 NASA RESPONSIBILITIES AND POLICY FOR DATA PROCESSING

The responsibilities of the Associate Administrator for Tracking and Data Acquisition, of which Code TN is a part, include[1]:

- Directing the planning, development, acquisition and operation of world-wide tracking, data acquisition, data processing and communications systems, facilities and services required in support of NASA programs.
- Providing staff direction and management of the NASA automatic data processing resources.

The first responsibility involves operating and improving a world-wide tracking and data acquisition network and its associated data processing equipment, representing 23% of NASA dedicated (Category B) ADP resources.

The second responsibility relates to the effective use of ADP in NASA, insofar as responsibility therefor has not been delegated to NASA field centers or program offices, and resides with the ADP Management Branch of Code TN. It involves assisting in decisions on ADP investment policy,

coordination of NASA-wide ADP matters, staff review of all planned data processing acquisitions, maintenance of an equipment inventory, administering standards, coordinating purchases to obtain volume discounts, redistributing surplus equipment and funding the development of techniques or equipment for general NASA use.

It is NASA policy to encourage the development of applied technology which in addition to its primary usefulness for astronautics may also have commercial, industrial or consumer applications of benefit to the American public. A familiar example is the Dow-Corning material, developed for reentry body heat shields, now used in cookingware.

The development of standards for a NASA minicomputer family falls under the second responsibility, that for staff direction and management of NASA ADP. It could also involve the first (operating the world-wide data acquisition network), in that the network employs many small dedicated computers subject to replacement by minis. Establishing standards for minicomputers where none exist could conceivably benefit users of the machines other than NASA in the Federal Government and in the public at large, thus adhering to the NASA policy on the secondary beneficiaries of applied technology.

1.2 PURPOSE OF THE DEVELOPMENT PROJECT

The overall objective of the development is establishing sufficient specifications or standards for minicomputer hardware and software to provide NASA with realizable economies in quantity purchases, interchangeability of minicomputers, software, storage and peripherals, and a uniformly high quality. The standards will define different minicomputer system component types, each specialized to its intended NASA application, in as many levels of capacity as required. The standard family of minicomputer systems will have the following properties:

- The hardware built to a given standard by various manufacturers will be physically and logically interchangeable
- The function and performance required of a subsystem at

a given level of the family will be met by the subsystem at the next higher level. This will permit system growth by substituting components of the next higher performance level

- Software for a computer of a given standard will run without modification on any computer built to that standard or to the next higher standard

- Program development software, to run on large computers, will be used to prepare ready-to-run programs for the minicomputer family

- There will be separate executive software for each specialty type, or branch, of the family; it will be partitioned so that required portions can operate on all members of the branch down to the smallest

- Family hardware and software components will be testable to ascertain that they meet their respective standards.

1.3 PRINCIPAL BENEFITS AND BENEFICIARIES OF THE RESULTS

The major benefit arises from the interchangeability of different manufacturers' products designed to the same standard. This will enable volume purchases of minicomputer systems, with the assurance that they will find application in NASA, and the lower prices of quantity procurements. It will also simplify the job of maintaining minicomputer systems by reducing the number of parts to be stocked, vendors dealt with, equipment types for maintenance trainees to learn, etc. An added benefit will be simplification of the choice of a minicomputer system to fit a given application by reducing the number of distinct equipments to choose from. The existence of standard interfaces as part of the minicomputer family will obviate the need for their detailed design in the future, simplifying the system design process. Further, the designs will be capable of expansion without massive revision because of the upward compatability of minicomputer family components.

Not only hardware but also software is to be interchangeable, at least within the minicomputer family branches. The benefits of fully exploiting this software interchangeability, such as by common use of a collection of applications modules, executive programs, etc., could be enormous, because

of the redundant programming obviated thereby. Interchangeability will also simplify the job of developing the minicomputer programming aids which will be designed to run on large computers and produce code for the minis. Interchangeability of programs, currently obtained by specifying identical hardware at the time a system design is laid down, greatly simplifies the design of systems which must automatically exchange programs or data, such as in a network. Adopting a standard minicomputer family extends interchangeability throughout NASA and makes possible for the first time a network of like extent.

To obtain the full effect of these advantages, the NASA standard components must be as economical to acquire and operate as units offered for general sale. This requires a large number of installations - perhaps many more than likely in NASA alone - to distribute the manufacturers' capital costs, particularly those for support services for the standard components. The way to acquire this large usership is to adapt the standards to the needs of a potentially large market, both within and outside the government. We feel NASA minicomputer applications are not so rare as to preclude taking this approach. Doing so, however, will require identifying and basing standards on the needs of users in NASA and elsewhere, as detailed in Section 5.

1.4 SCOPE OF THE DEVELOPMENT

The development of standards for a family of minicomputer system components breaks down into two phases:

- Phase 1 - an immediate effort to specify minicomputers for a particular application. The example of data acquisition and processing in NASA networks is used throughout this Appendix.
- Phase 2 - a longer-term project to develop standards for the remainder of NASA applications.

Both phases will run concurrently at the beginning of the development, with major emphasis on Phase 1. Phase 1 will conclude with the procurement of components for the particular application (i.e., network data acquisition), leaving Phase 2 the sole activity of the development. However, a new Phase 1 could be commenced at any time the need arises, as in the case of a consolidated procurement of minicomputers for some other application. The components specified in Phase 1 must conform to the standards developed in Phase 2; hence, the need for concurrent effort on the two phases.

It should be recognized at the outset that the family of components to be specified in this development will not be suited to all possible applications of minicomputers in NASA. Such a goal conflicts with reasonably limiting the number of members and should be considered an ideal of which 80% - 90% attainment is the practical limit for purposes of this project.

Two major tasks confront the principals in this development. The first is determination of the component types which make up the family and of the parameters which must be specified to assure required compatibilities.

The second is that of determining the major NASA applications for which the branches of the minicomputer family are to be specialized and the ways in which component characteristics will differ from branch to branch. The first branch of the family has already been determined to be concerned with NASA network data acquisition. Identification of additional branches may be postponed until the data acquisition branch is completely specified at the completion of Phase 1.

Phase 1 will consist of a 3-5 manyear effort requiring less than twelve months for completion. Phase 2 will be 20-30 manyear effort requiring up to three years to complete.

1.5 ORGANIZATION OF THIS APPENDIX

This appendix consists of eight sections: summary, background, minicomputer description, statements of the problems, development project description, representative families, management plan and proposals to perform the development.

Section 2, Background, describes the evolution of the minicomputer and presents material on NASA use and inventory of the devices.

Section 3, Minicomputers - Detailed Description, discusses the features of minicomputers and their software, relating them to the classes of uses to which they have been applied.

Section 4, Statements of the Problem, outlines minicomputer system components and their parameters.

Section 5, Description of the Development Project, describes the main tasks and suggests methods for their conduct. It is the work statement for the development.

Section 6, Representative Families and Guidelines, sets forth family structures which are pertinent to the development, for illustrative purposes.

Section 7, Implementation Plan, describes the actions necessary to manage the development of and promulgate minicomputer system standards.

Section 8, Proposals to Perform the Development, contains information to guide the preparation and evaluation of proposals or (if submitted by a NASA Center organization) plans to perform the development tasks.

SECTION 2. BACKGROUND

2.1 ORIGINS

The technological development that led to the emergence of the minicomputer was the wide-spread commercial availability of integrated circuits in the mid-1960's. The advent of this electronic development resulted in substantial progress in computer technology. It made feasible the manufacture of computers at substantially lower cost and enabled independent manufacturers to jump quickly into the computer market with a significantly new product. In 1965, the minicomputer industry emerged with its first substantial sales of about $30 million. With rapid growth in sales, number of available machines and number of participating companies, the industry clearly established itself as a distinguishable segment of the computer industry. In the last half of the 1960's annual sales have increased at an average rate of over 40 percent per year and, by the end of 1971, the installed base reached over 25,000 units.

The word "minicomputer" probably came from a paper presented at the Fall Joint Computer Conference in 1968, entitled "The Mini-Computer -- A New Approach to Computer Design."[1] At that time, the authors described an experimental computer designed by IBM, called MINI. MINI had no arithmetic or control logic but had 512 bytes of 2-μsec storage, an instruction set of eight instructions, an IBM Executary for bulk storage, a CRT display, a printer/keyboard, and a common carrier Dataset. MINI was used to demonstrate the feasibility of programming arithmetic and logic functions by using only fetch, store, and branch on zero or not zero instructions and

by replacing special-purpose input/output equipment with general-purpose data flow input/output.

The term "minicomputer" was applied to a host of small general-purpose computers introduced during 1969 to satisfy the demands of the scientific, data communications, and control computer markets. By the end of 1969, a number of manufacturers also began to use "minicomputer" to classify small general-purpose computers aimed primarily at the commercial processing market.

2.2 MINICOMPUTER DEFINED

AUERBACH Standard EDP Reports[2], have roughly defined a minicomputer as a computer with the following characteristics:

- Costs less than $25,000 when introduced for a minimum stand-alone configuration that includes some type of input/output, such as a Teltype ASR 33 with paper tape attachment
- Provides at least 4K words of memory
- Performs calculations under stored-program control
- Can be programmed in an assembly or higher-level language
- Can be used by a broad range of users and is not restricted to specialized applications.

Although the preceding definition presents some difficulties, notably in minimum memory size and maximum price, and is arbitrary, it is sufficient to the purposes of this Appendix.

2.2.1 Mainframe

In general, the minicomputer offers fewer features than full sized computers. This is particularly true of the lower cost models. As may be seen from the accompanying chart, taken from the AUERBACH Standard

EDP Report series[3], some do not even have arithmetic units capable of multiplication and division, which must consequently be performed by subroutine. Short word-lengths, usually of 16 bits, are the rule in order to save on logic costs. They result in the use of elaborate addressing schemes and multi-word operations, both at the expense of through-put. An almost universal restriction is due to the provision of a single data channel to the memory. The result is a virtual stalling of the CPU during I/O of the block transfer variety, which monopolizes the single channel.

Characteristics	Minimum	Features Available* Average	Maximum
Memory Word length (bits)	8	8 or 16	18
Size (words)	1,024 - 4,096	4,096 - 32,768	1,024 - 65,536
Increment size (words)	1,024	4,096	8,192
Cycle rate (kHz)	125	571 - 1,000	1,250
(Cycle time, μsec)	(8)	(1.0 - 1.75)	(0.8)
Parity check	No	Optional	Standard
Memory protect	No	Optional	Standard
Direct addressing (words)	512	512 - 4,096	All of core
Indirect addressing (type)	No	Single/multi-level	Multilevel
Central Processor General-purpose registers	1	1, 2, 3, 4	16 (in core)

Figure 1. Minicomputer Characteristics

Characteristics	Features Available* Minimum	Average	Maximum
Index registers	0	1	15 (in core)
Hardware multiply/ divide	No	Optional	Standard
Immediate instructions	No	Almost half yes	Yes
Double-word instructions	No	Almost half yes	Yes
Byte processing	No	Half yes/half no	Yes
Input/Output			
Programmed I/O channel	1	1	1
I/O word size (bits)	8	8/16	18
Priority interrupt lines	1	1 standard, up to 64 optional	4 standard up to 256 optional
Direct memory access	No	Optional	Standard
I/O maximum transfer rate (DMA), words/sec	125,000	400,000-600,000	1,250,000
Other Features			
Real-time clock	No	Optional	Standard
Power fail/restart	Optional	Optional	Standard
Largest disc (megawords)	No	0.1-3.8	14.5
Assembler	Yes (not macro)	Yes (not macro)	Yes (macro)
Compiler	No	Basic Fortran	Basic Fortran, Standard Fortran, Algol, Focal, Basic
Operating system	No	No	Real-time, foreground/ background, time sharing
Applications packages	No	No (except by special contract)	5-types offered are typesetting, data concentration, data acquisition, plotter or display control, instrument control, hybrid analog-digital, pulse height analysis, inventory control, sort/merge, report generation.
Purchase Price**			
Computer with 8K words of core and Teletype Model ASR 33			
8-bit words	$ 6,900	$11,180	$16,400
12-bit words	$ 8,200	$12,500	$16,800
16-bit words	$12,100	$19,900	$33,000
18-bit words	$23,600	$24,600	$25,600

* Exceptional minicomputers are not included because the features offered are atypical.
** Price bears no direct relationship to the features listed in the column because a particular system does not have all of the features noted.

Figure 1. Minicomputer Characteristics (Contd.)

The more powerful (and more expensive)minicomputers nevertheless have a surprisingly large range of desirable features. Many levels of interrupt processing and direct memory access are two such features which enhance the mini's I/O performance. In addition, memory protection, privileged instructions and realtime clocks incorporated in orders render them suitable for on-line, multiprogrammed control and timesharing applications.

2.2.2 Peripherals

Secondary storage devices for use with minicomputers show a more marked difference from their full-sized counterparts than do mainframes. This is principally because their electromechanical technology has undergone very little cost improvement since the introduction of the minicomputer, or indeed since the invention of the electronic computer. In order to be more nearly commensurate in price with the low cost mainframe, which has decreased in cost by five orders of magnitude since 1952[4], storage devices offered for use with minicomputers have less capacity than full-performance versions (see Figure 2). These inexpensive peripherals exhibit a much higher cost per bit of capacity than is typical of large units, indicating that the mini-disc and -tape suffer from diseconomies due to their small scale. Even these lower performance units cost in the neighborhood of what a minicomputer mainframe does. Other low cost peripherals in great variety are available and include line printers, CRT and hardcopy keyboard terminals and card and paper tape handlers.

2.2.3 Software

Minicomputer systems software tends to be arranged in a hierarchy of operating systems, having different hardware requirements. Typically, a system designed to run on a machine configuration of 4,096 16-bit words of core storage will offer a basic assembler supported by a range of utility routines and possibly a Basic or Fortran compiler. A system designed for 8,192 words of main memory will include a compiler and an assembler with a more comprehensive range of facilities (such as

Typical Minicomputer Disc Systems

Type Device	Fixed Head Disk	Fixed Head Disk	Removable Disk	Removable Disk	Removable Disk
Model Number	RS-11	RS-64	RK02	2311	2314-B1
System	DEC PDP-11	DEC PDP-11	DEC PDP-11	IBM System/360	IBM System/360
Capacity (millions of bytes)	0.512-4.096	0.128-0.512	1.2-4.8	7.25-58	87-233
Average Access (msec)	17	16	80	75	60
Transfer Rate (bytes/sec)	125,000	125,000	90,400	156,000	312,000
Purchase Price ($) Min. Max.	 14,000 77,000	 6,950 20,450	 12,900 33,900	 50,380 223,595	 111,385 201,385
Cost Per Byte ($) Max. Min.	 0.0274 0.0189	 0.0543 0.04	 0.0107 0.00705	 0.00695 0.00389	 0.00128 0.000865

Comparison Between Cassette-Cartridge and Standard Tape Systems

Characteristic	Cassette-Cartridge Systems	Standard Tape Systems
Max. On-Line Storage (Millions of bits)	1-10	1,200
Data Transfer Rate (Thousands of bits/sec.)	2-10	300
Price (with computer interface)	$2,000-$6,500	$10,000

Figure 2

pseudo-operations) and automatic desectorizing, a facility which allows the programmer of a machine with paged addressing to write programs as if the whole of core store were directly addressable. For larger configurations, typical offerings include a time sharing operating system and a real-time operating system to permit the concurrent execution of real-time and batch-processing tasks. The recent availability of inexpensive disc storage units has led to an emphasis on disc operating systems for minicomputers. Typically a full disc operating system offering multiprogramming requires at least 16,384 16-bit words of main memory.

The great difficulty with minicomputers has been in the preparation of application programs. With little storage and I/O capacity typical of most installations, little in the way of aids to the user-programmer can be run on the minicomputer itself. Neither has extensive software necessary to prepare and debug code for the minicomputers been available for users to run on large-scale host machines of their own, having ample facilities. As a result, applications programs have in general been prepared with such assistance as the target minicomputer itself offered and consequent low productivity by individuals charged with the task.

A common assumption is that minicomputer users, possibly well-educated in natural science or engineering, tend to have scant computer science background. Whether or not this is true in NASA has not been established. If it is true, however, then the spectacle of a scientist preoccupied for months with minicomputer programming problems must also be common in that agency. The result of this example of the fascination which hands-on programming holds for computer professional and novice alike may be the best program for a given problem solution. In general, however, it probably is not, compared with the result obtainable by a qualified programmer assisted by ample facilities resident on a large-scale system.

2.3 THE TECHNOLOGY THAT MADE IT POSSIBLE

Minicomputer prices have been declining at a relatively static rate since before the term minicomputer was coined, or about 1963. Figure 3 portrays this trend, which amounts to an annual decrease of 18% a year for computers identically configured. Figure 3 also shows the price differentiation between 8-bit and 16-bit machines, a more or less constant feature. During the same period performance as measured by the ratio of word length to main memory cycle time has also been increasing. The specific performance, or performance per dollar, equal to

$$\frac{\text{word length}}{(\text{cycle time})\ (\text{standard configuration price})}$$

is plotted in Figure 4, and reveals an uptrend of 50% a year. These trends, neglecting all but the mainframe and a single teletypewriter, reflect the revolution in semiconductor manufacturing technology which has occurred from 1965 to the present. They have also been observed, incidentally, among larger machines just prior to the advent of the minicomputer.

The upward trend in performance and downward spiral of minicomputer costs have been fueled by decreasing costs of logic circuits and memory cores and by improved speeds of operation for the cores*. For example, in 1965 a typical diode coupled transistor logic gate had a factory cost, when assembled on a printed circuit board, of about $2.70. In 1971 the same TTL gate packaged and mounted cost the factory $0.10, representing a compound reduction of 40% annually. During the same period the main memory core plane cost decreased from about 3.0¢ a bit to 0.5¢ per bit, equivalent to an annual drop of 27% a year. Figure 5 portrays these trends.

Not only did memory core elements get cheaper during the era of the minicomputer, they also were improved to operate six times as fast. Whereas cores of 50 mil outside diameter, capable of memory cycle times of 6 to 8 microseconds, were in use prior to 1964, machines introduced

---

* The material on core and semiconductor cost performance which follows is drawn from reference (5).

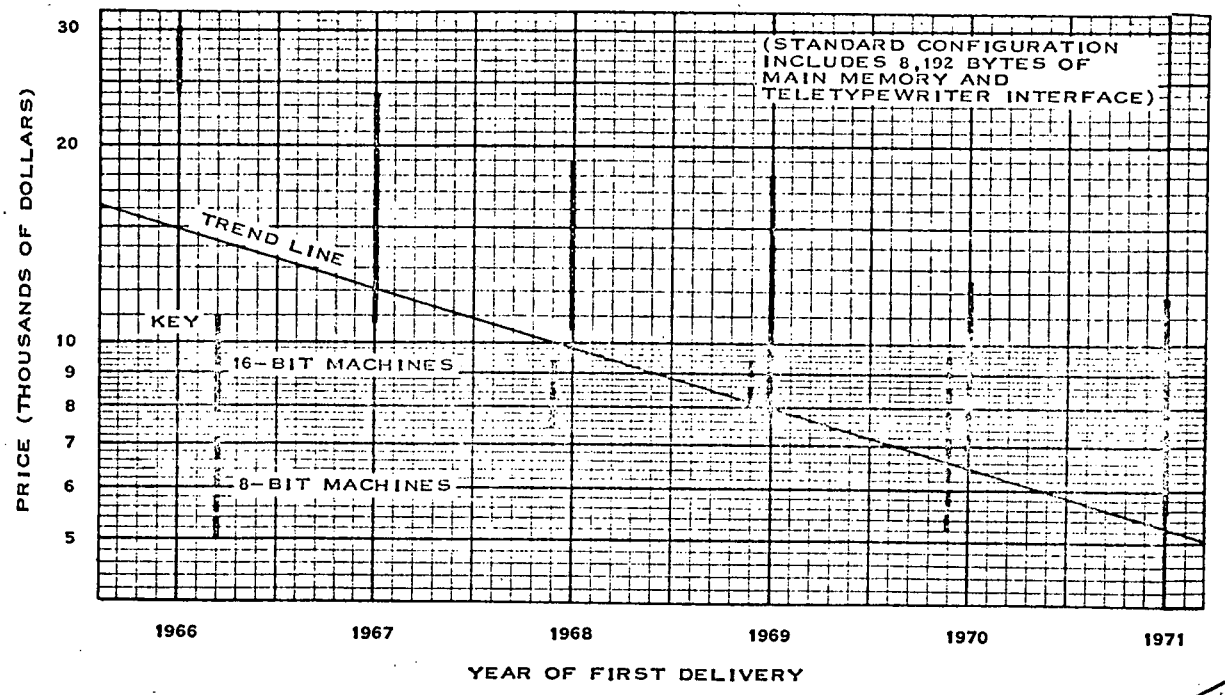

Figure 3. Standard Configuration Price Ranges

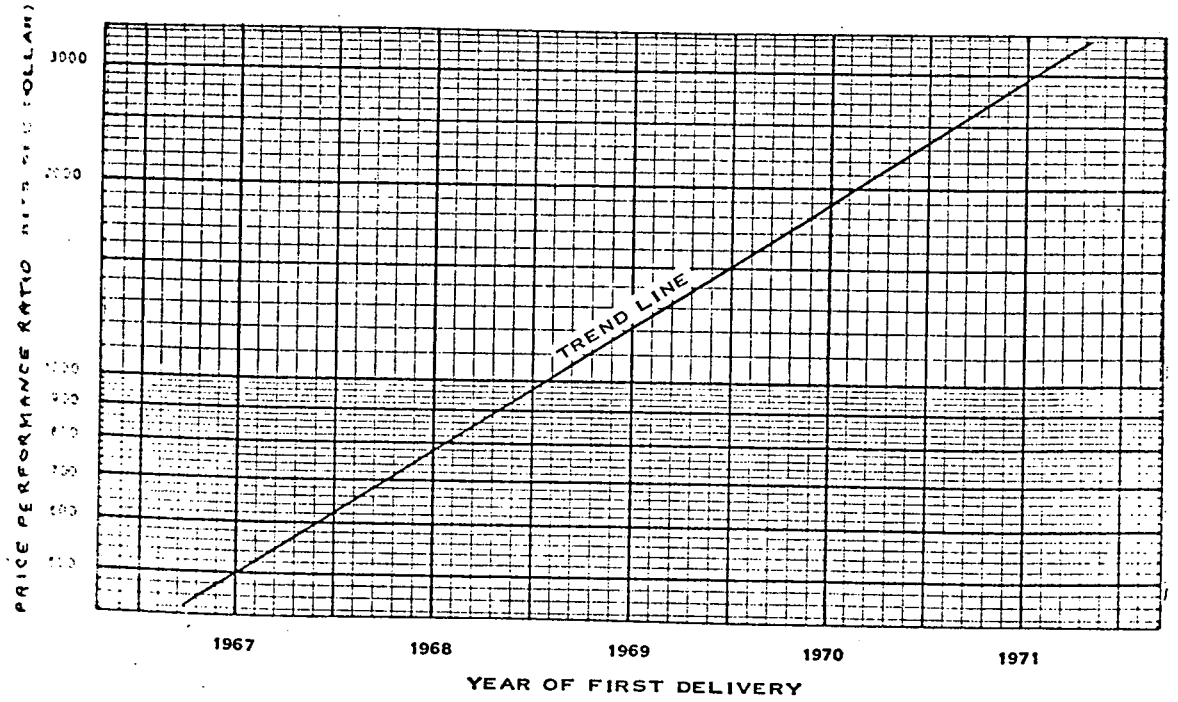

Figure 4. Trend in Price/Performance Ratio

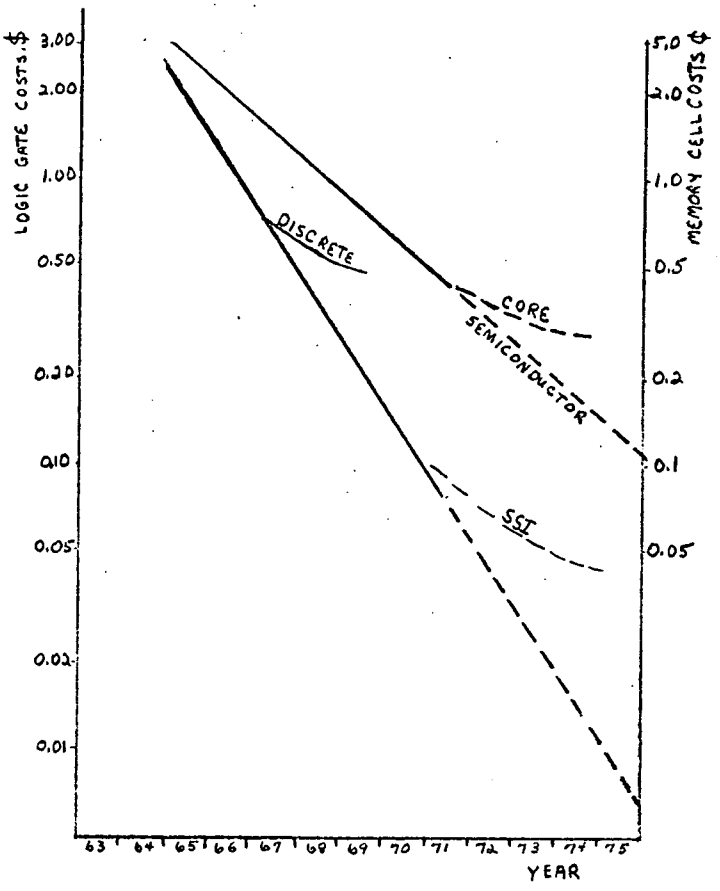

Figure 5. Trends in Logic and Memory Costs(5)

since 1969 have utilized cores of approximately 20 mil o.d. and had cycle times of about a microsecond.

Logic speeds did not improve. On the contrary, when integrated circuits were first introduced, they exhibited switching times of 40 nanoseconds, twice the 20 nanoseconds attainable by their discrete component predecessors. It wasn't until recently that 20 nsec gates have again become economical for routine use in minicomputer integrated circuits. The effect of the faster memories has been to require added CPU logic circuitry to keep up. For example, logic has been used in parallel adders and data paths, carry look ahead and faster decoders. At the same time, features which have been added to minicomputers have required more extensive logic circuits. The greater reduction in cost of logic circuits relative to core memory, shown in Figure 5, apparently has prevented the more lavish use of logic circuits from affecting the ratio of logic to memory costs of minicomputers. This is indicated by Figure 6, which shows that memory costs of listed minicomputers, representing three generations of logic hardware (discrete components, small scale integrated circuits, and medium scale integration), have remained steady at around the 37% of the total. This result does not depend on the selection of particular minicomputers but appears to be general with respect to the device.

	DDP-116	DDP-516	H-316	PDP-11	H 112
LOGIC	28%	25%	25%	24%	24%
MEMORY	33%	40%	37%	40%	44%
OTHER	39%	35%	38%	36%	32%

Figure 6. Factory Cost Breakdown for Minicomputers (5)

2.4 APPLICATIONS AND NUMBERS OF MINICOMPUTERS

At the end of 1970 approximately 25,000 minicomputers were in use in the United States, distributed as shown below:

Distribution of Minicomputers, Year-end 1970*

Petrochemicals	2,000
Electric Machinery Manufacturing	3,750
Other Manufacturing	4,000
Education	3,500
Printing and Publishing	750
Service Bureaus	2,000
Government	2,000
Hospitals	750
Laboratories	6,250
	25,000

Of these, it is estimated that 40% were employed in monitoring applications, 25% in controlling processes and the remaining 35% as stand-alone computers[7]. A further breakdown of these applications follows:

- Monitoring
  - off-line data collection 20%
  - discrete event data collection 10%
  - monitoring continuous processes 10%
- Control
  - continuous process control 12%
  - discrete events control (e.g., automated testing) 13%
- Stand-alone computing
  - autonomous problem-solving (scientific) 12%
  - small business, typesetting, computer aided instruction 12%
  - key entry systems, communications 11%

* Figures adapted from references (6) and (7).

Further growth in the numbers of minicomputers shipped has been forecast[7] as follows:

Year	Number of Units
1971	11,300
1972	14,000
1973	16,600
1974	19,600
1975	22,000

It has been estimated that minicomputers will account for about one-third of NASA's approximately 1200 computers by July 1973. While projections of the NASA inventory of minicomputers are not available, figures comparable to those reported above for the total market would range from 150 minicomputer acquisitions in 1973 to 200 in 1975. Thus, NASA could have on the order of 900 minicomputers by the end of 1975 if it acquired them at the rate predicted for the user community as a whole.

Detailed studies of the numbers of minicomputers in NASA and their applications are not available at time of writing, but it is assumed that laboratory and telemetered data acquisition applications outnumber others, which include:

- autonomous scientific problem solving
- automated checkout
- navigation and guidance
- simulation systems operation
- communications
- control of computer peripherals and "front end" processing.

2.5 PROBLEMS IN THE USE OF MINICOMPUTERS

The need for careful specification of functional requirements is not limited to large scale data processing systems. The minicomputer shares this need, albiet on a smaller scale, with larger systems. It has been estimated that the costs of rectifying a blunder in specifying the ADP system to do a given job can exceed the cost of the equipment by a factor of ten. Even so, prospective minicomputer owners, faced with a small budget for getting their new system up and operating, have in the past overlooked this crucial fact and slighted the important preliminary phases dealing with requirements definition and system specifications. A remedy for this condition is recourse to an original equipment manufacturer, whose function it is to deliver complete systems tailored to the intended application and guaranteed to work as specified. Alternatively, groups skilled in system analysis, specification and design can be established within user's organizations large enough to make such a solution economical.

Many minicomputer applications have fallen outside the precincts and purview of the large-scale organizational computer center and its pool of personnel with computer-related skills. Minicomputers used in laboratories for scientific problem solving and data acquisition exemplify this class of applications. Users of these minicomputers may have scant previous experience with computers and be poorly equipped to develop programs on their new purchases. In the past, minicomputers -- especially those at the low end of the size scale -- typically afforded such novice programmers only an assembler, and lacked compilers and debugging packages to make his work easier. In addition, storage was limited and the loading of programs proceeded at the very slow rate of ten bytes per second afforded by a paper tape reader. Since then, compilers have been added to the software offerings of most minicomputer manufacturers, and storage of relatively generous capacity is available, as are high speed input devices such as magnetic tape cassettes. These peripherals can run the cost of a minicomputer well up in the five figure range, however, and may defeat the objective of low-cost computing, especially if they are not needed for production runs of the programs, once written and debugged.

Hence the large-scale machine, if our minicomputer owner has access to one, offers an attractive alternative to writing programs on the minicomputer. Cross compilers, simulators, etc., operating on large-scale host computers for the purpose of producing code for minicomputers, have not been generally available for this purpose up to the time of writing.

The diversity of the minicomputer industry may present problems to the large institutional user. For example, the number of minicomputer manufacturers, which is declining somewhat now, hit a peak of over 60 in 1970; each produces several models. In addition, technological improvements cause the obsolescence and replacement of models in each manufacturer's line at intervals of two years or so. In 1970 Bell Telephone Laboratories was reported[8] to have 120 minicomputers in use, consisting of 34 models supplied by 12 different manufacturers; these numbers are undoubtedly larger by now. This raises questions of support: How can maintenance for this inhomogeneous brood be simplified? Can software development be shared? Can peripherals be swapped from one installation to another? How can one avoid building a special software interface for each mini requiring access to another computer? The larger the number of computing systems, the more pressing does the need for answers to these questions become, in order to use the investment efficiently.

Programming costs could also present problems to a large institutional user. Assuming for the moment a ratio of one programmer per owned minicomputer, NASA's estimated total of 900 minicomputers in 1975 would require annual programmers' salary costs in excess of $15 million. This figure, roughly 6% of the annual ADP budget, approximates the purchase price of the minicomputer hardware. At this magnitude, minicomputer programming costs may be acceptable, but the figure could be much higher if, for example, scientists rather than full-time programmers program the minis. Also, administrative costs were not included in the $15 million estimate and could account for a substantial increase. If means can be found to promote efficiencies in the application of programming resources, such as eliminating unnecessary duplication, they should be exploited.

2.6 NASA HEADQUARTERS ADP MANAGEMENT RESPONSIBILITIES

The relevant mission of NASA Headquarters in the role of ADP management is technical review of plans to acquire data processing systems. Accordingly, the adequacy of the analysis and design which go into the following features of such acquisition plans must be evaluated:

- Requirements definition
- System specification and design
- Equipment selection
- Implementation plans
- Support plans
  - Facility
  - Personnel
  - Maintenance
  - Operation
  - Replacement

An object of the review, in addition to that of preventing technical errors, is the greatest effectiveness possible in the application of ADP resources. Efficiencies which can be promoted from Headquarters include those due to utilization of surplus NASA equipment, lowered purchase costs due to large volume buying, the establishment of standards and underwriting development effort.

The particular problems arising from the use of minicomputers and singled out by NASA for solution are those described previously in the discussion of programming costs and high design, maintenance and replacement costs due to the diversity of minicomputer models. In other words, is there something that can be done to contain the costs of programming minicomputers? And can the effort which goes into hand tailoring each mini installation somehow be minimized?

NASA has suggested a compatible family of minicomputers as a solution to these problems. Their compatibility would be such as to guarantee the interchangeability of machines of equivalent power, without the need to revise physical interfaces. Programs would also be able to

operate on all minicomputer family members of power equivalent to or greater than that of the mini for which the program was written. The specification of such a family of minicomputers, their peripherals and software is the object of the development plan contained in this Appendix.

SECTION 3. MINICOMPUTERS - DETAILED DESCRIPTION

3.1 HARDWARE

3.1.1 Processing Unit

Normally, the central processor includes a program counter, an accumulator, a register which extends the accumulator to double length, and one or more index registers. Sometimes a condition register keeps track of processor status in respect to overflow, the operation mode, or the result of a comparison. Central processors vary most in the number of accumulators provided, the instruction sets implemented, the instruction decoding technique, and interrupt handling capability.

Most 16-bit processors use one-word instructions with the following format: a four- to six-bit operation code, a two- to four-bit modification field, and an eight- or nine-bit address field. Most of the operation codes are used for memory referencing instructions. Non-memory referencing instructions use additional bits of the instruction word as part or all of the operation code; thus the number of instructions is not necessarily small. Most have an instruction set of 64 to 100 instructions; some have many more, over 200. The modification field further defines the instruction, usually defining the addressing mode

(indexing, indirect addressing, or both) or specifying a literal (immediate addressing).

The basic instruction set usually includes the arithmetic operations of fixed-point add and substract; multiply and divide are implemented by subroutine but usually are available as optional features. Double-precision fixed-point add and subtract are sometimes provided. A few of the larger minis offer floating-point hardware as an optional feature but this feature is usually expensive, sometimes costing more than the processor. All offer some form of logical, compare, and shift operations. Many also offer byte manipulation instructions. The input/output instruction is usually very general and complex. It transfers control, status, and data words between the peripheral devices and the processor's accumulator. Commonly, the input/output instruction also provides control for optional features, which are addressed as input/output devices.

Several minicomputers include read-only memories (ROM) either as basic hardware or as optional equipment. Minicomputers that use ROM to store microprograms that define the processor's instruction set attain a high degree of flexibility but sacrifice speed of conventional instruction execution. Despite the high speed of ROM memories (up to 200-nsec cycle rate) and the high speed of microinstruction execution times, a conventional instruction, such as add the contents of a memory location to the accumulator, consists of several microinstructions, and the total instruction execution time increases.

3.1.2 Interrupt Systems

The interrupt systems for minicomputers are extensive and consist of internal and external interrupt levels. Internal interrupt levels, sometimes called traps, include power failsafe, memory parity, memory protect, and real-time clock interrupt levels. Usually the memory protect level can be inhibited from the operator's console, but the other internal interrupt levels cannot. The basic external interrupt system

usually consists of one, two, or four levels; additional interrupt levels can be added in modules of two, four, or eight levels up to a total of 16, 32, 64, 128, or 256 levels.

In an external interrupt structure with but one level, all interrupts, regardless of their cause, transfer control to a standard software routine that determines the cause and branches to a set of instructions to process the condition. The interrupt routine tests the status of devices in the order of the required response time for each device from the shortest to the longest. Thus, the device with the longest required response time will be serviced only after all other devices have been serviced. While the interrupt is being processed, no other interrupts can be serviced. The overhead time required to identify the cause of each interrupt, and the possibility of losing other interrupts while this function is being performed, are disadvantages.

External interrupt levels are under program control and levels can usually be individually disarmed or inhibited. A disarmed interrupt level ignores an interrupt signal. An inhibited interrupt level stores an interrupt signal but does not cause an interrupt until the inhibition has been removed. Some minicomputers provide enough levels (usually as options) to allocate a different level for each interrupt cause. The main advantages of multi-level interrupt systems are the reduction of the software overhead for interrupt identification and the ability to nest interrupts; i.e., a high-priority interrupt can disrupt the processing of one of lower priority.

A common hardware interrupt sequence is to suspend processing and execute a transfer of control instruction to an indirect address in a core location selected by the interrupt level. Sometimes, the hardware sequence includes storing the processor status before transferring control to the interrupt servicing subroutine. Some hardware provision is made to block out all interrupts until the interrupt servicing subroutine has stored the status of the processor, the contents of the accumulator, index register, program counter, overflow, and so on. In addition, hardware

provision is made to block out all interrupt levels of an equal or lower priority until the current interrupt level is released by instruction.

3.1.3 Memory

Typical memory sizes are from 1,024 to 65,536 eight-bit bytes or 4,096 to 32,768 16-bit words. Data is transferred between memory and the central processor via a memory bus. Few minicomputers can afford more than one memory bus; thus even with a direct memory access option, processing is suspended (i.e. the I/O device steals a processor cycle) when data is transferred between memory and high-speed peripheral devices.

Because only 8 to 10 bits are usually available for the operand address, most minicomputers use a paging technique to address memory. A page is 256 to 1,024 words long, and a memory referencing instruction can directly address only the first or current memory page. The current page is the page from which the instruction being executed was drawn. Indirect addressing increases the instruction execution time by one memory cycle per level, but indexed addressing usually does not increase the instruction execution time. In indexed addressing, one or more index registers supply the added bits necessary to address all of core. Usually, but not always, both indexing and indirect addressing are provided. In some cases, an address extension register or double word-length instruction provides for directly addressing core. In any particular minicomputer, the addressing techniques are usually a combination of several methods. For example, the Data General Nova has direct, multi-level indirect, and indexed addressing capabilities.

Addressing schemes are summarized in Figure 1. The limitations of each method indicated in the figure are not always apparent to the programmer because the system software (assemblers, loaders, and language compilers) frequently allows the programmer to write instructions as if the whole of main memory were directly addressable from every memory reference instruction.

Method	Description	Problems/Limitations
Direct Addressing	Instruction format includes field large enough to address any main memory location.	Number of required bits either curtails main memory size too severely or the instruction length slows execution time, and makes logic expensive, (because instruction extends over several words)
Indirect Addressing	Address in instruction specifies a memory word that contains the operand address (multilevel indirect addressing is sometimes permitted)	Time consuming due to extra memory cycle required to fetch address word.
Paging	Memory divided into sections called "pages", typically 256 words. Address in the instruction refers to address within the current or base page. Usually the base page is accessible by any instruction.	Overhead required to specify page. Constraints on organization of program to be met by programmer or software
Relative Addressing	Operand address interpreted as a displacement from address of instruction using it.	Constraints on program organization leads to problems when instructions are added or removed from an existing program
Indexing	Address specified in instruction is added to contents of an index register to obtain effective address	Overhead associated with setting up index register contents. Time consuming if index register implemented as a main memory location.

Figure 1. Minicomputer Main Memory Addressing Methods

Most minicomputers provide memory parity and memory protect as optional features, although some offer one or both as standard features; and some offer no memory parity option. The latter is infrequent except in cases where the core cycle rate is slow and memory has proved very reliable.

Memory protect is most frequently implemented by upper- and lower-bound registers that define the protected core area. Some increase the word length to include a protect bit in each word. Memory protect usually includes a protected set of instructions, such as instructions that load the upper- and lower-bound registers or change the interrupt status and input/output instructions.

Both memory protect and memory parity usually include an interrupt level to signal a protect violation or a parity failure. Because of the nature of core memory structure, adding a bit to the word length for memory protect or parity checking increases the cost and cycle time of memory far beyond the proportion of bits added to the word length.

3.1.4 Input/Output

Minicomputers include a programmed party-line input/output channel. The data channel is one word wide, eight bits for an eight-bit processor, 12-bits for a 12-bit processor, and so on. There are two methods for I/O transfers:

- Programmed input/output -- a computer instruction is executed each time a data word is transferred
- Autonomous block transfers -- a single instruction initiates the transfer of a block of data.

In both cases, the processor is free to perform other functions while the data transfer is in process and an interrupt signals the completion of the requested transfer.

Programmed Input/Output is basic to minicomputer. In this method, the input/output service sequence consists of determining which device requires service and the type of service required, of transferring the data word between the accumulator and the input/output device and between the accumulator and the appropriate location in memory, of counting the word transferred, and of testing to see if all words have been transferred. Thus, input/output transfer rates are affected by the memory addressing techniques, the instructions provided for controlling and testing counters, the number of interrupt levels, and so on. Typical minicomputers have maximum data transfer rates of approximately 50,000 bytes per second with this method.

Once initiated, autonomous block transfers proceed independently of any stored program. Special hardware controls the transfer by means of registers containing the addresses of the first and last main memory locations involved in the transfer (or equivalent information), and a pointer to the memory location currently involved (or, equivalently, a counter of the number of words already transferred). The controller terminates the transfer and generates an interrupt when the specified transfer is completed. Two kinds of autonomous block transfer capabilities may be provided. The Direct Memory Access (DMA) feature uses special hardware registers within the controller to control the transfers. The data-break or multiplexer feature uses dedicated main memory locations for this purpose. Of the two types, the Direct Memory Access feature is more expensive (because of the cost of the special-purpose registers) but faster, being limited in speed only by the rate at which data can be transferred between the data channel and the computer main memory. The data break or multiplexer approach is slower than a DMA channel because it requires several memory accesses for each data word transferred to update memory locations and to compare the progress of the transfer with the finishing state.

Most minicomputers can support an optional direct memory access (DMA) channel. The channel includes a one-word data buffer, a current address register, and a limit address (or word count) register

to provide for automatic transfer of blocks of data. Two pairs of core locations are dedicated to the channel to provide two sets of initial and limit addresses for the channel's current and limit address registers. The program need only initiate the block transfer and replenish the limit addresses as each block is transferred. Because most minicomputers have only one memory bus, transfers via DMA suspend processing (called cycle stealing) if the processor and DMA try to access memory simultaneously; the DMA channel has priority. Several devices can usually interface to a DMA channel, but only one device can use the channel at a time.

In the DEC PDP-11 family, first announced in January of 1970, DMA hardware is incorporated in most peripherals, and the design of the Unibus which connects them permits data transfers between peripherals independently of the processor. This departure from previous designs was occasioned by the use of fast semiconductor main storage. Apparently it was necessary to relieve the processor of all possible input-output chores in order to enable it to "keep up with" the rapid memory, which therefore was fitted with dual ports to allow the processor and a peripheral simultaneous access.

Most minicomputer manufacturers do not make their own peripheral devices but buy standard devices and provide the controllers for industry standard input/output devices: high-speed paper tape units, punched card readers and punches, line printers, magnetic tape transports, plotters, displays and Teletype Units ASR/KSR 33/35. Almost all minicomputers provide an optional mass storage device, disc or drum. Magnetic tape cassettes are being offered by a number of manufacturers for mass storage. In addition, some manufacturers offer extensive lines of analog/digital and digital/analog systems and/or data communications equipment.

3.2 SOFTWARE

The kind of software required by a system depends on its hardware configuration. Software for a system with mass storage, such as magnetic tape, drum, or disc is quite different from software for a system without mass storage. Many manufacturers provide modular software, and each module requires a specific minimum hardware configuration, number of memory locations, optional features, interrupt lines, mass storage, and peripheral devices. Generally, developing and running programs on a small hardware configuration is harder than for a larger system because less system software is provided, the facilities in the higher-level languages are less elegant, and the system is less automatic and requires more operator intervention, e.g., loading and unloading paper tapes in a prescribed sequence.

3.2.1 Minimum System Software

System software for the minimum hardware configuration should include:

- Diagnostic routines to check the operation of every unit in the system and detect malfunctions within those units supplied as spare parts
- Debug routines to diagnose errors in both source code and output code
- Input-output handlers
- Loaders capable of handling all systems software supplied as well as applications programs
- An assembler.

3.2.2 Assemblers

Current manufacturer-furnished system software universally includes an assembler. Many manufacturers supply more than one assembler for different hardware configurations, with the assembly language for a small configuration being a subset of the assembly language designed for a larger configuration. The following features characterize the effectiveness of assemblers:

- Number of passes of the source code
- Number of core locations occupied by the assembler program
- The number and types of diagnostic messages output
- Pseudo-operation codes provided
- Whether the output code from an assembly is in fixed binary format or in relocatable format
- The number and types of macros provided in the assembly language
- Whether the assembly language includes facilities for incorporating user-defined macros
- The extent to which macros can be nested

Most assemblers require a minimum of two passes of the source code to produce an assembled program, and a so-called one-pass assembler either leaves many references to be resolved by a loader or is really a two-pass assembler but does not require that the source code be read from the source code input device twice. The first pass checks the source code for syntactic errors and builds the symbol table in memory. The second pass completes the assembly and tags unresolved references for the loader. If the two-pass assembly program must be input twice, it will seriously prolong assemblies on small configurations with but a single paper tape reader. For larger systems with mass storage, this usually is not a problem.

The number of core locations required by the assembler subtracted from the total number of core locations in the system defines the maximum size of the symbol table, which indirectly determines the length of programs that can be assembled. Symbols are data and program tags or labels. For short word-length computers, each entry in the symbol table probably includes more than one world.

Pseudo-operation codes do not generate machine-language output code but direct the assembly program: define data, specify conditional assembly, equate labels, and so on. The data definition pseudo-operations

are especially important if the application areas require different data formats. The data formats provided should correspond to those required by the applications.

The number and types of diagnostic messages output determine the ease with which source code can be debugged. Most assemblers combine two or more error types into some of the diagnostic messages and print out the code along with the statement in error and its location in core. Usually the statement will give the programmer a clue as to which error was made, but a user must be wary of diagnostic messages where many errors are tied to one message. They indicate little more than the fact that an error was made.

A relocatable assembler produces machine-language output code relative to a base address. When the assembler program is loaded for execution the loader adds the base address appropriate for the current memory map; thus the program can be stored and run anywhere in memory. A fixed, binary assembler produces output code that must always be loaded and executed in the same area of memory.

A macro assembler treats certain functions that require several machine language instructions for execution as one instruction. The assembler takes one source-language statement, generates all the instructions needed to implement the statement, and inserts the group of instructions into the output code. Macros make programming in the assembly language easier and make source language programs shorter and less subject to error. If the macros provided are not appropriate to the particular application, however, this indicates the manufacturer has not had much experience with that application, and having the macro assembler is of no value.

The facility for incorporating user-defined macros into the assembly program is a powerful facility for tailoring an assembler to a specific application. Frequently required functions can be programmed and incorporated into the assembler and thereafter treated as system macros.

The extent to which macros can be nested determines the ease of programming user-defined macros, particularly complex ones. Nesting macros means that one macro can use another macro in its definition, the s econd macro can use a third macro in its definition, and this procedure can continue up to the number of levels allowed. This facility is important if different, complex application programs will be programmed, and the programs do not have complex functions in common but do have simpler functions in common. The system macros should include only the simple functions; each program will build up its complex functions from the simpler ones.

Another important characteristic of an assembler is its assembly speed. For applications where the same programs are run repeatedly, the assembly speed is of little importance; but if different, new, short problems will be run, assembly speed is significant.

3.2.3 Compilers

Fortran is the most common higher-level language offered with minicomputers. Currently, two Fortran languages are defined by USASI (United States of America Standards Institute); these are Basic Fortran (formerly Fortran II) and Standard Fortran (formerly Fortran IV). Manufacturers usually define their Fortran languages in respect to one of the preceding two languages. Despite standardization efforts, Fortran compilers are machine-oriented; and Fortran programs using the Fortran language for one system will not run on another system. A Fortran compiler for a particular system either does not include all of the features of a standard Fortran language or includes additional features not defined in the standard Fortran language.

As with assemblers, other important features are the hardware configuration required, the core storage required by the compiler, the size program that can be compiled, and the number and kind of diagnostic messages output. An important compiler criterion is the efficiency of the output code. In addition, the compiler should allow the incorporation of assembly-language subroutines.

3.2.4 Operating Systems

Operating systems for minicomputers are becoming increasingly important, particularly for systems that include mass storage devices. Most operating systems are of the foreground/background types; one or more real-time programs can be executed in the foreground and one batch program can be executed in the background. Batch background programs are sometimes priority oriented. Time sharing operating systems for minicomputers are also available.

Foreground/background operating systems make minicomputers suitable for real-time control applications, as well as increasing the efficiency of the overall computing system. Real-time programs are incorporated into the operating system and are executed in the background. The important feature of these systems for control applications is that new real-time programs can be debugged and prepared for incoporation in the operating system without closing down the system.

Time sharing operating systems are a variation of the foreground/background operating systems. Instead of real-time control programs being executed in the foreground, time sharing users execute their programs in the foreground, and batch programs run in the background.

Operating systems vary in complexity depending on the kinds of applications for which they were designed. Most manufacturers who include operating systems in their software packages offer modular systems with more features available as the hardware configuration increases in size. Operating systems perform all of the operations associated with the following functions:

- Communication between the operator and the system
- Input/output
- Interrupt system
- File manipulation

- Processor status
- Initiation of program execution
- Core assignment

User programs have access to the preceding facilities only through the operating system to insure against inadvertent destruction of the programs in core.

The manufacturer designing the operating system makes various assumptions on which the system is based. These assumptions or system parameters are based on the system hardware and the application for which the hardware will be used; and they include the average core storage required by a foreground or background problem, the maximum number of foreground problems, the maximum number of priority levels allowed, the type of programs that can be run in the foreground and background, the system software a user can utilize, and things of this sort. Thus, a particular operating system can be too big and complex, too small and simpleminded, or exactly right for a particular application, depending on the parameters used in the system design.

The operating system allocates memory for all program executed. Normally, the memory map includes three memory areas; the area assigned to the resident portion of the operating system, the area assigned to the resident applications programs, and the area assigned for temporary use by all other programs executed.

Operating systems also vary in the range of system software packages they can supervise, the ease of inserting programs into the batch stream, the protection safeguards for user's files, and the facilities for segmenting large programs to fit the system.

3.3 APPLICATIONS OF MINICOMPUTERS

While the general purpose computer has found application in an extremely wide range of problems, the minicomputer owing to its more restricted capabilities has been used in a lesser variety of ways. These have tended to concentrate on industrial uses where a suitable application,

data systems know-how and the not inconsiderable capital required to design and engineer a minicomputer system existed. The applications generally recognized, moreover, require minicomputers having somewhat distinct characteristics, so that a partitioning of minicomputer types along these lines is not inconsistent with actuality and convenient for discussion. These types are:

- Process monitoring and control
- Peripheral control
- Time sharing
- Autonomous problem solving
- Data acquisition and communications processing.

3.3.1 Process Monitoring and Control

Processes typical of industrial applications are petrochemical refining, chemical plants and even cookie baking. In these, the computer receives online inputs measuring the process underway, computes the required control values, such as valve positions, and generates outputs to effect the needed control actions. Analagous NASA applications are in the operation of space flight simulators and control of certain ground-and spaceborne experiments. Some applications, like monitoring of a building for industrial security, are confined solely to monitoring.

Most process control applications require very flexible input-output equipment capabilities, to accommodate a variety of monitoring and control devices. Because of the diverse data rates typical of the I/O devices, several DMA channels are preferable to a single, multiplexed input/output channel. Also, multiple priority levels will be needed to avoid excessive interrupt processing. A real time clock is generally a necessity, and logging and timing operations, and debugging features which aid in isolating problems in a highly asynchronous system, are desirable. If an interruption would be catastrophic to the process, the computer must have a high MTBF and a low mean time to repair. These applications are among the few which may require resistance to unusual environmental conditions, such as temperature extremes, vibration and the presence of abrasive or corrosive substances.

A mathematically oriented programming language should be available, and it should permit subroutining, to deal effectively with the wide variety of input/output devices. Utility programs should include ample mathematical routines.

Process control applications are more individualized than others. Consequently, a rather considerable amount of custom development has been necessary to adapt the minicomputer system chosen to its particular application.

3.3.2 Peripheral Control

Minicomputers are employed to control input and output equipment for large computers. In this role they take over these functions from the operating system, liberating it for the more demanding computation performed on the large machine.

In this application, it is important for the processor to respond quickly to interrupts, so multiple levels are required. Furthermore, it must have sufficient capacity to respond to its host without losing data in the worst possible interrupt conflict situation. Features which promote this ability are: scratch pad memory, general purpose registers and multiported memory.

Naturally, the minicomputer must have an expanded signal repertoire, with which to communicate with the large machine's peripherals. If it is controlling storage systems, power failure interrupting will be required to save data or prevent physical damage to discs.

Software requirements for the peripheral-controlling mini are not extensive. Data testing and moving commands should be included in

the language supplied, and the code generated from any programming language should be efficient.

3.3.3 Time Sharing

The response time of a system dedicated to a number of concurrent users is its primary figure of merit. As response entails swapping large program segments between central and auxiliary storage, features which tend to speed this traffic are needed. They include a high maximum input-output rate, assisted by one or more direct memory access channels. Many levels of priority interrupts are needed for rapid system response to user demands and to extraordinary situations, such as a power failure. Many of the general purpose computer's arithmetic features, such as floating point and multiply hardware, are advantageous in speeding user service.

Much file storage is needed in time sharing systems, which requires of the minicomputer system the ability to accommodate a quantity of disc or other extensive secondary storage. Memory mapping and memory protection features are indispensable for partitioning central storage efficiently and safely.

In addition to a time sharing executive, system software must include interactive procedure languages for programming and for file manipulation.

3.3.4 Autonomous Problem Solving

Each stand-alone minicomputer system has specific characteristics related to the nature of the problem being solved. Examples of these applications range from computer-assisted design through a dedicated information storage and retrieval system all the way to the first computer application, that of computing ballistic tables. The requirement for peripherals (input-output and storage), programming languages for computation, throughput capacity, etc., are all dependent on the application selected.

3.3.5 Data Acquisition and Communications Processing

Data acquisition systems are used as first level filters for large amounts of source data. Examples in NASA are decommutating, assembling and forwarding data from satellite-borne experiments. Because communications processing applications are similar in many respects to data acquisition, they are included in this category.

The primary consideration in this application is sufficient input-output bandwidth to accommodate the anticipated peak data rate. Computation requirements are not generally difficult to meet, but attaining them must be assured even during peak rate data transfers. For this reason, multi-ported memory and DMA are desirable in order to prevent the flow of data from interfering with processor operation, as it would in the case of a single-ported memory.

Extensive file manipulation is not usually required of the data acquisition system, obviating the need for disc storage and its operating system. An elaborate input-output structure is not required in most data acquisition applications, where a single multiplexed input channel will suffice. Similarly, the communications processor functioning as a "front end" to a large computer system need not be required to respond rapidly to complex demands either from its host or the communications channels serviced and can forego both the multi-level interrupt and elaborate input-output structures.

Other important features include bit and byte manipulation instructions which permit efficient data moving and formatting. Where a system is to operate unattended, power failure interrupt and an automatic restarting capability will be required. Memory parity and some form of hardware error detection capability may also be required. If statistical analysis is required, mathematical utility programs could be a necessary feature.

SECTION 4. STATEMENTS OF THE PROBLEMS

The basic problem to be solved in the subject development effort is that of establishing sufficient specifications and standards for minicomputer hardware and software to provide NASA with realizable economies in quantity purchases, interchangeability of minicomputers, software, storage and peripherals, and uniformly high quality. Inherent in this problem, as it is in the general problem of setting standards, is that of avoiding being more comprehensive and restrictive than necessary to achieve such goals. It is of fundamental importance, therefore, to state goals and objectives explicitly and to make them both attainable and demonstrably worth attaining.

The problem extends beyond the subject of minicomputer hardware and software to quality assurance and maintenance. It is difficult to establish meaningful specifications and enforceable standards in any one area without doing something similar in other areas. The ramifications of the basic problem, therefore, extend to virtually all technical aspects of computer and peripheral hardware and software specification, fabrication, integration and operation.

4.1 GOALS AND OBJECTIVES

The goals and objectives stated below are not mere euphemisms or "motherhood" resolutions. They are attainable, their attainment can be established, the path to attainment can be laid out and progress can be measured; they are not, however, easy or inexpensive to attain.

4.1.1 Goals

The overall, long range goals of NASA with respect to the subject development effort are as follows:

- To achieve economy in purchasing minicomputers by "batching" its purchases according to some arbitrary size-speed categories and exploiting quantity discounts and other large-purchase advantages.
- To achieve flexibility, enhanced availability and economy by providing for reasonable interchangeability of both hardware and software, between manufacturers or suppliers, and between successive technological generations.
- To achieve measurable performance, quality and reliability of product so as to make these properties independent of supplier and technology.
- To achieve economy in operation and maintenance by providing modularity in design and commonality in spares provisioning and maintenance tools, instruments, procedures and training (at the system level).
- To achieve economy and speed in developing applications software by providing NASA users and their programming suppliers with large-computer power on large computers to develop completely ready-to-run small computer applications software.

The foregoing goals are qualitative, general statements of intent. In the next paragraph these general goals will be translated into objectives, which are rather more specific but are still long-range and fairly comprehensive. These objectives will then be examined more closely to identify the specific problems that must be solved in meeting each objective. In general, these problems will consist of establishing

some minimal set of characteristics comprising a specification or standard necessary to meet each objective, and to provide numerical values to sets of those characteristics.

Before stating the objectives in broad terms, it will be useful to define a few terms that will be used in expressing the objectives.

4.1.2 Definitions

4.1.2.1 Subsystem Type. We shall speak of several kinds of subsystems or components of complete, operation systems developed to meet a specified end use (application). These kinds of subsystems or major components we will call "subsystem type;" they are as follows (not intended either as definitive or restrictive):

- Types
  - Processor (exclusive of read/write storage)
  - Storage
  - Displays
  - Operating software
  - Program development software
  - Application software.

4.1.2.2 Category. Each of the foregoing types can be realized in different ways, called "categories." Examples of categories are:

- Processor
  - Central Processor
  - I/O Processor
  - Peripheral Controller
  - Selector Channel
  - Programmable Controller
  - Micro-processor

- Storage
  - Main Storage (Core, MOS)
  - Random Access Auxiliary Storage (Disc, Drum)
  - Serial Access Auxiliary Storage (Tape)
  - Discrete Storage (Punch Cards, Magnetic Cards, Slides)
- Displays
  - Printers (Serial and Line)
  - Cathode Ray Tube (CRT)
  - Plotters
- Data Entry Devices
  - Keyboards
  - Light Pen
  - Joy Stick
  - Pantograph

4.1.2.3 Levels. Each type of subsystem can be classified according to some measure, generally indicative of size, capacity, speed, and at any given time, cost. We shall speak of several ranges for a particular type and category of subsystem or component as capacity level. Each level may involve several variations in which one or another parameter is emphasized according to application. Thus, at each of several levels there may be several variations. The basic idea in the development of such a family is to do so in response to specific applications. When a new application arises, and it does not fit well with variations that have already been developed, and the prospective "bug" is of sufficient magnitude to warrant a new variation, a new member may be incorporated into the family. The object is to provide common features to all members of a particular family at a particular level, and that the subsystem at the next higher level incorporate the logical sum of all the lower level features so as to make it able to substitute for any lower level item.

4.1.2.4 Test System Configurations. The levels discussed in the previous paragraph applied to hardware (or languages or software) of a particular type and category. Here we are speaking of systematic collections of several types and categories of subsystems and components to comprise a series of systems of increasing complexity. At the bottom end,

for example, we might have a simple collection of a central processor, a main storage, a teletypewriter, a representative application (bench mark) program, input data in machine-readable form, and a document describing the system and its uses for the several different classes of persons involved: purchaser-user, system engineer, programmer, test engineer.

At the other extreme we might have a network of processors of various types, main and auxiliary storage devices, display and data entry devices, operating (executive) software, test software, test data in machine-readable form, correct test results, instruments (e.g., oscilloscopes, recording voltmeters) and test documentation including acceptable test results.

The purpose of such a series of standard test configurations is to permit users to measure system response (application program running time, system response time, etc.) for a given candidate or set of candidate hardware.

The foregoing definitions are intended to be illustrative of the kind required to be more carefully advanced in the course of a development study, and are not intended to restrict such an effort. They will serve the purpose, in this Appendix, to permit further description of the development effort, and to permit meaningful goals and objectives to be established, and questions to be raised. These matters will be discussed in the paragraphs that follow..

4.1.3 Objectives

- The hardware built to a given specification by various manufacturers will be interchangeable in a minicomputer system (e.g., a PA22 built by DEC will be interchangeable with one built by Data General).
- The functions and minimum performance requirements of a subsystem at a given level in the family will be incorporated in the specifications of the subsystem at the

next higher level. Thus, equipment at a given level may be substituted for equipment at the next lower level of the family tree (e.g., a PA22 computer can replace a PA221, but not a PA213).

- Software designed to conform to a computer of given specification will run without modification on any computer built to that specification (e.g., an executive or application program will run on any manufacturer's PA 121).

- Software designed to conform to a computer at a given specification will run without modification on a computer built to the next higher specification on the family tree.

- Program development software (assemblers, compilers, program aids, etc.) will be written for (designated) large (NASA) computer(s) to convert (designated) source languages to the machine language of any of the processors covered by the specifications, as designated at assembly/compile time to the conversion program. The object is to provide a capability on a big machine to prepare ready-to-run programs for a minicomputer system, using all of the power and advantages that large machines have for normal (its own) program development software.

- Executive software will be written for each computer family tree, and will be comprehensive enough to cover all specified standard test system configurations. Executive software will be so partitioned that various subsets of modules can be used for appropriate levels of test system configuration complexity, for each level of the corresponding family tree. (It will be assumed that high-speed storage requirements will accompany each module, and that system response times can be estimated for various test system configurations using standard tests.)

- Standard tests will be designed and standard test data prepared, with the correct values being provided and tested by the test software. Bench mark programs will, therefore, not only provide general system check-out and running or response times, but also accuracy and precision (if appropriate) of test results. Instruments to test electrical and electronic circuit responses to test programs will be included. Test programs will be designed for various standard test configuration at all size-speed levels.

- Standard procedures and tests for measurement of reliability, maintainability and repairability will be developed.

- Specifications and standards will not include:

  - program running times
  - system response times
  - failure rates
  - reliability

  Such information, or data needed to calculate such information, will be measured using the standard test procedures. Equipment of various manufacturers, or particular total systems, can then be accepted or rejected based on needs and requirements of the procuring agency.

4.2 DESCRIPTORS AND MAJOR PARAMETERS FOR THE NASA STANDARD SERIES OF MINICOMPUTER SYSTEM ELEMENTS

This is the first of a series of problem areas, corresponding to the objectives, that are candidates for study in the proposed development project. The problem in each of these areas is essentially taxonomic -- classifying, naming, and organizing the elements into a systematic structure. The requirements for a particular application can then be analyzed, and specifications prepared and expressed in terms of the elements of that systematic structure.

In this first area, it is required to classify -- identify, name and generally describe -- each type of subsystem and component. It is then necessary, for each type, to identify, name and describe the particular parameters or descriptors used to describe formally a particular example. Then ranges for the parameters can be determined for each of the several levels that might be defined.

4.2.1 Processors

Categories of processors are:

- Central Processors
- I/O Processor
- Device Controller
- Multiplexor
- Selector Channel

- Microprocessor
- Arithmetic Logic Unit
- Communication Processor.

Typical parameters include:

- number of gates
- propagate time of gate
- number of instructions
- instruction execute time
- number of micro-programs
- number of registers
- register size.

4.2.2 Storage

Categories of storage are:

- Parallel-access storage. Fixed access/cycle time, random access, parallel word or block. Core, MOS.
- Cyclic storage. Drums, discs.
- Serial storage. Magnetic tape, paper tape, micorfirm tape.
- Discrete storage. Punch cards, magnetic cards, slides.

Typical parameters:

- Average access time; minimum, maximum
- Storage capacity
- Transfer time (read time, write time).

4.2.3 Displays

Categories of displays are:

- Serial (Character) Printers

- Line Printers
- Cathode Ray Tubes
- Plotters
- Card Punches
- Paper Tape Punches.

Typical parameters include:

- Display speed (rate in bits)
- Display symbols
  - number
  - type or complexity
- Display medium

4.2.4 Data Entry Devices

Categories are:

- Keyboards
- Switches and dials
- Light pen
- Joy stick
- Card transport and reader
- Tape transport and reader
- Disc transport and reader.

Parameters are:

- Transfer rates
- Symbol types.

4.2.5 Interface Standards.

The word "interface" is used both generically and specifically. However, authoritative definitions on specific meanings have not been made. Since "interfaces" are the focus of attention in the proposed work, we shall advance some tentative explicit definitions.

We see the word "interface" as being the qualifier in three key elements involved in data interchange:

- Interface data standards
- Interface dialogue procedures
- Interface conversion products.

Interface data standards include the definition of an individual data "packet", a description of the language to be used in a packet (in a broad sense), a definition of language and packet structures, the order in time and space of packet elements, the definition of symbols and codes, the designation of a transmission medium, and the specification of transmission facilities.

Interface dialogue procedures define the control mechanisms, actions, and message types and sequences needed to establish, confirm, maintain, and terminate interconnection and data flows (making use of the interface data standards above). These will include message addressing and also terminal "protocol," the ritual of "handshake," "ACK-NAK" (acknowledge/no-acknowledge), synchronization and other preliminary exchanges between terminals.

Interface conversion products comprise the means -- hardware, software, algorithms and tables -- needed to convert a message composed according to one data standard into a message(s) or record(s), identical in content, composed according to different standards or requirements.

Here we interpret "interface" to apply to data standards and dialogue procedures.

4.2.6 Processor Language

Here we speak specifically of the central processor's instruction repertoire -- the "machine language." It is a special case of interface, since it is the means by which human beings can "interface" with the machine. In the present context, the problem is to consider

establishing a standard minicomputer instruction repertoire, some subset of which would be used by any machine included in the standard series.

- Categories of instructions would include (for example):
  - Storage address
  - Branch
  - Loop control
  - Stack operations
  - Shift
  - Bit and Field
  - Immediate
  - Program control
  - Move
  - I/O
  - Interrupt
  - Register control.
- Parameters include:
  - Instruction world size
  - Instruction type
  - Instruction format.

4.3 DESCRIPTORS FOR STANDARD MINICOMPUTER SOFTWARE LANGUAGES

4.3.1 Assembly Languages

4.3.2 Procedure-Oriented Languages

4.3.3 Report Program Generators

4.3.4 Higher-Level and Special Purpose Languages

4.4 DESCRIPTORS FOR STANDARD MINICOMPUTER PROGRAM DEVELOPMENT SOFTWARE

4.4.1 Assemblers and Cross-Assemblers

4.4.2 Compilers and Cross-Compilers

4.4.3 Simulators, Emulators and Interpreters

4.4.4 Software Debugging Aids

4.5 DESCRIPTIONS FOR STANDARD MINICOMPUTER EXECUTIVE SOFTWARE

4.5.1 Interrupt Handlers

4.5.2 Error Monitors

4.5.3 Schedulers

4.5.4 Allocators

4.5.5 Input/Output Control Services

4.5.6 File Maintenance Systems

4.5.7 Data Managem ent Systems

4.5.8 Fault Recovery and Reconfiguration

4.5.9 Network Control Systems

4.5.10 System and Data Access Control

4.5.11 System Maintenance and Diagnostic Software

4.6 DESCRIPTORS FOR STANDARD MINICOMPUTER PERFORMANCE AND ACCEPTANCE TEST DESIGNS AND DATA SETS (BENCH MARKS)

4.6.1 Standard Test System Configurations (Hardware, Peripherals, Executive Software, Auxiliary Instruments).

4.6.2 Bench Mark Programs for Test Configurations

4.6.3 Standard Test Data - Input and Results

4.6.4 Auxiliary Test Instruments, Connections and Settings (to test hardware response and performance).

4.6.5 Test Documentation

4.7 DESCRIPTORS FOR SYSTEM RELIABILITY, MAINTAINABILITY AND REPAIRABILITY STANDARDS

SECTION 5. DESCRIPTION OF THE DEVELOPMENT PROJECT

In Section 4 we addressed very broadly and comprehensively the problems involved in achieving certain defined goals and objectives. It is not the intent of the development project described herein to solve in a single massive effort all of the problems and achieve all of the goals and objectives. A definitive discussion of the problem area is necessary to assure that whatever subset is selected for any single effort will be approached as a part of a larger and unified whole. It is the intent of this development project to address a particular and initial subset; that subset will be defined in this section. In fact, the project will be approached from two aspects:

- A short range aspect, with specific limited objective of providing guidelines representative of standards, specifications or requirements for several minicomputers and associated hardware suitable for data acquisition in stations of NASA networks. This is a large buy; reducing requirements down to a few different kinds of processors, stores, and peripherals will result in the achievement of some of the goals and objectives at an early date.

- A long range aspect using a much broader approach. The object in the initial effort will be to address the more important application areas (in the terms of dollar volume of minicomputers potentially to be purchased over a ten-year period) and formulate the structure of the "family tree" and the procedure to be used in establishing new "members of the family" as each need or stimulus arises.

As discussed elsewhere, the work of these two aspects is not independent; the interim results of one effort should be used in the other. For this reason the development project has been divided into two phases, the first of which concentrates (but not exclusively) on the short range aspects, and the second on the long range. The questions to be discussed in this section will, therefore, be divided into the same categories: short-range and long-range. Phase 1 work will be divided so that most of the resources are expended on short range, and few on long range. In phase 2, this ratio will be reversed. The phase 2 short range work will consist of developing specifications for additional family members, in response to specific large buys or important (in numbers) applications.

5.1 SHORT TERM STANDARDS AND SPECIFICATIONS

The essential difference between the substance of the short range and long range results is that long term results can and should aim at having a strong influence toward modular system construction using functionally small and specialized modules, whereas short term results must be content with definitions of modules compatible with present technology, architecture and production processes.

The standardization achieved so far within the industry has been achieved basically for the benefit of the supplier, to provide an existing customer with a new generation product compatible with the old generation software and physical data files (tapes, cards). Because smaller manufacturers made their equipment compatible with that built by large firms, additional standardization was achieved which benefitted the user. In the proposed development project, the aim is to achieve still more standardization for the benefit of users, particularly NASA

users. Since such standardization will tend to make users independent of their current suppliers, a program will likely meet with some opposition, especially from the larger suppliers. The short range program, therefore, must be executed with an awareness of this possibility and take present informal standards into account. User and NASA interests will best be served by stimulating and retaining as much industry cooperation as possible.

The short range program is, therefore, seen as providing guidelines of limited scope and depth. The goals and objectives of the development project will be similarly constrained, as described in the next paragraph.

5.1.1 Goals

Short range NASA goals are as follows:

- To achieve economy in purchasing minicomputers by developing specifications for several computers and peripherals to fit within the general, long range framework, for each "big buy" that NASA might make.
- To achieve interchangeability between the products of different suppliers built to the same specifications and standards.
- To achieve economy in programming the standard minicomputers by developing large computer development software to permit complete programming, translation, trial runs, debugging and checkout on one or more specified large NASA or commercial machines.

For the short range goals, low priority will be placed on defining and making standard measurements of performance, quality and reliability. This does not mean that low priority will be placed on achieving these properties in particular components -- simply upon standard means of measuring them. Similarly, low priority is placed on modularity and commonality in maintenance aspects. Both of these subjects will be covered in the long range work; in the short range, the aim will be to retain compatibility with the long range standards as they are developed.

5.1.2 Objectives

The short range objectives will, of course, be compatible with and related both to the foregoing goals and to the objectives stated in Section 4.1.3. They are as follows:

- To develop a specification for major subsystems required in the assembly of minicomputer systems for the data acquisition network. These subsystems may include the following:
  - Central Processor
  - I/O Processor
  - Main Storage
  - Tape Storage
  - Disc Storage
  - Printer
  - Keyboard
  - Cathode Ray Tube
- To develop preliminary specifications for central and I/O processors at not less than two levels such that processors built to the specifications of the higher level can be substituted for any of those at the lower level, and software developed for those at the lower level will run properly on those at the higher level.
- To develop requirements for computer program development software suitable for preparation of system and application programs for use in the data acquisition network.
- To develop requirements for minimum essential executive and utility software for the minicomputer systems for data acquisition applications.
- To develop requirements for test data and test software for all systems described above.

5.2 ACTIVITIES REQUIRED FOR DEVELOPMENT OF PRODUCT-FAMILY GUIDELINES AND STANDARDS

As suggested earlier, the proposed development work is not unlike a commercial product-line development program in its objectives. The principal difference is that in a product-line development, the

supplier is interested in interchangeability and compatibility in order to provide his customer with as complete a line as possible, and thus to capture the largest possible share of the customer's current and future business. In the proposed development project, the sponsor -- NASA -- is interested in interchangeability and compatibility between the product-families of different suppliers of the same kind of product, as well as between categories of products and successive generations.

There are several complementary activities of primary importance in a commercial product-line development program. These include, in approximate order of importance:

- Market Research
- Technical Research
- Intelligence - (Competitor Activity)
- Field Service.

There will need to be activities analogous to these in the development project. In the place of Market Research, for example, the activity required will be to examine current and future applications of small scale computers, within NASA, to develop from this information the types of products needed and to make estimates of the quantity of each kind that will be required.

Technical research covers several subordinate activities: circuit technology, systems technology, manufacturing technology,and programming technology. The role of these technical activities will be very much the same as in commercial product development. The activity analogous to intelligence on the competition will be intelligence on the present and planned product lines of industry, in both hardware and software. Corresponding to Field Service, NASA will have to consider providing maintenance of its user groups, including spares provisioning, maintenance personnel, and diagnostic, test and repair equipment.

These topics as they apply to the development project will be discussed in more detail in the paragraphs that follow. However, it must be emphasized, at this point, that the problem of developing a set of product-family guidelines for NASA users is a perfectly feasible thing to do. The activities required, mentioned above and explained below, to some degree, occupy every company that wants to get into the computer business and expand its share of the market. The result of a successful effort on NASA's part will be that several suppliers will voluntarily design and produce to be compatible with each set of product-family guidelines, instead of a single supplier.

5.2.1 Applications Survey

Although it is theoretically possible to develop the characteristics of a product-line independently of a market or applications survey, such a course is anti-survival in the commercial world. Now adherence to the product-family guidelines and standards to be developed will be voluntary to both suppliers and NASA users. Since such adherence on the part of both is essential to success, and since suppliers will use the standards only if their market research efforts show a definite market potential (in or out of NASA), it seems reasonable to assume that a comparable activity should be conducted as a part of the development effort.

It should be noted at the outset that we are speaking of minicomputer based systems when we speak of applications. Such systems may -- and probably will -- include large-scale systems comprising smaller minicomputer based systems in parallel or cascaded together and controlled by other minicomputers. Thus, applications requiring large-scale data processing or storage capability should not be arbitrarily excluded from consideration. Examples of such systems are shown in Section 6.4.

This activity -- the applications survey -- should be conducted even if only a relatively small "market" area is selected. The short-range goals, for example, will probably concentrate on data acquisition systems.

There are sufficiently numerous variations of such systems, of which satellite-telemetered data is only a single example, to suggest that an applications survey would be quite useful. The task would be to set down a list of such variations, develop representative system designs, draw up general characteristics for the subsystems, and make forecasts of the quantities required. These tasks require different techniques; they will be discussed in another section.

To summarize, an application survey is needed for both the short range and long range aspects of the project: a narrow survey for the short range, and a broad one for the long range. In addition, the survey should not be confined to small-scale applications of data processing and storage equipment. Large scale applications which permit the employment of multi-processor networks using small processors should be included, at least in the large-scale survey.

5.2.2 Technical Research

The subdivisions of this activity, mentioned earlier, include:

- circuit technology
- system technology
- manufacturing technology
- programming technology.

In commercial product line development work, it is not sufficient merely to design a new product for which a good potential market exists. It is also necessary that it be amenable to fabrication in a practicable and financially feasible way, and that the final product be reasonably reliable -- that is, operate properly for a specified number of hours with a high degree of probability.

The same criteria for technical soundness exist in a standards project. As in the previous section, it is necessary to consider the fact that the guidelines and standards cannot be enforced. If standard

products are to be realized, the standards must be practicable from various technical points of view.

The closer the product guidelines are to the product line a supplier might develop for himself, the more likely the "family" niche is to be filled in the numerous examples. Since this is really the object of the exercise, the technical objectives are to strike as close a compromise to nearly standard current practices as possible (e.g., 16-bit word sizes), to provide a clearly feasible evolutionary path, and for the long range, to work well within the expected states of the various arts.

The emphasis will clearly be on system engineering, since that technology is required to bridge the space between application and product and to predict with reasonable accuracy the trends in product design, computer system architecture (organization), executive and applications software, and other elements of perhaps a very large system (e.g., a continuous production process). System engineering is also required to consider the problems of interfacing diverse system elements, and to evaluate the many possibilities, and the matter of reliability, maintainability and repairability.

Programming technology is of approximately equal importance, since for the long run, executive software is an important subsystem that must be responsive to both the hardware and the nature of the application. Standards for program translation software -- assemblers, compilers, interpretors -- are of critical importance in providing software compatibility and low development costs for application software.

Awareness of developments and advanced work in circuit technology is important in the area of interface standards. The electrical characteristics of various kinds of devices will form the basis for some part of such standards.

In general, the level of technical competence required for developing standards is distinctly higher than that required for

commercial product development. The planning for the development project should take this fact into account.

5.2.3 Intelligence on Present and Planned Products

This activity should be much more easily conducted for the NASA effort than for a commercial project. It is reasonable to expect that suppliers will hope that NASA guidelines will tend to favor their own families or product lines. Such a possibility is more likely to exist when knowledge of at least the more important characteristics of current and planned product lines is available.

Although guidelines should be based upon the near-standards which generally evolve in a maturing industry, they should be sufficiently loose and flexible to permit promising new concepts to be brought to fruition. This may require obtaining some pretty sensitive intelligence on such concepts, and intelligence-gathering plans should take into account the sensitive aspects of proprietary information. Information gathering techniques, for example, might provide for soliciting opinions on candidate standards; opinions might include substitute or dual standards.

The utility of such information will be greatly enhanced if it can be organized in ways that will simplify the formulation of candidate guidelines and permit the evaluation and comparison of the candidates by a set of selection criteria or effectiveness measurements. Such organizations are related to the taxonomies, mentioned in earlier sections, which actually constitute the logical structures of the product-families and provide the basis of the definitions or descriptions of their members.

It is quite probable that the first organizations will make use of the survey tables used by trade publications to present summaries and comparisons of computers, and that the first product-families will resemble the product-lines of the larger manufacturers. These are, of course, sound starting frameworks, and provide us, at this writing,

with some idea of the appearance of both the structures of the guidelines, and the guidelines themselves. We have attached several such tables as Tabs to this Appendix.

5.2.4 Field Service

Field Service is that activity which has to do with the actual use of the guidelines and standards. It covers the selection, or application engineering, process, in which a particular prospective user starts with a requirement and ends up essentially with a system specification. It includes installation and test, maintenance, and design services for revisions or additions to the installation.

The purpose of the development project is to encourage NASA users to select equipment and system software from a range of products fabricated to NASA-promulgated guidelines and standards. It is, therefore, reasonable to assume that such guidelines and standards should cover not only the systems themselves but also the problems encountered by the user in specifying, testing and maintaining his system.

That these concerns for user problems were recognized in the early days of the computer industry is shown by the adoption of "bundling" (as it is now known) by the first suppliers (Remington Rand and IBM). These suppliers not only manufactured and sold the equipment, but they provided systems analysis services to assist in defining the problem, training classes at no cost to train programmers, complete proposals to spell out the system recommendation and the analysis to back it up, assemblers (and later compilers), installation and test, and maintenance services or contracts.

In the computer industry proper, "unbundling" has taken place, but only because industries of independent suppliers grew up to provide the users with the needed services -- which were just not available in the early days. A parallel situation exists today in the minicomputer field; unbundling in general, and the small size and youth of many of

the suppliers of minicomputers and associated peripherals have left a void in place of these user-related services for the small and (generally) specialized user. This is not to deny that these services are available, for sufficiently affluent users, from original equipment manufacturers, so called because of the complete systems they assemble from components and deliver tailored to the buyer's need, ready to turn on and operate. Minicomputer manufacturers also are increasingly aware of the void in customer services and moving to broaden them.

A guidelines program should therefore take into account the need for private industry to provide such services at a nominal cost. For NASA, this can be interpreted as being applicable either for establishing in-house service organizations, or the basis for inexpensive and relatively standard contracts for such services.

Examples are the development of guidelines for "Configurators" -- algorithms which assist in selecting hardware and software products from families for fairly standard application situations; standard parts for spares stocking; test points to make possible the use of standard diagnostics, test equipment and maintenance manuals; test software and data sets for both acceptance testing and maintenance and diagnostic purposes. Of course, the foregoing are examples of the kind of consideration that should be given to this activity in product-family planning; they are not offered as good answers to such considerations.

5.3 REPRESENTATIVE TASKS TO BE PERFORMED IN THE DEVELOPMENT PROJECT

The preceding section dealt with different kinds of activities or problem areas involved in developing a product-family. The parallel was drawn between such activities as they are normally conducted by a supplier of proprietary product line and as they would be accomplished in a NASA-sponsored development project. Since the end result -- descriptions of a product-line or family -- is the same in each case and has been achieved in the former, such a development project is clearly feasible.

In this section, we shall present some brief descriptions of several rather broad tasks which need to be performed. These tasks are:

1. Collect data
2. Organize and analyze data
3. Provide forecasts
4. Define typical systems for typical applications
5. Synthesize product-family guidelines and their interface standards
6. Develop application, test and reliability guidelines.

5.3.1 Collect Data

The data needed to do a proper job of setting guidelines covers a broad front. The data collection work is basic to the success of the development project; it may well be important enough to establish a separate library or data base.

The data to be collected should not be confined to mini-computers although attention should be focused there. Mini-computer design trends tend to follow large-scale computer practices, in both design and application.

A list of the kinds of data required should include:

- Manuals and descriptive literature on hardware, including processors, storage, input-output and communications terminal equipment
- Manuals and descriptive literature on languages, program development software, and executive or supervisory software
- Requirements, analyses, block diagrams, flow charts and descriptive literature on applications and applications software
- Documentation on interface standards, existing and proposed. These should include communications standards, electronic circuit standards, international code standards, and the interface standards set by large procuders for their proprietary products. Relevant military standards may also be appropriate, where they are not classified
- Maintenance manuals, diagnostic charts and techniques, repair practices, test setups and standards, test and diagnostic equipment
- Statistical data on production, installation and use of computer equipment. Computer installations by application type (concentrating on applications of interest to NASA users).

Data should include historical data and forecasts, as well as current information. Methods of collecting data should include conventional literature searches, but be well supplemented by more direct means of obtaining current information and opinions, such as questionnaires and interviews.

5.3.2 Organize and Analyze Data

This is perhaps the most critical and difficult task. Organization implies a structure; a structure ought to be functional; the function here is to store and retrieve information from which candidate guidelines and standards are drawn. Thus, the structure used for categorizing the information collected is closely related to the standards structure that the development project is to produce.

To review briefly: The object of the development project can be divided into three parts:

1. A structure which divides the entities dealt with -- hardware, interfaces, software -- into types, categories and family groupings.
2. Definitions for the parameters and characteristics which are used to describe the elements of the structure.
3. The numbers or measurements that are applied to the parameters, and that finally establish the actual guidelines or standards.

Execution of this task requires some explicit initial approximation of the first of these parts. Development of the second part can then be started by a preliminary analysis of the data. An example of the result of this procedure is given in Section 4.2; other examples of the first part only are given in Sections 4.3 to 4.7.

Refinements to organization and more detailed analysis of the data is followed by correlations of product characteristics (the numbers applicable to the parameters) with statistical data on production quantities, software usage and frequency of use. Trend analysis and forecasts of various kinds -- technological, production -- can then be made. The object of correlations and forecasts is to provide clues as to the limit that should be placed on both the scope and level of detail of the guidelines. Standards are neither needed nor desired for situations that fall in the tails of distribution curves, and overly detailed standards will tend to be restrictive and to impair healthy innovation and development. In an environment of voluntary compliance, the result of badly formulated guidelines is ineffectiveness.

5.3.3 Provide Forecasts

Guidelines and standards may be based upon and drawn from the present and past, but they should look to the future, both to be effective in a changing world, and to provide no obstacle to development and variation. Looks into the future are therefore important.

There are three basic types of forecasting methods that can be employed. One is qualitative in nature, in that judgment and opinion of qualified observers is involved, without regard particularly to the sources of their information or the logic of their reasoning. A second is the analysis of stationary time series, and involves the detection of problems and trends in changes of problems in historical or observed data. The third type involves the use of "causal models"; that is, the formulation, testing and use of relationships between sets of data believed to have common causes or effects.

The point, of course, is to be able both to pose questions about the long-range future -- what will future processes, technologies, products look like -- and to develop some information to provide credible answers relevant to the purposes of the development project.

This whole subject is rather thoroughly explored in a recent Harvard Business Review article*, "How to Choose the Right Forecasting Technique." The topic of product development in particular is discussed; the parallels to the proposed project can readily be drawn.

*HBR, July-August 1971, Vol. 49, No. 4, pp. 45-74.

5.3.4 Define Typical Systems for Typical Applications

Although guidelines and standards will more than likely find use outside of the NASA community, they are intended primarily to be used in that community. They will therefore be developed to cover certain application areas of particular interest to the NASA community.

It is desirable that the guidelines be no more comprehensive than they need to be; this applies to all types of components -- hardware, software, interfaces. The specific types and categories, and the magnitude of the families -- levels and quantity in each level -- should therefore cover only those to be used by NASA. The information needed to make these determinations -- types, categories, families -- will be drawn from functional system descriptions corresponding to the applications selected. Thus, it will be necessary first to decide which applications should be covered; second, develop typical system configurations for each application; and third, to develop some gross performance requirements for several levels of performance for each system.

This task involves collecting existing system designs, stripping them of non-essential information, and preparing block diagrams, tables and verbal descriptions in some standard format. The results will be used in the next task.

5.3.5 Synthesize Product-Family Guidelines and Their Interface Standards

The series of system configurations which were developed from typical NASA applications in the previous task form the basis of this task. These systems are resolved into their constituent hardware and software subsystems and components with their interfacing elements described, defined and measured. Corresponding elements are aligned, compared and placed in order. The results form the basis for the development of the various families and interfaces.

5.3.6 Develop Reliability, Test and Application Guidelines

This rather comprehensive task is included here to call attention to several rather important areas which can easily be overlooked in a standards project. Equipment which is built to be logically and physically interchangeable should also have comparable reliability and repairability characteristics. In other words, a user should expect a replacement for a component to have overall performance characteristics very close to those of the replaced item.

This task also considers the user directly, and develops the guidelines by which users of future computing systems can have their requirements converted into equipment selected according to the guidelines and standards.

Other aids for the NASA user are manuals which provide guidance on the installation and test of new equipment, on procedures for obtaining preventive maintenance as well as repair, what options are available for an expansion or change to his systems, how to go about arranging for various kinds of programming and design services, and perhaps a listing of organizations having equipment and applications related to his.

5.3.7 Summary

The tasks briefly described above are intended only as an indication of an approach that might be taken in a computer guidelines development project. A more definitive, explicit and comprehensive set of tasks can be expected in a formal proposal that a prospective contractor for the development project might prepare.

In the next section we shall present a simple example of what might result from the execution of such tasks.

SECTION 6. REPRESENTATIVE FAMILIES AND GUIDELINES

An attempt will be made in this section to give an example of how a standard NASA family of minicomputer subsystems might be structured. It is not necessary that we be either comprehensive or definitive, nor, in fact, that the different parts of the structure be consistent with each other. The purpose here is to present a starting point, and provide a vehicle or "straw man" for the generation and organization of ideas by those who will review and comment upon this appendix.

Only the following elements of a minicomputer system will be discussed in this section: the processor, storage, system software, languages, and meta-programs. If the job (of which this is an example) is done properly, input and output devices will have their own built-in interfaces for compatibility. We shall discuss system and minicomputer organization trends first, then specific applications, followed by a discussion of processing functions, families of processors, stores and system software, and finally, the subject of program preparation software.

6.1 GENERAL SYSTEM ORGANIZATION

On discussing families in other sections of this appendix, we have attempted to avoid reference to "minicomputer families", and to direct attention, instead, to families of processors, stores, input and output devices, and software. In doing so we are following a trend in computer architecture which can be observed in both large-scale computers and minicomputers. The trend is one in which the "computer" as a fully-integrated stand-alone system of hardware components is giving way to multi-processors, functional processors, micro-processors, firmware, and other modular components to permit built-to-order data handling systems.

We can identify three stages of evolution of minicomputer organization. These are, respectively:

- Conventional computer organization
- Unified bus organization
- Functional organization.

The levels of detail in each stage can be used as the basis for a taxonomic structure, and we propose to do so in this section.

6.1.1 Minicomputer Activities

The conventional architecture is illustrated in Figure 1 (Figure 2 of Reference 1). It shows the usual central processor connected to main storage and input-output devices through separate data buses, and the direct memory access (DMA) bus for high speed direct transfer. The associated hardware -- CPU, storage, I/O interfaces and control panel -- are packaged and marketed as a "minicomputer." In order to be able to handle a wide diversity of applications, minicomputers are almost always substantially over-designed for any one. This is good for general-purpose systems, which characterizes most large-scale systems. It is not so good for the dedicated application which characterizes the minicomputer-based system. Even so, they are not easy to adapt to specific applications, and there is little or no flexibility with respect to growth or new technology.

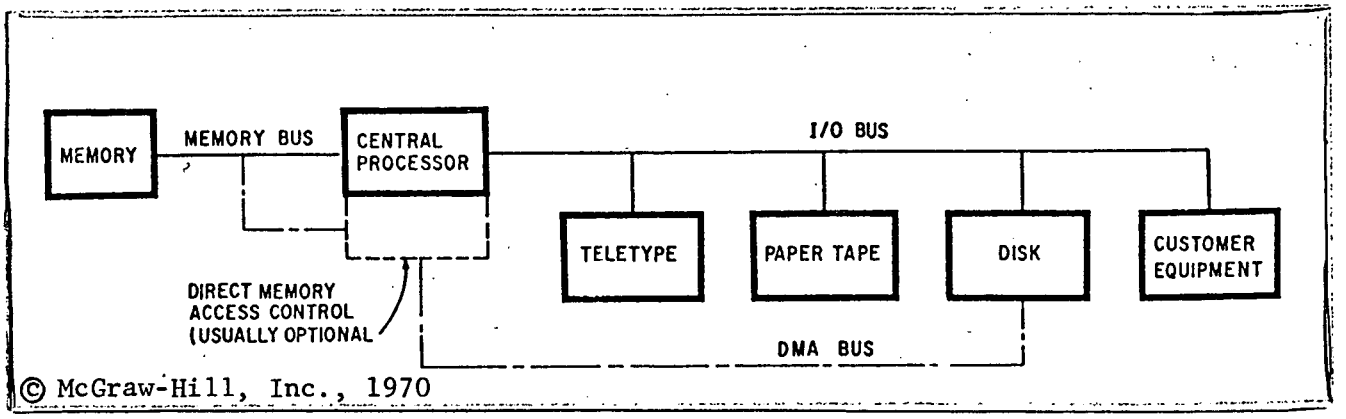

Figure 1. Conventional Architecture

A relatively new architecture -- the unified bus approach -- is illustrated in Figure 2 (Figure 1 of Reference 1). This type of organization removes the foregoing defects of the conventional approach. Each device has its own built-in asynchronous interface hardware, achieving flexibility. Furthermore, new technology equipment can replace obsolete modules, whether such modules are processors, stores, or peripheral devices.

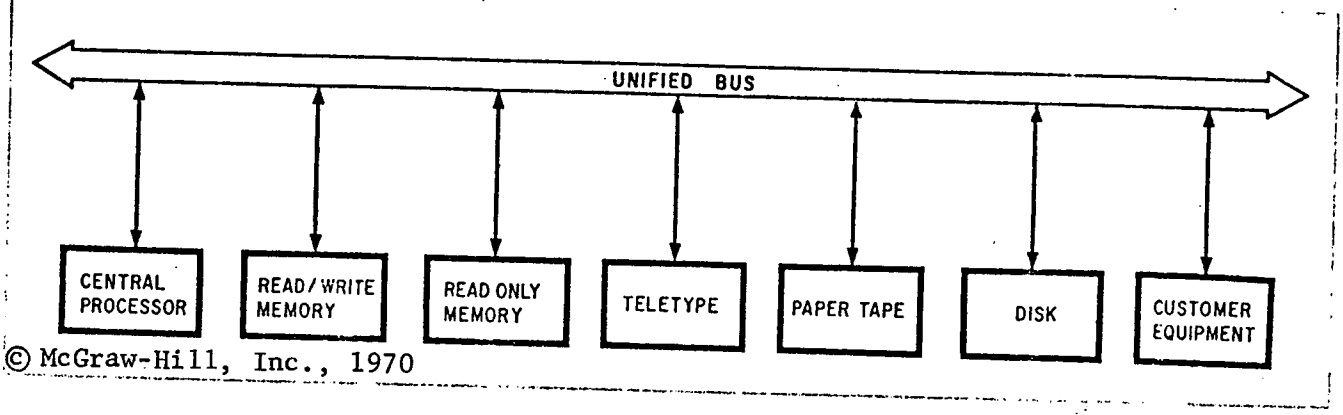

Figure 2. Unified Bus Architecture

To quote from p. 50, Reference 1 (applicable to the Digital Equipment Corporation's PDP-11):

> Communication between any two devices on the bus is in a master-slave relationship. During any bus operation, one device, the bus master, controls the bus when communicating with another device on the bus, called the slave. For example, the processor, as master, can fetch an instruction from the memory, which is always a slave, or the disk, as master, can transfer data to the memory, as slave. Master-slave relationships are dynamic: the processor, for example, may pass bus control to a disk, whereupon the disk may become master and communicate with a slave memory bank.
>
> When two or more devices try to obtain control of the bus at once, priority circuits decide between them. The priority level of the central processor is programmable, and masks anything of lower priority on the bus.

Other devices can have different priority levels, but their levels are fixed at the time of installation. A unit with a high priority level obviously always takes precedence over one with a low priority level, but of units with equal priority levels the one closer to the processor on the bus takes precedence over those farther away.

-----Electronics/December 1970

Another example of this architecture is that in the Lockheed Electronics Company's "SUE, the System-User-Engineered minicomputer." The organization for this minicomputer is shown in Figure 3.

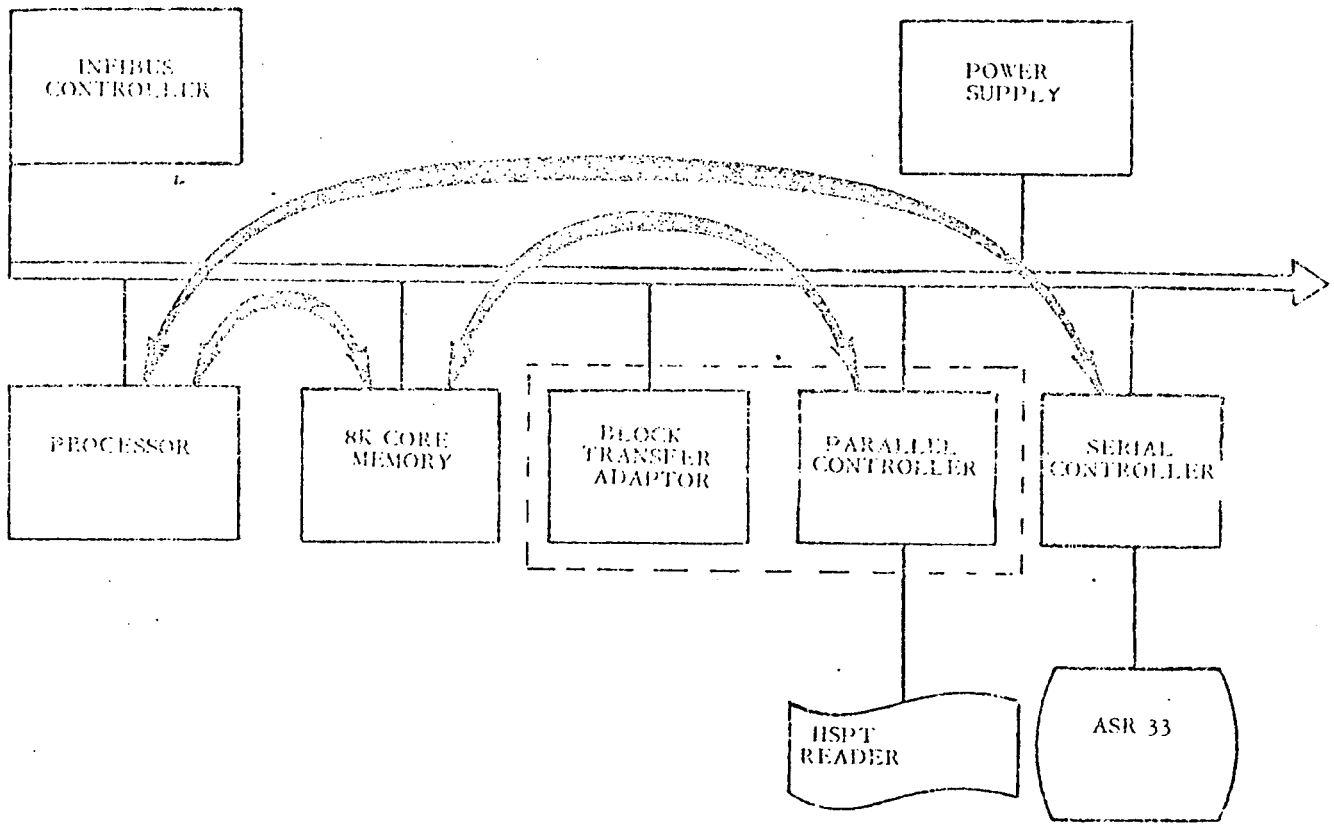

Figure 3. Unified Bus Architecture[2]

The characteristics of the unified bus type of architecture is summarized pretty well in the following paragraphs quoted from Reference 2:

> Each functional module -- processor, memory or device controller -- is on a pluggable multilayer circuit card. Because SUE modules are independent, asynchronous and have identical interfaces they can be put together in endless configurations starting from a minimal unit with 1K x 16 of memory on up to a 32K x 16 system loaded with peripherals.
>
> The user engineer can select the precise system modules required for any application; as requirements grow or change, modules can be added, deleted or changed without affecting the operation of other system components.
>
> Micromodularity ensures SUE against obsolescence. Large-scale integration is used throughout system modules wherever feasible. Advances in LSI technology can be incorporated as they occur in new functional modules and a SUE system can be updated by replacing the old modules with the new.

The unified bus concept was introduced commercially in about 1970 by Digital Equipment Corporation. A more recent commercial development is an extension of the same concept to lower-level and more basic functions. That is, the bus is connected to a much larger set of much smaller functional units.

The functional organization is illustrated in Figure 4 (from Reference 3). It is the organization of the GRI Computer Corporation's GRI-909 "Direct Function Processor". Although similar in structure to the unified bus, there is a distinctive difference in level of function connected to the data bus; this can readily be appreciated by comparing Figure 4 with Figure 5, which shows functional modules of the same general level. However, in Figure 4 they are connected to the bus, whereas in Figure 5 they are part of the central processor shown in Figure 2.

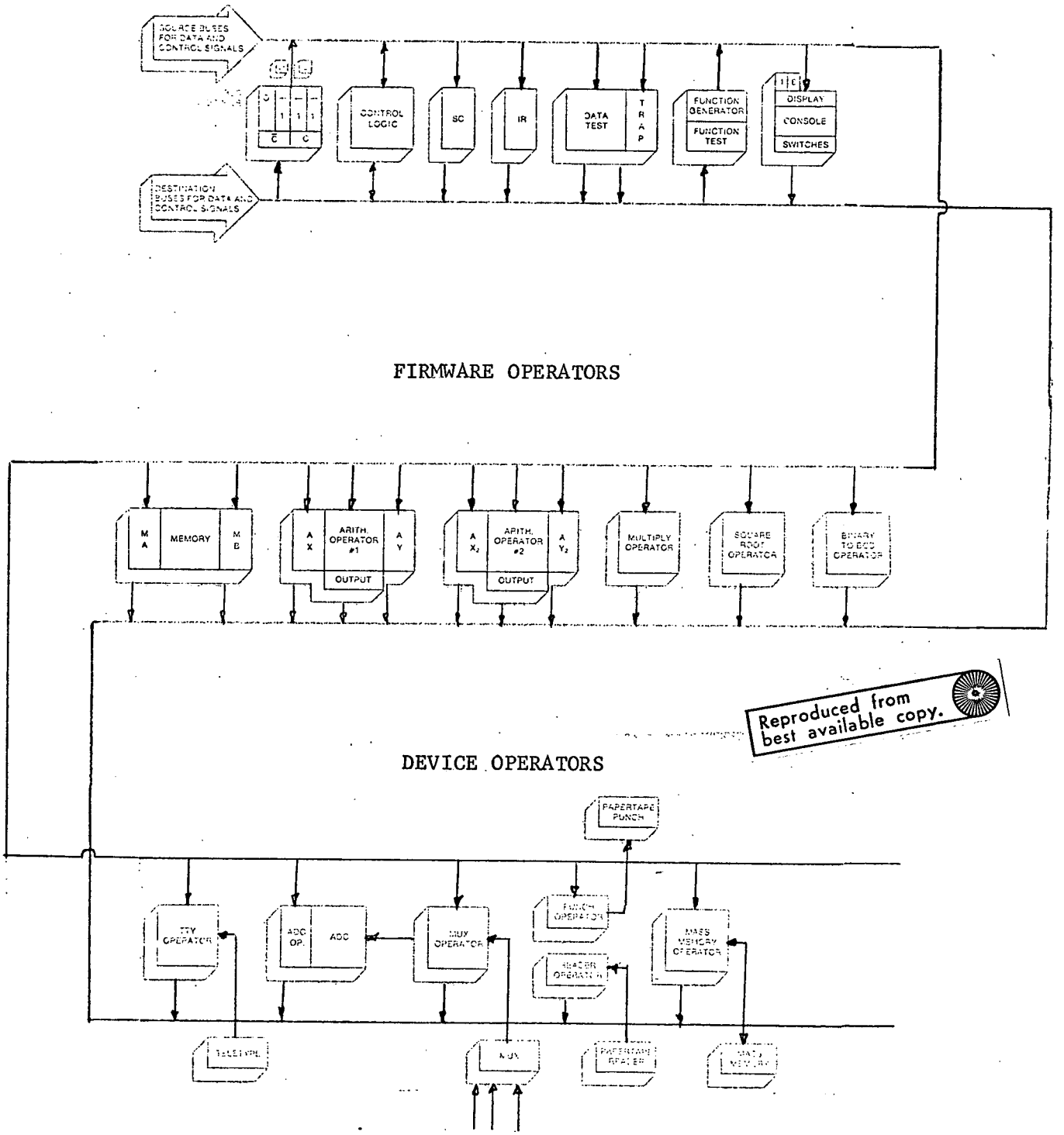

Figure 4. Function-Level Architecture

SC is a Sequence Counter; IR is an Instruction Register.

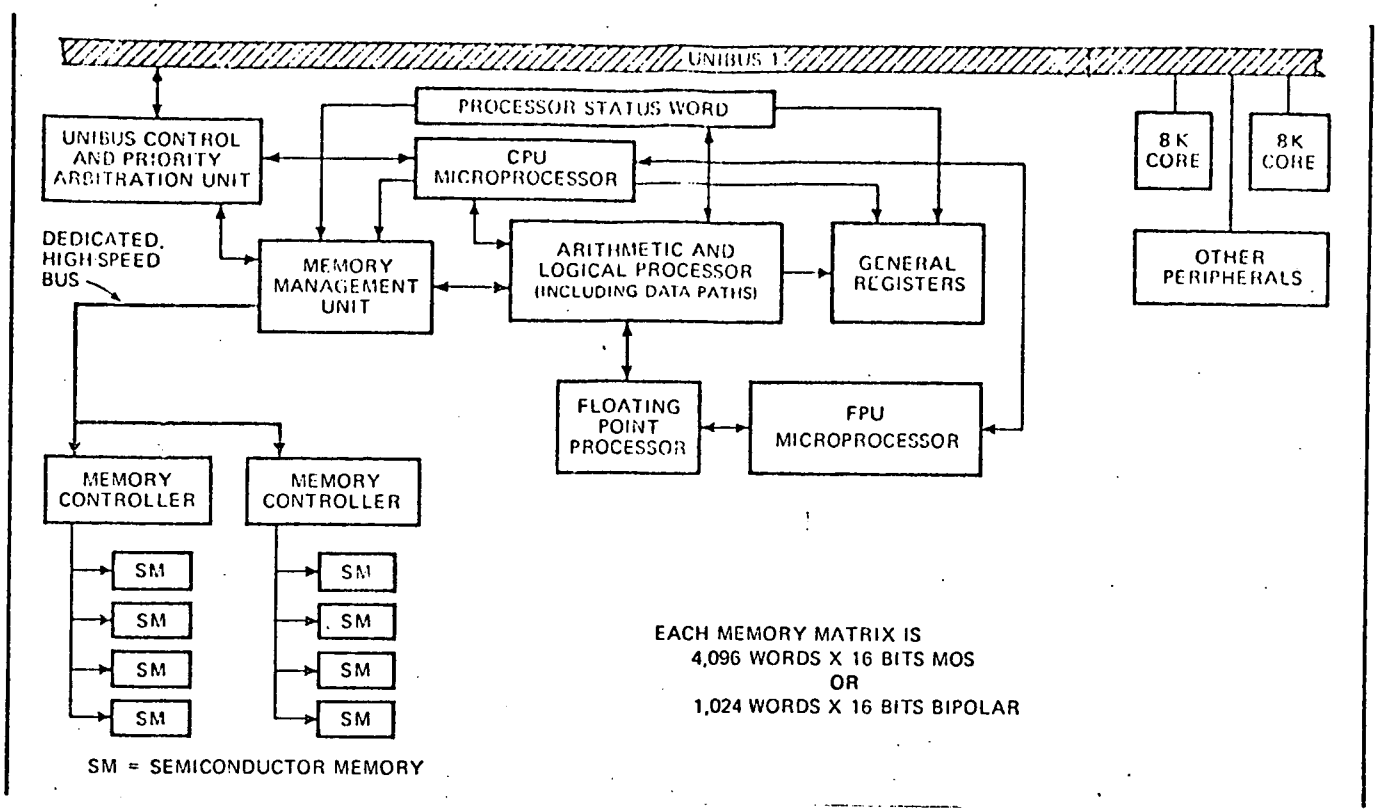

Figure 5. Unified Bus Architecture

*The lower-level functions are not directly connected to the bus.

The following paragraphs are quoted from Reference 3, and refer to the GRI-909 whose organization is illustrated in Figure 4.

> Bus Modifier - The bus modifier is designed to take information from an input device and move it to an output device, while performing simple operations on the data as it passes through.
>
> This programmable path provides a means of getting any data word, from any functional component of the system, appearing on the destination buses back to the source buses to be delivered to any other functional component in the system.

It can transfer a data word from one device to another, complemented or not complemented, and can modify the data in one of the following ways: (1) no modification, (2) incremented by one, (3) shifted left one bit, (4) shifted right one bit. A two's complement (negative) number can be obtained, on the fly, by combining the one's complement operation with the increment by one operation. Associated with the Bus Modifier is a Link Bit through which data can be shifted for testing one bit at a time, and an Overflow Bit for tests involving incrementing data.

The GRI-909 Computer is the first of a new class of general-purpose digital computers. Called a Direct Function Processor, it represents the next logical step in the use of computers as wholly-dedicated system controllers. The internal architecture and the machine language of the Direct Function Processor greatly enhance the system designer's ability to utilize computer control effectively and economically. Its programming language is oriented functionally, not arithmetically. The GRI-909 can be programmed in the kind of functional terms a systems engineer must use to define system operation. It keeps the responsibility of the system program in the hands of the designer by providing a means of relating systems functions directly to computer instructions.

The intrinsic modular design of the Direct Function Processor permits a variety of machine configurations ranging from highly economic, minimal processors for systems requiring simple data manipulations, to large configurations combining powerful computing instructions with a variety of peripheral devices. Hardwired firmware operators can be added in the form of plug-in modules to provide virtually thousands of computer instructions. These modules, or firmware operators, provide a flexibility and expandibility unequaled by conventional computer designs. Data transfer between system or computer devices is accomplished directly, in a single operation, without temporarily storing the data in special input/output registers or accumulators.

6.1.2 Minicomputer Networks

In the previous section we discussed trends in minicomputer architecture, and used the trend as the basis for structuring families of

computer components. The assumption in each example of architecture was that the sum total of any set of parts still comprised essentially a single minicomputer, and that the newer approaches permitted more flexibility, growth, and technological updating than the conventional, more monolithic minicomputer architecture. The possibility of attacking multiple processors or duplicate functions of any kind on the bus was not discussed.

In this section the natural extension of flexibility, growth and modularity into networks of systems will be briefly explored. The reason for doing so is to show that there is a practical upper limit to the level for the "family concept" if these new architectures are indeed trend-setters. We believe this to be the case, and that there will be no higher levels of computer families; that is, integrated computer architectures which incorporate the capabilities of all of the subordinate-level architecture. Instead, for the dedicated applications which we address within the NASA community, these larger systems -- mini or maxi -- will be built-to-order for the application.

These larger built-to-order systems will, in fact, be of two distinct (but increasingly overlapping) types: a system comprising a multiplicity of processors connected to a common interface (e.g., unified bus); and a system comprising a multiplicity of systems of the first kind. The fact that these two overlap can be more readily appreciated if we permit more than a single "unified" bus -- and this is already happening in such systems as the PDP-11/45.

An illustration of multi-bus architecture, where the busses are categorized by transmission needs rather than by device function, is shown in Figure 6 (Figure 1(c) of Reference 4). A simple 2-system network of two such multiple bus minicomputer systems is shown in Figure 7 (Figure 1(d) of Reference 4). The 2-system network of Figure 7 has the property that the two systems can share the data in the semi-conductor memories with

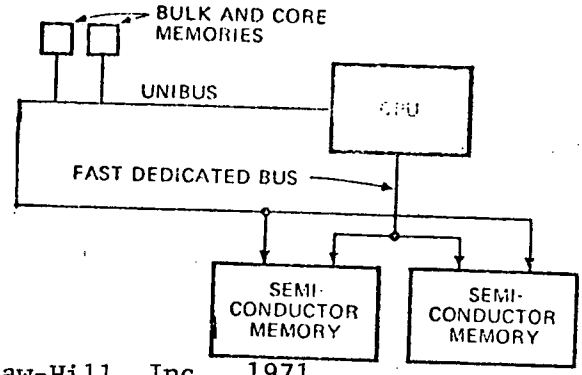

Figure 6. Multiple unified bus architecture showing both busses connected to two-part semiconductor very high-speed storage.

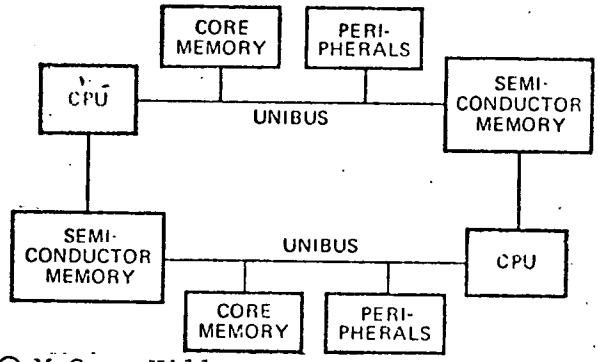

Figure 7. Multiple System Network

virtually no conflict or data-time penalty. Furthermore, the high-speed semi-conductor memories provide a data-passing and parameter-passing buffer between the CPU's so that the larger size core memories can also be shared.

This concept is easily extended to many systems. An example of a 3-system network is shown in Figure 8 (Figure 5 of Reference 5). In this example, System C has two processors, one of which controls all of the peripheral equipment and handles the communications terminals.

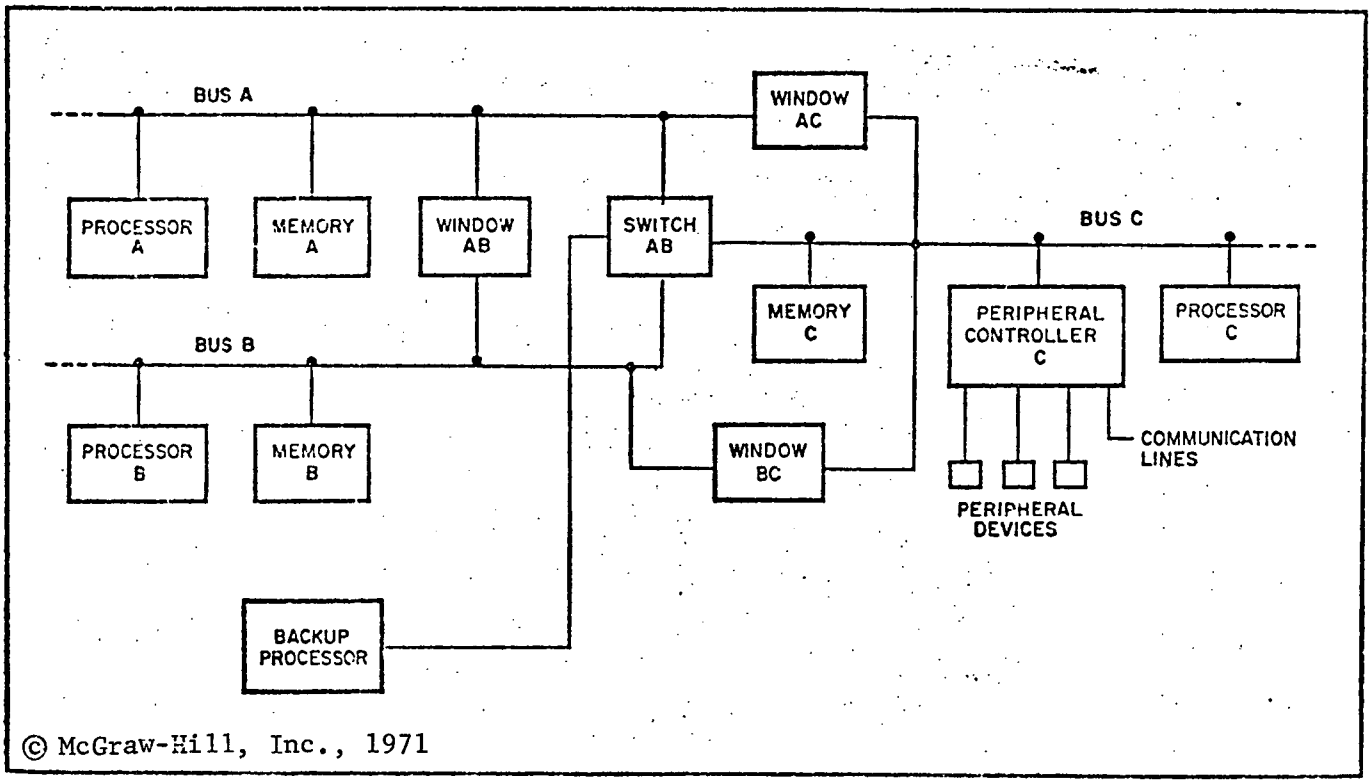

Figure 8. An Example of a 3-System Network

A variation of this approach is shown in Figure 9 (on pg. 1-1 of Reference 6). This is an illustration of the Hughes H-4400 Multi-processor system. While these multiprocessors cannot be classed as minicomputers, we are addressing multi-processor and multi-system architecture here. The concept is different from that of Figures 7 and 8 in that there is a single

switch instead of many, and a multi-part memory (up to 16) providing for many (8 x 16 = 128) processor-memory paths. This architecture exemplifies the trend to specialized processor modules, which will be reflected in the "family" to be described later.

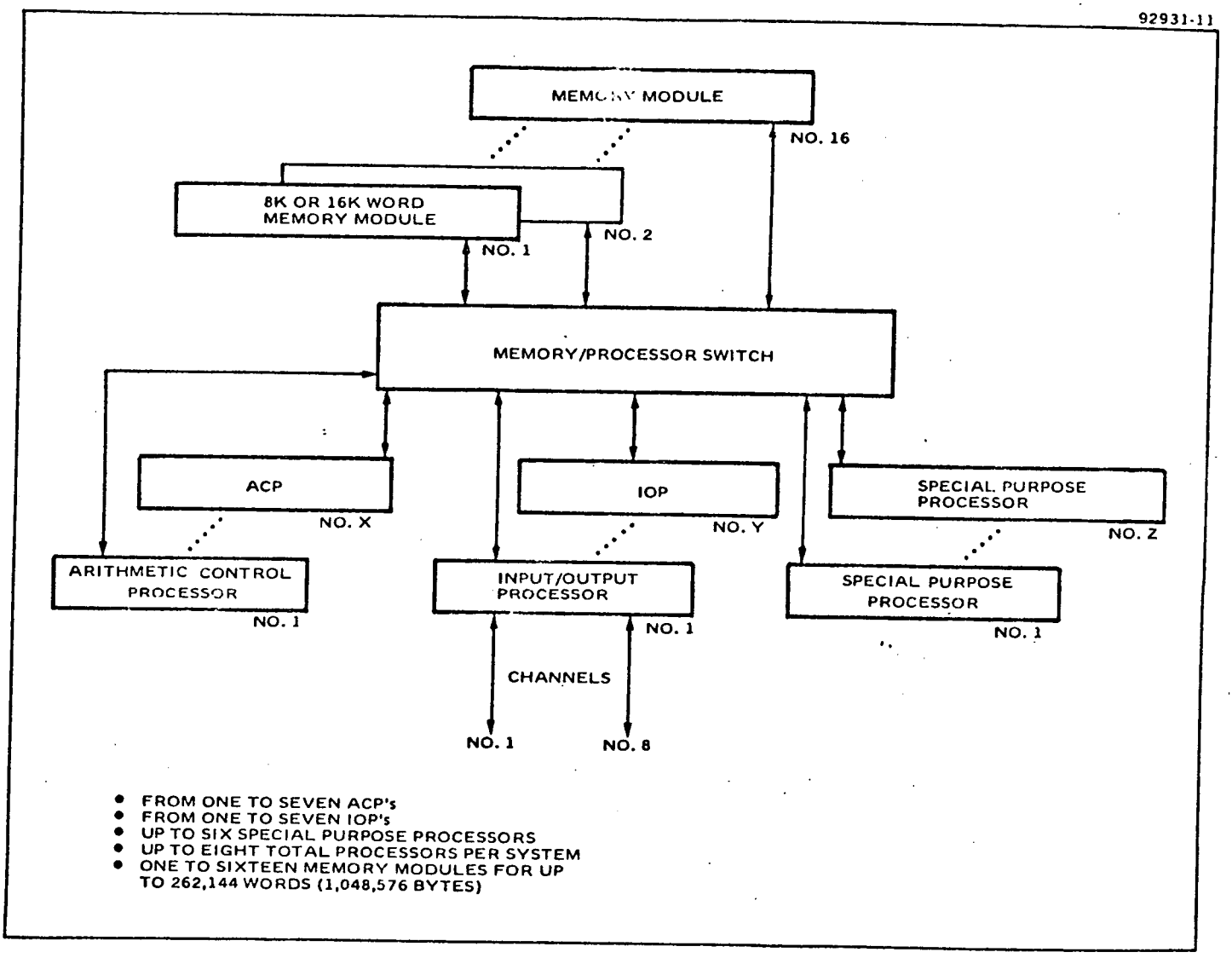

Figure 9

The types of modules include Arithmetic Control Processors (ACP), Input/Output Processors (IOP), memory modules, and a Memory Processor Switch (MPS). Modular growth or custom system tailoring is achieved by the addition of the desired module types in any combination from the unit computer (one memory, one ACP, one IOP and one MPS) to the maximum configuration of 8 processors and 16 memory modules. Subsystem modularity provides further tailoring of each element.

The Arithmetic and Control Processors are the actual computing elements. If a system is composed of multiple Arithmetic and Control Processors, each processor may operate with equal system stature, assigning itself program tasks as required. The processors may also operate as dedicated processors with one being the executive with a master/slave relationship to all other processors.

Input/Output Processors provide the communication link between the computer and external devices. Once initiated by an Arithmetic and Control Processor, the Input/Output Processor will execute its stored program and operate independently until its program has been completed. The Input/Output Processor may contain from 1 to 8 channels in any combination of four channel interface types.

Special processors necessary to satisfy a specific system's requirements may also be connected in the same way as an ACP or IOP. These processors have direct access to memory and can communicate to the rest of the system in the normal fashion.

The Memory Processor Switch resolves memory conflicts on a fixed priority basis. It also supervises the executive assignment of the Arithmetic and Control Processors and directs system interrupts to the appropriate processor. The system status and real-time clocks also reside within the switch module and are accessible under program control.

A more complex and general example of a multiple system network which incorporates minicomputers and larger scale systems is shown in Figure 10 (Figure 7 of Reference 5). It is a network envisioned for use of students at Carnegie Mellon University.

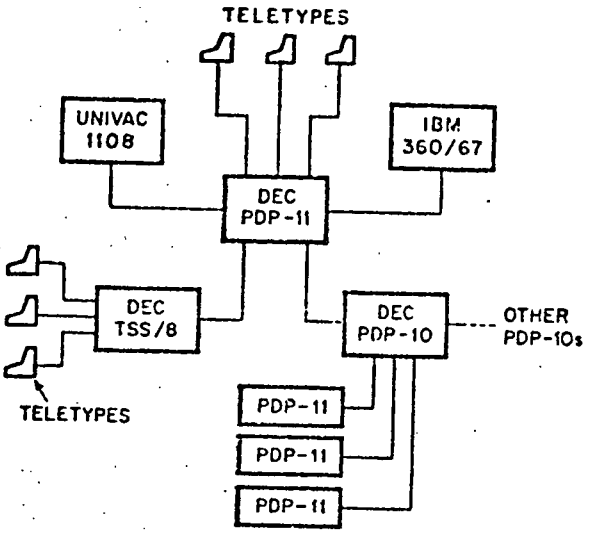

Figure 10. An example of a complex network involving large and small computer systems.

Of course, many network configurations are possible, as shown in Figure 11 (Figure 10 of Reference 7). There is no particular need for a central station or dominant computer system; in fact, network control may "flock" -- migrate between nodes. As larger and more complex networks of exclusively minicomputer systems or components are achieved, the distinction between the total capabilities of such a system and those of a large-scale computer system tends to vanish. For this reason, families of minicomputer processors and stores must be considered possibly to be applicable to applications which traditionally have required large computers. These, however, are systems designed for specific applications; we shall address particular examples of systems and networks in the next section.

Network geometries include (A) star, (B) multidrop, (C) loop, and (D) multiconnected. Geographical locations are the same in each case and the central station is indicated by the letter "C."

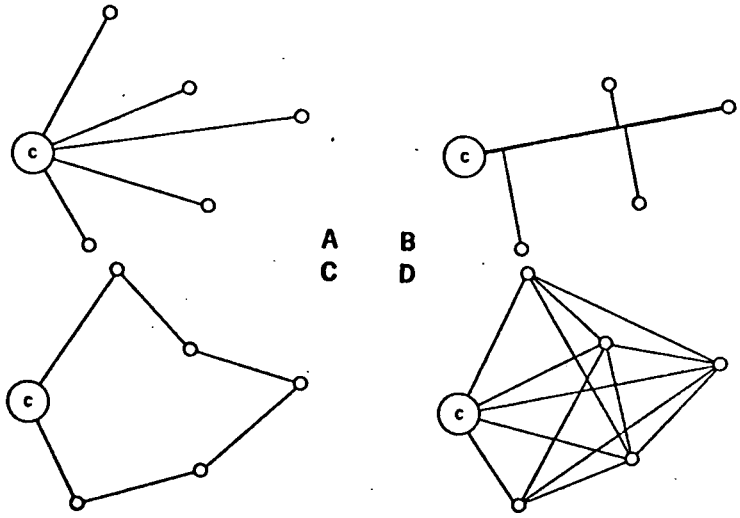

Figure 11

6.2 SPECIFIC APPLICATIONS FOR SINGLE AND MULTIPLE MINICOMPUTER SYSTEMS

The range of applications in which minicomputers can be used is very large and increasing rapidly. That range, in fact, is beginning to include many of the applications in which only large-scale computing equipment can be used. In these areas, groups or networks of minicomputers or components are increasingly being applied.

In this section we shall examine several areas of application of importance to NASA, and which, in general, account for an extremely large fraction of small systems in use today. The particular application areas are Data Acquisition, Equipment Controllers, and Communications. Each of these applications imposes upon the computing equipment certain requirements that are more or less characteristic of it, and which therefore "drive" the design of members of a family. In the development project a study of which this section is representative, will be conducted to determine which application areas should be included in the development of a family structure and guidelines.

These application areas are quite thoroughly discussed in the literature (see bibliography). Our purpose here is to demonstrate the procedure of extracting major design requirements to permit rational selection of "family" members.

6.2.1 Data Acquisition

A typical data acquisition system is shown in Figure 12 (Figure 1 of Reference 8). In this example, data is drawn automatically from sensors. A specific example is shown in Figure 13 (Figure 1 of Reference 9), showing an electrolytic tinning line in a steel mill.

The basic function performed by a data acquisition system, as the name and the diagrams imply, is to collect and record information from one

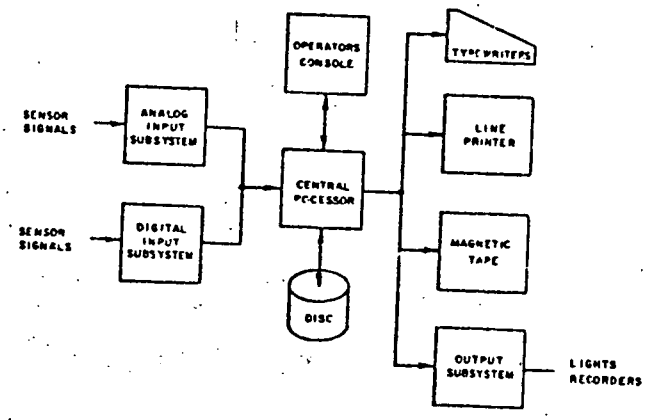

Figure 12. Data Acquisition System Components

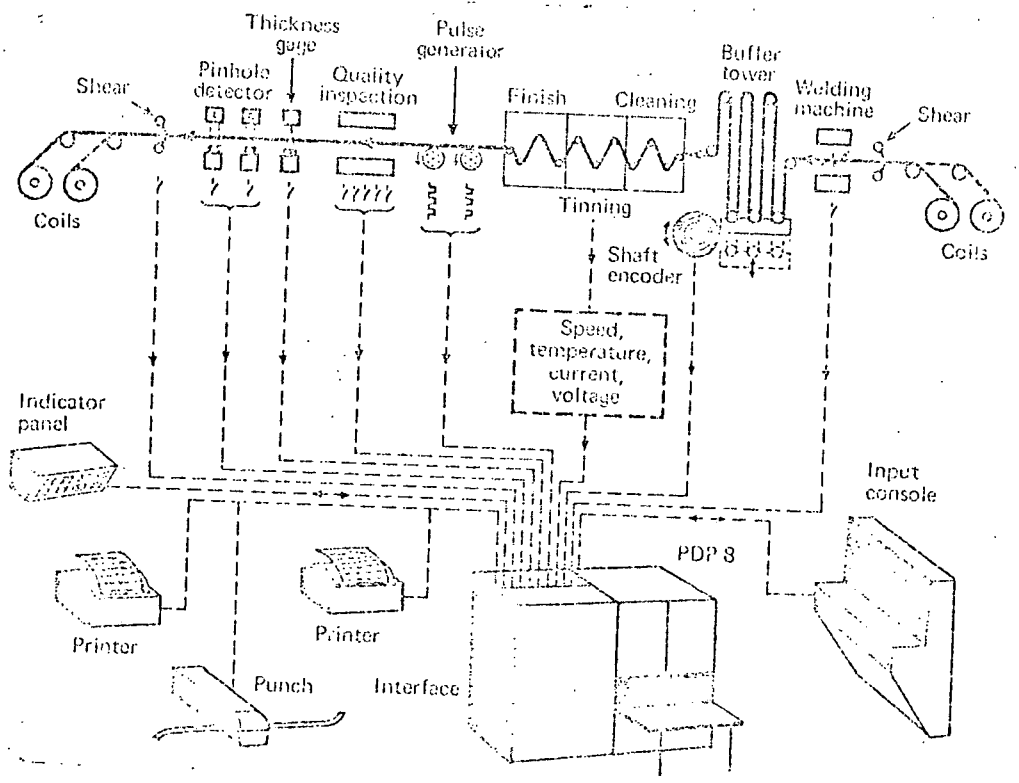

Figure 13.* An industrial data acquisition system for an electrolytic tinning line in a steel mill

* Reproduced by permission of Control Engineering

or more sources and record (log) the information in machine-sensible form for subsequent processing, analysis, cataloging and storage. The sources may be remote, as for example an observation satellite, a meteorological station, a branch office, a factory machine station; or very close to the computer. There may be many sources and relatively few kinds of data, as in a meteorological network, or a few sources and many kinds of data, as in a space experiment. In all cases, however, the job is to collect and permanently record data of a transient nature. Processes which provide monitoring and control information, performed with a view to influencing the process measured in real time is not included. A given system may, of course, incorporate such functions in addition to that of data acquisition. In this discussion, however, we speak of data acquisition as a one directional, continuous flow of periodically sampled information from one or more sources to one or more recording or storage media. Intermediate data processing may include such operations as serial-to-parallel conversion, demultiplexing, formatting, error detection or correction, data reduction and smoothing, and blocking. Analysis functions used for monitoring or control purposes will be included under Equipment Control Applications.

Thus defined, data acquisition applications impose requirements such as the following on a system. The parameters used are considered to be representative rather than definitive. Measures used will be in comparative rather than absolute terms, relative to the state of the art in small-scale processors and storage units:

- Interrupt Capability: low
- Number of Registers: high
- Instructions: high power in shift, bit, field, move, register control
- Arithmetic: simple
- Microprogram Capability: important for special conversion, edit and move routines
- Main Storage: moderate size, high speed, memory overlap and multi-part desirable
- System Software: simple.

The emphasis on all characteristics is on bandwidth -- a high throughput rate is required of all components.

6.2.2 Equipment Controllers

This relatively large area includes process control, which was perhaps the first important application market for minicomputers. It also includes peripheral controllers, which is perhaps the most recent important application. A third area of rapidly increasing importance is direct control of machine tools and factory equipment, using programmable controllers. It also includes monitoring and "alarm checking" applications, in which measured parameters are continuously or periodically compared with standards or limits.

In this application, one-way throughput capacity is de-emphasized in favor of computation and generation of real-time information for feedback or control purposes, either directly and automatically or indirectly through mechanical alarms and displays. In general, two-way flow of information between the computer system and an external system is required, typically status information flowing to the processor, and control signals flowing to the external system (including indirectly via a human operator). Reduction and recording of information (logging) is secondary (in this discussion); such functions are generally included in a given equipment control application, but are covered under Data Acquisition, Section 6.2.1. This distinction in definitions is consistent in this exposition, since the two application areas (if included in NASA applications) will generate distinct "family" members. In a particular application involving both data acquisition and equipment control, appropriate selections can be made from the "family".

Examples of three major equipment control applications are shown in Figures 14, 15 and 16. Figure 14 (Figure 2 of Reference 9) shows

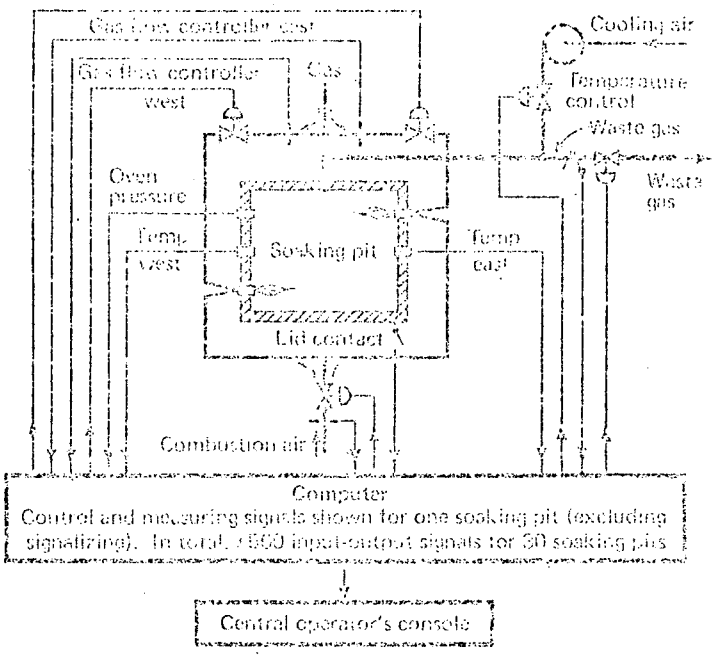

Figure 14.* System for Control of 30 Soaking Pits in a Steel Mill

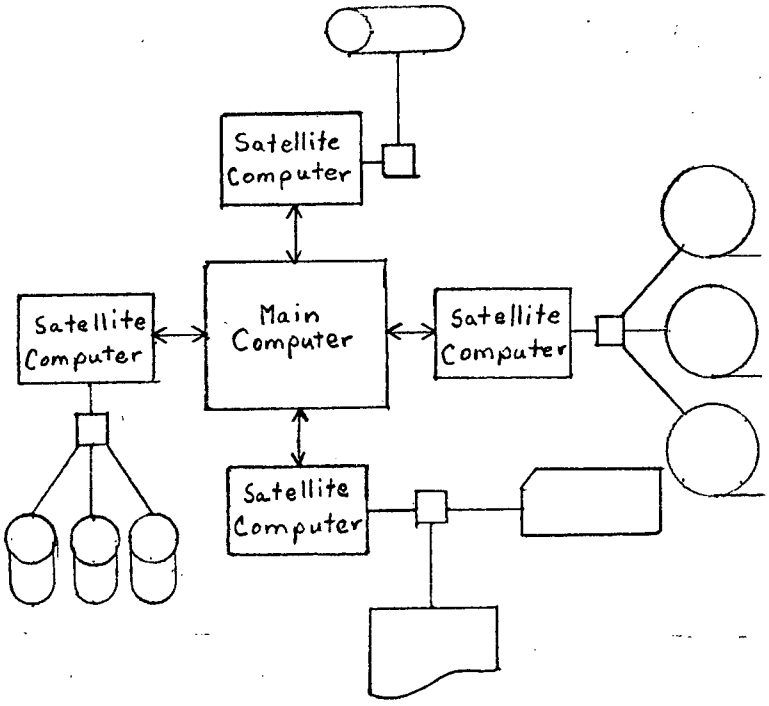

Figure 15.** Minicomputers Used in Place of Peripheral Controllers[14]

* Reproduced by permission of Control Engineering

a system for controlling soaking pits (to reheat steel billets for rolling operations). The two-way flow is clearly shown. Figure 15 (Figure 1.31 of Reference 14) shows minicomputer systems being used in place of hardwired controllers for control of peripheral devices in a large-scale computer system. Figure 16 (Figure 1 of Reference 10) shows the functional block diagram of Allen Bradley's PDQII applied generally to control of factory equipment, as well as a remote computer that can control, in turn, the PDQII program.

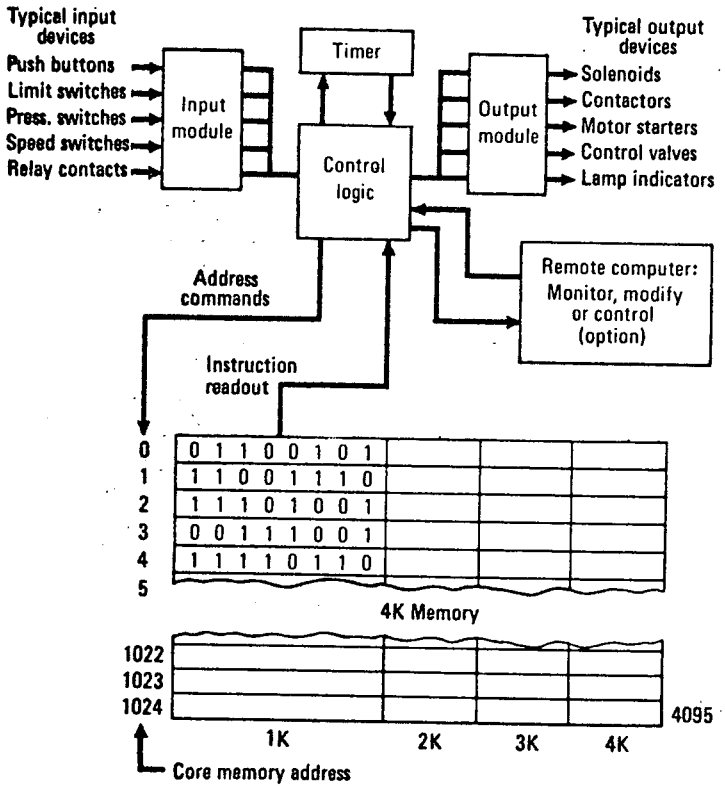

Figure 16.* Factory Machine Control, Showing Two Levels of Control

* Reproduced by permission of Control Engineering

Equipment control applications impose the following requirements on a computer system:

- Interrupt Capability: Moderate
- Number of Registers: Moderate
- Instructions: Moderate power in branch, loop control, stack operations, logical instructions
- Arithmetic: Moderate
- Microprogramming Capability: Important for high-speed, fragment control loops; multiply, divide, floating point or special arithmetic
- Main Storage: Moderate size and speed.

6.2.3 Communications

From the point of view of the overall applications, Data Acquisition and Communications are quite different. From the point of view of the computer, they are quite similar. The difference between the two is epitomized by the words Acquisition and Communications: the one implies capture and storage, the other implies reception and transmission. From a functional point of view, we may find that a close scrutiny will reveal only small differences, and that the impact of Communications applications on the processor family will be the same as that of Data Acquisition. Nevertheless, we shall look here at the three basic types of applications of minicomputers to communications. These are:

- Line concentrator
- Front-end processor
- Message switch.

As in Data Acquisition, the main flow of information is into and out of the computer; there is, unlike Data Acquisition, little substantive data processing performed on the incoming data. However, there may be, as in Equipment Control, some flow of information back to the source. This return information may be to establish "line protocol" -- the exchange of information between terminals needed to establish and confirm a connection -- or it may be to indicate successful receipt by an acknowledgement signal ("ACK"), or unsuccessful receipt by a non-acknowledgement signal ("NAK").

As in the other two application areas, a Data Acquisition system may also act as a communications processor if the sources are remote and transmitted over long data-links. Similarly, an Equipment Controller may include communications functions if the equipment controlled include remote batch terminal equipment. With a properly designed "family" of processors, stores and system software, selections can be made for any given combination of applications.

Of the three types of applications mentioned above, the Line Concentrator is perhaps the simplest. The processor terminates several relatively low speed transmission lines, consolidates the information and sends it along a lesser number of relatively higher speed transmission lines. Little processing of the transmitted data is performed. An example is shown in Figure 17 (Figure entitled "Concentrating Lines of Reference 11), in which the characters of many low-speed, asynchronous lines are combined and arranged in serial form in a high-speed, synchronous line. The start-stop bits of the asynchronous data are stripped off in the process, thus achieving a real increase in transmission efficiency. Code conversions to incoming data may also be made to provide uniformity. A complementary application is that of de-concentrator, in which the process is reversed.

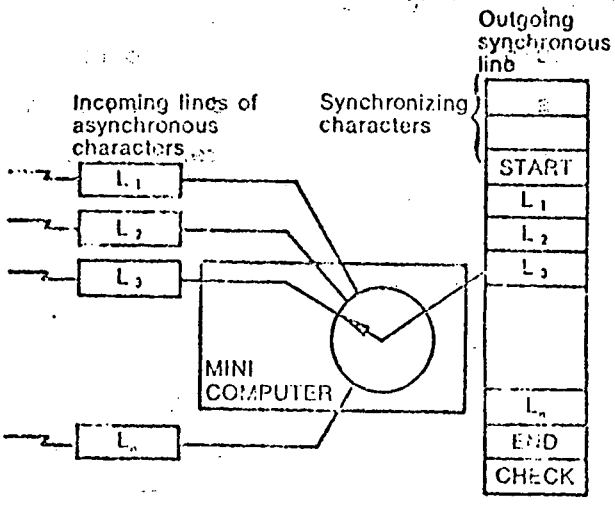

Figure 17. Line Concentrator

Two kinds of Front-End Processors are illustrated in Figure 18 (Figure entitled "Eliminating Computer Interruptions in Reference 11. The functions of the communications controllers (which may be programmable controllers, or minicomputers) in this application include that of terminating a multiplicity of data transmission lines and concentrating them. In this application, the Front-End processor is almost identical functionally to the Line Concentrator.

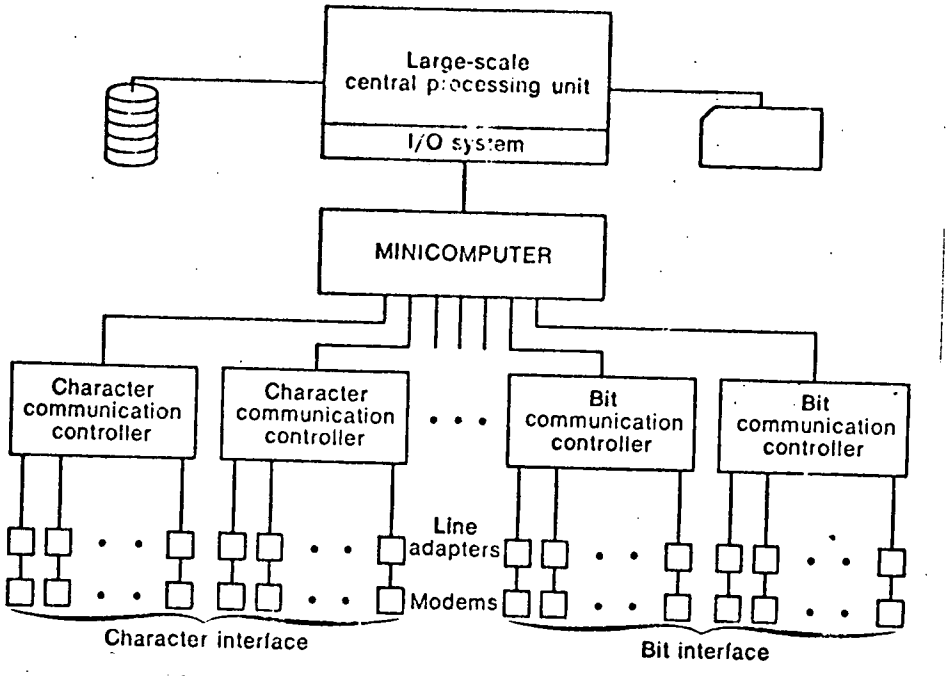

Figure 18. Front-End Processor

A second version of a Front-End Processor is that of the minicomputer shown in Figure 18. It takes the output of the controllers and feeds them into a large processor; it serves to relieve that processor of the many interrupts and routine message receiving (and transmission) operations, thus substantially relieving its processing overhead. Typical of such functions are error detection and correction, requests for transmission, code conversion, message assembling, decrypting. Some of these functions may be shared with the communications controllers.

A more powerful version of a Front-End Processor is one in which the controller functions are accomplished by a single processor interfacing directly with the transmission line terminals. The front-end processor in such an application will assume all terminal, line and message control functions.

The third communications application is that of the message switch. This application is illustrated in Figure 19 (Figure 1 of Reference 12), by the "IMP", the ARPA networks Interface Message Processor. The processor in this application includes all of the functions covered in the previous example, and performs the additional major function of forwarding messages intended for a host computer different from its own. The IMP configuration is shown in Figure 20 (Figure 7 of Reference 12). The IMP handles messages between host computers, prefixing a header to each message containing the destination and the sender's (user's) identification. The message passes from IMP to IMP until it reaches its destination.

Although the IMP is classed as a minicomputer (See Reference 8, page 20), message (store-and-forward) switches have been developed using very large computers. The question here arises as to what a very large store-and-forward processor can do that cannot be accomplished by several smaller processors at the same site. If the advantages are small, they might well be overcome by the redundancy and fail-soft aspects of the multiple small processor design.

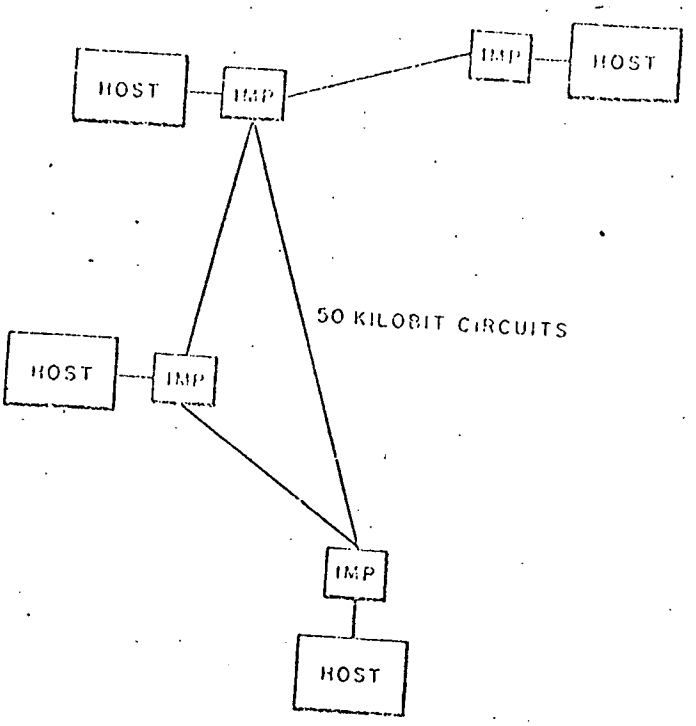

Figure 19. ARPA's Interface Message Processor as a Message Switch for Host Computers

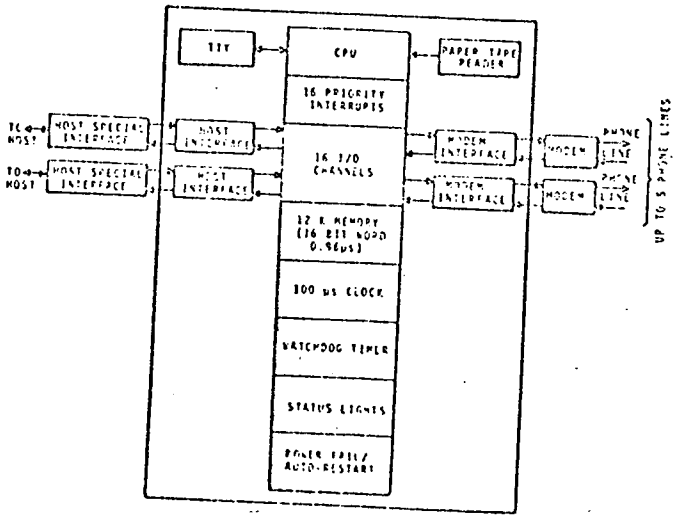

Figure 20. IMP Configuration

The three types of applications discussed above appear to be hierarchically (vertically) differentiated rather than categorically (horizontally) differentiated. We, therefore, have a rather large range in the values of the parameters descriptive of any capability. However, with respect to Data Acquisition and Process Control, we can make the following summary:

- Interrupt Capability: High
- Number of Registers: High
- Instructions: High power in shift, bit, field, move, register control
- Arithmetic: Simple
- Microprogram Capability: Important for protocol, code conversion, and move routines.
- Main Storage: Moderate size, high speed.

6.3 FUNCTIONS AND FAMILIES

In the development project which we are attempting to describe in this appendix, a more thorough investigation of applications in the NASA community will be made in the data gathering and analysis tasks. Functions and design features required of data processing systems can then be identified and used to define families, standards, or guidelines for selection or specification of equipment by NASA users. As an example of this process, we shall in this section derive a representative set of functions, and from them, define a family of processors. A similar procedure can be applied to developing families of (or guidelines for) storage systems and of systems software.

We shall assume that many NASA applications can be described by one or a combination of two or more of the applications of the preceding section. We shall resolve these applications into their component functions at several levels of complexity. These functions, properly organized, will form the skeleton of our representative family.

6.3.1 The General Structure - An Overview

The intent of the structure we present here is to serve as an example of what might be produced in a fairly large-scale, intensive standards development program, and not as a first iteration of what will be produced in such an effort. At the same time, we must admit to the fact that the structure presented contains some "editorial" content -- some opinion. That opinion is expressed in a number of ways; viz.:

- The structure is functional. It is not based on technology, embodiment, or size and speed
- The structure is based on the use of standard modules and a unified bus; that is, a bus to which subsystems may be logically connected, and along which travel signals of standard size and format.

We present a structure, incorporating the foregoing technical opinion, as a consequence of following the steps presented in Section 5.

The inclusion of technical opinion in this example will serve to prevent it from being either sterile or puerile -- both typical properties of examples that are not well thought out. However, it must be emphasized that the example is not provided as a vehicle for the expression of our opinion; the main purpose of the opinion is to provide reasonable substance to the example. A lesser purpose is to provide a reasonable take-off point, or "straw-man", for a full-fledged development effort.

The structure is presented in Figure 21. The diagram shows three functional levels: the applied systems level, the modular subsystem level, and the functional component level. The top level, applied systems, includes general purpose minicomputers which are made up of internal processors and storage subsystems only. It also includes special purpose controllers which would cover most current peripheral controllers, and programmable controllers, which include read/write memory. The structure thus provides the possibility of including one or more standard minicomputers (of a range of sizes) and programmable controller configurations, as well as micro-programmed special-purpose equipment with read-only memories.

It is, however, at the middle level, that of the modular subsystems, that we envision most standard items of hardware and software to be established. These modular building blocks, designed to interface on one or more unified busses, together with standard system software packages, will be used to build up systems of the desired nature, complexity, throughput and response time. Although ranges of sizes and speeds are not precluded, we envision large throughputs and short response times in applied systems being achieved through parallelism rather than through inclusion of very high-speed or large capacity family members.

The basic elements of this middle level include the internal processors, storage subsystems, interface processors, and the user's hardware and software. We are concerned here primarily with the first three of these.

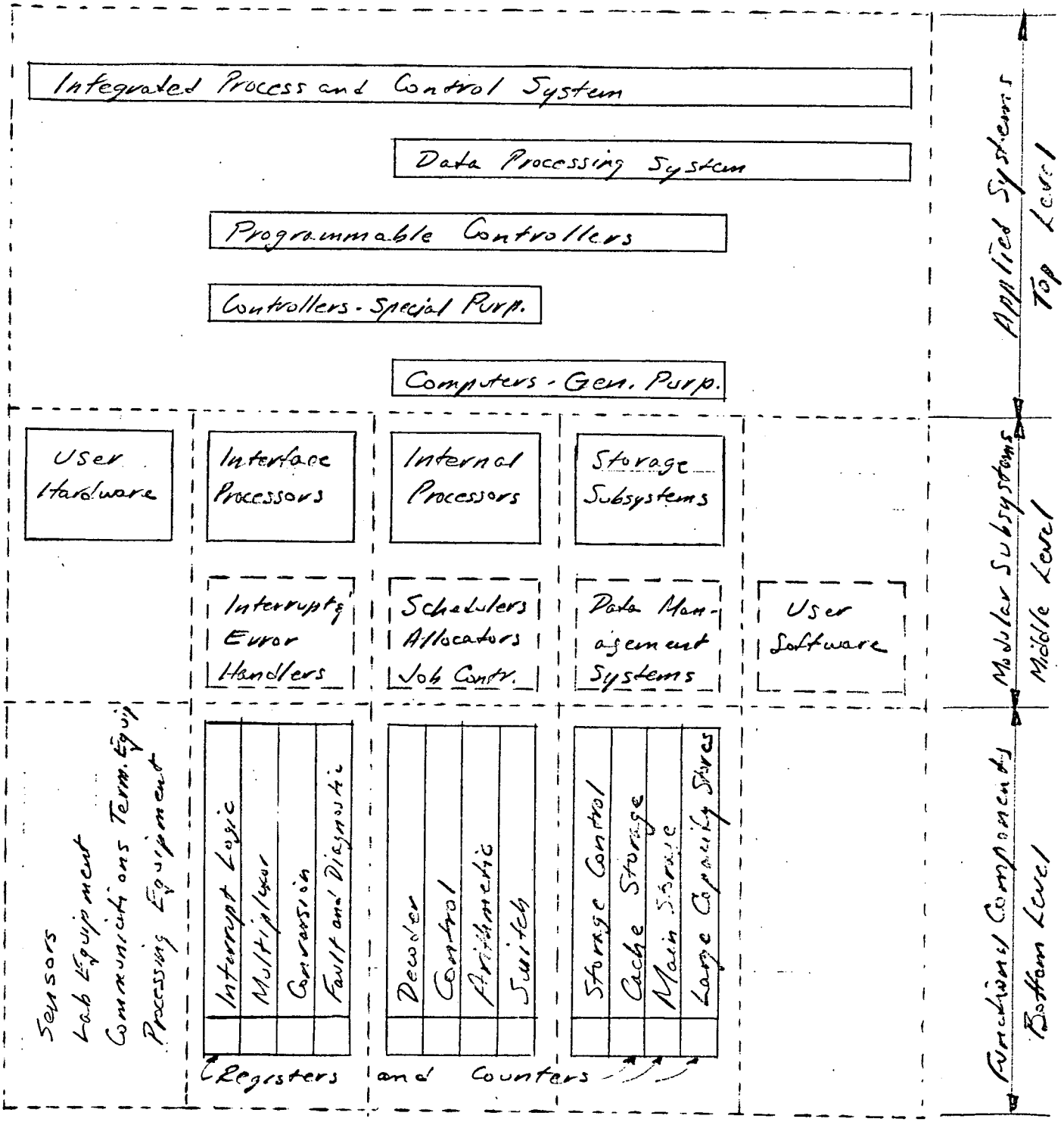

Fig. 21. Structure of family of computers, subsystems, and functional components.

By internal processors we mean the hardware or firmware (read-only memories and micro-programs) which is more commonly referred to as the Central Processing Unit, or CPU. We prefer not to use that nomenclature here, because we are not speaking of a unit, but rather of some collection of functional units named at the bottom, or functional component, level. Internal refers to the fact that these processors are conceived with functions internal to the computer system, as implied by the names of the functional components shown in Figure 21.

The interface processors, by contrast, cover functions involving user equipment external to the computer system. Referring back to the applications described in previous sections, the types of processing that might be included in this area are analog-to-digital conversion (ADC) from sensors, digital-to-analog conversion (DAC) for control of process equipment, transmission line protocol, error detection and correction, and correct transmission acknowledgement-no-acknowledgement (ACK-NAK) of communications equipment, data acquisition or communication line multiplexing or concentrating. Processors -- hardware and firmware -- to drive equipment more closely related to data processing installations (such as cathode-ray-tube terminals and plotters) are also included.

The third category covers storage subsystems, again categorized functionally rather than physically or technologically. Included might be stacks, cache memories, main stores, large-capacity stores, on-line stores, and archival stores in which permanently recorded data can be taken off or put on line (tapes, discpacks, possibly photographic plates).

In each of the foregoing instances, the modular aspects are stressed, so that modules can be added or removed to change the size or capacity of a system, and new technology modules of a given function can replace obsolete examples of the same function without providing special interface hardware or software.

The bottom level comprises the assemblies that go to make up the modular subsystems. Whereas examples of the second level are the modules shown in Figures 3, 13, and 16, examples of the components of the bottom level are shown in Figures 4 and 5. As implied by both of these figures, the particular technical or physical embodiments of these components are not relevant. Whether hard-wired sequential and combinatorial logic are used, or micro-programmed read-only memory, it is the function, and the nature of the inputs and outputs, which are to be standardized in the proposed development project. It is the intent to make the structure as independent of technology as possible, rather than to base the structure on a particular technology. Furthermore, by using as the foundation of the structure a standard bus, or set of busses and switches, and an indefinitely large set of modules, we have made the structure independent of an organization or control hierarchy.

We shall discuss the representative family in a little more detail in the next section.

6.3.2 The Structure of the Computer Families

We -- and, initially, the development project -- are concerned with three families: the internal processors, the interface processors, and storage subsystems. Additional, but later, consideration can be given to software. For example, an operating system may be segmented and modules incorporated in various subsystems (with the possibility of implementing such executive modules in read-only memory using micro-programming), and even perhaps in the functional components. Thus, a processing subsystem might be varied by augmenting it with several versions of very high-speed executive controls embodied in read-only memory.

Examples of functional components of modular subsystems are shown in Figure 21; further examples can be found in Figures 4, 5, and 9. The Burroughs D-machine, or Interpreter (Reference 13) provides additional examples: Logic Unit, Memory Control Unit, Decoder Unit, Micro-program Memory, Micro-program Buffer, Switch Interlock. An example of storage control processors is given in Figure 22; it comprises Main Storage Control, Buffer Storage Control, and Channel Buffer Control.

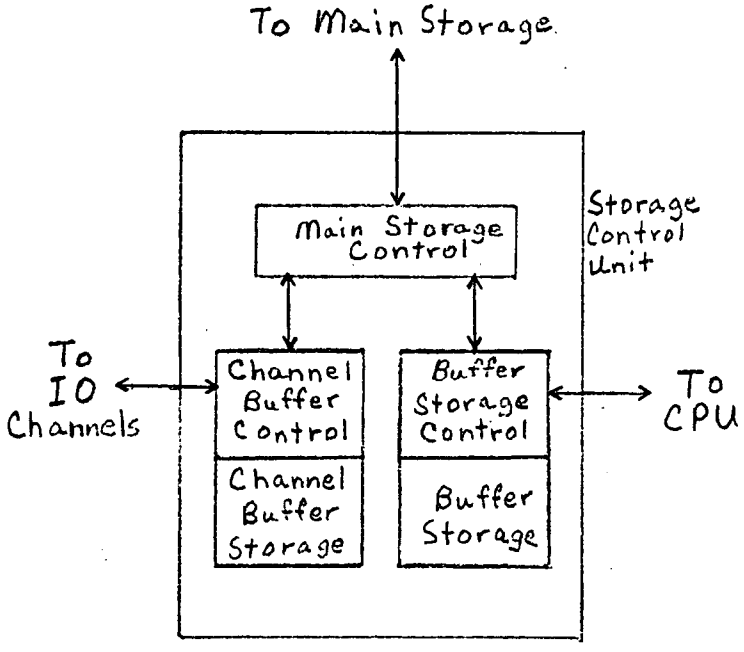

Figure 22.* An Example of a Storage Control Processor[14]

Using Figure 21 as a skeleton, and pieces of some of the diagrams mentioned in the preceding paragraph, it would be possible to draw fairly elaborate and detailed examples of structures of the three types of families we have hypothesized. In practice, aside from obviating the work involved, there is some advantage in having such detailed examples in the implicit form in which they now exist. Figure 21 is detailed enough to give a clear indication of what a full-scale development project might come up with; there is perhaps already enough detail for workers in this field to take exception to, as if it were presented as a final product of investigation and study, rather than as an example. The other diagrams -- Figures 4, 5 and 9 -- and references will provide enough real-life substance to the arbitrarily selected "functional components" so that the reader can mentally construct his own standard structure. Finally, Figures 13, 14, 16, 18, and 20 will provide examples of the top-level, applied systems which might be synthesized from the standard modules the reader will have mentally selected.

In the development project, much, much more detail will, of course, be necessary. However, such detail will be established by selecting from among specified parameters mentioned in the descriptions of Section 4.2 The final result might be a definition or description of a structure and its elements that may well bear a resemblance to such descriptions as those of the Digital Equipment Corporation's PDP-11, the Lockheed Electronics Company's "SUE" (System-User Engineered Minicomputer), the GRI Computer Corporation's GRI-909, and the Burroughs Corporation's Interpreter (D-machine).

We must emphasize that we do not endorse these machines, nor do we advance them as candidates for standards. We present them here as explicit (but limited) examples of a possible structure of standard families of processing and storage modules.

SECTION 7. IMPLEMENTATION PLAN

7.1 Previous sections have dwelt on the activities of the group charged with specifying the NASA family of minicomputer systems. The size of the development group has been approximated, the duration of short- and long-term efforts postulated, and goals and types of tasks pretty well defined. Beyond creating an organizational form for the group, which can be a part either of a NASA Center or of a contracting firm, there remains only the detailed planning of the development. This, by no means a trivial task, is left to the contractor or NASA Center which will perform the work because of the great variety of approaches possible. A specific plan for this part of the implementation can only serve as an example, perhaps an irrelevant one, for the developer to follow or ignore as he sees fit.

7.2 There are, however, important goals for NASA management external to the development itself, both before and after the commencement of technical work. These are, briefly:

- Establishing a concensus on the proposed activities of this Appendix among the members of the NASA ADP community who would be affected by them
- Revising the Appendix as indicated by the concensus
- Staff action to obtain funds
- Preparing an RFP (or administrative document) and selecting a contractor (or Center organization)
- Monitoring and guiding the development
- Publishing the specifications and guidelines for their use
- Establishing a group to update or supplement the specifications.

Completion of each of these actions constitutes a milestone of the implementation plan. The means of reaching them is discussed in the following paragraphs.

7.2.1 Establishing a Concensus on the Proposed Activities of This Appendix

This action is necessary to assure the widest possible use in NASA of the minicomputer family components, once they have been specified. Their use cannot be compelled - to do so would contradict NASA policy on Center and Project autonomy, which extends down to individuals, designed to encourage innovation needed for progress in applied technology. Therefore, adhering to a standard family of minicomputer components must be agreed to in principle by a reasonably large segment of the presumptive user community in NASA before specification of the family is undertaken. Prior to giving its assent, the user community is entitled to a say in what components will form the family, what steps will be taken to specify capabilities suited to their applications, what justification for bypassing standard components will be accepted, and so on.

These ends can be served by circulating the Appendix, accompanied by a statement of policy on implementation, for review and comment by

appropriate NASA officials and their technical staffs. A suggested means of securing substantive comment representative of group concensus, superior to correspondence because of the interaction permitted between opposing views and the consequent maturity of the conclusions reached, is the working session or symposium.

7.2.2 Revising the Appendix to Agree With the Concensus

Assuming NASA readers of the Appendix agree in principle on using specified minicomputer components in preference to others, the substantive changes they suggest should be incorporated. At the time that this is done, the Appendix should be removed from the parent document and supplemented to stand alone as a plan for the development. To obtain further constructive comment, the revised and supplemented Appendix, or Plan, is then reviewed by members of the academic community selected for their familiarity with systems of small computers, the intended applications, transportable software, the development of standards, etc. Suggestions of merit so obtained are likewise incorporated in the plan.

7.2.3 Staff Action to Obtain Funds

This must be accomplished sufficiently far in advance, if the work is to be done under contract, to assure that funds will be available in FY 1973. Postponement until the following year would probably render most of the 200 minicomputers NASA estimates it will acquire in FY 73 incompatible with any minicomputer family that is finally specified.

7.2.4 Preparing an RFP and Selecting a Contractor (or Center Organization)

The Plan will contain ample material from which to draft a request for proposal (RFP), which in final form will be used to elicit bids for developing the minicomputer family specifications under contract. It would be advantageous to offer a preliminary version of the RFP to interested industry members for comment; alternatively, a small contract can be let to refine the document as necessary. When the bids are in, they can be

evaluated with the help of Section 8, which follows, and if necessary with the assistance of a contractor qualified to judge the proposals' merits. Award of the contract is then made based on these evaluations and standard criteria used in such cases.

If a Center assumes responsibility for the development, it may elect to proceed in-house rather than under contract. If so, the Plan will serve as a basis for preparing the detailed program, alluded to in paragraph 7.1, for the conduct of the development.

7.2.5 Monitoring and Guiding the Development

Headquarters, or the NASA Center which assumes responsibility for the development, will administer and monitor the contract, if that is the medium chosen for performing the work. This will involve establishing close liaison between NASA users of minicomputers and the development group to ensure the latter access to valid data on application of the devices. In addition, the short-term work of the group, to specify minicomputer system components needed for a selected application, will require that special arrangements be made with the design team concerned with the application. The latter must agree to participate in the development of standard minicomputer system specifications to the extent required. In practice this will mean incorporating design revisions suggested by the development group to broaden the applicability of equipment and software.

The technical monitor must not only promote liaison, but also provide guidance to the development effort. Such items as the functional divisions within the minicomputer family, the order in which family branches are developed, the overall schedule, etc., will require NASA decision. Because the interests of more than a single Center or Program Office are affected, some Headquarters involvement in these decisions is required.

7.2.6 Publishing the Specifications and Guidelines for Their Use

Once the specifications for the minicomputer family components have been developed, they must be made available for use throughout NASA. This is done by publishing them officially in appropriate administrative or technical publications distributed to organizations potentially concerned with acquiring, modifying, using or disposing of computing equipment. These publications also provide a ready means for updating or supplementing the specifications, which will be required. Guidelines for the use of the specified minicomputer components will also be required and may be published with the specifications or separately. The guidelines would indicate when the standard components should be used, when their use could be waived, and what information is required to obtain a waiver. They would present other information to help the user decide if the standard components suit his application.

7.2.7 Establishing a Group to Update or Supplement the Specifications

Although the development effort will conclude in three years, the need for maintaining and adding to the specifications will persist as long as minicomputer technology undergoes rapid change. As the current high rate of change is expected to continue unabated into the next decade, it will be necessary to provide for necessary modifications and additions to the specifications after their 3-year development. A group should be founded and charged with this responsibility early in the development. By working closely with the development group, they can acquire the skills necessary to carry on its work after the termination of the development. Their function will be to analyze exceptions granted to the use of the standard minicomputer family components and determine when supplemental standards - for new equipment types or for new applications - are required. They should be aware of technical progress in minicomputer systems technology and of the plans of government and industry standards organizations.

SECTION 8. PROPOSALS TO PERFORM THE DEVELOPMENT

This section sets forth the major technical considerations which apply in evaluating proposals to perform the development. Considerations of an administrative nature, (for example, the contractor's ability to perform or whether he qualifies as a small business) are covered in existing directives and hence omitted here. What is discussed are the major points we feel contractors should glean from the request for proposal and respond to.

At time of writing, no RFP has been prepared; and, indeed, none will be if NASA undertakes the development without contractual assistance. The nature of the development described in this Appendix is expected to change somewhat owing to the critical review to which it will be subjected in the search for a concensus described in Section 7. Consequently, our remarks in this section are confined to aspects of the development which

seem immutable, although it is understood that they will not thereby entirely avoid the risk of change-induced irrelevance.

8.1 UNDERSTANDING OF THE PURPOSE

The goal of the development is specifications for a family of minicomputer components which are to a degree interchangeable. However, the purpose behind the goal is achieving economies due to such interchangeability from quantity purchase of components, simplified design of systems of components, and more easily developed and sharable software. These economies are not attainable unless the number of different types of each component is small, hardware components are genuinely interchangeable and software transportable within the bounds set in previous sections. Hence, the developer, whoever he is, must guard against diluting these subobjectives throughout the course of his work, even in the face of technical difficulties. Concern for attaining actual economies should illuminate his analysis of the NASA requirement for minicomputer systems, the cost of the development and the value of its benefits.

8.2 UNDERSTANDING OF THE PROBLEM

The problem, as defined in previous sections, is that of specifying the NASA family of minicomputer components: determining the family structure, constituent component types and their places in the structure; identifying the parameters to be specified for each component; and determining the values of the parameters. An understanding of the problem implies, besides comprehension of this problem-objective, recognition of the technical problems which will be met enroute to its attainment. For example, an obvious pitfall is the specification which gives a strong advantage to one manufacturer. Another difficulty is posed by the conflicting objectives of generality, needed to assure a long-lived specification, and of detail, to assure the components' interchangeability. A thorough treatment in a proposal of anticipated technical problems reveals an insight which could forestall unpleasant technical surprises in the development.

8.3 PROPOSAL CONTENTS

The following outline is intended to be representative of the contents of proposals received.

8.3.1 Understanding of the Problem; Goals and Objectives

8.3.2 Technical Approach

- General Description
- Restatement of Goals and Objectives - Exceptions, Reservations, Revisions
- Description of Major Difficulties Expected - how to approach
- What standards or guidelines will look like
- Outline of Tasks - New proposed task list (detailed description in next section)
- Description of Methods and Techniques
- Approach to Testing or Evaluation of Results - how good are they? Measures of Effectiveness.

8.3.3 Detailed Descriptions of Tasks

For each task:

1. State Task, Subtasks or Task Units
   - Data collection
   - Analysis
   - Forecasts
   - User guidelines
     - Actions
     - Products
   - Synthesis of Product Guidelines (Standards)

2. Explain relationship of task to end result
3. Explain how task will be executed
4. Description of output - report or techical memorandum.

8.3.4 Project Plan

- Task Schedule
- Task Interdependencies
- Reports and Technical Memoranda

8.3.5 Estimates of Time and Cost

REFERENCES

Section 1

1. NASA Management Instruction 1101.1E, July 15, 1971.

Section 2

1. D. C. Hitt, G. H. Ottaway and R. W. Shirk, AFIPS Conference Proceedings, Vol. 33, Part 1, 1968.

2. Jean Bartik, "Minicomputers Turn Classic," Data Processing Magazine, January 1970.

3. AUERBACH Computer Technology Reports, Minicomputer Notebook, AUERBACH Publishers, Inc., Philadelphia, 1970, Introduction, pages 9 - 10.

4. Ibid, page 4.

5. D. L. House and R. A. Henzel, "Semiconductor Memories and Minicomputers," in Computer, March/April 1971.

6. R. M. Davis, in address to IEEE Symposium on Minicomputers at National Bureau of Standards, Gaithersburg, Maryland, March 1, 1972.

7. EDP Industry Report, June 25, 1971, pages 3 - 6.

8. R. K. Jurgen, "Minicomputer Applications in the Seventies," IEEE Spectrum, August 1970, page 38.

Section 5

1. Harvard Business Review, July-August 1971, Vol. 49, No. 4, pp. 45-74.

Section 6

1. Electronics, December 21, 1970.

2. "SUE, System User Engineered Computer," Lockheed Electronics Company, Inc., 1972.

3. "GRI-909 Computer" (brochure), G-R Industries, Inc., 1969.

4. R.J. Clayton, B.A. Delagi and R. Elia-Shaoul, "Evolution Breeds a Minicomputer That Can Take on Its Big Brothers," in Electronics, October 11, 1971.

5. W.B. Riley, "Minicomputer Networks -- A Challenge to Maxicomputers?" in Electronics, March 29, 1971.

6. "Hughes H4400 Computer System," Hughes Aircraft Company (FR 69-11-1245), 1969.

7. Spectrum, February 1971.

8. R.A. Henzel, "Some Industrial Applications of Minicomputers," in Computer, September/October 1971.

9. H. Broekhuis and M.S. Jongkind, "Planning and Managing Process Computer Projects," in Control Engineering, February 1971, p. 60.

10. Control Engineering, April 1971.

11. J. Barth, "Using Minicomputers in Teleprocessing Systems," in Data Processing.

12. F.E. Heart, R.E. Kahn, S.M. Ornstein, W.R. Crowther and D.C. Walden, "The Interface Message Processor for the ARPA Computer Network," AFIPS Conference Proceedings, Vol. 36, 1970.

13. "The Interpreter" (TR 70-2), Burroughs Corporation, February 16, 1970.

14. H. Katzan, Jr., "Computer Organization and the System/370," Van Nostrand Reinhold Company, 1971.