CR-152059

N78-70018

PRELIMINARY STUDY FOR A NUMERICAL AERODYNAMIC SIMULATION FACILITY

FINAL REPORT

BY: N. R. Lincoln
A. A. Vacca
R. A. McHugh
R. W. Johnson
D. B. Bonstrom

OCTOBER, 1977

Prepared under Contract No. NAS2-9457 by:

CONTROL DATA CORPORATION
2800 Old Shakopee Road
Minneapolis, Minnesota 55440

for

AMES RESEARCH CENTER
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

510

PREFACE

This Final Report presents the findings of the Preliminary Study for a Numerical Aerodynamic Simulation Facility (NASF) as determined by Control Data Corporation. The document consists of six sections. Each section addresses a specific aspect of the Control Data study, and each section includes appendixes, a reference listing, and bibliography where appropriate.

Sections 1 and 2 discuss current and future electronic technology and architecture technology as a basis for valid recommendations. Section 3 gives an analysis of existing flow models and the measurement criteria to be used for benchmarking the project. Section 4 highlights the objectives relative to the design of the facility. Section 5 discusses a preliminary system design which could meet the NASF objective. Section 6 gives an example of a configuration to establish requirements for constructing the site.

In addition to this Final Report, a separate Summary Report presents the salient findings of this preliminary study and summarizes the first phase of a program for the development of a Numerical Aerodynamic Simulation Facility.

CONTENTS

3<

4<

5<

6<

SECTION 6. PRELIMINARY SITE REQUIREMENTS

7<

8<

SECTION 1

ELECTRONIC TECHNOLOGY SURVEY AND PROJECTIONS

INTRODUCTION

The electronic computer industry, in over 30 years of existence, has enjoyed phenomenal success and growth. Growth has been in virtually all aspects including economics, applications, and technology. The raw computer power has also increased almost beyond comprehension from Eniac to today's supercomputers. As computers become faster and more powerful to meet today's needs and as new and more complex applications are introduced, greater computing power will be required that will lead to tomorrow's supercomputer. The state of the art in designing and building computer components will be a significant factor in the development of these computers.

The early advances in performance were largely attributable to component technology changes and improvements. Vacuum tubes, transistors, and integrated circuits contributed tremendous advances in computer speeds. However, today's computer technology is approaching some very real limits to such things as reduced component size, faster switching times, and greater component density as primary means of achieving additional computing power.

A problem such as computational aerodynamics challenges computer designers to search for whatever techniques may be found to match the machine to the problem, and to the extent possible, the problem to the machine. The Preliminary Study for a Numerical Aerodynamic Simulation Facility is a project to propose methods to accomplish these matches. This survey and report on electronic technology is the part of this study intended to provide a technology platform on which to base computer architectures proposed for a computational aerodynamics design facility.

9<

OBJECTIVE

The objective of this survey is to provide a summary and evaluation of the current state of computer component technologies and projections as to the future of these technologies for possible application to the computational aerodynamics design facility.

SCOPE

TECHNOLOGIES SURVEYED

A survey of all technologies, which may pertain to such a facility, is not possible because of limited resources nor is it prudent to investigate all, since some are well established and fairly stable. Those technologies which are relatively new and/or were deemed to have potential application in the proposed facility were investigated and are included in this report. They are as follows.

- Charge Coupled Devices (CCD)
- Magnetic Bubble Memories
- Electron Beam Addressed Memory (EBAM)
- Large Scale Integration (LSI) of Semiconductor Circuitry
- Josephson Devices
- Advanced Rotating Mass Storage
- High Speed Data Channels
- Advanced Test Equipment

EXTENT OF INVESTIGATION

To the extent possible consistent with time and funds available, the above technologies were investigated to determine the following.

- Relative merit
- Limitations
- Cost
- Risks
- Operational implications

METHODOLOGY

SOURCES OF DATA

The amount, types, and sources of data vary considerably by technology and depend on many factors such as age, relative interest to the industry, and the degree to which Control Data has independently pursued the technology. In general, the following sources were used, as applicable, for each technology.

- Control Data project files
- Technical journals, bulletins, and periodicals
- Vendor and industry surveys
- Specialists within Control Data

APPROACH USED

The subject technologies were each placed in one of three categories relative to the type and degree of investigation to be applied. The categories and the approach used for each are as follows

- New technologies which have been actively investigated by Control Data for considerable time, these are CCDs, bubble memories, EBAM, and LSI. For this category, a thorough, although nonexhaustive, utilization of all four of the above sources was made. Material was summarized and evaluated by specialists who have been directly involved with a given technology.

- New technologies which are not directly pursued by Control Data; these include Josephson devices and TELD/MESFET portions of LSI. Sources of information for this category were publications and vendor contacts; data was extracted and evaluated by a specialist in a related field.

- Technologies which are not new in themselves but are evolutions of prior art, these are high-speed data channels, advanced rotating mass storage devices, and advanced test equipment. The first two of these technologies were each assigned to a specialist to obtain an assessment of the evolution to date and to project further evolution into the 1980's. Advanced test equipment is an exception in that it is ancillary to the system, yet is very crucial to future high performance devices. Test equipment specialists within Control Data were consulted, and some information was gathered from producers of test equipment.

OBSERVATIONS

CHARGE COUPLED DEVICES

General Description

Charge coupled device (CCD) memories have been available for over 2 years. Development has progressed to a point where CCDs warrant serious consideration in a storage hierarchy. They are presently available in 16-kbit devices with 65-kbit samples just becoming available. The expectation of 256-kbit chips by the early 1980's is high.

The CCD is a metal-oxide semiconductor (MOS) device. The CCD cell is a MOS capacitor, and a shift register is composed of a series of these capacitors. Data is stored in a cell as the presence or absence of a charge and is moved through the shift register by the application of clock pulses to overlying clock lines.

The manufacturing process is N-channel MOS (NMOS) technology, but for the storage cell, the process is simplified by three factors.

- No metal/oxide contacts

- No diffusions (for source or drain)

- No metal lines

The peripheral circuitry, however, is normal NMOS technology.

Relative Merits

Because CCD technology is relatively new and several semiconductor manufacturers are engaged in it, the rapidly increasing user interest and application should cause some fairly rapid advances. Since CCD manufacturing processes are similar to those used for MOS microprocessors and random access memories (RAMs), CCD technology should get impetus from these markets. By the early 1980's, an 18-Gbit memory appears a reasonable expectation. It should operate at a rate better than 1·Gbit/s. Access time is anticipated to be 1 to 2 ms.

Assuming a normal technology improvement pattern for CCDs, the availability of a 256-kbit device by 1980 seems likely. Such a device would make feasible a storage system with the following characteristics. The failure rate projections are based on experience with, and history of, MOS circuits in general since a CCD is an MOS device. Experience with CCDs may reveal different failure mechanisms, but this is not expected to cause a significant difference in overall failure rate.

Capacity	1.8×10^{10} bits, solid-state, volatile, refresh required
Physical size	1.6 to 2.1 m^3, dependent on type of packaging and cooling
Power	43 kW, operating 14 kW, standby
Data rate	1.6×10^9 bits per second
Access time	2 ms, worst-case 1 ms, average
Chip	256 kbit, dynamic MOS

Failure rate	5.8 failures/month, no SECDED
	0.6 failures/month, with SECDED

The size of the basic CCD building block, that is, the chip or integrated circuit, permits flexibility in configuring a memory system. Such a memory can range from bit-serial to many bits in parallel. This also allows increasing total bandwidth of the memory by paralleling bits.

Limitations

Perhaps one of the biggest drawbacks to utilization of the CCD in a storage system is the dynamic nature of the device. That is, the data which is stored as a charge in a MOS capacitor must be periodically refreshed or the charge dissipates and the data integrity is lost. Typically, refresh is accomplished by cycling the CCD memory at some minimum rate while it is not being accessed, increasing the clock to normal rate for data access. Implemented in this manner, refresh has no effect on data transfer rate after access has been accomplished but adds to the total memory access time analogous to rotational latency in disk storage access. This degrades overall performance somewhat, although proper design can minimize the impact. In addition, the CCD memory is volatile, that is, it loses all data on loss of power unless a back-up power source, such as battery, is provided. Fortunately, the low standby power requirements of the CCDs (about 1 μW/bit expected) eases this problem.

The organization of CCDs, at least at present, is a limitation, although to a moderate degree. In order to achieve optimum performance from this type memory, data block size is highly dependent on chip organization. This requires matching system design (block size) to chip organization. Unfortunately, vendors today have little, if any, commonality in chip organization, and it seems that no standard is emerging.

Although CCD technology is relatively new, this can be a limitation since problems generally accompany new technology. It seems inevitable that system implementation of CCDs, in the near future at least, can expect growing pains as the technology matures.

Cost

Based on chip cost comparisons, a CCD memory using the 65-kbit devices available today would have an estimated one-third cost, on a per bit basis, versus using 16-kbit devices. Likewise, with the introduction of the 256-kbit chip, expected in the early 1980's, the system cost should have the same ratio when compared to that using the 65-kbit devices.

Risks

Because CCD technology is a phase of MOS technology which is relatively well-established, risks are not as great as with a totally new technology. Some potential risks, although not very probable, are in the category of less-than-normal development relative to semiconductor development in general. The above projections of cost, density, data rates, and so on, are based on what have been normal trends in the semiconductor industry. If for some reason CCDs receive less attention, these goals may not be realized.

Operational Implications

CCD memory is a volatile storage media requiring refresh to maintain data integrity. System implementation must take this into account. For nonvolatile storage, the system design would have to provide auxiliary power for emergency power failure as well as for any normal power-down.

Considering the above characteristics, CCD memory could provide a medium to large storage function. Cost per bit is a limitation on the large-scale end where CCD cannot compete with disks. Speed and access time are limitations on the small-scale end where more suitable RAMs are available. CCDs fall in the gap between high-speed RAM and disk storage, making it a candidate for intermediate or backing storage for a high-speed computational engine.

MAGNETIC BUBBLE DEVICES

General Description

Magnetic bubble devices are a class of integrated circuits which use mobile magnetic domains for digital storage and processing. The basic binary element in such devices is a localized flux reversal in a thin flat film, the magnetic orientation of which is normal to the plane. This localized reversal, termed a bubble, is cylindrically shaped, energetically stable, and can be moved about in the medium with very little expenditure of energy.

Binary data is usually represented as a coded bubble/no bubble pattern. Controlled existence and motion of such patterns are generally provided by metallic overlay patterns deposited on top of the bubble medium. Typical control overlays include conductor patterns, soft magnetic elements, or combinations of both. In each case, the overlay serves to define storage cells and introduce local magnetic field gradients which manipulate the bubbles in coherent fashion. For example, the popular shift register bubble memory uses a rotating in-plane field to activate the magnetic overlay and current pulses to write, erase, and access various locations. Data output can be accomplished in a variety of ways, but magnetoresistors formed in the magnetic overlay layer are most generally used.

Bubble devices may be classified as either logic or memory depending on function. The logic devices utilize the magnetostatic interaction of bubbles to perform prescribed operations on the data. Despite the fact that a complete set of logic functions and even a bubble computer have been described in the literature, such devices tend to be slow and sensitive to their magnetic environment. It is probable that bubble logic will remain a laboratory curiosity in the foreseeable future.

Bubble memory devices are of three general types: conventional linear shift registers, two-dimensional shift registers (for example, lattice devices), and random access memories (RAMs). The RAMs have tended to be cumbersome and complicated, therefore impractical for use in conventional computer systems. They will not be considered here.

Lattice devices are a recent and potentially important development. A closely packed array of bubbles (reminiscent of a hexagonal crystal lattice) is manipulated en masse without a fine featured overlay. Mutual repulsion and a confining border are used to define the lattice positions, and very small bubbles can be used. Since no voids are permissible, information is stored either by varying the wall state of the bubbles or by using multiple layers, one to define the lattice and the second to carry the information. Generally, a single I/O port is provided. Access to individual rows or spots in the lattice is by block movement of the entire storage area. Although these devices offer great potential returns in cost and density, it is unlikely that a large, high-speed version will be available for many years. Several organizations (IBM, Rockwell, and Univac) are investigating them, but no fully functional lattice device has yet appeared in the literature.

This leaves the linear shift register memory as the only viable choice offered by today's bubble technology. The linear register can be organized in a variety of ways including the single serial chain, the serial-parallel-serial (major/minor loop), self-decoding, dynamic ordering, and hierarchical loops. Of these, the major/minor loop organization is the most popular and furthest developed. A complete set of functional elements has been described and the access time is reasonably fast. Additionally, the memory can be made flaw-tolerant quite easily (increasing yields), and the processing and wiring complexity is low.

The major/minor loop organization uses a number of small parallel shift registers (minor loops) for the storage of data. These surround and are interconnected by a primary shift register (major loop) which contains the read/write/erase apparatus. In operation, data is manipulated serially in the major loop but transferred to and from the minor loops in parallel, one bit into each loop. The advantage, of course, is that the access time becomes primarily a function of minor loop length rather than chip capacity. Defective loops can be ignored with the straightforward application of a PROM I/O map.

Merits

The growing interest in magnetic bubbles may be traced to a number of features which, in some ways, combine the virtues of both magnetic and semiconductor devices. Bubbles are nonvolatile, radiation-resistant, fairly dense, and have solid-state reliability. Power consumption is low (a few μW/bit), and zero standby power is required. The storage medium is inherently digital, and as a result, phenomenally low hard error rates have been measured (10^{-14}). Bubble memories also have reasonably fast access when compared with rotating storage, and the cost is projected to be low.

Compared with conventional integrated circuits, bubble devices are extremely simple structurally. The most common circuits require only two delineated layers and one precise mask alignment. Although fine-line lithography is required, there are no complex structures or diffusions, and the processing is free from ionic contamination problems. Further, the completed devices usually require very few external circuit connections. This kind of simplicity generally allows bubble circuits to be very dense, very high capacity, and eventually very low in cost.

Bubble memory technology has reached a point where a large, relatively high performance memory system can be envisioned in the next 2 to 3 years. Feasibility of all components and circuits has been demonstrated. Development of a memory system with the following characteristics is foreseeable, yet represents a significant effort.

Capacity	2×10^{10} bits, solid-state, nonvolatile
Physical size	1.8 to 3.9 m^3, dependent on type of packaging and cooling
Power	20 kW, maximum 12 kW, typical 0 kW, standby
Data rate	0.2×10^9 bits per second

Access time	5 ms, worst-case
	0 to adjacent blocks, or if next block is queued
Chip	10^6 bits, major/minor loop, epitaxial garnet, 2-μm bubbles

While bubble memories have existed for a relatively long time in the laboratory, they have not been implemented in large configurations sufficiently to enable reasonable error rate estimates. The above reference to 1 error in 10^{14} bits represents laboratory measurement at the bubble device level. While it is an impressive figure, allowance should be made for other system contributions to errors. Nevertheless, bubble technology appears to have a potentially low error rate.

Limitations

On the other hand, bubbles have their limitations. Of the three candidates for auxiliary storage (CCDs, EBAMs, and bubbles), bubbles are clearly the slowest both in terms of access time and data rate. High-speed operation (¼1 MHz) has been shown to be possible but not practical, given today's packaging and electronic limitations. Low cost, while promised, has never been demonstrated. Wafer production continues to be a costly manual operation and no progress will be made until manufacturing is scaled up. Bubbles also are vulnerable to temperature and magnetic fields but these problems can be minimized by suitable package design.

The bubble medium itself is relatively complex and costly. The most suitable materials today are the rare earth-substituted iron garnets which are grown epitaxially on single crystal, nonmagnetic garnet substrates. The liquid phase epitaxy process requires high temperatures and very costly materials. Despite excellent control and high growth yields, this type of substrate will always be an order of magnitude more costly than silicon and is two orders of magnitude more costly today. The eventual successor to the garnet films may be the amorphous transitions metal films. These materials, prepared by conventional vacuum deposition, provide a suitable range of bubble properties and are extremely cheap. If the current problems with temperature sensitivity, repeatability, and dielectric integrity can be resolved, the entire technology will be revolutionized. For the present, however, these materials appear to be at least 5 years away.

Status of Technology

Most skepticism about the viability of bubble technology can be answered by an examination of 1977 activity. More than 20 companies are engaged in bubble development and seven products have been announced. Earlier problems with chip yields, packaging, and reliability appear to be diminishing rapidly and at least 17 memory systems have been described in the literature. Most of these initial offerings have been conservative in performance, but they are serving to test customer acceptance and develop manufacturing techniques.

The laboratory activity is a better reflection of today's state of the art. Recent developments include one million bit chips, working devices at 1 MHz, single level (no mask registration) circuits, and wide temperature (-40°C to +120°C) operation. Smaller and smaller bubbles are becoming practical, particularly with electron beam and X-ray lithography, and cost projections are plummetting. A likely manufacturing cost for garnet film devices is 0.01 cent per bit by 1980.

Table 1-1 summarizes commercial bubble technology today and expectations for 1980. These estimates are based on interviews with bubble development groups and publications in the literature.

Prognosis

It is obvious that bubble memories are a technical reality, and continuing progress in cost and performance is likely. In spite of this, bubbles have been slow to emerge from the laboratory and the acceptance in the computer industry has been limited. There are a number of reasons for this. First, and most important, a great uncertainty exists in cost. The bubble medium is very expensive and will remain so until production volumes are established. Further, the search for lower cost alternatives will continue at a relatively low profile until a production scale motivation exists. A second factor is performance. Most of the early memory products have been slow and fairly expensive. A favorable point on the cost/performance curve can be reached only with larger, faster chips. Other

TABLE 1-1. BUBBLE TECHNOLOGY TRENDS

Bubble Medium	1977	1980
Material	Garnets	Garnets
Bubble diameter	4 to 6 μm	1.5 to 3 μm
Chip Design	**1977**	**1980**
Organization	Serial, Major/minor loop	Major/minor loop
Fabrication technology	Optical, two-level	Optical, one-level
Chip density	1.5×10^5 bits per cm^2	1.5×10^6 bits per cm^2
Chip capacity	10^5 bits	10^6 bits
Shift rate	100 to 200 kHz	100 to 500 kHz
Memory Systems	**1977**	**1980**
Chips per module	1 to 16	1 to 16
Average access time	1 to 4 ms	0.5 to 4 ms
Data rate/channel	50 to 150 kbits/s	100 to 500 kbits/s
System capacity	10^5 to 10^7 bits	10^5 to 10^9 bits
Operating temperature	0° to 50°C	-40° to 120°C
Cost	0.05 to 0.2 cent per bit	0.01 to 0.05 cent per bit

factors include the unavailability of integrated support electronics, lack of standardization and multiple sources, and of course, the lack of a product commitment by IBM. One should also note that bubbles are a dynamically moving technology. Many potential manufacturers have resisted settling upon a specific technical level since improvements are being announced almost routinely.

The future, nonetheless, seems fairly bright for magnetic bubbles. Almost certainly they will settle into that portion of the memory market that requires reliability and nonvolatility. The wide range of capacities, organizations, and performance also suggest that bubbles will have a more universal appeal than other media such as the electron-beam-addressed memory (EBAM) and fixed-head rotating storage. Only time will tell. however, whether this medium offers enough to displace existing technologies from their accepted applications.

ELECTRON-BEAM-ADDRESSED MEMORY

General Description

Electron-beam-addressed memory (EBAM) technology, per se, is not particularly new; it dates back to the 1950's as the Williams tube. This was a 5-inch CRT with a capacitance target and secondary emission detection. The results were relatively low signal levels, short storage lifetime, and little success. By the late 1960's, however, the semiconductor industry had developed MOS technology which was found to provide a good target with high gain for an EBAM.

The data pattern is a two-dimensional array on the target, and it is addressed by deflection of an electron beam. Data is stored as the presence or absence of a positive charge along narrow tracks on the MOS target. Charge storage is accomplished by the application of a bias voltage to the target while the beam is swept along the track during a write or erase cycle. In a subsequent read cycle, the beam is swept over the same track. Where a charge is present, a current is produced to a sense circuit. Where a charge is absent, no significant current is produced.

Relative Merits

EBAM technology has been drawing increasing interest, particularly from a technological viewpoint. It is a blend of several technical fields such as semiconductors, vacuum tubes, and high and low voltage electronics. If solutions or improvements for the problem areas mentioned in the following Limitations paragraph can be found, EBAMs have considerable potential as on-line mass storage devices.

The MOS target is a simple semiconductor which provides high yield and low cost. It is also a relatively nonvolatile storage media. On loss of power, data integrity is retained for a few weeks, sufficiently retrievable. Therefore, this type of storage device would not require backup power, provided it had sufficient capability to shut down in an orderly manner upon loss of power.

Although the basic EBAM storage building block (the tube) has a rather large capacity, some configuration flexibility exists in multitube systems. This permits sharing of electronics for lower cost/bit and paralleling data bits for higher bandwidth, if necessary. However, EBAM transfer rates are about 10 Mbit per second or greater.

Additionally, like the CCD, an EBAM is a totally electronic device. It does not have some of the negative aspects inherent to most electromechanical devices.

Limitations

While functional EBAMs exist today, they are not yet sufficiently established for building an economical and reliable system. Cost-effective production must be proved, and quantitative data is needed on reliability, error rates, pattern sensitivity, and so on. In addition, to decrease costs while increasing capacity and performance, three significant problem areas must be addressed.

- The basic target uniformity and life must be improved. The electron beam is a source of radiation, and as such, causes damage to the target with time. In order to avoid this target fatigue, present EBAMs rotate or permute the data to average the usage of the target area.
- The electron optics will require upgrading to accomplish submicron beam diameters in production. The small beam will be necessary to achieve density goals. Also, a two-stage deflection system would have a deflector with a large number of lenslets requiring high-precision tooling.

- The cathode needs further development. As the beam is made smaller to achieve greater density, the brightness must be increased. Conventional dispenser cathodes used at present could possibly reach 100 Mbits per tube. To go beyond 1 Gbits as projected, some other technique is required. Field emission cathodes have been applied in a few areas but would need effort to bring this technology to a practical level.

Tubes with a capacity greater than 10 Mbits and an access time better than 30 μs have appeared in recent literature (reference 1-1) but have not as yet been demonstrated as viable system components. Future developments could conceivably produce tubes with 1-Gbit capacity by 1980.

Regarding electron beam memory technology in the context of the auxiliary backing storage considered as a requirement for a special computation facility, consider the following major characteristics of this memory technology in a general sense, in order to highlight the attributes of EBAM memories which can be expected to limit system reliability.

- Precision high voltage power supplies seem to be an inherent necessity with this memory. The acceleration potential of the electron gun is commonly 10 kV, which must be accurate to about 1 volt. This stable voltage is needed because a spread of only 2 volts in electron energy will cause the beam to assume an elliptical cross-section, radially oriented, at the edges of the target area, since slower electrons are deflected further. The high voltage system, in addition to being physically large, must therefore be constructed of components whose aging characteristics are measured in hundreds of parts per million.

  Voltages of a few hundred volts are required in the deflection system as well, with a precision of about 0.01 percent. Together, the relatively high voltages require a large distribution system and constitute a maintenance hazard and reliability limitation.

- The current memory mechanism is one of linear charge storage and linear beam displacement. There is no threshold for the basic memory element as there is for almost every other memory type.

  For example, a semiconductor memory latch assumes a minimum energy state in its set (or reset) condition, and a fixed amount of energy must be supplied to reverse its state. A similar condition exists for a magnetic core or a magnetic bubble. An MOS memory cell may employ linear charge storage, but such storage occurs on one element of a transistor, a highly nonlinear device. CCD memories also use linear charge storage but with regularly interposed nonlinear sensing elements to requantize the charge pattern.

The use of a linear storage mechanism means that sneak or stray currents or areas of surface contamination can directly and cumulatively reduce the signal strength by reducing the amount of charge stored.

- The lack of a photolithographically defined memory cell has been thought to be an inherent design advantage for EBAM memory, because memory fabrication is simplified and because the memory capacity could in principle be adjusted after assembly to provide an arbitrary signal to noise ratio or to compensate for isolated defects.

  In practice, however, this simpler construction may result in more complex operation, since the cell definition is accomplished by beam deflection. Critical definition has thus been made a part of the memory device rather than part of the fabrication system.

  It may also result in practice that the cells become defined on the silicon dioxide surface in an EBAM, by virtue of the fact that the cell boundaries are commonly written with charge to form a conducting background grid. There is some evidence that cell boundaries become difficult to redefine with the persistence of this grid.

- Perhaps the most attractive feature of EBAM technology is the promise of very fast access time. The basic beam deflection time should be no greater than 50 microseconds and might well be reducible to about 20. It seems unlikely that all of this promise will be realized inasmuch as the need for data permutation will likely persist. This can be a substantial access time penalty for a memory system which encounters high traffic. In such an application, using a common permute algorithm and sequential uniform reading, the access time penalty could grow to several hundred ms.

- The projected EBAM tube is a high vacuum envelope of large size (50 cm). The maintenance of a vacuum seal, microphonics, and out-gassing of internal surfaces are but some of the classic lifetime and reliability limiting characteristics of vacuum tubes.

- While erosion of the electron source will limit EBAM tube lifetime, the more critical aging (and reliability) mechanism is target fatigue. At present fatigue rates and beam currents, constant use of a single spot would result in a lifetime of about 5 seconds. Thus, uniform usage must be sought through the permute mechanism. Since the effectiveness of this procedure depends on the program or usage patterns, fatigue is an obvious reliability factor.

  The controlled damage of the EBAM target from electron beam radiation is similar to damage done to MOS wafers during electron-beam metal deposition in orthodox semiconductor processing. The fact that this processing problem has persisted in that industry for some time suggests that its understanding is less than complete. The integrity of the EBAM storage media directly depends on it being well understood or at least predictable.

- Because the electron beam can be deflected by stray electromagnetic fields, considerable shielding of the tubes is required. Such shielding may reduce EMI as a reliability consideration but at the expense of volumetric efficiency. The present tube/shield combination is roughly 0.007 m^3, and the peripheral high voltage system and electronics add significantly to that volume.

- It has been known for some time that EBAM is not well suited for smaller capacity memories. This results from the physical bulk and cost of the elementary memory components. This factor in itself is not a handicap for the NASA-Ames application because a large backing store is needed (2×10^{10} bits).

- A related performance factor is, however, crucial for this application. The physical bulk of a single tube and the press for reduced costs will drive an EBAM system in the direction of larger capacity tubes. For example, a memory constructed of 2×10^4 EBAM tubes (1 Mbit each) is physically too large for this application, whereas one made from 2×10^4 bubble or CCD chips is not. The data rate from each tube may not be easily adjustable, since there is only a single beam (a segmented target does not give multiple concurrent signals) and increased sweep rates may be difficult at higher densities where the basic signal energy is reduced. Therefore, one concludes that the ratio of data rate (per second in bits) to memory capacity is about 0.25 to 1.0 at present (Microbit model 700) and will diminish to 0.015 with future generation products (Microbit model 950). This ratio is the maximum attainable by a system, since it is the ratio characteristic of the single EBAM tube.

  Corresponding ratios for bubble and CCD chips are approximately 1.56 and 7.8 at the 256-kbit chip capacity, and 0.4 and 2 at the 1-Mbit chip capacity. Volumetric considerations do not exclude the use of either of these chip sizes or types for the present application.

  The desired ratio for the backing store, needed for most effective utilization of the complete computational system, is $(1.6 \times 10^9)/(2 \times 10^{10}) = 0.08$. Thus, both CCD and bubbles can be made to fit, but 128 Mbit EBAM tubes cannot unless they achieve a data rate of 10.24 Mbits per second. If the 950 tube is built at the 2 Mbit per second expected, a 2×10^{10} bit system would have a maximum data rate of only 312 Mbit per second, and this would be attained by operating all of the 156 tubes in such a system simultaneously, so that queuing or permuting of idle tubes could not occur. The transfer time of an 8-kword block would then require 1.64 ms as opposed to 0.32 ms for either CCD or bubbles. The 1.32 ms difference is offset by any access time advantage enjoyed by EBAM, but an overall performance problem will remain.

Cost

At the present state of development, EBAM tubes and electronics have a fairly substantial cost. The tube technology needs refinement, and the electronics are complex because of requirements such as restoring and permuting of the data. Multitube systems share these electronics to keep cost/bit down; this dictates large memories. Cost today is on the order of 0.1 cent per bit for a memory of about 100 Mbits. If the problem areas discussed above can be properly resolved, EBAM costs should reach the 0.001 to 0.01 cent per bit range. This is perhaps several years away from production reality.

Risks

Control Data is actively pursuing, as well as following, EBAM technology. It appears that system commitment to EBAM at present would be risky from both a cost and technological viewpoint. However, by the time a design freeze is required on a project such as that considered in this study, the stability of this technology should be clearly identified.

Operational Implications

EBAMs will perhaps always have very large capacity because the basic module, the tube, must have a large capacity, and the electronics must be shared by multiple tubes to make the system cost-effective; this means large capacity memories. On the other hand, EBAMs offer extremely fast access times for their size although not fast enough to replace RAMs.

However, target development to eliminate the problems associated with partially destructive read and fatigue is needed. Until this happens, system usage of an EBAM must take into account the time required to restore data after some number of reads and to permute data blocks periodically.

LARGE-SCALE INTEGRATION OF SEMICONDUCTOR CIRCUITRY

Large scale integration (LSI) has become a household phenomenon over the past 10 years with the realization of a host of consumer products such as hand calculators, TV games, various auto components, and so on. However, the high level of integration achieved in these areas (upwards of 1000 gates) was only possible because of the low speeds required (and consequent low power). High performance requirements, particularly as one strives for extremes of performance, cannot trade-off these factors, a priori. It is essential, however, to accomplish the logic functions, given the speed imperative, with as low power as possible because of the implicit limit on the scale of integration imposed by power dissipation. In addition, LSI for high performance has no fallout for broad-based consumer applications. This limits the technology as surely as power dissipation or photolithography in a market-driven economy. The resources needed to push the technology ahead are simply needed elsewhere.

Why bother to push for a larger scale of integration then, when contemplating a super processing engine? The Control Data CYBER 76, which is one current leader in the ability to push jobs out the door, gets by with discrete transistors. The STAR-100, which leads for another class of applications, uses SSI (roughly four gates per package). As one looks to the future, however, it is quite clear, both from experiment and analysis, that significant increases in performance and reliability can only be effected with a move to more integration. The reason, in simplistic terms, is that in order to achieve any significant reductions in propagation time through a logic chain, one must stay on the chip for as many steps as possible. A typical critical logic path (for example, a 48-bit add) requires that a signal pass through 11 to 15 stages of logic. In the case of SSI, the signal must leave the IC and traverse a length of inter-chip pc board wiring for each stage. The total delay is the sum of the gate delays plus the board delays, and the latter typically average 1 to 1.5 ns per run in controlled impedance systems. Thus, a practical limit with SSI, even with zero gate delay, is 15 to 20 ns. Of course, architectural alternatives, such as splitting the cycle or pipelining, can be invoked to accommodate this circuit technology problem. For a given architecture, however, LSI will be inherently faster.

It is also more reliable, and this can easily become the dominant consideration for very large ensembles of electronic parts. The reason for this superiority lies in the reduction of chip bonds, solder joints, and circuit real estate as one goes to a higher scale of integration. In addition, because the internal LSI gates do not have to drive external lines, they can run at lower power with reduced heat dissipation and emitter area per unit of logic.

Storage Technology

Storage technology can be categorized as a specific case of LSI technology. Its unique structure allows storage to reflect the maximum component density offered by a respective technology.

To date, this technology has resulted in drastic modifications to traditional architectural as well as component selections. With the advent of dynamic RAMs, CCDs and fast registers, system bandwidth objectives have been dramatically increased.

Table 1-2 summarizes both present and projected storage capacity versus performance. 256x1 devices with access delays of less than 10 ns are currently in the final stages of development. A 4X-density improvement is on the drawing boards but with less than 10 ns access delay requirements, it will take time to effectively produce this product. High-performance RAMs (20 to 60 ns access time) are being competed for by several technologies. Dimensional-processing and power-scaling have produced NMOS static RAMs in the sub-60 ns range while conventional bipolar techniques offer 4K RAM candidates in the 20 to 40 ns range. There is reason to believe that during the next 4 to 6 years, product density will increase to 16 kbits.

With respect to moderate performance technology (100 to 200 ns), 16K dynamic RAMs are currently being offered by several suppliers. Most suppliers see visibility to 65K RAMs in the next few years, but methods of achieving the 4X-bit density are not well defined. Scaling, shrinking, and new photolithographic apparatus are deemed necessary to achieve this density as well as improved materials and circuit design techniques.

TABLE 1-2. STORAGE CAPACITY VERSUS SCHEDULE AND ACCESS DELAY

	Register File	High Performance Storage	Moderate Performance Storage
Device access	5 to 10 ns	20 to 60 ns	100 to 200 ns
1976 to 1978	256 x 1	4K x 1	16K x 1
1980 to 1982	256 x 4	16K x 1	65K x 1

If anything, storage objectives tend to be underestimated while circuit technologies tend to the bullish end of the scale. All agree, however, that 22.6 to 25.8 mm of area still defines producible boundaries to both circuitry and storage. Storage technology continues to more effectively utilize the allocated area. Because of this dynamic density growth, effective system designs must include sufficient addressing capacity while not requiring physical alterations.

Limits of LSI

The following describes the limits of large-scale integration and the factors that set the limits.

In attempting to increase the scale of integration, semiconductor technologists come upon three major barriers· chip area, photolithography, and power dissipation. All of these barriers manifest themselves as economic hurdles; that is, costs escalate rapidly as one attempts to go beyond them. The slope is not infinite, however, so a group to whom parts cost is not compelling can venture well beyond these limits. If the Republic of Japan or the USSR, for example, were to decide that some aspect of the national honor or security were at stake, application of appropriate resources could be brought to bear and the barriers would yield.

Chip Area

Depending on the complexity of the process, there is an upper limit to chip area beyond which the yields drop dramatically. The probability of defects in the semiconductor substrate and epi compounds with that of process flaws due to dust or mishandling in the subsequent steps. Each is area related and the yield typically drops off sharply at areas of 25 to 35 mm^2 for MOS processes (6 or 7 steps) or 12 to 16 mm^2 for bipolar processes (12 to 14 steps). In silicon, there seems to be a skirt on the high end; that is, the probability does not drop to zero for large chips but has some residual value because the defects are not randomly distributed (reference 1-2). But the yield of these very large chips is quite uneconomical and unpredictable. Be that as it may, one reads of examples of good chips which are 10 to 20 mm on a side (for example, in large CCD imaging circuits). A second factor, which mitigates this effect somewhat, is that certain types of defect are ineffective unless they happen to fall in a critical area. The emitter and base regions in bipolar circuits and the gate region in MOS are the most sensitive. Flaws within the bonding pad areas, on the other hand, may be reasonably ignored.

Photolithography

The masking and etching steps comprise the most critical steps in most semiconductor processes. Two types of optical limits come into play here, the resolving power of the various photo/etching systems (how fine a line can be formed) and the field over which this resolution is operative. Both factors affect the sensitivity to misregistration, but for LSI, the second type is probably dominant. For example, a sensible yield of individual high speed transistors can be obtained from a photolithography system which is quite intolerable for LSI. Typical production systems today can readily produce 3-μm line widths over a 4- or 5-mm field.

The diffraction limits are closer to 1.5 μm, but lines of this size only occasionally survive the subsequent processes, primarily because of misregistration. These arts are favored by heavy research and development support by the semiconductor industry, so continual improvement can be expected. But optic systems are up against an increasingly difficult barrier as the diffraction limits are approached.

A number of laboratories are exploring alternative technologies. Reference 1-3 describes the general activity around the world. However, these developments will not become operational for LSI logic for 5 to 10 years.

Power Dissipation

Power dissipation poses a somewhat different hurdle. If means can be provided to drain off the chip heat, reliable operating temperatures can be maintained. These means become cumbersome to some systems, however, if the power per circuit goes much above 5 W. Air-cooled systems are particularly difficult. The circuits must be spaced less densely, which can offset the speed advantage being sought. Even with well-engineered cooling, however, a high-power chip is less tolerant of flaws in the cooling path, such as bonding flaws or air gaps. This translates to a real but somewhat soft barrier to LSI circuit size. It means that, given an existent cooling system, a lower power technology can use a large scale of integration, or if the scale is fixed, a less elegant cooling system can be used.

Reliability of LSI

Component failure rates of LSI technology product offerings are expected to maintain the impressive rates of their predecessors (0.1 to 0.05 failures/million hours). Process improvements which improve reliability will be essentially offset by increased complexity. Overall system reliability, however, will be improved significantly due to the total reduction in subassemblies of critical technology.

Current density will become a very critical parameter due to emphasis on die reduction and component shrinking, coupled with high performance requirements. Since current migration has been correlated to temperature (T^4 actually), emphasis on maintaining low operational temperatures, below 65,C to 70,C, is a must. Random defects associated with shallow emitter implantation/diffusion will be reflected as leakages and slow degrading failures. Breakdown voltages will be lower due to component scaling (in all directions) and impedances will be higher, giving rise to increased concern over static charge.

Just as one begins to believe that all failure modes are known, a new one is uncovered, much like disease control in the medical field. So it will be with semiconductor reliability. As implantation techniques,, E-beam mask generation, oxide isolation, and component scaling become commonplace, new failure mechanisms of concern may arise. It is difficult, if not impossible, to predict what they may be. It is probably safe, at this juncture, to assume that they will be related to temperature and dimensional scaling.

Some Comparative Technologies For The 1980's

The lead time requirement for development of a large processor, such as this program contemplates, does not allow consideration of any circuit technology which is not now in virtually a production status. Prudence demands, therefore, that the plan implement the high performance technology which is being produced now. The pace of the semiconductor technology requires constant vigilance. However, a lot can happen between now and the hardware design freeze. Technologies which particularly bear watching are the efforts toward realizing high performance circuits in N-MOS, GaAs, and Josephson devices (cryogenics). It is also important to predict the rate and amount of improvement likely to come about in the classic bipolar technology being planned, since this is the most likely implementation circuitry.

Field-Effect Transistor (FET) Technologies

Included in the FET category are metal-oxide-silicon (MOS) as well as field-effect devices which do not use oxide to insulate the gate (MESFETs and JFETs). The most robust, well-developed, and likely-to-be-of-interest MOS technology is N-MOS. The most successful memory and microprocessor products are of this family, and a great deal of research is being invested in its continual improvement. One rarely hears of subnanosecond implementations, however, even though some such activity is known to exist.

A fairly recent report (reference 1-4) on some IBM research in this area indicates that demonstration devices were made which exhibited 100-ps switching delays. Extraordinary fabrication methods were needed to realize the 1-μm gate lengths, and precise implantation was needed to produce the 200-ohm substrate material. Generally, these implicit techniques are somewhat far removed from manufacturability. But the extraordinary resourcefulness of the semiconductor industry could well change this pace of development if proper incentives become available. The main problems are achieving the 1-μm line widths over a full chip area (probably at least 2 to 3 mm on a side) and controlling the material properties precisely so that the thresholds do not wander. One-μm line width requires E-beam lithography at present, and suitable step, repeat, and duplication processes have yet to become available (reference 1-3).

A case for a form of MESFET has been well presented (reference 1-5). This realization requires GaAs base material, which has provided a number of fast microwave devices. Figure 1-1 from reference 1-6 illustrates the attraction of both the normally-on (same power 10X speed) and normally-off (same speed at 1 percent power) MESFETs. In GaAs technologies, the scale of integration is limited by the allowable chip area. Defect densities are inherently higher in intermetallic compounds, so the probability of nonzero yield drops off sharply with size. A circuit 1 mm on a side is a signal achievement for GaAs.

Another problem with GaAs is that no planar isolation techniques (such as Isoplanar or ILO) have been developed. Individual gates or circuits reside on mesas, which means that metal runs between them must climb up and down hills. This has proved to be a serious hurdle in silicon technologies and severely limits the degree of integration that is feasible.

On balance, therefore, subnanosecond MESFET's are not expected to become a significant contender for LSI applications in time to intercept the needs of the pending hardware requirements of this program. This technology is most likely to find application in various microwave areas and high-speed counters. These are really equivalent to SSI/MSI logic applications but could provide the economic basis to perhaps solve the technology's LSI problems by the latter 1980's.

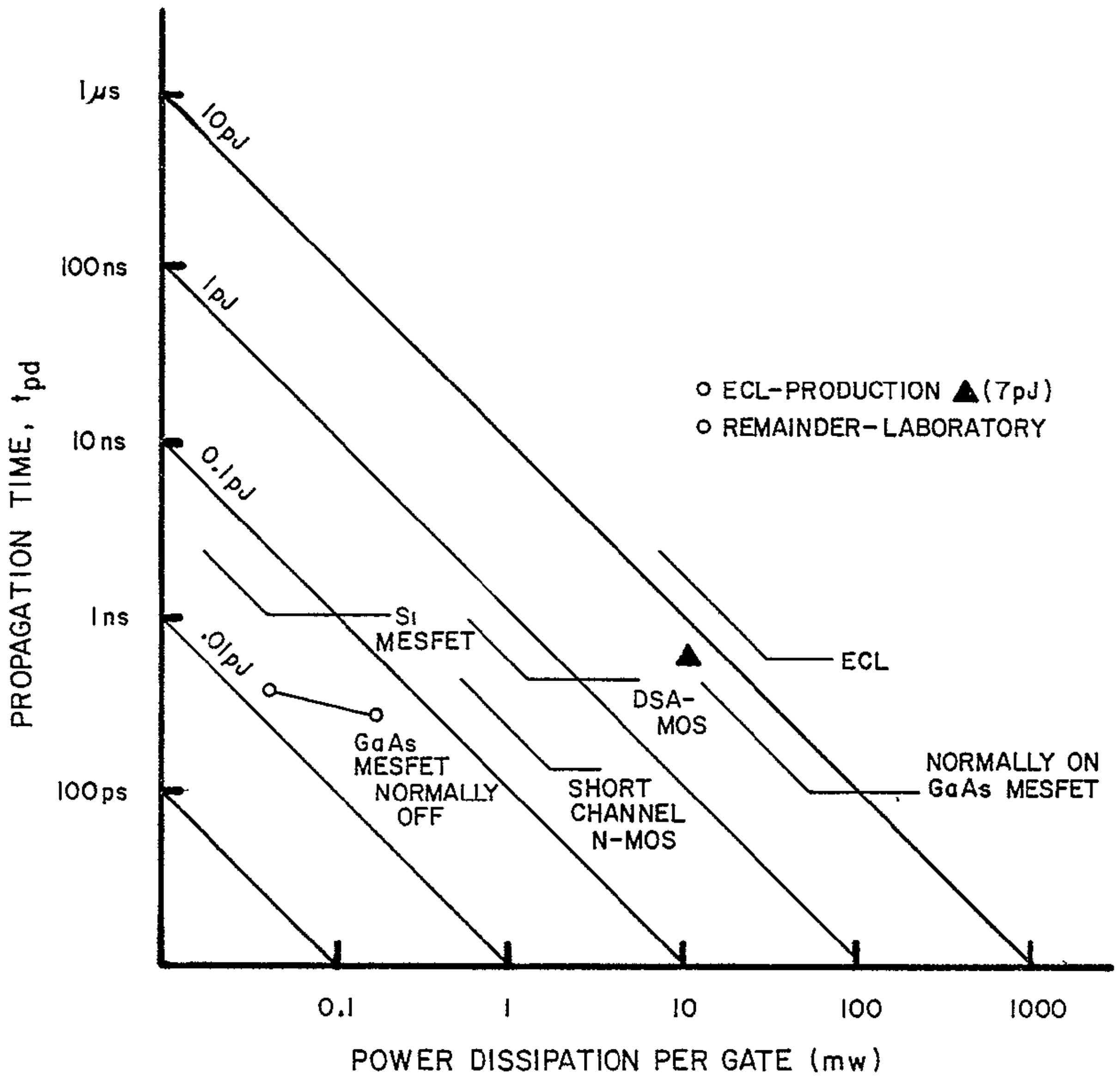


Figure 1-1. Speed-Power Curves for High-Speed Logic Types

Other GaAs Technology

Another form of GaAs technology which has been demonstrated in laboratories (reference 1-7) as extremely fast circuitry (15 to 100 ps) is transferred electron logic devices (TELDs). These devices utilize Gunn diode technology with initial applications being directed toward the microwave industry. TELD logic, unlike Josephson devices, can operate at normal ambient temperature. A relatively good understanding of the properties of GaAs exists, and fabrication of GaAs material, although not easy, is being done regularly by several manufacturers.

However, to date, GaAs technology has progressed slowly as compared to silicon technology. Controlling the doping-density/active-layer-thickness product of the epitaxial film is difficult. A probable device speed factor is 50 to 100 ps, but device dissipation is about 200 to 300 mW. Power consumption, isolation difficulty, and material anomalies presently restrict the usefulness of TELD. One additional factor limiting progress with GaAs is the apparent development funding level. Major semiconductor suppliers are, at best, dabbling in the technology while the majority of R and D allocations remains in silicon-related technologies.

Very little is documented on cost since the technology is in experimental phases. Like any new technology, the risks are high. This is an area of technology which should be monitored rather than pursued. Should breakthroughs emerge sufficiently to make GaAs a viable candidate for computer applications, 3- to 5-ns minor cycles would be conceivable. The probability of this occurring in the next decade is considered unlikely, and in its present state, the technology does not lend itself to computer applications.

For silicon, it is necessary to estimate the improvement possible over the next decade in the conventional production process. Before turning to ECL, however, mention should be made of integrated injection logic (I^2L), a variant on conventional bipolar technologies. References 1-8 and 1-9 give a good account of its versatility. This relatively recent development promises to be one of the most dense bipolar technologies. It is also characterized by a significantly lower speed power product than either TTL or ECL, over its range of operability, which is quite wide. Early hopes for using it for high-speed logic have not materialized, however; the 1 pJ curve was found to tail off toward higher power as one drops below 10-ns gate delays (figure 1-2). To be sure, this barrier may prove to be movable, but the technology looks much more competitive compared with TTL or n-channel MOS, which incidentally represent much larger markets.

For large processors utilizing LSI, the technology first appears to have enormous potential. Architecture configurations such as those considered in this study require the ultimate in performance at the basic logic element level. I^2L application potential will be assessed with these requirements in mind.

Present random logic density for I^2L is 120 to 150 equivalent gates per mm^2. With standard processing adjustments, 150 to 175 gates per mm^2 are achievable. As E-beam mask generation and compatible processing become a reality (1978 to 1980), 200 to 250 gates per mm^2 will become a reality. Die sizes of 20 to 25 mm^2 remain realistic reproducible upper boundaries for this same time period. Provided these estimates are realistic, 1980 I^2L technology would have a respectable die density of 4000 to 6500 equivalent gates. Even if a 4 to 6 flexibility ratio of ECL to I^2L is considered, the 700 to 1600 gate per array density would be commendable (ECL logic, because of the emitter dot, collector dot, and complement output, offers a significant degree of logic flexibility over current I^2L configurations).

ECL, by contrast, has a modest 12 to 15 gates per mm^2 for present gate arrays with visibility to 25 to 40 gates per mm^2 speculated for the same time frame. Arrays of 600 to 1000 gates appear difficult but plausible for the early to mid-1980's.

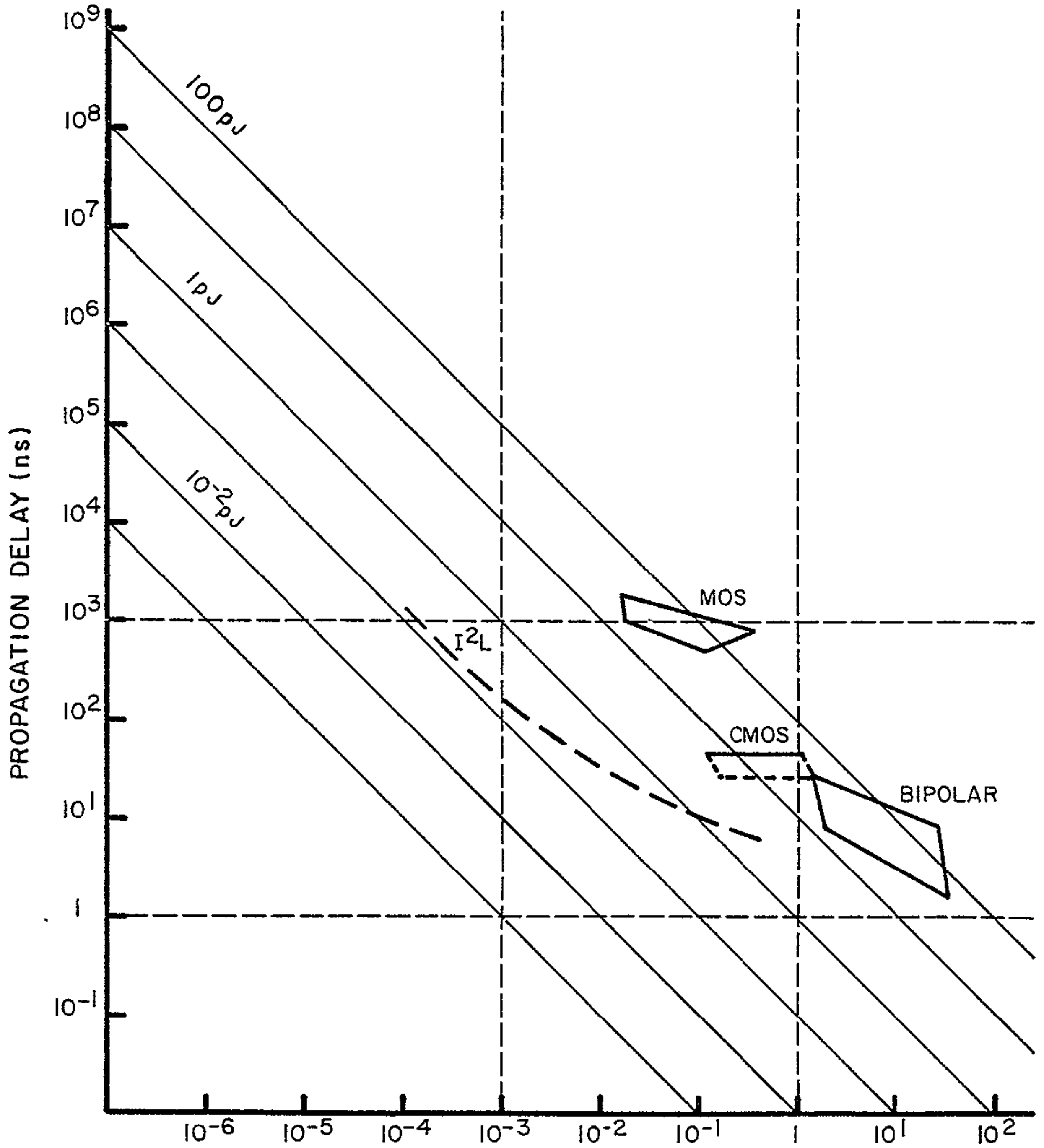


Figure 1-2. I^2L Speed-Power at High Performance

If I^2L component density is not sufficiently impressive, the reported speed-power products certainly are. Figure 1-3 typifies observed and projected speed-power products for I^2L technology. However, the higher performance (6 to 10 ns stage delay) is achieved utilizing more complex processes. The process would require some or all of the following: epitaxial layers, oxide isolation of componentry, two-layer metallization, Schottky barrier technology base-enhanced implantation, arsenic diffused or enhanced emitters (collectors), and 2-μm photolithography. No major supplier has seriously speculated further performance improvements beyond 4 to 6 ns per stage. It is conceivable, with submicron technology becoming a reality, that one might reasonably project I^2L stage delays of 2 ns.

Considering these projected delays, a minor cycle time can now be estimated. For complex architecture such as that proposed, a worst-case path would require 12 to 14 stages of logic and a total logic count of 40K to 50K gates. With partitioning effects considered, 10 to 16 arrays would be required. One could expect a minor cycle of 26 ns to 30 ns.

Applying the same requirements to ECL, 40 to 100 arrays would be required to achieve the logic function. Because of the logic flexibility associated with ECL, 7 to 9 complex stages would be required in a minor cycle. With 600 ps assigned to the more complex functions (400 for basic gates), 7 ns to 12 ns appears achievable.

ECL technology, because of its logic flexibility and performance, presently offers the best opportunity of achieving additional system performance. Current ECL LSI achieves subnanosecond performance. The technology utilizes oxide isolation, several implantation (versus diffusion) steps, very shallow geometries (1 to 1.5 μm epitaxial thickness), self-align masking (nitride-oxide passivation/masking) and 2-μm minimum spacings. This technology, as of 1977, is state-of-the-art and does not yield acceptable quantities because of the lack of product maturity. It is presently estimated that up to an additional year may be required to establish sufficient stability for this technology (ISOPLANAR).

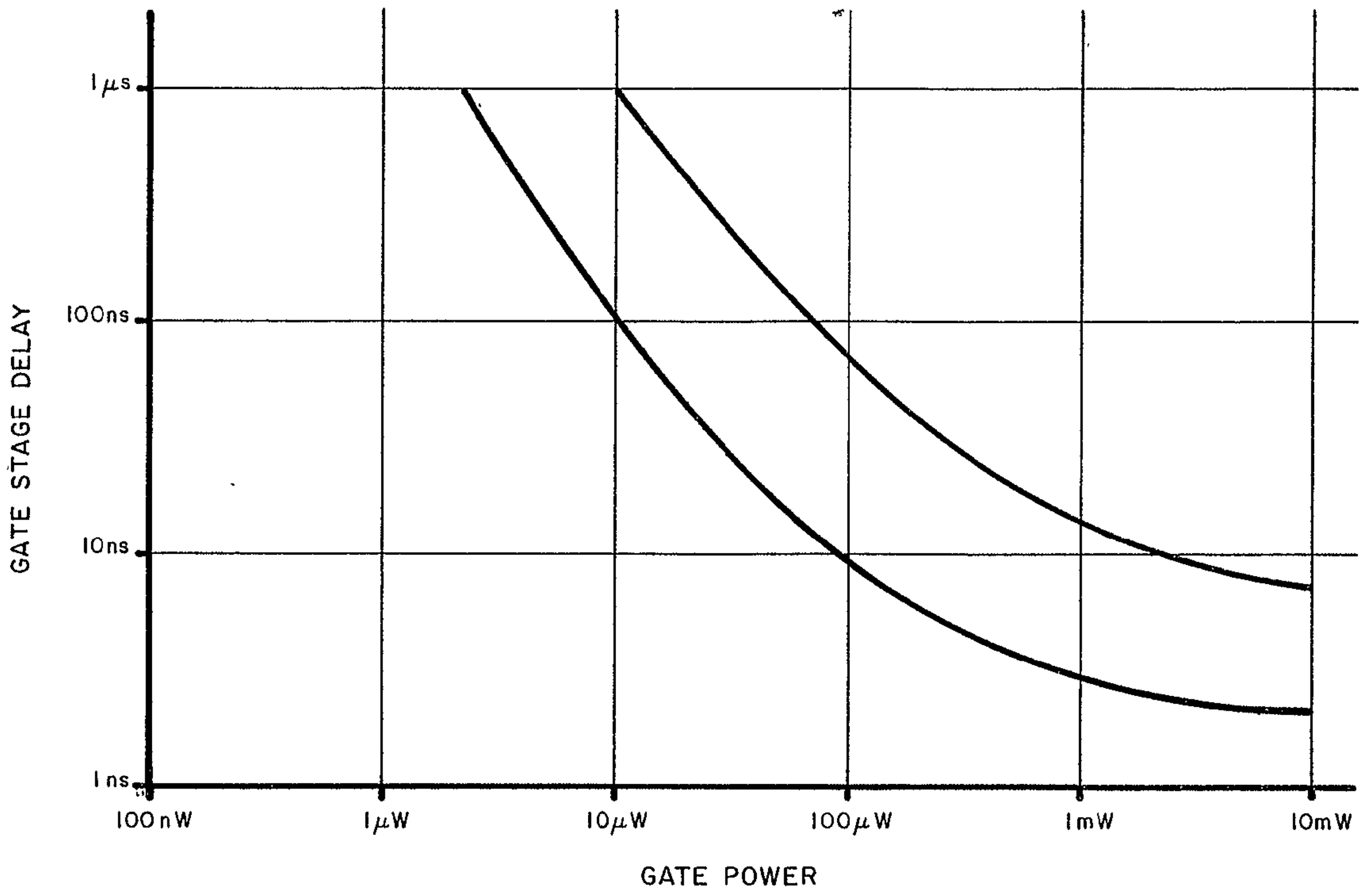


Figure 1-3. Speed-Power for I^2L Technology

Beyond this, and in order to achieve additional performance, options are seen as follows:

- Additional process sophistication such as shallower diffusions or ion implantation steps, thicker surface dielectric oxides for reduced capacitance, and more use of implantation techniques to shape junction boundaries and to improve repeatability.

- E-beam masking may become a reality such that submicron device geometries can be developed in production quantities. Direct E-beam to silicon surface development will require improvements in surface flatness or corrective techniques built into the E-beam generation to make allowances for surface anomalies. Submicron technology, to become a reality, will also require additional sophistication in nitride-oxide and metallization etching techniques. Plasma or dry etch techniques are already a reality for selected process steps.

  E-beam mask generation or elimination of the 10X to 1X and step-and-repeat procedures will become commonplace by 1980. This will serve to enhance repeatability as well as to allow for finer geometry devices if direct wafer masking does not become an effective production tool by this time.

- Increased circuit density will be required to reduce the number of critical off-chip stages required in logic networks. In 1977, 5 mm per edge silicon die appears to stress the state-of-the-art for high performance silicon processing technology. For the past decade, epitaxial surface technology has not progressed in terms of die size as has nonepitaxial surface technology such as NMOS or PMOS. I^2L technology utilizes epitaxial surface techniques, but the thicknesses and processing steps are less critical than those required for subnanosecond performance. As increased processing techniques are imposed on devices, as has been the case for performance-driven technology, the die size restriction for volume production remains. In 1980, an objective size of 5 mm per edge appears to be reasonable, certainly no larger than 6.5 mm per edge.

- Performance may also be improved by reducing circuit RC time constraints. A few of the potential enhancements suggested above address reducing capacitance. Reducing the resistance usually implies increasing the power dissipation. Every effort will be made to improve performance without increasing power dissipation. In fact, a reduction in power dissipation per gate will receive every priority. Present arrays display a figure of merit of some 7 pJ at the switch level. Objectives for the next effort in the 400 ps X 5 mW or 2 pJ range appear plausible.

- Scaling of technology is also a possibility. Scaling refers to the simultaneous reduction of all parameters in an equal ratio, such as concentrations of diffusion/implantation, diffusion/implantation depths, surface dimensions while maintaining aspect ratios, oxide/nitride thicknesses, power dissipation, voltage supplies, and so on. Immediately a host of technological traps and horrors may arise, but the concept must be given appropriate attention. This must not be confused with mask shrinking, which only applies to surface dimension and has been used successfully on several RAM products by more than one manufacturer.

While its inherent density and low power make I^2L highly attractive for LSI, the apparent speed barrier all but eliminates it for this application. For the extremely high performance systems, however, it falls short of ECL technology by some 2 to 4 times. Most technological advancements focused on improving I^2L performance can also be applied to ECL.

Because of the ostensibly wide applicability to consumer products and other high volume uses, extensive resources may be invested in I^2L development. In this situation, subnanosecond capability could fall out, although the barriers at present seem insurmountable.

Prognosis

Improvements to performance will require advances in processing technology as well as possible minor modifications to traditional circuit design concepts. Some form of emitter coupled logic (ECL) appears the most plausible candidate at the present. Silicon material appears proper for at least one more generation, and lower speed-power products must be a primary concern. Additional usage of ion implantation will yield possible improvements in transistor junction profiles resulting in higher gain-bandwidth devices. E-beam mask making will, hopefully, allow for scaling of present state-of-the-art oxide-isolated ECL devices. Five-mm per edge technology will allow 300 to 500 ECL gate density or an equivalent of 500 to 1000 gate functions per die. Performance switch objectives will be in the 250 to 400 ps per stage. PC board densities with increased packaging density and lower speed-power product silicon technology should result in 4 to 5 fold improvements over present ECL LSI technology or 0.8 to 1.3 million gates per square meter. Prototype modules reflecting this technology appear reasonable for 1980 to 1981. The prototype developed for this time frame would display practical technologies that utilize manufacturable processes and techniques for the 1980's.

JOSEPHSON DEVICES

General Description

Josephson tunneling in superconductors was first considered in 1962 (reference 1-10), although the suitability of the resulting devices for computer usage was not recognized until later (reference 1-11). A Josephson junction in its elementary form consists of two superconductive film strips which overlap and are separated by a thin insulator. The substrate material is commonly glass or silicon, and typical superconductive films are lead alloys. Liquid helium is often used as the cryogenic agent, such that normal operating temperature is 4.2°K. Current can pass from one conductor to the other, tunneling through the insulator. Such conduction will be superconductive (without voltage drop) until the current reaches some maximum level I_m. The value of I_m can be reduced by application of a magnetic field in the junction insulator. If I_m is reduced below the operating level I_g, the device switches to its voltage state (reference 1-12) and assumes operation on a voltage state load line.

The device cannot be switched back to a superconducting state with a magnetic field, but rather the device current must be reduced to I_{min}, at which point superconductivity is resumed. The nominal voltage present on a junction is 3 mV, and switching commonly occurs in a few tens of picoseconds (reference 1-13). The switching of the junction from superconductive to voltage state, and the current steering that results, is reminiscent of the operation of the cryotron, which was the subject of considerable earlier work (reference 1-14).

A three-device cell has been operated with current transfer times of about 600 ps, and a resulting cycle of about 1 ns. Cell area tends to be larger than that for a corresponding semiconductor circuit but can be made comparable if high-current densities are used (reference 1-15), however, detrimental resonant effects can result from such high-current densities (reference 1-16).

Relative Merits

Several logic circuits have been demonstrated using the basic Josephson junction, including a four-bit multiplier (reference 1-17) and a single-bit adder (reference 1-18). The adder had a 500-ps propagation delay and a 10-ns reset time. Herrell (reference 1-19) has described gate geometries suitable for logic use and measured speed-power products of 5 fJ and propagation delays of 170 ps.

Several memory devices have been described using a Josephson junction as the basic element. The most ambitious has been a single junction DRO memory cell which stores a single quanta of magnetic flux in the junction area (references 1-20 and 1-21). A flux shuttle shift register memory has been proposed (reference 1-22) in which flux quanta in Josephson junctions would be transferred from device to device. Calculations indicate that transfer times of 10 ps and power dissipation of 10^{-18}W may be possible, although prototype devices have apparently not been built.

Estimates have been made for memory systems made with memory devices as described (reference 1-23). Four-kbit circuits are estimated to take chips of roughly 6 mm on a side and would yield memory cycle times on the order of 2 ns, although no such LSI circuits have been fabricated.

Limitations

The large-scale integration of Josephson devices will be necessary before computer components can be realized with these elements. This results from the fact that immersion in, or close contact with, a liquid helium bath will be needed for all superconductive elements, so that the composite hardware volume must be restricted to that of cryogenic containers.

No large-scale Josephson circuits have been reported to date; the four-bit multiplier is the most complicated component reported so far. After full wafers of more complicated chips are fabricated successfully, yields must be determined and packaging schemes must be devised.

A considerable number of technical references of the various aspects of Josephson junctions have been compiled as a part of the RADL assessment of these devices for potential use in advanced computers to be built in the 1980 time frame. Approximately 80 percent of these articles have been authored by personnel of International Business Machines, with the remainder being written by a few universities and by Bell Laboratories. No references have been found by an industry source which can be considered a potential vendor of such components to Control Data Corporation. This evidence, which is circumstantial, is a strong indicator that work on Josephson devices has not progressed to the stage where a product can be contemplated. No commercial vendor for Josephson devices is presently known, and it seems unlikely that one will develop before the mid-1980's.

The problems with fabricating Josephson junction devices include film adherence and corrosion, particularly for lead alloys (reference 1-24). Temperature cycling between 275 and 4.2°K produces stress in thin films which can be destructive (reference 1-25). The insulator thickness determines the threshold current, and it may be necessary to control oxide thickness to within a fraction of an angstrom (reference 1-26).

Further, while the switching speed of the junction logic gates are impressive, much of this advantage is lost by the need for reset procedures. Junction memory may be the most promising application.

Considerable engineering must attend the packaging and checkout of large cryogenic computer modules, since the bringing out of test and signal leads through Dewar walls and the complication of cryogenic immersion will allow even less convenient repair than in the case of present day LSI. Equipment maintenance will be similarly affected.

Cost

At this point, quantitative cost information does not exist. It certainly seems reasonable to expect this technology to produce a relatively costly implementation, at least in its early life. Even if the LSI devices themselves compare with other LSI technologies, the cost of cryogenic packaging and associated checkout and maintenance procedures would inevitably affect costs adversely.

Prognosis

It is reasonable to expect that an extensive period of development must follow early laboratory work before components can be offered for use in systems. A further period of system design will be needed with any cryogenic computer device because no large computing systems have been constructed with devices which require low-temperature environments, and the nature of electronic design and checkout can be expected to be complicated severely by the prospect of such a radical departure in component packaging.

In summary, one must conclude that Josephson devices will not be a realistic alternative for use in computing systems which are to become operable in the 1977 to 1985 time frame.

ADVANCED ROTATING MASS STORAGE

General Description

Over the past 12 to 15 years, disks have provided the bulk of on-line mass storage. Disk technology has consistently provided the best combination of capacity, performance, and cost per bit. Even though the future will bring some new technologies to the market, disks will remain a very important member of computer storage hierarchy for some time. In order to understand the future of disks, one should examine the historical trends and the technologies which led to the current position. From this, some judgement can be applied as to what might be expected in the future.

Capacity

Probably the most important parameter by which disks are measured is that of capacity. What has happened to disk drives over the past decade is indicative of the emphasis placed on capacity. In the early 1960's, Control Data introduced the original 852 with a capacity of about 2 megabytes. This was succeeded in the mid-1960s by the 854 which increased the capacity to 7 megabytes. The late 1960s saw the introduction of the 841 disk which had a capacity of about 30 megabytes. In the early 1970s, the first 844 with 100 megabytes appeared, followed shortly thereafter by the 200-megabyte version of the 844.

Disk drive capacity is determined by two parameters, the areal recording density, as measured by bits per unit of area, and the total surface area available. In addition, two parameters affect the areal recording density. The first parameter is the density of bits along the track, or bit density, and the second parameter deals with the spacing between tracks, or track density. These two parameters are limited at any point in time by the technology available to design and produce read/write heads, smooth recording surfaces, accurate head positioners, and precise mechanical structures.

Magnetic recording technology has come a long way since the introduction of the first disk drives. Six-fold and four-fold improvements in bit density and track density, respectively, have had a multiplying effect on areal density. There does not appear to be any theoretical reason why a trend of increasing recording density on disk cannot continue. It does appear, however, that the need to develop some new technology must be met in order to continue to make significant advances.

Simply stated, increases in recording density are achieved by reducing and precisely maintaining the distance between the head and the disk surface. Up to this point in time, heads have been designed to fly at lower and lower altitudes, with computer simulation programs allowing a proposed design to be evaluated via the computer prior to building and testing. The flying height of heads has undergone a reduction of four or five to one in the past decade.

Experimental work indicates that recording densities may be reaching the limits of technology represented by the combination of the flying head and oxide disks. It may be necessary to develop technology based on a nonflying head combined with a plated disk to double the present densities. While it is impossible to forecast technology breakthroughs, it seems reasonable to forecast areal densities which would raise disk pack capacity to over one billion bytes by the early 1980's.

Media

As increases in recording densities have continued, the mechanical tolerances required for pack interchangeability have become increasingly tight, so much so that the fixed-media drive has returned to the marketplace. An examination of pack usage over the past 10 years has shown that the numbers of packs per drive has dropped from about six to slightly over one, indicating that the user is doing far less pack swapping today. Economy and reliability, in addition to improved system utilization, are bringing fixed-media disks back.

On the other hand, it is recognized that there are certain situations in which pack interchangeability is a very desirable feature. Consequently, development work aimed at providing a low-cost drive offering pack interchangeability continues. The key to being successful here is the reduction of the fixed mechanical and support costs associated with the typical stand-alone drive. It would be misleading to say that a removable media drive can achieve the same cost per bit as a fixed-media drive.

Costs

When one examines the costs related to building a disk drive, it becomes apparent that there is a high initial cost involved in the mechanical structure, the power supplies, the motor, and the read/write electronics. For the drives used on a CDC CYBER class machine, these costs are fixed and represent a significant portion of the total cost. It is interesting to point out that drives separated by about 10 years and more than an order of magnitude in capacity were priced the same, if one removes the effects of inflation.

This high fixed cost is one of the reasons why disks will lose out to the emerging technologies in the small capacity range. By the mid 1980's, it is likely that disks with capacities below 25 megabytes will not be offered even for minicomputers. However, in capacities above that, cost per bit reasons will dictate the continued use of disk technology. At that point, single drives with capacities above a billion bytes will be available at a cost per bit of 10 percent to 25 percent of what users are paying today.

Access Time

Access time is affected primarily by two factors, both of which will probably show little improvement in the future. The first of these is the positioning of the heads in a movable head device; this part of access time does not exist in a fixedhead device. Early drives used electric positioners and operated in the 100-ms range. These were followed by hydraulic positioners with speeds around 75 ms. The present technology, and the one expected for the foreseeable future, uses a voice coil positioner operating in the 30-ms range.

Effective positioner speed can be improved by use of multiple heads per arm, thereby reducing the length of the stroke. An additional benefit occurring from this approach is that it increases the size of cylinder, thereby improving performance in the case of sequential read or write operations. Even with the projected improvements, a limit of 15 to 20 ms for average positioning time appears to be the limit for disks in the future.

A second parameter affecting access time is that of rotational latency delay. Presently, disks operate at 3600 rpm, yielding an average delay of 8.3 ms with little reason to expect any improvement in that area, primarily because of mechanical limitations.

Prognosis

Disk technology is relatively mature and stable; it is, and will continue to be, a reliable, cost-effective mass storage device for several years. Other emerging technologies, such as CCD and bubble memories, challenge some applications of disks, but their biggest threat seems to be in the cost/access gap between rotating storage and RAM. Disk storage development may be past the knee of an improvement curve, but advances should still be possible. For some time, perhaps well into the 1980's, disks should remain the workhorse for on-line bulk storage.

HIGH SPEED DATA CHANNELS

General Description

In considering the flow of information through a computing engine, it is useful to bring in the concept of machine balance. The rate at which the arithmetic or computational part of the machine can generate results must be balanced with the rate at which the memory can supply and absorb operands. The relevant problem data (raw data, partial results, answers, parameters, and so on) of large problems, such as this project is concerned with, typically cannot be contained in the directly addressable memory for cost or architectural reasons. A good design must therefore further balance the bandwidth of the backing storage with that of the main storage. This includes the capacity and speed of the channel connecting the two levels of storage, since the backing memory can be expected to be physically separate from the main machine structure.

Balance is a somewhat complex term, since it implies smooth, uninterrupted flow between all three servers, across a breadth of problem types. Of particular interest in this part of the design is the technology and methods of effecting the interchassis information transfer. For that reason, it was considered worthwhile to include a section in this report addressing these technologies.

The three basic types of computer system architecture used in today's machines (reference 1-27) are:

- Channel to central processor (for example, IBM 360/370)
- Channel to memory controller (for example, XDS Sigma)
- Channel to main memory (for example, CDC CYBER)

The distinguishing characteristic is the way in which the I/O information is fed to the main memory. Many variants within these three main branches can be cited, but (at least to one classifier) the essential differences lie in the way the machine interfaces with the outside world. Each approach has its advantages and advocates, but in any case, the problem of designing suitable channel hardware to attach a remote store must be solved.

The classic approach (a multiple conductor, ready-resume channel) runs into two typical barriers. The ready-resume (typically on a word or byte basis) procedure imposes a substantial overhead on the data traffic. To sustain a memory port which is capable of moving 128 bits every 10 to 20 ns, one must provide roughly twice that bandwidth for a simplex channel to a storage device 3 m away, if a response to each word is required. Clearly, one must think in terms of moving blocks of data if a balance is to be achieved. As the backing storage requirement for capacity grows, another problem confronts the classic channel, which has by now become fixed in length and perhaps even tuned to the rates needed. That is, the ability to connect the number of storage units is limited by the space needed to attach or route the cables and perhaps even the floor space needed to contain the units is larger than that swept out by the cable radius. This problem increases with increasing data rate because of the greater need to keep the cable lengths down.

Furthermore, a large special-purpose computational facility will almost certainly have an associated general-purpose station (computer system) to provide user interface and general processing utilities. The backing storage provides the primary information exchange facilities between the two processors, and this makes its connectability even more important.

In consideration of these looming problems, a number of firms have research efforts addressing ways to improve this technology. At Control Data, the effort is System Communications. A common theme that runs through most of this research is to explore the use of single-wire (or even wireless) interconnections. The communications industry has spent, and continues to spend, an enormous amount on research which supports these technologies. The strategy is to utilize this foundation and direct some offshoots into data processing systems needs. The change to single-wire, serial transmission has profound system implications. It is a completely disjointed concept to the established interprocessor communication methodologies. Fortunately, however, most systems designers have had to become conversant with the use of common carrier facilities (networking) over the past 10 years. So at least in the case of distant couplings, means have been developed to utilize serial links. The use of such techniques for closer coupling, for example, between a process in central memory and "his" disk space, conjures up visions of severe performance degradation because of the serializing-deserializing and buffering overhead. Indeed, some such degradation may occur, but perhaps some other source of degradation, which the classic design was or chose to be ignorant of, is avoided. These resolution questions are the heart of high-bandwidth channel research.

A number of experiments are underway to probe these system's implications, but for the purposes of this report, the discussion will be confined to the transmission technologies themselves. These can be roughly divided into coax, optical fibers (light pipe), and wireless methods (light pipes being considered a form of wire). Wireless facilities usually mean microwave or optical carriers because of the extensive bandwidth requirement.

Coax Transmission

The largest body of technology (and research) probably attends the coax option. This results from the extensive capital being invested in the cable TV (CATV) industry which in turn was an outgrowth of the common carrier establishment. This background provides a wide variety of cable types, in-line repeaters, power supplies, taps, multiplexing, and interface modules. The basic system bandwidth is 300 MHz divided into 30 MHz upstream and 270 MHz downstream. Most one-way facilities have 300-MHz bandwidth with as much EMI and RFI immunity and environmental protection as one wishes to pay for. Tables 1-3 through 1-8 summarize the components for wideband transmission.†

†Detailed information for tables 1-3 through 1-8 is documented (reference 1-28).

TABLE 1-3. WIDEBAND TRANSMISSION COMPONENTS: COAX CABLES

	Shielding		Outside Dimensions	Loss per km	Cost per km
	Type†	dB	cm	dB at 150 MHz	$
RG-174	SB	50	0.254	300 dB	164
ADF59PV	FB	80-100	0.635	75	98
59FD95P	DB	110+	0.635	75	330
ADF6PV	FB	80-100	0.762	59	120
RG-11	SB	50	1.046	33	
11FD95P	DB	110+	1.046	33	590
PI75412J	SS	200+	1.046	33	400
PI75500J	SS	200+	1.270	28	490
PI75750J	SS	200+	1.905	20	980
PI751000J	SS	200+	2.540	11	1570
300 MHz Fiber Optics	-	200+	1.270	10	2000

†Shielding types

SB	Single copper braid, 95-percent coverage
FB	Aluminum, polyester foil with 60-percent braid
DB	Double copper braid
SS	Solid aluminum sheath

TABLE 1-4. WIDEBAND TRANSMISSION COMPONENTS· DIRECTIONAL TAPS

Unit	Cost	Bandwidth	Insertion Loss	VSWR In	VSWR Out	Isolation
470 35 3- dB		50 to 225 MHz		$\leq$1.22:1		$\geq$25 dB (T-O)
Outdoor	11.00	Plug in tap valves.				$\geq$20 dB (T-T)
5510 11,14,17 22,26,30		50 to 225 MHz	3.5,1.8,0.9,0.4, 0.3,0.2	1:22:1 (thru)		$\geq$25 dB (T-O)
Outdoor	6.50	Plug in: 2,4 output plate		1:25:1 (tap)		$\geq$20 dB (T-T)
566 Active		$\pm$0.25 dB	$\leq$0.75 dB (thru)	$\leq$1.22:1 (thru)		$\geq$40 dB (T-O)
Outdoor	27.54	$\leq$33 dBmv for 12 channels input		$\leq$1. 3.1 (tap)		$\geq$18 dB (T-T)
DMT-2 7,10,14, 18,22,26,30,34		10 to 300 MHz	3,1.5,0.7,0.5	1.22:1	$\geq$10 MHz	$\geq$20 dB (T-O) $\geq$20 MHz
Outdoor						$\geq$10 dB (T-O) 10 MHz
T-0600 Splice	7.50 +	5 to 300 MHz $\pm$0.5 db	Tap Loss	$\leq$1.22:1		$\geq$25 dB (T-T)
Outdoor	Conn					D$\geq$15 dB
T-16 One Out	9.95 +		Tap Loss -10 1.25 -14 0.5			
Outdoor	Conn					

TABLE 1-4. WIDEBAND TRANSMISSION COMPONENTS: DIRECTIONAL TAPS (Contd)

Unit	Cost	Bandwidth	Insertion Loss	VSWR		Isolation
				In	Out	
T-26 2 Out Outdoor	10.50 + Conn		-10 2.0 -14 1.25 -18 0.5			
T-46 4 Out Outdoor	11.00 + Conn		-10 3.5 -14 1.25 -18 0.75			

NOTE: Directivity (D) equals isolation minus tap loss.

TABLE 1-5. WIDEBAND TRANSMISSION COMPONENTS: POWER SPLITTERS

Unit	Cost	Bandwidth	Insertion Loss	VSWR		Isolation
				In	Out	
932U-(2W)	3.22	5 to 300 MHZ $\pm$0.25 dB	$\leq$4 dB	1.22.1(10-300)	1.38.1	30 dB
Outdoor				1.28:1(5-10)	1.5:1	28 dB
933U-(4W)	5.94	5 to 300 MHz $\pm$0.25 dB	7 dB	1.25:1(10-300)	1.25.1	30 dB
Outdoor				1.28:1(5-10)	1.34:1	
936U-(2W)	2.32	5 to 300 MHz $\pm$0.25 dB	$\leq$4 dB	1.22 1(10-300)	1.38·1	30 dB
Indoor				1.28:1(5-10)	1.5:1	28 dB
556U-(2W)	2.20	5 to 300 MHz	4 dB	1.22.1(10-300)	1.38:1	30 dB
Indoor				1.28:1(5-10)	1.5:1	29 dB
533-(3W)	3.18	50 to 220 MHz	$\leq$3.5 dB; $\leq$6.5 dB	$\leq$1.22:1	$\leq$1.22:1	$\geq$20 dB
Indoor						
937U-(4W)	5.25	5 to 300 MHz $\pm$0.25 dB	7 dB	1.25:1(10-300)	1.25:1	30 dB
Indoor				1.28.1(5-10)	1.35:1	
2SBV-P(2W)	14.40	10 to 216 MHz	$\leq$3.5 dB			$\geq$20 dB
Outdoor						

TABLE 1-5. WIDEBAND TRANSMISSION COMPONENTS: POWER SPLITTERS (Contd)

Unit	Cost	Bandwidth	Insertion Loss	VSWR In	VSWR Out	Isolation
MS-2V (2W) Indoor	5.20	10 to 216 MHz	3.5 dB			26 dB
MS-4V (4W) Indoor	8.30	12 to 216 MHz	6.9 dB			26 dB

57

TABLE 1-6. WIDEBAND TRANSMISSION COMPONENTS. 12-CHANNEL AMPLIFIERS
(50 to 100 MHz and 170 to 220 MHz)

Unit	Cost	Bandwidth	VSWR In	VSWR Out	Max. Gain Opt. Gain	Distortion Character	Noise	Power Supply
VICOA 425 Mainline Amp.	319.20	50 to 225 MHz ±0.25 dB	1.33·1	(14 dB) 1.5:1	26 dB 22 dB	+35 dBmv Output -87 dB Cross Mod	≤10 dB	19 to 30V 0.6A
VICOA 433 Bridging Amp.	329.00	50 to 225 MHz ±0.5 dB	1.22.1	1.38:1	34 dB 31 dB	+47 dBmv Output -63 dB Cross Mod	------	19 to 30V 0.8A
VICOA 461M Line Extender	95.40	50 to 225 MHz ±0.5 dB	(16 dB) 1.38:1	1.38:1	24 dB as required	+44 dBmv Output -57 dB Cross Mod	------	19 to 30V 0.3A
B-T 1249 Bridger	295.00	50 to 220 MHz ±0.33 dB	(20 dB) 1.2.1	(16 dB) 1.38.1		+43 dBmv Output -57 dB Cross Mod		18 to 30V 0.65A
B-T 1252 Mainline Amp. (AGC)	386.00	50 to 220 MHz ±0.33 dB	(14 dB) 1.5.1	(16 dB) 1.38.1	29 dB 27 dB	+54 dBmv Output	9 dB	18 to 30V 0.4A
B-T 1242 Transporter	305.00	50 to 220 MHz ±0.25 dB	(20 dB) 1.2:1	(16 dB) 1.38·1	24 dB 22 dB	+51 dBmv Output -57 dB Cross Mod	≤13 dB	18 to 30V 0.55A
B-T 1244 Line Extender	79.00	50 to 220 MHz ±0.75 dB	(15 dB) 1.4:1	(14 dB) 1.5:1	18 dB	+46 dBmv Output	≤13.5 dB	18 to 30V 0.25A

58A

TABLE 1-7. WIDEBAND TRANSMISSION COMPONENTS: 21-CHANNEL AMPLIFIERS (50 to 250 MHz) AND 28-CHANNEL AMPLIFIERS (50 to 300 MHz UNIDIRECTIONAL)

21-CHANNEL AMPLIFIERS (50 to 250 MHz)

Unit	Cost	Bandwidth	VSWR In	VSWR Out	Max. Gain	Opt. Gain	Distortion Character	Noise	Power Supply
VICOA 210 Mainline Amp.	377.00	50 to 225 MHz ±0.25 dB	1.28:1	1.5.1	26 dB	22 dB	+32 dBmv Output Cross Mod=-90 dB 2nd Order=-80 dB	≤10 dB	19 to 30V 0.8A
VICOA 214 Bridging Amp.	377.00	50 to 225 MHz ±0.5 dB	1.22.1	1.38:1	34 dB	31 dB	+50 dBmv Output Cross Mod=-57 dB		19 to 30V 0.8A
VICOA 216 Line Extender	147.55	50 to 225 MHz ±0.5 dB	1.38:1	1.38:1	24 dB	as required	+45 dBmv Output Cross Mod -57		19 to 30V 0.32A

TABLE 1-7. WIDEBAND TRANSMISSION COMPONENTS· 21-CHANNEL AMPLIFIERS (50 to 250 MHz) AND 28-CHANNEL AMPLIFIERS (50 to 300 MHz UNIDIRECTIONAL) (Contd)

28-CHANNEL AMPLIFIERS (50-300 MHz UNIDIRECTIONAL)

Unit	Cost	Bandwidth	VSWR		Max. Gain	Opt. Gain	Distortion Character	Noise	Power Supply
			In	Out					
VICOA 251 Mainline Amp.	438.10	50 to 280 MHz ±0.25 dB	1.38:1	1.5:1	25 dB	22 dB	+32 dBmv Output Cross Mod=-92 dB	≤11 dB	19 to 30V 1.1A
VICOA 256 Bridging Amp.	440.05	50 to 280 MHz ±0.5 dB	1.38:1	1.5·1	34 dB	24 dB	+44 dBmv Output Cross Mod=-64 dB		19 to 30V 0.8A
VICOA 260 Line Extender	159.25	50 to 280 MHz ±0.5 dB	1.5:1	1.5:1	24 dB	as required	+40 dBmv Output Cross Mod=-67 dB		19 to 30V 0.8A

TABLE 1-8. WIDEBAND TRANSMISSION COMPONENTS: BIDIRECTIONAL AMPLIFIERS

Unit	Cost	Bandwidth	VSWR		Max. Gain	Opt. Gain	Distortion Character	Noise	Power Supply
			In	Out					
VIK-2-300-10 Mainline Amp.		F=50 to 300 MHz ±0.25 dB	<16 dB	<16 dB	29 dB/22 dB		49.5 dBmv Output Cross Mod -57 dB	≤11 dB 276	18 to 60V 1.2A to 0.4A
		R=5 to 24 MHz ±0.25 dB	<18 dB	<18 dB	18 dB/7 dB		32 dBmv Output Cross Mod -92 dB	≤12 dB 24	
VIK-2-302-10 Bridger Amp.		F=50 to 300 MHz ±0.5 dB	<16 dB	<16 dB	34 dB/24 dB		44 dBmv Output Cross Mod -64 dB	≤11 dB 276	18 to 60V
		R=5 to 24 MHz ±0.5 dB	<18 dB	<18 dB				≤12 dB 24	1.2A to 0.4A
VIK-2-303-10 Dist. Amp.		F=50 to 300 MHz ±0.5 dB	<16 dB	<16 dB	44 dB/34 dB		+44 dBmv Output Cross Mod -64 dB	≤11 dB 276	18 to 60V
		R=5 to 24 MHz ±0.5 dB	<18 dB	<18 dB	10 dB/7 dB			≤12 dB 24	1.2A to 0.4A

This bandwidth will easily carry 50-Mbit data and probably 100-Mbit data or 200-Mbit data if the increased error rate can be tolerated. Several firms have run test system data over installed cable TV systems. Probably the best known are the experiments with Manhattan Cable and a group of New York banks (reference 1-29). RADL has run a number of similar tests using the Jonathan, Minnesota and Bloomington, Minnesota cable TV systems. Data trunks using both single channels (6 MHz per TV channel) and multiple channels (24 MHz) have been implemented (references 1-30 and 1-31). Generally, the components were found to operate quite satisfactorily and to provide a considerable opportunity for improved data transfer techniques.

Optical Fibers Transmission

The other single-wire technology which is advancing rapidly is optical fibers (light pipes). At least one excellent account of the status of this technology has been published (reference 1-32). In summary, the components needed for effective use of light pipes for broadband data transmission are the fibers themselves, the transmitters and receivers, and suitable in-line components (if certain networks are to be realized).

Until about 1976, fibers with usable attenuation and bandwidths for broadband data (50-Mbit) transmission were not commercially available. The available fibers could only reliably support about 10 Mbit. The new graded-index technology provides fibers with about 300-MHz bandwidth, which is sufficient for 50-Mbit and perhaps even 100-Mbit transmission. This bandwidth (300-MHz) may prove to be a plateau in the fiber development, since it is sufficient to serve the cable TV requirements, which is expected to be a primary market. However, institutions such as Bell Labs will continue to push the research to increase bandwidth for the needs of telephony.

Transmitters and receivers are available in various capabilities but are as yet somewhat costly. The costs can be expected to follow the normal semiconductor learning curve improvement, once a sufficient market develops. Fifty-Mbit transmission currently requires a lasing diode, which, as a class, has lifetime problems. LEDs may be used, but their spectral spread results in pulse spreading of about 4 ns/km. Therefore, a 1-km link, 50-Mbit transmission (20-ns period) experiences a substantial jitter in the pulse edges. This translates to an inferior error rate, which may still be tolerable, especially for shorter lengths. Receivers are usually PIN diodes or avalanche photo-diodes (APD) and as yet are somewhat expensive. Sensitivity is adequate although the noise factor of subsequent stages can still be a determining factor because of the nature of the diodes.

Several research groups are attempting to develop some of the directional couplers, taps, and switches which coax systems have come to require. Usually this work is called integrated optics and involves creation of planar optical wave guides which couple to the transmission fibers. Typically, fine line parts are etched to capture the evanescent wave of the optic power moving through the guide. Since this is a linear process, an optical wave can be coupled in also, thereby resulting in a directional coupler. This work is high technology to date and will probably not produce any commercial devices for 5 to 10 years. However, it does bear watching as success could be quite valuable.

The other common in-line component needed is a suitable repeater. Optical repeaters have some special problems such as external powering, wider dynamic range capability, and the ability to cope with dc codes. These do not require any breakthroughs to solve but translate to higher costs. Offsetting this apparent disadvantage is the superior loss characteristics optical fibers, so fewer repeaters are required.

Wireless Transmission

These technologies (primarily microwave and optical links) become of interest when the data channel must traverse spans for which wire links are impractical. An example would be from one site to another, where the right-of-way to string a cable cannot be obtained. These are infrequent cases but are worth mentioning as possibilities. Wireless link is generally only feasible if the two linked systems are capable of communicating over serial transmission facilities.

Again the cable TV and MDS (microwave distribution system) industries are the prime users of these links. Equipment for a wide range of bandwidth power and other capabilities are available, including optical links of various types. RADL has used both on an experimental basis with good success. The optical links are as yet somewhat gimmicky, in that their primary advantage is requiring no FCC license to operate. On the other hand, microwave common carriers exist in most states and can provide individual channels, along prescribed networks, or in a region-wide broadcast mode as in the case of MDS. However, this technology is an end-case for the purposes of this study and is only noted in passing.

ADVANCED TEST EQUIPMENT

General Description

Approximately a quarter century of growth in the computer industry has led to many and often complex changes in electronic components, digital devices, methods of packaging, and testing techniques. It was a relatively simple task to test discrete components of the earlier generation devices, but medium- to large-scale integration has introduced components with incredibly large amounts of functional capability. Also, semiconductor components of today have increased in speed to the point where testing is becoming a challenge.

Since this study is looking into large-scale computing systems for the 1980's, it was deemed appropriate to at least get some idea of what could be expected for test equipment and test techniques. As subnanosecond gates, perhaps with switching speeds below 100 ps, become available, the question of how to test them must be asked.

The approach to testing has undergone changes as technology advanced. With SSI devices, it was possible and practical to do all testing in one or two operations. This generally consisted of dc and ac testing; dc testing determined if the device functioned and at the correct dc levels while ac testing checked limits on switching times. In general, all gate inputs and outputs were available on external pins of the device package.

With the level of integration in semiconductors today, accessibility via package pins is very limited relative to the amount of function contained in the package. Also, circuit speeds virtually rule out functional testing at the device speed. For these reasons, Control Data has taken a three-mode approach to testing.

Functional Test Mode

The dc testing, as mentioned earlier, is essentially a functional test. A given set of signals is applied to the inputs statically, and the outputs are checked to see that the correct function was performed as defined by a truth table for the device. The set of inputs is then changed repeatedly until all functions of the device are verified. This test includes dc level checking.

Pattern Test Mode

Some types of circuits can be pattern sensitive, that is, certain failure mechanisms may only reveal themselves upon subjection to a unique configuration of data passing through the device. This is particularly true for array types of circuits such as memory devices but can also be true for such things as an ALU or shift network.

A pattern test is designed to generate patterns considered to be worst-case, and the tester subjects the device to these patterns. The test is to determine if the device can process these patterns correctly at rated speed.

Transition Test Mode

If the device functions correctly in the functional and pattern tests, it can then be checked for actual switching speeds. This is typically done on a sample basis to determine if a given lot is within specification. The test consists of applying ac inputs according to specification and checking whatever outputs are accessible on the package for specified rise and fall times.

Test Equipment Configuration

Considerable versatility and flexibility is required in test configurations because of the wide range of devices tested and the extensive function in most. The configuration used by Control Data is basically a computer-driven interface. The interface is as general-purpose in nature as possible and may include some control functions. For the most part, however, the computer does all controlling and checking.

Availability

Test equipment of this nature tends to become rather unique and specialized. Even if a tester is designed which can be used by multiple manufacturers, the market is still very small; very few companies are operating at the leading edge of the technology such as this project requires. Therefore, test equipment such as this does not, and probably will not, exist as off-the-shelf or standard products.

In general, at least some producers of test equipment will accept special orders for such testers but will charge for development as well. On the other hand, it appears that the needed technology for actual measurement exists. For example, sampler technology available from at least two manufacturers extends up to 14 GHz. This translates roughly to rise times on the order of 25 ps.

The next decade will strain testing technology, and it appears that some creative approaches may become necessary. However, at this point there is no reason to believe the problems are insurmountable. Special computer-driver testers, operating in modes similar to those described above, should provide sufficient capability.

CONCLUSIONS AND RECOMMENDATIONS

INTRODUCTION

The foregoing discussions of technologies for high-speed computers have been conducted without regard to a particular processor design. Therefore, it seems desirable to identify those elements that are relevant to the Navier-Stokes computational facility, as envisioned by Control Data Corporation, and to offer some remarks that put the technologies in context of the overall project.

SELECTING A TECHNOLOGY

One of the most difficult problems in computer architecture and design is selecting a specific technology for a product. The architect must make choices dependent on predicted futures, current state of development, industry-wide acceptance, and a wealth of other intangible factors. The lead time required to produce a complex entity, such as the Numerical Aerodynamic Simulation Facility (NASF) means that many choices have to be made in 1978 for a computer to be operational in 1982, based on sometimes sketchy research data and hazy forecasts. This process leads to a risky guessing game of anticipating learning curves, production volumes, and costs, with expensive consequences awaiting one who errs, even slightly.

The mention of the word risk in the context of an expensive computer development may cause one to hesitate in proceeding. The fact is that if a greater than 1-gigaflop performance is required in 1982, there is an element of risk involved in adoption of technologies. Plainly, the low-risk, extant technologies of MECL 10K and MOS memories are inadequate to meet the gigaflop goals. Thus, one must venture into new technologies now under development, and perhaps not yet even tried out, to attempt to meet aggressive performance goals.

Significant Factors

It is not accidental that there are, in this report, many references to the state of the industry and very often to what IBM is doing. The direction and attention a technology such as bubbles gets from the industry depends on what IBM is doing or wants to do with that technology. Important features such as density, cost per bit, and transfer rates are affected by IBM's intentions more than all other end-users combined. For other technologies, with low volumes and extremely high performance (such as Control Data's ECL LSI), the willingness of the developer to invest time, money, and critical manpower resources can dominate the development.

The willingness of a given technology vendor to commit development and manufacturing resources to a project will be directly affected by profit considerations. More importantly, considerations such as return on investment, cash flow, and cost recovery curves may influence a vendor to avoid a particular high technology project (with high per-unit profits) in favor of a low technology, high volume, bread and butter product that does not absorb key resources and cash.

The probability that semiconductor memory and circuit technologies will achieve the performance and density goals is also dependent upon the evolution of relatively new processing techniques and computer-aided development tools. Processes such as electron beam mask generation, or even wafer etching, are under development with their state of maturity somewhat in question. The software and mainframe horsepower necessary to perform circuit design, layout, and verification, and hence circuit employment and machine design for large complexes of hardware, are just reaching the point where they can be effectively employed in support of emerging technologies. In the future, it is quite possible that a new technology will not even emerge without first having in place a great deal of software and hardware power to produce the plans, block diagrams, masks, printed-circuit techniques, and interconnect schemes. Semiconductor vendors in the past have not been equipped or motivated to develop some of these processes, thus requiring the CPU developers to take on a greater responsibility in these areas. Given the profit motives and limited resources possessed by the semiconductor manufacturers, as stated previously, a tightly coupled cooperation between the computer developer and the semiconductor industry in general may be the only way in which high technology will continue to be aggressively pursued for supercomputers in the future.

The impact of not gambling on the success of a new technology may be worse than the risk that the technology will not yield desired results or be seriously delayed. For example, choosing to adopt a new and not quite mature technology which provides a 4X circuit density over an extant, mature circuit family may mean that the parts count of a supercomputer can be reduced sufficiently to provide reasonable reliability rather than marginal operation.

To view only the circuit technologies in an isolated sense is an error. One must take into account the concomitant risks of new packaging, power, cooling, and interconnection systems which are needed to support the electronic technology. Although a 4-year period exists for the development of a Navier-Stokes computational facility, several areas of mechanical and packaging development move much slower than do the electronic developments. Thus, the lead time from the point an electronic technology decision is made to the moment when a mature mechanical supporting system is available can be excessively long.

THE USEFUL TECHNOLOGIES

Based on the studies conducted so far, it is possible to identify some of the technologies that would be needed by a Navier-Stokes engine in 1981 or 1982.

Memories

Regardless of the architecture being considered, it seems obvious that there is a need for a selected hierarchy of memory for the Navier-Stokes facility.

High-Speed Buffer Memory

Most architectural alternatives being considered have built into them a need for cache, register file, or working storage buffers that must be quite fast and somewhat dense, with cost being an ancillary consideration. The characteristics of these kinds of memories are that they are accessed at the minor cycle rate of the target computer mainframe, or even at a rate two or three times faster. They also require storage for 64 to 1024 words of intermediate or buffered data. To utilize normal logic, flip-flops for this function are a possibility, but the density needed could cause an unacceptable expansion of real-estate, which would affect costs, floor space, and even reliability. A better alternative would be to acquire a memory circuit with at least 1000 bits of storage, which could be accessed at 10 ns or less (assuming a 10-ns clock). Current samples of high speed ECL RAMs show that this objective is reasonably attainable with existing technology developments.

High-Speed Working Storage

The high-speed buffer memories usually suffer from a lack of density and high cost in exchange for their performance characteristics. The amount of working storage needed for data to be processed by the CPU is estimated to be between 32 and 64 million 64-bit words. Thus, some form of memory chip other than the buffer memory chip will be required. The design of this level of memory system can tolerate a slower access time than required by the buffer memory system. There are two important points to consider.

- The large volume of memory required means that it will physically be spread out. The resulting distances for transmission of address and control and the return transmission of data are going to require several CPU cycles to traverse. For example, if memory access time were one minor cycle and transmission time (round trip of address to and data from the memory) were four minor cycles, the total would be five minor cycles. To meet cost, power, or density requirements, it might be desirable to use a memory with a two-cycle access time. This would then make the memory access time six minor cycles, instead of five, for only a 20-percent resultant degradation in random access performance.

- In large measure, the performance of the working storage memory is going to be dependent on the bandwidth of data transfer, rather than the random access performance. Memory access time, therefore, becomes a small additional overhead on the total data transfers made for a computation.

  Although bandwidth is a key to the performance of such processors, the time does arise when some random processing is required. The ideal situation would, of course, be to have a working storage of 64 million words with an access time of one minor cycle. Unfortunately, the achievable density of ECL RAMs in the 10-to-40-ns range in the next 4 years appears to be 4000 bits. The physical space needed to contain 64 or even 32 million words of such technology would be quite large, imposing even higher transfer overhead, not to mention the fact that the power, cooling, and cost considerations make such an approach quite impossible for this project's objectives.

A possible approach is to produce a hybrid memory having in some instances fast access at increased cost, and in other instances, slower access at reduced cost with increased density. The current projections for ECL high-speed static RAMs indicate that the first objective can be met. What is needed to complement that is a 16K RAM with about 100- to 160-ns access time.

Although considerable density and cost advantages apply to the CCD memory projections, it is felt that CCD access times are too slow to meet the working storage requirement for the Navier-Stokes machine.

Backing Storage

An examination of the data volumes and processing time requirements of the Navier-Stokes facility indicates the need for another level of memory in the hierarchy which possesses faster access times and transfer rates than the available bulk storage (disk and tape), but is much cheaper, lower powered, and more dense than the main working storage. This type of memory is used to swap the working data for one run, or job, quickly out of the main working storage while loading a new job and to store data associated with a job while it is waiting to be loaded into the main working storage or waiting for transfer to bulk storage.

For example, if a given set of runs each takes 5 minutes and requires a working set of data of 32 million words, the time required to swap data to bulk storage at 40 million bits per second would be about 1 minute or 17 percent of the total job time (this assumes that the next job cannot start until all data is in place, nor may the current job be swapped until the job itself is fully complete). When such jobs are being test run, the actual run times will probably be about 1 minute, thus making the swapping overhead about 100 percent. It should be obvious in such circumstances that twice as many jobs could be run if the swap time was reduced to zero. This is where the backing storage or swapping storage concept emerges. Usually computer architects use a rule of thumb that the ratio of swapping storage to main working storage volumes should be between 8 to 1 and 16 to 1. Thus a 32-million word system should have about 256 million words of backing storage. This storage acts as a slush box. Data is swapped from main storage at high speed and then drained from this level to the appropriate bulk storage at lower data rates while a job is actually running in the main working storage.

The CCD, bubble, and EBAM technologies discussed in this report are suitable for this application because:

- Swapping implies block transfers of fairly large, contiguous regions of memory.
- Access time for block transfers is less significant than transfer rates, as far as performance is concerned.
- The storage device must be fairly compact, with reasonable cost and power requirements, for this range of technology.
- This breed of technology fills the gap that exists in the performance, cost, and density curves for memory systems, in exactly the spot that second-level storage has been projected since the early 1960's.

None of these three technologies can be demonstrated today with the capacities and transfer rates (1 to 4 Gbits/s) that seem to be required by the NASF. However, projections show that by the 1981 to 1984 time frame, a mature version of at least one of these technologies will be available to build a backing storage for the NASF. If one were to guess at this point on which technology offers the best chance of meeting the project objectives, one would have to say that CCD seems most likely. Primarily, this arises from the fact that operational units of CCD are available at about one-fourth the necessary density, with sufficient bandwidth to meet transfer rate requirements, acceptable cost, and performance specifications. EBAM technology, while probably under development for a longer period of time (since the late 1960's), possesses some attributes (known wear-out characteristics of cathode and target) that on the surface makes it appear less desirable, despite the fact that the media can be nonvolatile under power-out conditions and that the access time is quite fast, even for large blocks. Finally, bubbles, while possessing this same nonvolatility, may be extremely difficult to push to the high transfer rates desired for a backing storage device.

Bulk Storage

As always, there is a significant requirement for a stable technology to provide a nonvolatile, high-density, low-power, and low-cost storage media for vast arrays of data and results. As stated in this report, for the foreseeable future rotating mass storage (disk technology) will remain the most viable candidate for this storage task. While recording technology and disk manufacturing are no longer yielding dramatic increases in storage capacity, it is felt that the reasonable upgrades in capacity presently envisioned by 1981 will provide an economical bulk storage facility for the NASF. Further, the transfer rate and access time characteristics that are expected in this same time frame are adequate if, and only if, a reliable and effective backing storage mechanism is available. With the work seen in CCD and bubbles, real backing storage characteristics are at hand, thus relieving disk storage technology from the burden of having to provide that function as well.

Archival Storage

The state-of-the-art in archival systems is in a great state of flux at this time with mass storage tape systems, terabit laser memories, and video recording techniques vying for the position of number one. The actual operational evidence to date of all these candidates gives one a disquieting feeling that this area of technology has not yet arrived. Since the NASF operation can begin (and probably will) without a vast, optimal archival device in place, it is strongly recommended that the delivery of a final storage device be phased into the plan at a date later than 1982 to allow for this area of technology to mature. A smaller scale archival device with a known set of characteristics should be in place for purposes of software development and system test by 1981, but it need not be replaced in the system for at least 2 years, during which time the facility is getting up steam as an operational entity.

CPU Circuits

Some obvious conclusions can be drawn from this technology report regarding the circuits to be used in the mainframe of the NASF.

- They are going to be fast (in the subnanosecond range).
- They will comprise a family of high performance LSI (at least 300 equivalent gates per die).
- They are going to require creative packaging concepts to power, cool, and interconnect.
- They are going to provide a higher degree of reliability for supercomputer systems than has been available at this time.

Less obvious are some pertinent details related to Control Data's approach to the construction of the Navier-Stokes facility.

- It is Control Data's judgement (stated in the architectural alternatives and system design sections of this report) that it is better to build a single unit (whether control, arithmetic, and so on) out of the fastest technology available (with the attendant risks discussed previously) than to build two units operating in parallel out of a safer technology which has one-half the performance of the former technology.

- The first generation of high-speed technology now under development and study possesses speeds and densities that are marginally adequate to meet the needs of a Navier-Stokes computer.

- New development of extensions to ECL LSI, with the now mature software tools and design approaches, could yield a circuit family more suitable for the NASF with a CPU availability in 1981.

- With the design automation tools now in hand, it is possible to develop the total system design and CPU design to a considerably detailed level for a family of circuits without identifying some of the detailed characteristics such as speed, density, and power requirements, as long as some lower limits or thresholds of performance pain can be established by the NASF architects and designers.

  This makes it possible to postpone a final decision about the actual circuit to be used until the mid-1979 time frame, whence the results of many developments, prototypes, and tests will be available to affect selection decision. In RADL's judgement, this is the best way to ensure that the most advanced technology will be utilized in the NASF mainframe. After consideration of the developments identified in this technology report, it is believed that the thresholds could reasonably be established as 400-ps gate delay, with 500 equivalent gates per die and 400 die per square foot of real estate. This would appear to yield a system which could be designed to have a 10-ns CPU clock cycle, which would yield an overall performance for the CDC NASF machine to match the requirements of Ames researchers.

REFERENCES

The following is a list of all references that are called out in the text of this section.

Reference Number	Reference
1-1	Hughes, William C.; Lemmond, Charles Q.; Parks, Harold G., Ellis, George W.; Possin, George E.; and Wilson, Ronald H.: A Semiconductor Nonvolatile Electron Beam Accessed Mass Memory. Proceedings of the IEEE, Vol. 63, No. 8, August 1975, pp. 1230-1240.
1-2	Warner, R. M., Jr.: Applying A Composite Model To The IC Yield Problem. IEEE Journal Of Solid-State Circuits, Vol. SC-9, No. 3, June 1974, pp. 86-95.
1-3 †	Key, J.: Impact Of Microfabrication On Future IC's. CDC Internal Report JK/MR No. 023.77, March 1977.
1-4	Fang, Frank F., and Rupprecht, Hans S.: High Performance MOS Integrated Circuit Using The Ion Implantation Technique. IEEE Journal Of Solid-State Circuits, Vol. SC-10, No. 4, August 1975, pp. 205-211.
1-5	Van Tuyl, Rory; and Liechti, Charles: Gallium Arsenide Spawns Speed. IEEE Spectrum, Vol. 14, No. 3, March 1977, pp. 41-47.

† References marked by a dagger are CDC internal references.

Reference Number	Reference
1-6	Ishikawa, Hajime; Kusakawa, Hirotsugu; Suyama, Katsuhiko; and Fukuta, Masumi: Normally-Off Type GaAs MESFET For Low Power, High Speed Logic Circuits. ISSCC Digest, 1977, pp. 200-201.
1-7	Upadhyayula, Chainulu L.; Smith, Rene E.; Wilhelm, James F.; Jolly, Stuart T., and Paczkowski, John P.: Transferred Electron Logic Devices For Gigabit-Rate Signal Processing. IEEE Transactions On Microwave Theory And Techniques, Vol. MTT-24, No. 12, December 1976, pp. 920-926.
1-8	Hart, C. M.; Slob A.; and Wulms, H. E. J.: Bipolar LSI Takes A New Direction With Integrated Injection Logic. Electronics, Vol. 47, No. 20, October 3, 1974, pp. 111-118.
1-9	Warner, R. M., Jr.: I-Squared L A Happy Merger. IEEE Spectrum, Vol. 13, No. 5, May 1976, pp. 42-47.
1-10	Josephson, B. D.: Possible New Effects In Superconductive Tunneling. Physics Letters, Vol. 1, No. 7, July 1, 1962, pp. 251-253.
1-11	Matisoo, J.: The Tunneling Cryotron – A Superconductive Logic Element Based On Electron Tunneling. Proceedings Of The IEEE, Vol. 55, No. 2, February 1967, pp. 172-180.
1-12	Jutzi, Wilhelm, and Schunemann, Claus. A Cryogenic Memory Cell Concept With Two Josephson Junctions Of Very High Current Density. Scientia Electrica (Switzerland), Vol. 21, No. 3, 1975, pp. 57-67.

Reference Number	Reference
1-13	Koyanagi, Masao; and Nakamura, Akira: Josephson Devices And Their Applications. Journal Of Electronic Engineering (Japan), No. 113, May 1976, pp. 31-34.
1-14	Buck, D. A.. The Cryotron – A Superconductive Computer Component. Proceedings Of The IRE, Vol. 44, April 1956, pp. 482-493.
1-15	Broom, R. F.; Jutzi, W.; and Mohr, TH. O.: A 1.4 mil^2 Memory Cell With Josephson Junctions. IEEE Transactions On Magnetics, Vol. MAG-11, No. 2, March 1975, pp. 755-758.
1-16	Jutzi, W. An Inductively Coupled Memory Cell For NDRO With Two Josephson Junctions. Cryogenics (Great Britain), Vol. 16, No. 2, February 1976, pp. 81-88.
1-17	Herrell, Dennis J.: An Experimental Multiplier Circuit Based On Superconducting Josephson Devices. IEEE Journal Of Solid-State Circuits, Vol. SC-10, No. 5, October 1975, pp. 360-368.
1-18	Herrell, Dennis J.: A Josephson Tunneling Adder. Digest Of The INTERMAG Conference, 1974, pp. 864-867.
1-19	Herrell, Dennis J.: Femtojoule Josephson Tunneling Logic Gates. IEEE Journal Of Solid-State Circuits, Vol. SC-9. No. 5, October 1974, pp. 277-282.
1-20	Gueret, Pierre: Storage And Detection Of A Single Flux Quantum In Josephson Junction Devices. IEEE Transactions On Magnetics, Vol. MAG-11, No. 2, March 1975, pp. 751-754.

Reference Number	Reference
1-21	Zappe, H. H.: A Single Flux Quantum Josephson Junction Memory Cell. Applied Physics Letters, Vol. 25, No. 7, October 1, 1974, pp. 424-428.
1-22	Fulton, T. A.; Dynes, R. C., and Anderson, P. W.; The Flux Shuttle – A Josephson Junction Shift Register Employing Single Flux Quanta. Proceedings Of The IEEE, Vol. 61, No. 1, January 1973, pp. 28-35.
1-23	Anaker, W.. Superconducting Memories Employing Josephson Devices. AFIPS National Computer Conference, Vol. 44, 1975, pp. 529-534.
1-24	Hawkins, Gilbert; and Clarke, John: Nb-Nb Thin-Film Josephson Junctions. Journal Of Applied Physics, Vol. 47, No. 4, April 1976, pp. 1616-1619.
1-25	Lahiri, S. K.: Metallurgical Considerations With Respect To Electrodes And Interconnection Lines For Josephson Tunneling Circuits. Journal Of Vacuum Science And Technology, Vol. 13, No. 1, January/February 1976, pp. 148-151.
1-26	Anaker, W. Superconducting Tunneling Circuits For Computer Applications. International Electron Devices Meeting, 20th Technical Digest, December 1974, pp. 5-8.
1-27	Computer Systems Architecture. Special Report 100.0000.700, Auerbach Publishers, June 1974.

Reference Number	Reference
1-28†	Larson, E. R.: Wideband Communications. CDC Internal Report, Project 95244, FY72.
1-29	Common Carrier Status For Cable TV? Data Communications, Vol. 6, No. 2, February 1977, p. 9.
1-30†	Larson, E. R.: Annual Report CY74 CATV Demonstration, CDC Internal Report, Project 4.1.1.2, 1974.
1-31†	Larson, E. R.. CATV/MDS Coupling Annual Report CY75. CDC Internal Report COP-75-021, January 1976.
1-32	Jacobs, Ira; Miller, Stewart E.. Optical Transmission Of Voice And Data. IEEE Spectrum, Vol. 14, No. 2, February 1977, pp. 32-41.

† References marked by a dagger are CDC internal references.

BIBLIOGRAPHY

A 100 Megabit Electron Beam Memory System. Technical Proposal Prepared For Advanced Research Projects Agency By Control Data Corporation, 8 March 1976.

Baechtold, W.; And Broom, R. F.: Complementary Logic Circuit. IBM Technical Disclosure Bulletin, Vol. 18, No. 3, August 1975, pp. 921-922.

†Bagdons, D. R.. 50 MB Data Set Evaluation. CDC Internal Report COP-76-017, September 10, 1976.

†Burke, R. G.. Contention Channel Characteristics. CDC Internal Report COP-74-010, August 19, 1974.

Chan, H. W., Lum, W. Y.; and Van Duzer, T.: High-Speed Switching And Logic Circuits Using Josephson Devices. IEEE Transactions On Magnetics, Vol. MAG-11, No. 2, March 1975, pp. 770-773.

Declerco, Michel J., And Laurent, Thierry: Design And Technology Of DMOS E/D Logic. ISSCC Digest, 1977, pp. 114-115.

†Doyle, B. H.: CCD Memory Technology. CDC Internal Report SEL-75-024, 5 December 1975.

†Doyle, B. H.: CCD Memory Feasibility Study, CDC Internal Report SEL-76-019, 15 July 1976.

†Doyle, B. H.. 16K CCD Status Report. CDC Internal Report SEL-76-011, 2 April 1976.

†Doyle, B. H.: 65K CCD Status Report. CDC Internal Report SEL-76-021, 31 August 1976.

†Entries marked by a dagger are CDC Internal references.

Feth, George C.: Memories: Smaller, Faster, And Cheaper. IEEE Spectrum, Vol. 13, No. 6, June 1976, pp. 37-43.

Henkels, W. H.. An Elementary Logic Circuit Employing Superconducting Josephson Tunneling Gates. Digest Of The Intermag Conference, 1974, pp. 860-863.

Herrell, D. J.; And Zappe, H. H.: Self-Resetting Logic Circuit. IBM Technical Disclosure Bulletin, Vol. 17, No. 4, September 1974, pp. 1204-1205.

†Larson, E. R.; Allen, G. R.; And Dreher, R. D.. Wideband Communications. CDC Internal Report, Project 95244, FY73.

†Larson, E. R.. CATV Data Communications. CDC Internal Report COP-76-011, July 1976.

†Larson, E. R.: Coaxial Cable Considerations For Digital Communications. CDC Internal Report COP-76-010, June 1976.

†Larson, E. R.: Figer Optics Communications. CDC Internal Report COP-76-013, July 1976.

Lin, Hung Chang; Halsor, Jack L.; Benz, Harry F.· Optimum Load Device For DMOS Integrated Circuits. IEEE Journal Of Solid-State Circuits, Vol. SC-11, No. 4, August 1976, pp. 443-452.

Magerlein, J. H.; And Dunkleberger, L. N.: Direct-Coupled Josephson Full Adder. IEEE Transactions On Magnetics, Vol. MAG-13, No. 1, January 1977, pp. 585-588.

Mause, Klaus. Multiplexing And Demultiplexing Techniques With Gunn Devices In The Gigabit-Per-Second Range. IEEE Transactions On Microwave Theory And Techniques, Vol. MTT-24, No. 12, December 1976, pp. 926-929.

†Entries marked by a dagger are CDC Internal references.

Ohta, Kuniichi, Morimoto, Mitsutaka; Saitoh, Manzo; Fukuda, Terumasa; Morino, Akihiko; Shimuzu, Kyozo: A High-Speed Logic LSI Using Diffusion Self-Aligned Enhancement Depletion MOST. ISSCC Digest, 1975, pp. 124-125.

Owen, Scott, And Nordman, James E.: Application Of Integrated Circuit Technology To The Fabrication Of Large Numbers Of Niobium Based Josephson Junctions. IEEE Transactions On Magnetics, Vol. MAG-11, No. 2, March 1975, pp. 774-777.

Pugh, Emerson W.: Storage Hierarchies: Gaps, Cliffs, And Trends. IEEE Transactions On Magnetics, Vol. MAG-7, No. 4, December 1971, pp. 810-814.

†Schiebe, L. H.: High-Speed Trunk Data Set User Interface Specification. CDC Internal Specification, January 27, 1977.

†Schiebe, Lowell H.: Network Trunk Groundrules. CDC Internal Report COP-74-004, May 16, 1974.

Schuenemann, C.: Fast Josephson Current Switch Logic. IBM Technical Disclosure Bulletin, Vol. 18, No. 2, July 1975, p. 551.

Sweet, Allen A.: Understanding The Basics Of Gunn Oscillator Operation. EDN Magazine, May 5, 1974, pp. 40-47.

Yao, Ying L.: Electrical Characteristics Of Two-Level Single Ground Plane Interconnection In Josephson Tunneling Logic. Proceedings, 26th Electronic Components Conference, April 1976, pp. 125-130.

Zappe, Hans H.: A Subnanosecond Josephson Tunneling Memory Cell With Nondestructive Readout. IEEE Journal Of Solid-State Circuits, Vol. SC-10, No. 1, February 1975, pp. 12-19.

†Entries marked by a dagger are CDC Internal references.

SECTION 2
ARCHITECTURAL SURVEY AND PROJECTIONS

OUTLINE OF THE PROBLEM

The problem, simply stated, is to develop a computational facility sufficiently powerful to support the design and analysis of aerospace vehicles in the 1980 to 1990 timeframe. This implies solving the Navier-Stokes equations for three-dimensional flow at subsonic, transonic, and hypersonic velocities.

Given our current understanding of computer-based solutions of the above problems, some general statements can be made about the computational requirements.

- No general-purpose computer now in existence is capable of providing the power needed to accomplish complex three-dimensional flow simulations. Based on experience gained in two- and three-dimensional flow simulations using Illiac IV and the CDC 7600, a sustained rate of 1 billion floating-point operations per second will be necessary to support use of the NASF for engineering work in the 1980's. This is in the range of 100 to 300 times the performance of the CDC 7600 on current flow simulations.
- The architecture and technology for supporting systems such as interactive graphics, archival storage systems, and high-band-width I/O subsystems are fairly well understood. Available general-purpose hardware and software appear sufficient for the needs of the proposed computational facility. The problem here is to get a proper assessment of the cost-versus-features profile.
- The architecture and technology that would make possible the construction of a processor capable of solving the three-dimensional aerodynamic problem are now at hand, in the form of parallel processing concepts and high-performance LSI technology.

- The development of a specialized machine to perform the Navier-Stokes solutions in the optimum fashion appears to offer the highest return for the invested effort, if:

  Development risk in terms of dollars, time, and expectations of performance can be reduced or contained, and

  The special architecture and design of such a computational facility does not preclude efficient adaptation to new algorithms as they reveal themselves in the 1980 decade.

- The effective power of any hardware architecture must be complemented by rethinking algorithms and restructuring traditional solution codes to match the specialized computer structure.

- The development of a truly effective computational facility thus requires an interactive design involving algorithms, supporting software (such as compilers), computer architecture, and technology.

- The facility, although specialized to this general task, cannot completely abandon the techniques and codes of the past. In some manner, the traditional FORTRAN inventory of program material should be capable of efficient operation on the computational facility, though at a somewhat reduced performance level compared to the optimized and restructured algorithms.

In summary, the required computation rates represent an enormous extension beyond that available from the best general-purpose engine extant today. Clearly, a machine tuned to this problem is needed. Tuning necessarily implies degraded performance against other classes of algorithms. The solution methodology of the subject period must be defined well enough so that it lies within the tuning range of the machine design, and the design must be capable of reaching the required performance for the defined class.

A significant portion of the first phase effort was directed at this matter of problem definition. Two approaches to the solution in use today were identified and were called implicit and explicit. A third approach is currently in the early stages of development and is termed the spectral analysis solution. This approach was assessed to be outside the class of interest in this study. Representative programs from both the explicit and implicit solutions were studied in order to establish base-line performance data and to assess the amenability of the programs to parallel computation methods. These programs are described in section 3 of this report, and their current computation rates are tabulated there. The problems for which the NASF is to be used will also include three-dimensional extensions to these example algorithms. This expands the computation time beyond feasibility on even the fastest general processors of today (7600, STAR-100, Cray I). A baseline time to compute can only be extrapolated. The extrapolation is not all that straightforward, however, and a considerable analytic effort was expended during the first phase to illuminate this area.

Beginning with the two-dimensional forms of the Navier-Stokes codes, the current versions running on the 7600 are achieving computation rates from 2 to 3 megaflops (millions of floating-point operations per second) as a sustained average. Extrapolations to three-dimensional models indicate that this rate will remain about the same. The initial projection of a computational requirement of 100 times the 7600 would thus yield a system of about 200 to 300 megaflops. This rate, however, will not achieve the goal of a complete 100x100x100 grid solution in 7 to 12 minutes. As described in section 3 of this report, the best guesses for computational load due to calculations in three-dimensions is much greater than anticipated at the outset of this study. The conclusions reached from these simplistic projections are that a computation rate of at least one gigaflop (one billion floating-point operations per second) be sustained throughout the job.

The special processor requirement definition is the principal task of this project phase. Such a facility, of course, requires the existence of a considerable amount of processing support facilities. The general system strategy is that the special processor would supply none of this, since such capability unnecessarily complicates its operation, particularly software. Some initial efforts were made to define these requirements with the idea of bounding the critical operation support systems. The volume of input and output data argues strongly for graphics facilities, and the requirements for solid compilers, system control and accounting software, and datacomm capability combine to indicate the need for a substantial front-end.

With the assumption that most, if not all, front-end functions can be performed by standard products, both hardware and software, the need to precisely define these system components at this early phase (5 years before target installation) is small. That is, if nothing special is required other than configuration and sizes or quantities, the only real need for defining the system front-end is to determine gross performance and overall sizing. This can be refined and actual products can be filled in as the program progresses. This level of definition has been employed in the system design and site requirement sections of this report. Standard products existing today are used to get a realistic base sizing. These products most likely would not be delivered with a system in 1982; they would be replaced by their then current counterpart.

TECHNOLOGY INFLUENCES

Section 1 of this report assesses the ranges of relevant circuit and storage technologies which are expected to be current in the 1985+ processing era. It is, of course, axiomatic that the cost effectiveness of the hardware design is largely determined by the correctness of the architect's judgement in anticipating the appropriate circuit and storage building blocks. To some extent, he can provide technology extensibility, that is, the ability of the design to incorporate unanticipated technology variations, but this is relatively narrow in scope.

Comparisons that become most important to look at involve the storage hierarchy. An open-ended address structure and addressing mechanism is a relatively low-cost way to retain considerable flexibility in adapting to unpredicted storage opportunities (or problems depending on your outlook). The most cost-effective mix of fast, extended and auxiliary storage, however, is strongly influenced by the precise cost-performance tradeoffs in the object span of time.

The choice of proper logic circuitry is usually the more controversial even though of less consequence. The primary considerations are the proper assessment of two major factors. The first of these is to predict the best likely speed-power product. During the small scale integration (SSI) era (1965 to 1980), the major logic families generally started in the 100- to 1000-pJ range (10-ns switch time, 100-mW power). Continuing technology improvements drove this down to the 10- to 20-pJ range. Early medium-large scale integration (MSI/LSI) circuits started in this 10- to 20-pJ range and have generally moved downward as new techniques were developed. Generally, those techniques which showed ways to allow a greater scale of integration were the principal causative agents. The circuitry chosen for this engine must, first, correctly intercept this trend. The particular speed and power chosen, which is a choice among various semiconductor options, then results largely from the degree of parallelism the designer feels is manageable.

The second factor is reliability. Whatever circuitry is chosen for this engine will be used extensively. Exotic circuit families which provide leading edge speed-power products generally cannot be used because of the lack of manufacturing level support. This lack of support manifests itself in reliability and availability problems. Given a reasonably established technology, the logic reliability is determined by the care taken in the cooling design and conservatism in the timing margins. Therefore, the device technology profoundly affects the physical architecture also, since this is principally determined by the cooling and power distribution structure.

Like most other fields of design, successful processor architecture demands a continuing interchange between the architect and sources knowledgeable in semiconductor technology. Furthermore, such sources should represent some breadth or the designer may become blind to legitimate alternatives. This is particularly true when the basic building blocks are changing as rapidly as they are for the computer designer.

The Electronic Technology Survey and Projections in section 1, summarizes observations on what can be expected in storage and circuit technology for the mid 1980's, plus comments on what would be desirable.

STANDARD AND SPECIAL SYSTEM COMPONENTS

As noted elsewhere, the overall facility design strategy calls for providing utility processing capability to relieve the special processor of most of the general-service tasks attendant upon operation of the facility. For this capability, machines with fully supported standard software are the best choice.

COMPILERS

The Navier-Stokes Solver (NSS) will require one higher-level language for programming solutions to the Navier-Stokes problem. If the operating system resident on the NSS can be reduced to a series of firmware functional modules, perhaps little effort will be needed for developing a higher-level language used in systems programming for the main computing hardware. Later in this report, the rationale for adopting an extended version of the FORTRAN language as the main programming language for the NSS will be discussed.

Different strategies for the design and implementation of this compiler could be considered.

- Generate a compiler that executes on the NSS and compiles code for the NSS. This strategy has several variants.

  Build a software simulator for the NSS to run on an existing, standard-product computer so that the compiler can be completely checked out long before the NSS is operational.

  Design and program the compiler during NSS construction and await NSS operation before beginning checkout of the compiler.

  Write the compiler in yet another higher-level programming language such as PASCAL, debug the compiler on an existing standard system and then compile in PASCAL for the NSS so that final FORTRAN would run on the NSS.

  Create a simulator which executes on the NSS and reproduces the execution of an existing machine (for example, CDC CYBER 176). A compiler could then be written to execute on the 176 to produce code for the NSS, and the resulting compiler could be checked out on the 176 which exists now. When the NSS becomes available and the simulator is debugged, the compiler could be moved to the NSS.

- Generate a compiler that executes on the front-end processor, producing object code to be executed on the NSS. Two variants are:

  Use the front-end (parsing phase and source code optimizers) of an existing compiler system (with appropriate NSS extensions)

  Build a brand new compiler from scratch.

Considering the above possibilities, the best approach appears to be the construction of a compiler to execute on the front-end processor, producing code for the NSS, and built whenever possible with existing software modules on that front-end processor. Obviously, any compiling scheme that relies on the availability of the NSS hardware for checkout of the language processor is going to delay the appearance of a stable language system for production programming. Using existing hardware and software will reduce resource requirements and risks in the compiler development.

GRAPHICS

Most users solving large scientific problems have realized that some sort of graphics facility is essential to efficient operation. Not only are the modeling time and cost reduced, but the overall system response to the user is greatly enhanced. The user is less harassed by details, and the occurrence of errors is much less frequent. As much as 90 percent of the cost (in terms of effort on the part of the user plus machine costs) can attribute to the pre- and post-processing tasks. Tables 2-1 and 2-2 summarize case studies done by an auto company (for analysis of a door). Figures 2-1 and 2-2 illustrate an example drawn from a mechanical engineering firm. Clearly, analyses of this type must contemplate capable graphics facilities.

The software which provides graphics capability has provided an interesting challenge. Its complex nature and natural independence of function from that of the special processor argues strongly for it residing elsewhere on one of the ancillary systems.

TABLE 2-1. COMPARISON MAN-HOURS USED FOR A TYPICAL AUTO-DOOR STRUCTURE ANALYSIS

	Estimated Manual Process	Actual Hours Making Use Of Maker X Body Design System	Estimated Hours Using Graphic Structures Pre/Post Processor
Outer panel	165	55	10
Inner panel	180	60	12
Front extension	228	76	15
Rear extension	222	74	15
Hinge reinforcement	96	32	6
Intrusion beam	9	3	1/2
Reinforcement at belt (outer panel)	9	3	1/2
Reinforcement at belt (inner panel)	45	15	3
Window stop	1	1/2	1/6
Reinforcement, inner panel lower front		Negligible	
Reinforcement, rear extension upper		Negligible	
	955 Man-Hours	318 1/2 Man-Hours	62 1/2 Man-Hours

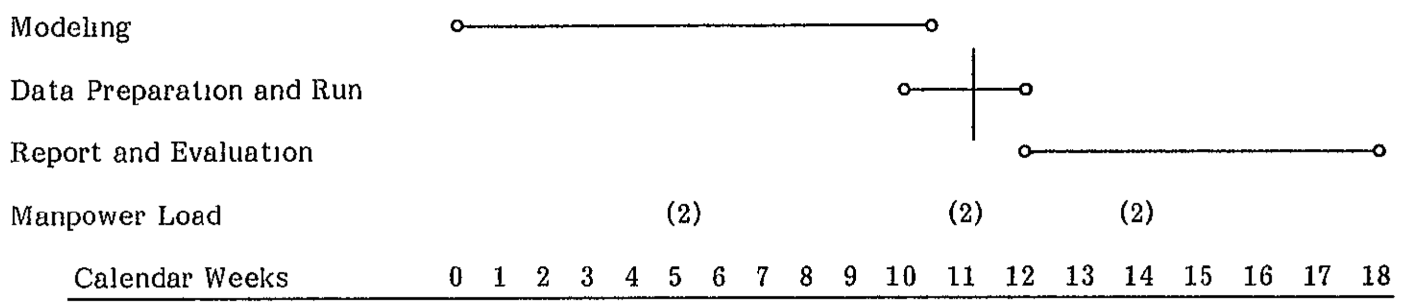
TABLE 2-2. COSTS AND TIMING COMPARISON OF COMPUTER ANALYSIS VERSUS PROTOTYPING APPROACH

Operation	Timing	Costs
Graphic structure analysis	13 Weeks	$12,455
Current design structures approach	30 Weeks	$19,200

COMMUNICATIONS AND OTHER UTILITY FUNCTIONS

A second, logically independent function would be the datacomm facilities to be provided. That is, if users are to input requests or be sent reports remotely via common carrier facilities, this front-end function should be apart from the special engine. Front-ending is becoming a well-developed art, and by 1985, it should be almost a commodity. Little is seen to be gained from intergrating such facilities back into the special computation processor, in view of the enormous potential for increased operating software complexity.

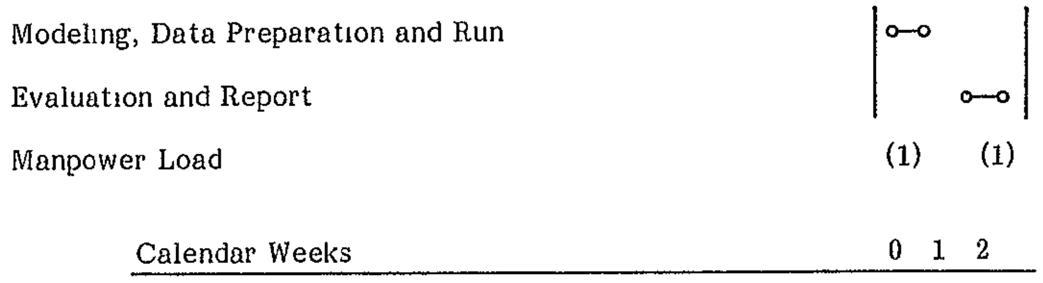
Other functions which need to be provided, but which should not be allowed to disrupt or complicate the special processor, are the various service station utilities. Magnetic tape, low speed disks, conventional printers, and unit record equipment are all facilities which should be handled off-line (from the special processor) to avoid the need for redeveloping a lot of complex software.



Cost

Modeling	1 @ 1200/wk, 1 @ 800/wk	20,000	
Data Preparation and Run	↓	4,000	
Evaluation and Report		12,000	
	Engineering	36,000	92%
	Computer Runs	2,400	6%
	KP, KV, ETC.	600	2%
		39,000	

Figure 2-1. Typical Design Cycle For A Mechanical Structure (With Manual Data Preparation and Output)



COST

	Engineer		Computer (Est)	
Modeling, Data Preparation	1 @ 1200	1200	5,000	10 HR @ 50 + 3 HR COMP.
Run		–	2,400	2 HRS
Evaluation and Report	1 @ 1200	1200	1,300	2 HR @ 50 + 1 HR COMP.
		2400	8,700	
Total Cost		$11,100.		

Figure 2-2. Typical Design Cycle (Same Problem With Interactive Graphics)

94A

Major data base or archival storage, however, should probably be accessible on a shared basis between the special engine and the ancillary processors. The management of such facilities is best off-loaded from the special processor, of course, but the need for broadband access paths is evident.

In summary, a substantial amount of ancillary processing capability is indicated. The general distribution of function ground rule should be to save the special processor for its principal function. All utility functions should be done off-line. For these support functions, standard systems components should be used insofar as possible. Software stability and dependability is particularly important which implies using standard, supported products.

REVIEW OF POSSIBLE SPECIAL PROCESSOR (SP) ARCHITECTURES

Even the most sophisticated estimates of the circuit and storage technology to be available in the next 5 to 10 years do not permit consideration of a conventional single instruction stream, single data stream (SISD) processor. Some form of parallel computation structure must be used.

Two major paths to parallelism are currently used in today's computer systems, the single instruction stream, multiple data stream approach (SIMD) and the multiple instruction stream, multiple data stream approach (MIMD). These two approaches appear to cover all areas of interest for this problem. References 2-1 to 2-8 furnish examples of the discussion extant on this topic. These basic approaches to architecture can be examined by examples of each.

CDC 6600/6500 (MIMD) APPROACH

MIMD is, at its simplest, more than one processor executing programs that in some way are coordinated with each other (figure 2-3). In the early days of computing, coordination, communication, and sharing of resources was accomplished by tying together two or more processors via a disk file or tape that could be read or written by all attached processors. Later, multiple processors were attached to a common high-speed memory which provided the communications and coordination mechanism. On the CDC 6600, 10 peripheral processors (PPUs) were attached directly to the memory of the central processor (CPU). Subsequently, the CDC 6500 provided two CPUs connected to the main high-speed memory, along with the 10 PPUs. These basic configurations are found on today's CDC CYBER Series computer systems.

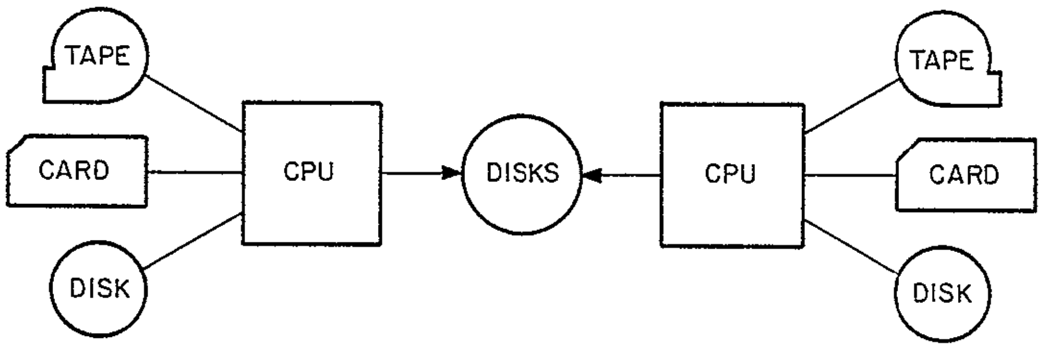


Figure 2-3. MIMD (Multiple CPUs Sharing Disk)

Such configurations can create engineering problems and a high degree of system problems to the extent that they are not carefully thought out and consciously constrained to a tight set of groundrules. The engineering problem is related to how large amounts of memory can be shared (via ports) with a multiplicity of attached devices (or processors) and still provide performance. Engineering tradeoffs must be made regarding fan-out and fan-in of physical cables, priority networks, and most importantly, methodologies of efficiently dealing with contention for memory access.

In the 6600 family of computers, the engineering problem was resolved by effectively making all processors (CPUs) synchronous in their electronic access to the memory. In addition, the 10 PPUs were limited in their bandwidth requirements by the time-multiplexing scheme used to implement those 10 processors. Since the 10 PPUs were in effect multiplexed together, only one real port was required from memory for all 10 processors. This solution resulted in a major reduction in memory access electronics and forced a necessary discipline on the programming options.

The systems problems of harnessing the parallelism of 11 processors (10 PPUs and one CPU in the 6600) or 12 processors (10 PPUs and 2 CPUs in the 6500) were dealt with by narrowly defining the functions to be performed by each class of processor. The 10 PPUs were assigned to tasks that can be easily separated from an operating system structure. A task is some portion of a job. For example, reading cards or opening a file were tasks that could be performed with a great deal of independence of other job activities, and thus could be assigned to a separate processor. To simplify operating system structure and user interface, the multiple-CPU CDC 6500 used its two central processors to handle two independent jobs. This approach required little or no coordination between the two CPUs.

The independent job approach is the simplest and cleanest method for controlling a multiprocessing structure. In the CDC 6500, it provided an economical means of sharing hardware. Thus, when two jobs could reside in the same (fairly expensive) memory, both could be executed simultaneously. In theory, this approach provided a better utilization of the PPU memory and external resources such as disks and other high-speed peripheral devices. There was one severe limitation in this approach. That was in the case of the single job that required all of the available high-speed storage (as would even a modest Navier-Stokes solution) while being almost entirely compute bound for key calculations. In this instance, no hardware or software devices were available to permit two totally independent arithmetic processors (CPUs) to coordinate on a specific and totally absorbing computation. So the performance became that of a single processor.

Customized programming of the two CPU configurations for one single calculation (one single algorithm) has permitted customers to access the power of both CPUs simultaneously, but at the cost of decreasing the operating system efficiency and excluding for a prolonged period of time all other general-purpose users of the system.

MULTIPLE MINICOMPUTERS FOR THE MIMD APPROACH

With the advent of very cheap, fairly fast minicomputers, the multiple-mini (figure 2-4) approach has been receiving close attention. The same engineering and systems problems that Control Data encountered in the early 1960's are once again the dominant constraints on effective application of multiple minicomputers to vast general-purpose problem solving, in spite of the extravagant claims to the contrary by inexperienced architects.

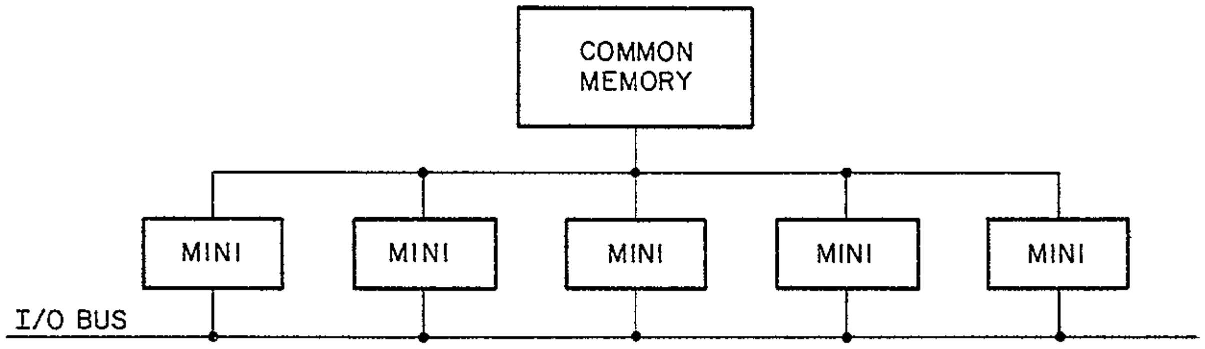


Figure 2-4. MIMD (Multiple Minis Sharing Memory)

Multiple-mini systems are comprised of two or more miniprocessors strapped together in a given configuration and generally coordinated through a series of software conventions with or without hardware assistance. In the most straightforward case, each minicomputer possesses its own memory and cooperates with other minis via a normal I/O channel. When operating as a collection of independent and somewhat asynchronous task processors (like the 6600 PPUs of 1963), the result can be a very effective processing system. The amount of data and coordination information passed between the minis in such a configuration does not strain the I/O channels or require much overhead for cooperative ventures such as I/O control, data blocking/deblocking, input editing, output formatting, and peripheral device allocation.

As the volume of data that is to be passed between minis increases or as the data structure required by one mini no longer matches the data structures used by another mini, the data handling requirements begin to exceed the I/O channel capability. A common solution to this dilemma is then to construct high-speed, random access storage and attach all of the miniprocessors to that memory by a collection of memory ports. Thus, the structure becomes even more similar to the CDC 6600, with 10 PPUs each having its own 4096 words of memory attached (for coordination reasons) to a large common memory.

Several major differences arise between the minisystem implementation and the CDC 6600, however. At the outset, the 6600 hardware was designed to limit the PPU to task operation and to reduce hardware ports and associated control. Today's multiple minisystems are being proposed to solve the computational problem by attempting to synchronize a multiplicity of processors for a single large calculation, thus yielding the parallelism needed by problems such as found in aerodynamics simulation.

Immediately, one is confronted by the need for a high-data bandwidth between each mini and the common memory, implying independent ports for each processor and a consequent growth of hardware. In addition, if each miniprocessor is truly inexpensive and off-the-shelf, its need for memory access will be electronically, as well as software-wise, completely asynchronous with all other members of the multiple-mini system. Resolving this problem of contention for memory among groups of 8 to 64 processors becomes a monumental hardware headache, as well as a system impediment. A common solution is to propose a general-purpose memory cross-bar mechanism which permits access to all memory modules by all processors, with some means of establishing priority of access while guaranteeing a worst-case delay for any given mini which has low priority. Such cross-bars tend to be costly when implemented, with hardware escalating at a rate of the trunk-width squared or even cubed. It is not uncommon for the cross-bar cost to exceed the combined cost of the minis and memory.

The general thesis of the ensemble-of-minis (or micros) approach is: For a given computational requirement, the collection of processors costs less than a unified processor, and the collection of N processors is capable of roughly N times the performance of an individual member.

Performance Estimates of Multiple-Minis

To examine multiple-minis, first consider performance. If the mini configuration is loose enough so that each mini is in fact doing an independent task (for example, one in each major city) the combined performance of N of them obviously is N times the work of one. At the other extreme, where the nature of the calculation is such that none can calculate until it receives all the preceding calculations of the others, the overall output is completely determined by the interelement communications mechanism. The possibility for deadlock is almost certain. So the key to the performance lies in the nature of the problem and the programmer/scientist's perception of the problem, since it is with his translation of the problem that the processor, ensemble or unified, must cope.

A parameter of processor design which influences the potential performance of an array of processors is the concept of data distance. As long as the data in the problem is well-behaved, which means the solution requires few data moves to nearby processing elements, the array can function efficiently. As the transmissions become more frequent or must pass through several nodes, the effective solution rate is severely reduced, because each data movement takes a quantum of processing time. Much work is currently being done to decrease this time loss. One such attempt to increase the connectivity is to increase the number of nearest neighbors as the logarithm (base 2) of the number of processors. Sixteen processors would have four nearest neighbors, and 64 processors would have six, and so on. A 16-node processor then could take up to four cycles to transfer a data item, 64 can take 6, and so on, from one node to the other receiving nodes. During this time, no computation is taking place and all processors, except those involved in the transfer, can be idle. In this case, the super array processor is reduced to one medium speed minicomputer.

Most problems are larger than the array of processors, in which case, the distance of the data to the array is even greater because the data needed is kept on a backing store (for example, disk) and everything comes to a complete stop until the transfer is completed.

On the other hand, the pipeline (vector) processor is intimately connected to a very large, immediate access memory. Any data item can be fetched in one memory access.

What does this all mean? Simply, as an array processor structure grows larger and larger, the time to transfer data between elements grows at a faster rate. In other words, doubling the number of processors (nodes) does not double the computation rate (cut solution time in half).

On the other hand, doubling the number of pipes in a pipeline machine will almost exactly double the data rate at much less than double the cost, assuming the memory size is not doubled (only the pipelines). (A doubling of memory is implicit in an array machine. Cutting local or node memory in half so as to keep the same total for twice the number of nodes means that the number of data transfers between nodes will probably increase by more than a factor of four during which no work is being done.)

Reference 2-1 states that Minsky and Papert have observed that for the general problem, the gain in performance is more like $\log_2 N$. That is, 64 processors will be 6 times as effective as one processor, largely because of the attendant effect that most processors stand by most of the time. Others (reference 2-2) dispute this conjecture because of its implied generality. Examples are cited where the N times factor is approached. However, these usually turn out to be cases where a large part of the problem naturally breaks up into independent tasks, and an analyst can be found who is able to discern this inherent parallelism. So the performance side of the equation increases no more rapidly than N, perhaps not at all, and in some middle cases as $\log_2 N$.

On the other side of the coin, the cost goes up at least linearly with N (assuming large quantity buys are in effect for all cases). Adding to this increase is the amount of communications and control circuitry, which increases at some faster rate. On the other hand, considerable evidence obtained for the case of a unified processor shows that the cost increases with the square root of capability (so called Grosch's Law). Figure 2-5 gives some calculated points for a unified design, in this case, a 6400 arithmetic unit. Paper designs were done using TTL and ECL bit-slice circuits and two versions of high performance ECL-LSI. The last points used extrapolated costs (based on quotations), and the first two used large quantity circuit and real estate prices. The performance criterion was taken as the possible minor cycle, which in CDC parlance is the time needed to complete a long add, that is, the floating-point add of two 48-bit mantissas. The theoretical slope of 1/2 is seen to be quite within the error, in any case, the slope is well under 1.

A similar reasoning can be developed using more widely extant processors. Consider the 8080. Signetics offers an 8080 emulator made from TTL LSI circuits (3000 series). The cost is about 2.5 times an equivalent standard MOS 8080, and the performance is about 5 times (the range is 2 to 9).

The PACE 16-bit single chip processor affords another example. A one-board emulator (TTL) exists and gives about a 10X performance improvement for roughly 3.5 times the cost.

These examples are all cases in point to support the general thesis that a unified processor (that is, a fixed or given architecture) can be designed with a fairly wide range of performance, depending on the circuit technology used to implement it. Although it may not be theoretically conclusive, this evidence supports the general trend of Grosch's Law, namely that performance goes up a factor of four for each doubling of price, that is, a slope of 1/2 on figure 2-6. This says that even for the case of a set of independent tasks, for hardware cost alone the unified processor approach is the less expensive. The

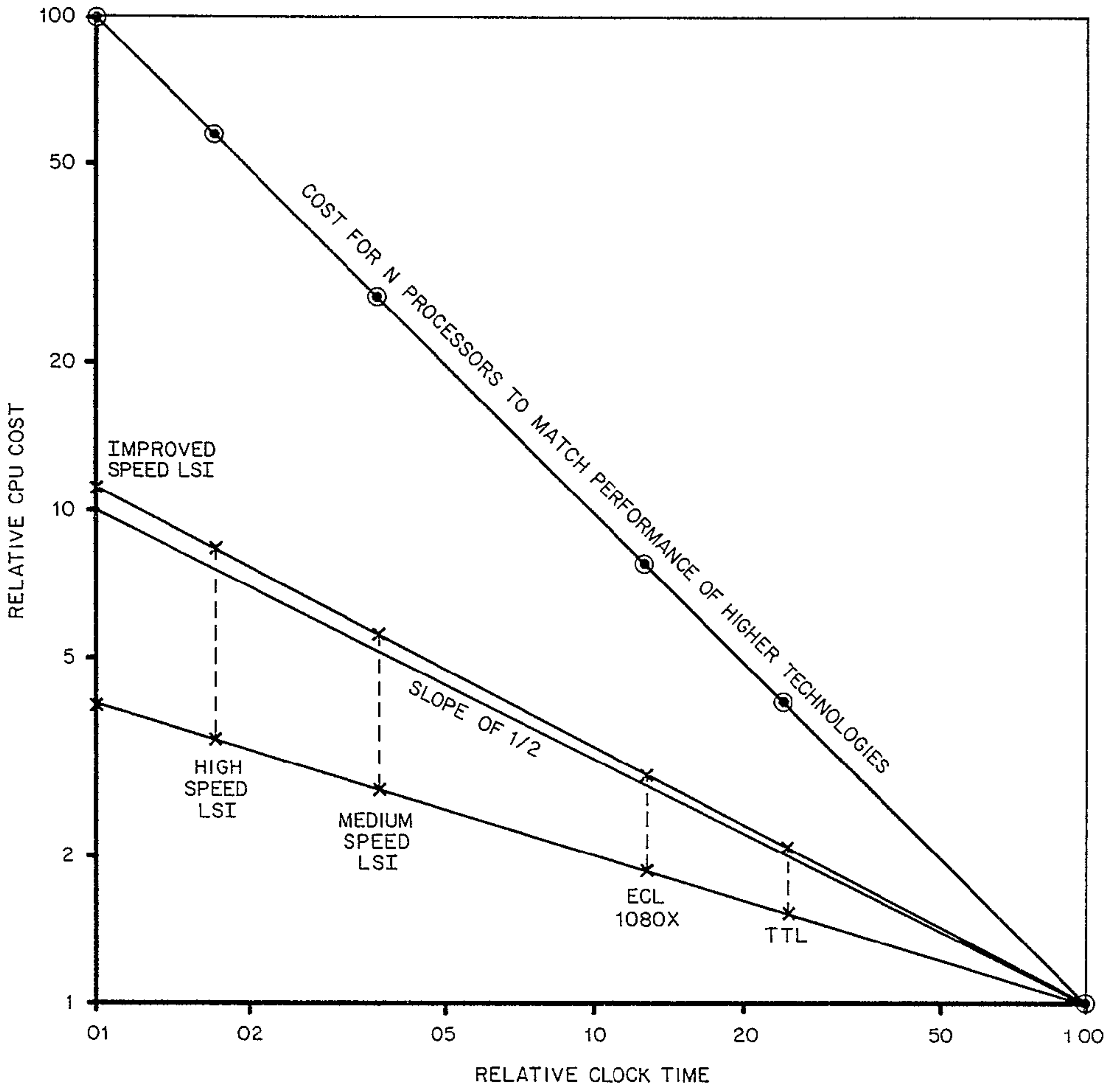


Figure 2-5. Cost Comparison of N Low Performance Processors To Higher Technology Processors (CDC CYBER Architecture)

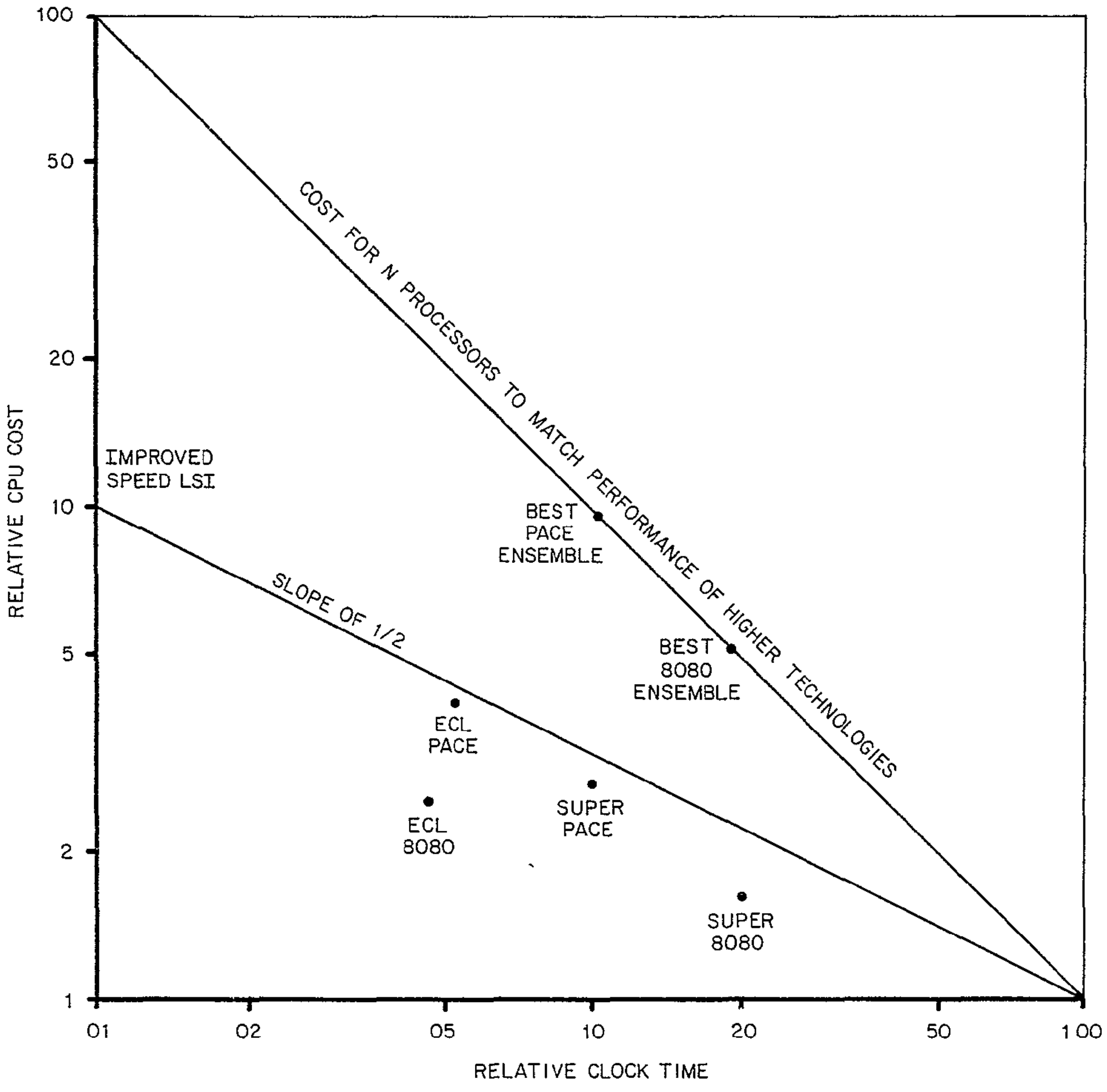


Figure 2-6. Cost Comparison of N Low Performance Processors To Higher Technology Processors (Other Architectures)

counter argument is that the software complexity needed to manage the multiprogrammed unified processor dominates the cost as well as limits the performance. The converse is the case of a multitude of minis attempting to reduce a single large problem, which is also a formidable programming endeavor. However, in this case, not only is the control software complex (indeed it is to date incomprehensible), but the hardware cost is also overwhelming.

Variable Architecture

One may say that the above is all very well and good but the objective is to replace a very costly and complex, unified processor with a multitude of small, inexpensive processors of different architecture (for example, replace a STAR-100 with masses of 6800's). Note what that implies, in terms of the amount of CPU hardware.

In vector mode, STAR produces a 64-bit add result every 20 ns. A 6800-based processor would take in excess of 1 ms. This is a ratio of 50,000 to 1. Thus, it would take at least 50,000 6800's to equal the STAR-100 vector rate in add mode. Of course, each 6800 must have its own memory, buffers, communication, I/O, and so on. The cost of 50,000 processor boards would approach $15,000,000 even without ensemble central control and communication.

The above 6800 system is obviously impractical without a higher performance processor. Consider a processor with a 1-μs integer instruction time and a 5-μs floating-point average instruction time. These numbers are picked to keep the processor cost down, and are typical of what can be expected for extrapolations of today's minis. Suppose further that the processor is a 32/64-bit machine so as to be suitable for scientific use. At least 250 units of this capability would be required; an estimate derived from Minsky's observation becomes quite beyond reason. But even in this most optimistic case, the cost of such a configuration would probably exceed that of the LSI STAR-100 (about 2500 LSI circuits or 375,000 gates). Again, when the ensemble control and communication hardware is added (which is included in the reference unified processor), the balance is even more heavily tipped.

A major point to note from both these exercises is the amount of multiprocessing needed to duplicate one STAR pipeline. Most current analyses of multiprocessor systems implicitly assume relatively small numbers (8, 16, 64, typically). If the application requires greater performance, more capable processors are implied or specified to keep the number down. The studies typically begin by describing 8080's or 6800's (everyone knows how inexpensive these devices are) but end by alluding to multiple minis (or maxis) or narrowing the application to appropriate jobs (for example, character manipulation). The bait-and-switch fails as an approach is made to real scientific problems. One should bear in mind that this analysis of the problem clearly indicates that a STAR pipeline itself falls short of the computational requirement by an order of magnitude. The problem is more like 500,000 6800's or 5000 Nova's in terms of computational speed. Faced with this, any competent computer designer immediately turns to increasing the performance of the individual processor as much as possible, thereby decreasing the number to at least a comprehensible, if not manageable size.

A major drawback to using the previously described mini configurations for high performance computations is the inherent problem of communication. At the highest level is the software coordination necessary to cause two or more processors to work on a single, key calculation. Assuming that all processors have universal access to all data in a given problem (via their cross-bar connection to the large centralized memory), partitioning work among the various processors for the following, not uncommon computation is difficult at best.

```
     DO  10  K=1,400
  10 X(K)=X(K)+Y(K)*(R*X(K+I)+T*Z(K+I+J))
```

The computation could be partitioned in terms of X(K) where each processor could take 1/16 (if there were 16 processors) of all the values of X, Y, and Z into its own memory, and then compute the corresponding result X(K). Note that all processors could not start until all processors had isolated their particular values of X, otherwise, a late arriving processor might read a newly computed value for X(K+I) that had been stored by one of the others in the mini configuration.

In the general approach, this scheme falls apart, however, as the DO loop termination goes from 400 to 400,000. At some moment, there will be too many values to be stored in the private memory of each mini. At this point, through software or hardware semaphors, all minis working on the computation must be put in some form of lock-step to prevent them from tampering with each other's data prematurely. Nearly all real scientific calculations present the mini configurations with such a situation.

CDC FLEXIBLE PROCESSOR (MIMD) APPROACH

Another MIMD approach is using Control Data's Flexible Processor (FP). The FP is a digital microprogrammable processor which has been designed specifically for multi-dimensional array processing. Since the units are microprogrammable, they combine the advantages of special-purpose hardware in efficiency and speed of software changes to the computational algorithms.

Each FP can be used as a building block in a modular system design where each FP is accordingly programmed with appropriate algorithms to perform specific tasks which are assigned to it. Moreover, programming of the various FPs in an array can be changed dynamically under control of a processing system which calls prescribed programs from memory, as from a rotating disk file.

The features of the FP make it suited for systems applications in a wide variety of scientific disciplines. One such area of application is digital image processing. Typical computational functions which can be solved with this processor include image matching, correlation, spatial transformation, registration, radiometric corrections, change detection, statistical classification, and various enhancements. Applications in the field of image processing include cartography, reconnaissance, management of earth resources, nondestructive testing, and biotechnology. Additionally, the FP is suited for control of image display and recording equipment.

Hardware and Software Description

The FP features high arithmetic computation rates, considerable character-handling capabilities, an advanced input-output structure, and semiconductor register file memories for data storage. Each unit features arithmetic logic capable of a 0.125-microsecond addition of 16-bit or 32-bit operands, a 0.250-microsecond fixed point multiply of 8-bit bytes, and a 1.125-microsecond fixed point multiply of 16-bit operands.

Flexible Processor Characteristics

- Microprogrammable - random access microcontrol memory
- 32-bit or 16-bit word lengths
- Array hardware multiplier
- 16-level hardware priority interrupt mechanism; 3-level mask capability
- Specialized logic for square root and divide
- 8-MHz file buffered word transfer rate; 16 word by 32-bit or 16-bit input file buffer
- 2-MHz direct memory access word transfer rate
- 1-MHz register-buffered word transfer rates
- Dual 16-bit internal data bus system
- 0.125-μs clock cycle
- 0.125-μs 32-bit addition, 0.250-μs byte multiplication
- Register file capacity up to 4128 16-bit words
- Hardware network for conditional microinstruction execution; four mask registers and a condition hold register

Related Examples

Two examples are described here to illustrate the versatility and data handling capacity of FP configurations. The applications described are:

- Digital change detection
- Digital scan converter memory system

The following examples demonstrate the capability of FP systems to perform complex processing functions at high throughput data rates. These systems are cost effective and therefore deserve attention in similar applications.

Digital Change Detection

The 4-channel modular change detection system performs automatic digital change detection at a nominal processing rate of 830,000 pixels† per second from each of two input images. Each pixel is encoded to 8 bits, hence the average processing rate exceeds 13 million bits per second. The system was developed in a program with the Rome Air Development Center.

Extraordinary performance, cost-effectiveness, and growth potential are achieved with modular architecture incorporating an array of 40 CDC FPs, each of which is microprogrammable. Each FP is microcoded to perform a specific computational algorithm which is assigned to it. High speed interprocessor communication and data transfer permit the use of combined parallel and serial configurations yielding optimum system performance.

The system is designed to perform change detection in real time. The two data sets are matched and correlated in a totally digital mode. Then the mission image data is spatially transformed (digitally) to register precisely with a reference image. A difference image is created by subtracting gray-scale values (tonal values) of the mission image from the reference image on a point-by-point basis.

† A pixel is a picture cell or image sample element.

This difference image is further enhanced to emphasize difference detail and to remove noise. A unique feature is false alarm suppression with a feature-oriented processing technique.

Processing is carried out in four change detection processors, each of which is assigned to one of four image channels. There are nine FPs per channel programmed variously for processing algorithms, two FPs for output control, and two FPs for output overhead (a total of 40 FPs). A large digital buffer stores 9 million bits of mission image data.

Processing of image data proceeds in pipeline fashion from the input source, through the various sections of the change detection processor, and to output devices (figure 2-7). Relative image distortions are corrected via a feedback control loop which operates on the memory address. The process is supervised with a CDC SC-1700 Computer. A library disk provides diagnostic and real-time algorithm firmware.

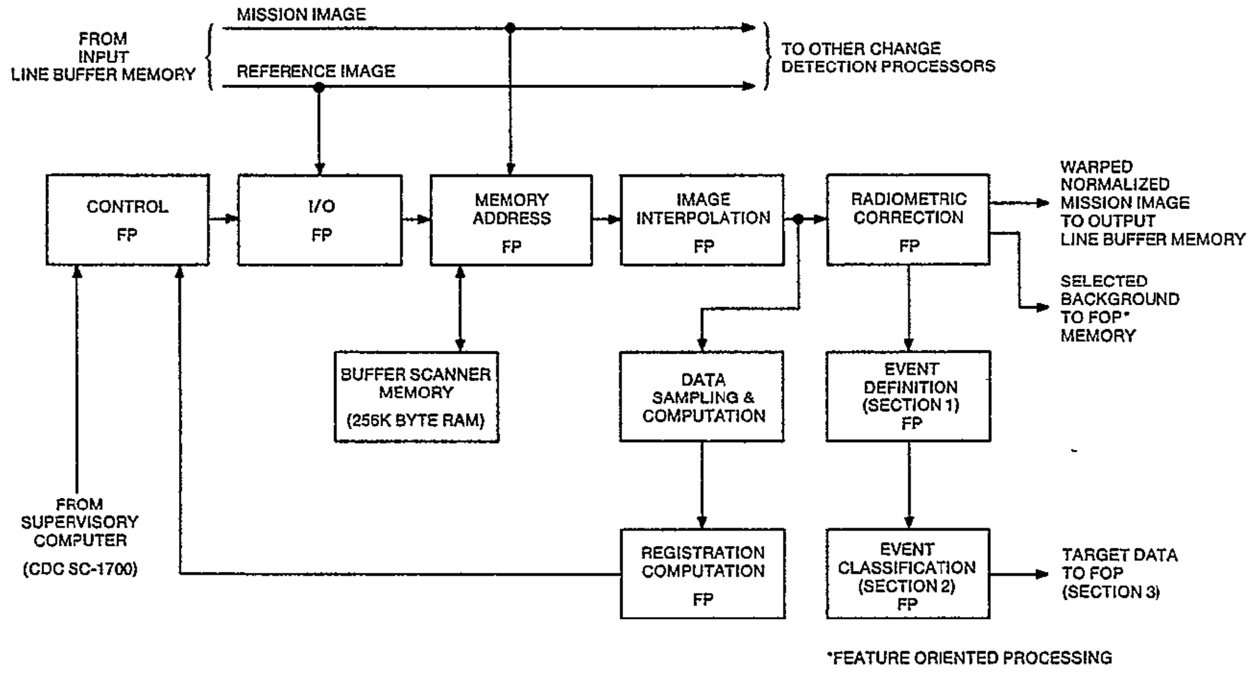

MISSION IMAGE
FROM INPUT LINE BUFFER MEMORY
REFERENCE IMAGE
TO OTHER CHANGE DETECTION PROCESSORS
CONTROL FP
I/O FP
MEMORY ADDRESS FP
IMAGE INTERPOLATION FP
RADIOMETRIC CORRECTION FP
WARPED NORMALIZED MISSION IMAGE TO OUTPUT LINE BUFFER MEMORY
SELECTED BACKGROUND TO FOP* MEMORY
BUFFER SCANNER MEMORY (256K BYTE RAM)
DATA SAMPLING & COMPUTATION
EVENT DEFINITION (SECTION 1) FP
FROM SUPERVISORY COMPUTER (CDC SC-1700)
REGISTRATION COMPUTATION FP
EVENT CLASSIFICATION (SECTION 2) FP
TARGET DATA TO FOP (SECTION 3)
*FEATURE ORIENTED PROCESSING

Figure 2-7. Block Diagram of Change Detection Processor (Single Channel)

Original scene and difference image data can be written on any one of three 9-track high speed digital tape drives, operating at 8000 bpi. These tape units can also be used as input data sources to the system.

Digital Scan Converter Memory System

A digital scan converter memory system provides refresh for a three-color video display system at an average rate of 78 million bits per second. The image frame consists of 512 lines of 640 pixels per line with each pixel encoded to 8 bits. The image is line interleaved and is refreshed at 60 fields per second or 30 frames per second. The system configuration (figure 2-8) includes a CDC FP and external memory which performs the following primary functions.

- Perform controller functions such as memory address computation, load gray-scale table-look-up memory, and so on
- Load and/or modify contents of the scan converter memory
- Translate operator inserted keyboard commands into alphanumeric and graphics overlay data

The capacity of the scan converter memory is 640 pixels per line by 512 lines per frame by 8 bits per pixel. The memory is organized into four banks each having a capacity of 40 kwords (k equals 1024) by 16 bits per word with two 8-bit pixels packed into each word. The number of banks can be modularly expandable up to a maximum of 16 banks. Thus, four independent 640 x 512 x 8 digital scan converters could be controlled by one FP. With some modification, this memory capacity could be extended to a single high resolution display of 1280 x 1024 x 8. In addition, each bank is expandable to a maximum capacity of 64K x 16. Thus, with 16 banks the maximum capacity is 1,048,576 words (16-bits each).

Since the memory is random access, it may also serve a dual role as an extended capacity random access storage for data processing calculations. The memory cycle time for either read or write is 300 ns.

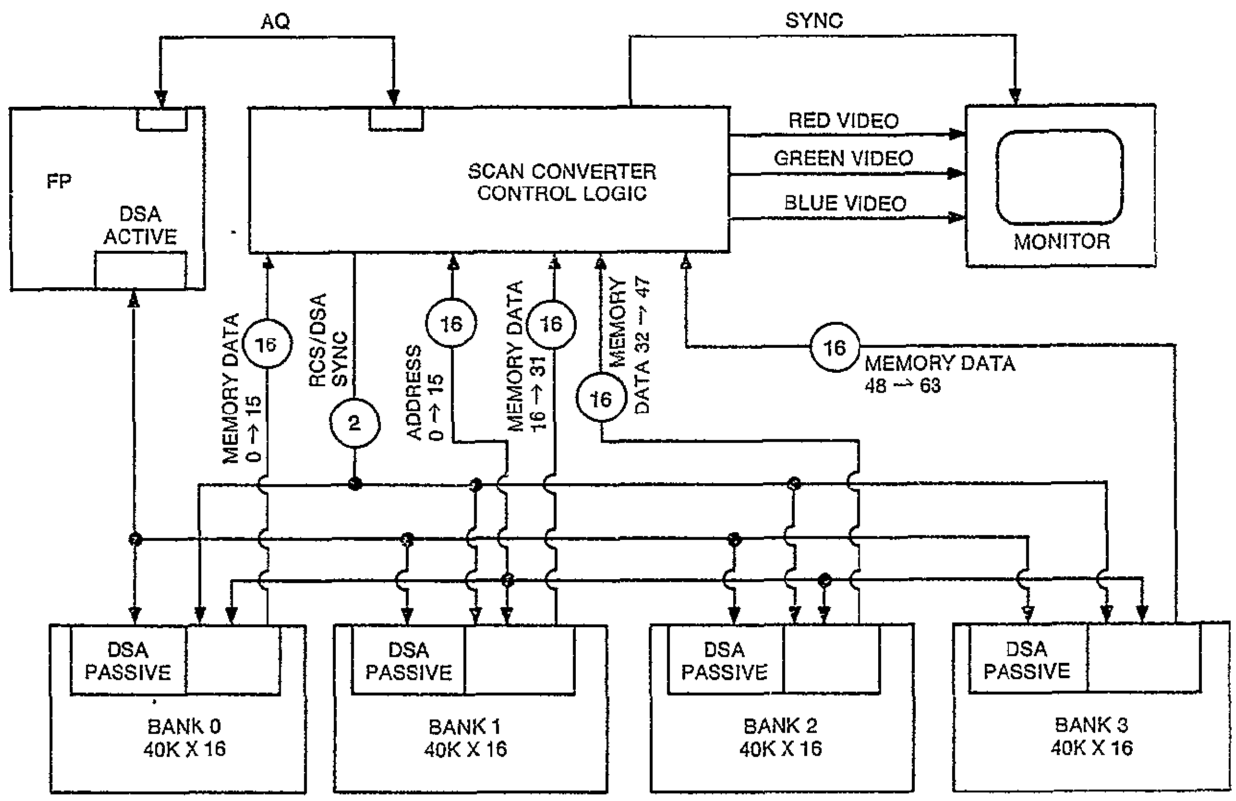


Figure 2-8. Digital Scan Converter Memory System

Keyboard and trackball or light pen inputs are translated by the FP-Processor so that memory contents can be appropriately modified to show an updated cursor position on the CRT display. This allows image pixel interrogation with the cursor which is especially useful for image analysis and interactive editing.

Image zooming and offsetting is implemented by appropriate keyboard entry into the FP.

Level slicing and image feature enhancement is implemented by a table-look-up memory.

Scrolling of continuous strip images into a moving window format from the CRT display can be accommodated by keyboard call of an appropriate FP microprogram.

THE ILLIAC STRUCTURE (SIMD) APPROACH

A review of the problems accompanying a totally independent, parallel processing approach leads one naturally to scrutinize the single instruction stream, multiple data stream (SIMD) approach very carefully. Perhaps the most famous (in terms of publicity and professional publications) example of this approach is the ILLIAC IV (figure 2-9). The underlying motivation for its architecture is as important today as it was in 1963. If, as seen in the MIMD approach.

- For the real computational environment, parallel processes must be put in some form of lock-step to avoid computational errors, and
- Electronic lock-step is desirable to provide well structured access to I/O ports for a process, and
- Reliability and cost are a major function of the amount of hardware in place, and
- Instruction fetching, sequencing, decoding, and checking are a substantial portion of the nonmemory hardware, then:

a multiplicity of instruction streams becomes optional (because of the lock-step nature of computations), becomes expensive (in terms of hardware), and becomes undesirable (in terms of software control and reliability). Hence, SIMD is the better path to parallelism.

These arguments are universally true for all SIMD architectures, but they do not imply a requirement for an array processor of the ILLIAC type.

The array processor concept appears to arise from two aspects, engineering and theoretical.

If one is going to construct a large number of parallel units, then one becomes concerned about engineering economies. These economies are not strictly cost of hardware, but involve considerations such as manufacturability, parts and type count reduction for reliability and maintainability, and interconnect reduction within a processor and between processors.

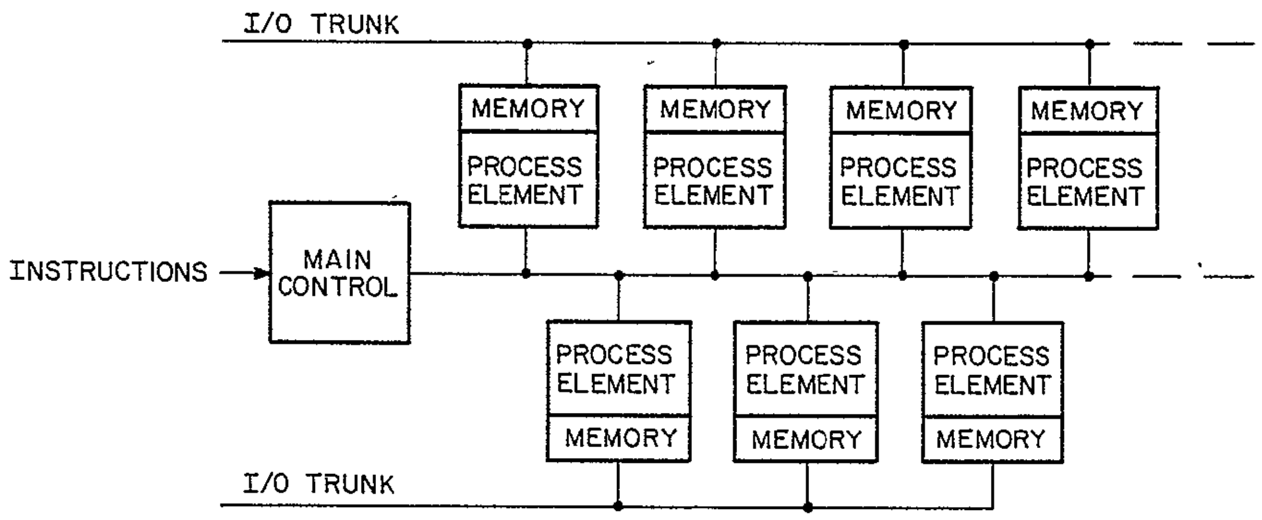


Figure 2-9. SIMD (ILLIAC Structure)

Such economies affect performance as well as cost. For example, if the output of an arithmetic element must be connected to 128 other arithmetic elements, the result data must pass through several levels of fanout (assuming one logic level for each four-way fanout). Each level of logic not only takes circuits which cost money and reliability, but also delay time as data passes through and between each level of fanout. Note that fanouts are wasteful since they perform no useful function on the data but are needed to provide linkage between units of logic.

A competent designer therefore turns to finding ways of reducing interconnects between elements and most certainly between individual processors. Thus designers usually are led to specify local memory for each processor and to minimize connections to adjacent processors as in the ILLIAC IV.

Reducing interconnects, as suggested previously, is possible only if there is a theoretical basis for constraining memory and processor interconnect as severely as it was done on the ILLIAC IV. A great deal of work has been performed in this area to establish the absolute minimal interconnect requirements. The two-dimensional array network manifested in the ILLIAC IV is a result of this theoretical effort. A single instruction stream, interpreted by the main control unit, directs the synchronous and simultaneous behavior of 64 identical arithmetic processors, which can either be computing the same function as all other processors or can be idle.

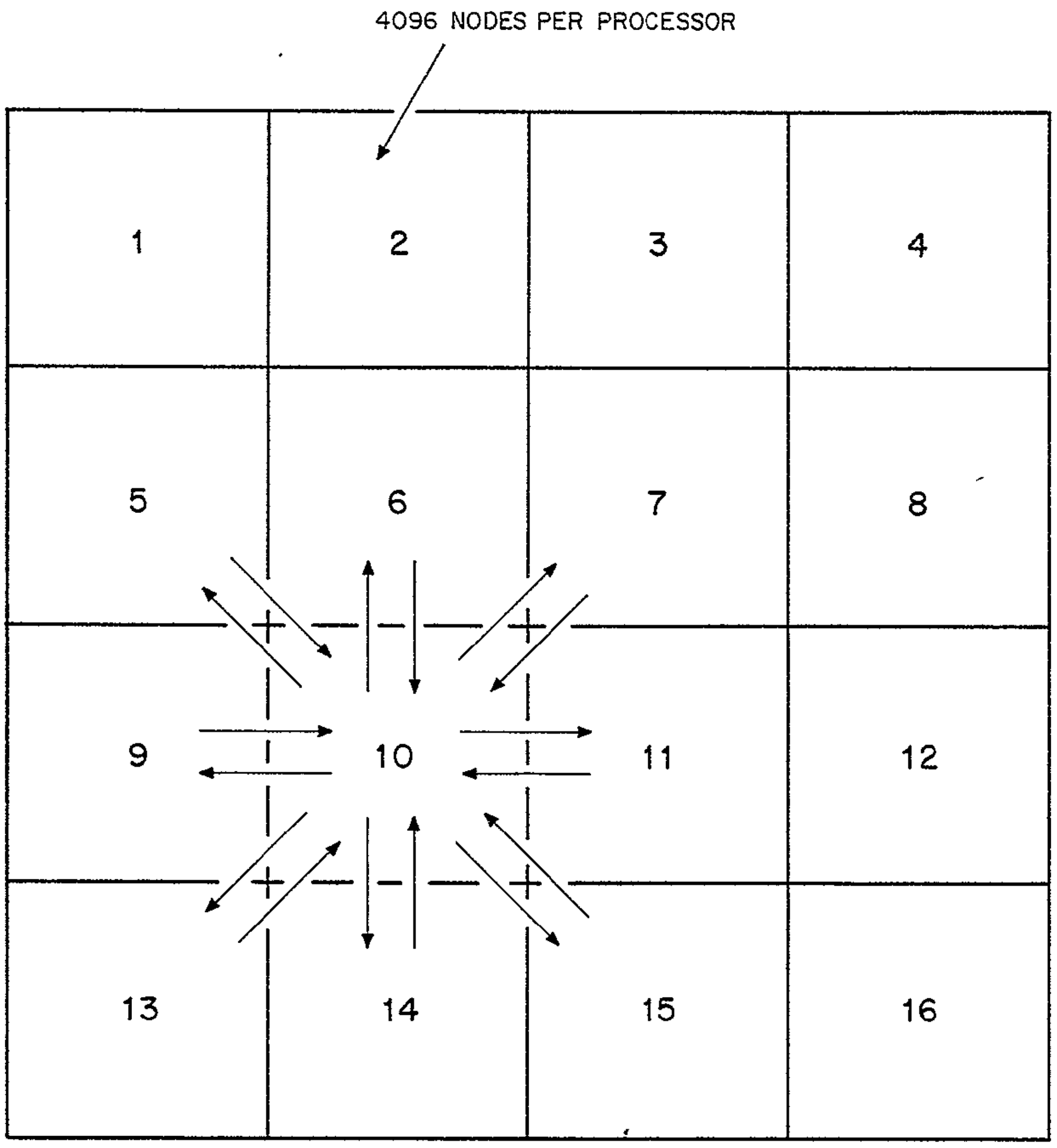
The major drawbacks to the ILLIAC IV have been its limited interconnectability, its limited memory per processing element, and consequent poor programmability quotient. These limitations are very important and should be examined more closely.

Interconnect of Hardware, Interaction of Data

In all the cases considered so far, whether configurations of independent minis or synchronous array processors, one consideration dominates the architecture. That is the need for data and computational results to interact at a very high rate. For example, if a task is to perform the multiply of two linear vectors X and Y many times (Z=X*Y*X*Y*X*Y), and each vector was 16 elements long, then 16 processors could each perform the operation many times for its own pair of operands, X(I) and Y(I). If this represented the totality of the computational load, the arrangement would be very effective, and no processor would need to share or communicate data with another.

In real life, however, few calculations of this type are encountered. Instead, after a period of computational work (called a quantum), one processor must trade some data with other processors. For example, the explicit solution of the Navier-Stokes equation involves the computation of quantities (pressure, velocity, and density) at a node as a function of data at surrounding nodes.

Thus a collection of processors (figure 2-10), each given a set of initial conditions for all nodes surrounding it, can compute for a time, producing a new set of values. At that point, however, all quantities computed must be passed to other processors which will use the new data for the next time step to compute new values for their nodes. Assume that each of 16 processors is calculating data for a block of 4096 nodes, for a total of 65,536 nodes. Further, assume that new values are calculated for eight characteristics. At the end of each time step (quantum), data for 512 of the points must be distributed to all



16 PROCESSORS

DATA SWAPPED EACH QUANTUM BETWEEN PROCESSOR 10 AND SURROUNDING PROCESSORS 5, 6, 7, 9, 11, 13, 14, AND 15 = 4 X 10^6 BITS

Figure 2-10. Solution by a Collection of Processors

processors computing nodes at the boundaries of the 4096-node block in the given processor. In addition, all boundary processors must swap data back into the given processor to provide new values for computing the next time step.

If the problem structure is such that a contiguous block of 4096 nodes is held by one processor, then a processor holding central nodes would have to communicate with eight other processors, one at each face of the square block of data and one at each corner. The data bits (assuming a 64-bit floating-point number) to be exchanged per processor are 512*2 (for two-way interchange)*8(characteristics)*64 equals 524,288 bits. The total I/O bandpass for a single processor will be needed to transfer eight of these 1/2 million bit quantities every time step, or a total of over four million bits for each quantum of work.

The total time for a time step becomes then $t=Q+4*10^6/IOBW$. As the quantum of work (Q) that can be done before an exchange of data is necessary becomes smaller and smaller (as in a turbulent flow model), the input/output bandwidth (IOBW) becomes the limiting factor on actual computational speed. This notion is called problem bandwidth and demonstrates the fallacy of measuring only the raw compute horsepower of a system, whether parallel or serial. In this example, a 10^8 operations/second processor with a 16-Mbit I/O bandwidth would require 1/4 of a second for data trading while yielding 6103 results per node (1/4 sec x 10^8 operations/sec x 1/4096 nodes equals 6103 results/node) of computational power in 1/4 second.

Limited Memory Per Processor

There is no question that if the amount of directly addressable memory can be kept small, the hardware interfaces can be greatly simplified and random access memory speed can be maximized. This approach, however, is at the cost of overall problem efficiency. For example, in the 4096-node case previously described, each processor needs at least 32K to store the data points. There would be insufficient memory in an ILLIAC IV to handle all characteristics of all data points. Sixty-four processors could handle 1024 nodes with fewer characteristics perhaps, but in general some of the data must be held in secondary storage. A complete time step then not only requires an exchange of data between processors, but must include one or more exchanges of data with the secondary storage device, further reducing the effective problem bandwidth.

Programability Quotient

All of the computer architectures discussed (configuration of minis and ILLIAC) suffer miserably from programming arthritis, an ache in every joint or interface. Because most of the architectural models have been driven by engineering expedience, rather than philosophical ground rules, it is difficult to approach a problem solution with them in a consistent, non-ad-hoc fashion. In fact, optimum solutions in the main have become tailored, hand-tuned exercises for even the best of the lot.

This is primarily because the instruction set provides no high-level specification of the parallelism desired, and because engineering simplifications (such as small local memory and restricted interconnects) have placed artificial boundaries around the computing resource that make even handcoding difficult.

There is no reason why an ILLIAC structured machine could not be implemented in such a manner as to enhance programmability. As long as the SIMD structure is maintained, a compiler should be able to produce reasonable optimum code if the machine does not have the constrained memory size and connectability of the ILLIAC IV.

VECTOR PROCESSING FOR THE SIMD APPROACH

Two SIMD style machines that were based on philosophical models are Texas Instrument's ASC computer and CDC's STAR-100 computer. The former uses the FORTRAN language as its basis, implementing in vector form the three-level DO loop of FORTRAN. The STAR-100 was conceived around Kenneth Iverson's A Programming Language (APL) philosophy of problem solving (reference 2-9).

Aside from their basis in philosophical constructs, these two machines share other attributes. Arithmetic processing speed is achieved through pipelining (figure 2-11), wherein a new set of results can emerge from an arithmetic unit every clock cycle despite the fact that the arithmetic operation itself may require several clock cycles to complete. This is accomplished by accepting two new input operands every cycle and stepping them through a series of segments in the pipeline where the partial arithmetic operation is performed every minor cycle. The totality of processing is therefore accomplished at several stages with new data at each stage, much like an automobile assembly line.

Another common attribute is that the interprocessor interconnect occurs via memory references rather than hardware interconnect between the processors. Thus, the 4 32-bit pipelines of the ASC and the 4 32-bit pipelines of the STAR-100 receive their data from memory and replace results in memory. Any new calculation can access any result from any previous calculation since all data is stored in a homogeneous, universally accessible memory.

Optimized programming, while still requiring a rethinking of the problem solution algorithms, is made easier since there are no artificial barriers such as small units of discontiguous memory and special interconnections with which to deal.

Machine instructions provide higher level of control or description of parallelism. As an example, consider programming a matrix multiply operation. In the mini configuration discussed previously, one must devise a scheme to fragment the matrix data among the individual processors, construct a program out of basic load, store, add, and multiply instructions for the processors, create a control protocol for starting up and shutting

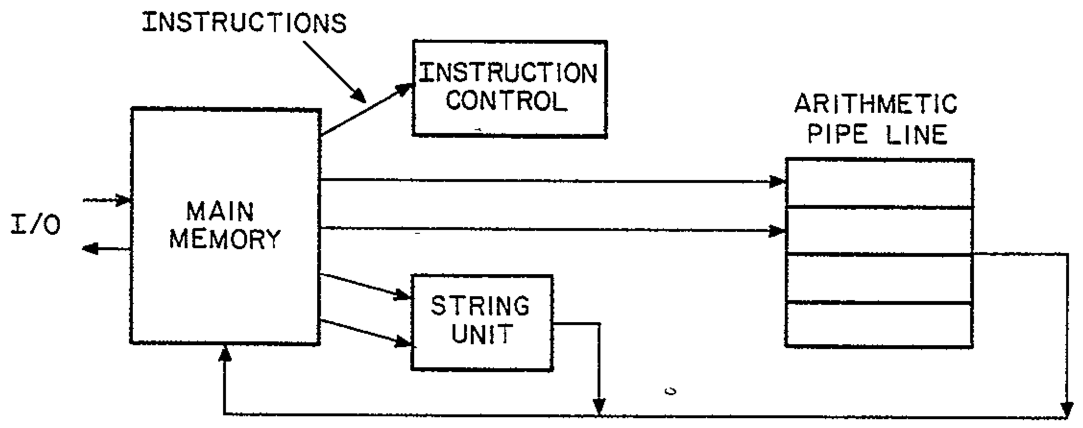


Figure 2-11. SIMD (STAR Structure)

down, and develop a process for passing to one of the processors the partial add and multiply products, so that a final sum can be formed. If the amount of data exceeds the local memory available, a whole new level of program complexity must be introduced. In the STAR-100 computer, a single instruction is used (inner product) to form the elements of the new array. Input data can reside anywhere in memory, and depending on the machine configuration, anywhere from one to eight arithmetic units could be employed in computing the inner product, completely invisible to the programmer.

HYBRID COMPUTING

One of the alternatives proposed for supercomputing has been the marriage of a high-speed digital computer to an equally high speed analog computer. Since the solution of the Navier-Stokes equations is meant to yield simulated flow field results, which are in themselves analogs of the real-world airfoil being modeled, the use of an analog computer seems quite natural for the computation engine. One of the main arguments made by analog proponents is the elimination of false solution hunting which has characterized past digital partial differential equation (PDE) solvers. While it is true that the analog system can be set up to constrain the solution to the domain of interest, it is not true that analog computers will not also go on wild goose chases pursuing imaginary or uninteresting solutions.

A review of the current state of analog/digital hybrid yields the following observations.

- The degree of precision required for the Navier-Stokes solver is beyond that available in known analog, digital hybrids. This is due to the transducing problems of converting analog-to-digital and digital-to-analog signals with the extreme accuracy needed by the digital machine to complete the solution.

- The assemblage of parts for a complete Navier-Stokes engine would be quite enormous, particularly as complex geometries and boundary conditions will be required for production jobs.

- The strapping together of the hybrid components will necessarily be fairly customized, with a great deal of logic, hardwired, so that the analog machine can meet the performance requirements. It appears that the variable setting up of integrator, differentiator, and data path networks probably cannot be done by program control from the digital machine because of these performance constraints.

- Even with a great deal of hardware wired together, problem setup could be quite substantial in terms of programming, hardwiring, and debugging.

- The examination of intermediate results of a given solution (such as a single Jacobian at time step 100) quite often gives the problem developer a great deal of insight into the behavior of that model. Such intermediate data is sometimes not existent in the analog system or not readily available to the outside world, because it is contained in hardwired paths imbedded deep within the analog machine.

The state of the digital computer and solution methodology compared to the corresponding hybrid technology leads to a recommendation for a purely digital approach to a machine to be fully operational in 1982.

To sum up this discussion on CPU architecture for solving the Navier-Stokes equations for flow field modeling, a parallel architecture is dictated and an SIMD structure is indicated as the best approach for performance and programmability.

MEMORY ARCHITECTURE AND ADDRESSING

Of the several architectural approaches discussed, one alternative consists of providing interprocessor communication via a large, high bandwidth memory system to which all processors are connected. Since this memory system is a key element in the more attractive proposals for the NASF, it is appropriate to discuss the memory system operation in more detail. To this end the following discussion is provided on memories and memory access mechanisms, beginning with the smallest building block (the memory chip) and proceeding outward to the total memory hardware complex.

Memory Chip

Figure 2-12 is a schematic of a typical bipolar memory chip available in 1977. It contains connecting pins for transmitting data from the chip, data into the chip, address data, a read/write signal (indicating in which mode the chip is being accessed), and a signal called chip select. Herein lies the first lesson in practical memory building. The chip is constructed according to standard manufacturing practices and occupies approximately 100 mm^2 (0.16 in^2) of circuit-board space. With normal manufacturing techniques, this permits approximately 16 pins to be used for connecting the chip to the circuit board. Thus employed, 10 pins are needed to address 1024 locations on the chip, with four more pins used for one bit of data in and data out: the two control signals and two signals for power and ground. Thus, there are limitations on how much memory can be packaged in a single location regardless of how much density can be achieved on the silicon chip itself. If this standard package is retained, certain techniques (such as multiplexing address bits) can be used, but these normally affect performance adversely. The alternative is to utilize a newer package (such as one with 24 pins) which has more connectivity, at the cost of more circuit board consumed and more complex routing patterns for signals. This problem of connectivity continues to plague the design at levels of the system, chip, bank, module, and system as will be seen later.

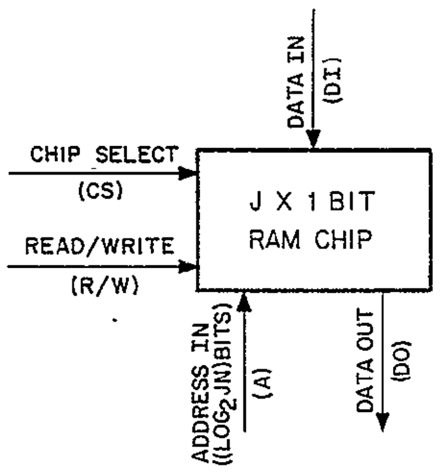


Figure 2-12. Single Memory Chip

In this simple example, the function of the chip select is of no use since access is to only 1024 bits on a single chip.

4-Bit Memory

A single-bit memory system is rather ineffective for differential equation solution since a word size of 32 bits is the minimal amount desired for the mathematics involved. Thus, the engineer must organize the single chips into aggregates that can be addressed as words. Figure 2-13 shows a typical assemblage of the basic chip into a 4-bit wide memory system, with J words of data contained therein. Note that the address, chip select, and read/write strobe are fanned out four ways, once to each chip. In this case, if this were the extent of the memory required, all chip selects could be biased true since all of the chips will be active all of the time.

It can be assumed that the four output and input data bits are being transmitted to some processor via a bus where some useful work can be done with them.

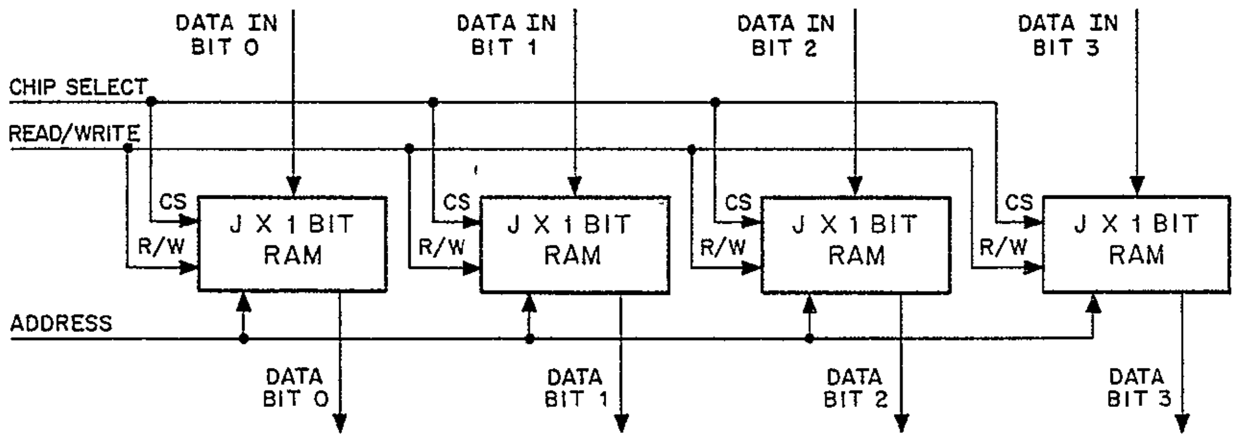


Figure 2-13. 4-Bit Memory by J Words

<u>2J by 4-Bit Memory</u>

For most applications, of course, 4-bit words are not sufficient, but it should be obvious that the word width can be extended infinitely depending on the fanout requirements for the various control and address signals. For example, some high performance groundrules prohibit fanouts greater than eight per driving circuit. In such a case, there is a need to interpose an eight-way fanout circuit into the system for each control and address signal, with each leg of the fanout attached to eight chips.

If the memory is to be expanded in the other dimension, however, additional logic is required in the system. Figure 2-14 shows a network which is capable of storing 2J words, each 4 bits wide. This is accomplished on the read side by attaching both data outputs for a single bit (one from each chip) to the same bus. The active chip select signal then determines which of the two chips for that bit will have its data on the bus. Note that one bit of the incoming address is channeled to the chip select and not sent to the address inputs of the storage chips. Later, the counterpart of this scheme can be seen for the total memory system, wherein the chips are replaced by large modules of memory.

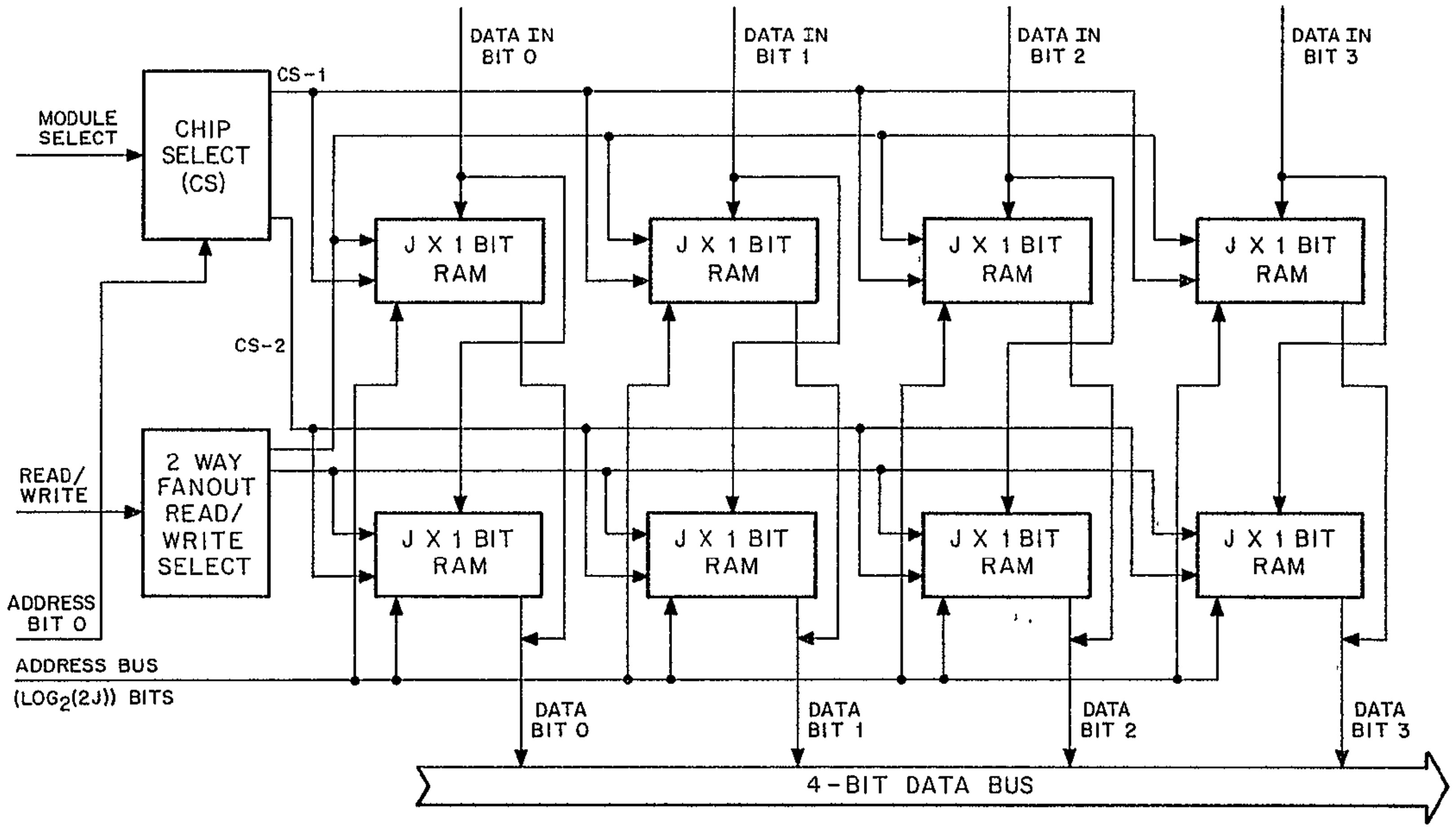


Figure 2-14. 4-Bit Memory by 2J Words

In this example, it can be assumed that after a certain number of nanoseconds has elapsed following the transmission of a stable address to the network, the correct data at that address is available on the 4-bit data bus shown at the bottom of figure 2-14.

A Single Memory Module

The network discussed above can be extended into a module (figure 2-15) with width, M, (number of bits delivered in a cycle on the bus) and depth, N, (number of words of bit-width M stored in the module) until practical limitations such as transmission line lengths for the busses (data and control), fanout circuit requirements, packaging, and cooling become dominant in the design. The fact that a maximum practical size exists for a given storage device technology is an important part of this discussion and will be reiterated many times.

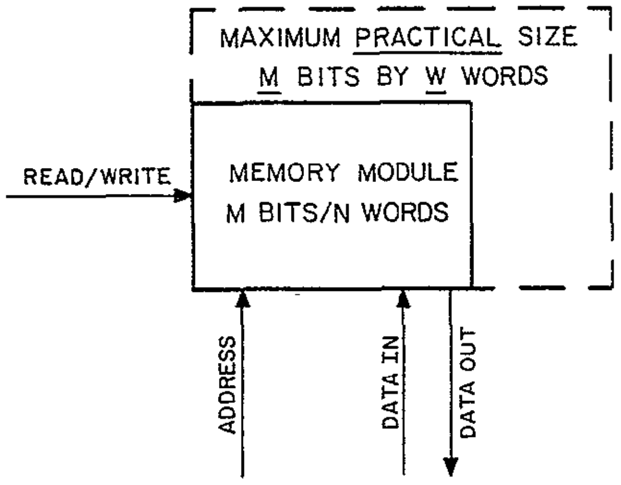


Figure 2-15. Single Module Of Memory

If the computer or processor needs only the size memory which can be housed in a single module, and if the cycle time of the computer clock is matched with the memory, then the system is in balance, a desirable or necessary system aspect discussed later in this section. This means that the CPU performance is not limited by the memory performance. For example, popular microprocessors based on the Motorola 6800 system can operate with a clock cycle of 500 nanoseconds. In this mode a simple instruction takes 1 microsecond with a memory that can be accessed in less than 500 nanoseconds. Speeding up either the memory or the processor by itself will not affect the performance of the system since they are so well matched; both memory access and processor speed must be improved by the same amount to yield any noticeable gains in system performance.

As seen in figure 2-15, simple interconnects are required between the processor and the memory module when a single module is sufficient. A direct connection exists between the data busses and the processor busses. The address and read/write lines are also directly connected. This idyllic state can continue until the user of this system discovers that W words (the maximum allowed due to physical constraints) are no longer sufficient for his problem solution. This situation, of course, is more typical for the supercomputer user than for any other varietal. In this case more memory modules must be added to the system.

M by 2N Memory on a Bus

Figure 2-16 shows two modules (there could be more) attached to the processor on a bus. In this system the processor places an address on the address bus which is sampled and decoded by each module. The module containing the data is selected by decoding and places bits on the output data bus to be transmitted to the processor. This bus structure is commonly found in current microprocessors such as the 6800 and 8080. As stated previously, when the memory and processor are matched in speed, this system is quite

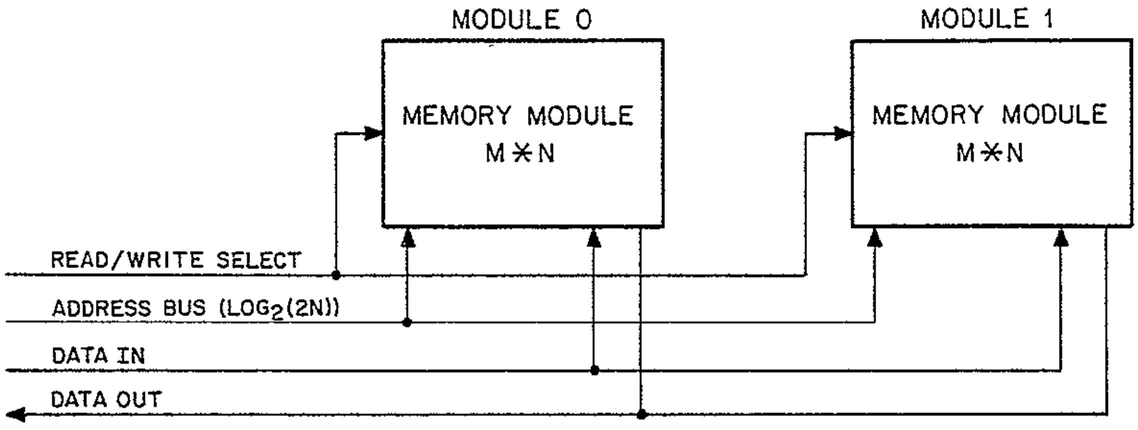


Figure 2-16. M by 2N Memory via Bus

adequate for interconnecting memory modules. Obviously, a physical limitation exists for the number of modules which can be simply connected to a bus without adding hardware for fanouts and drivers.

As the amount of memory required in each module and the number of modules required to provide additional storage grow, the memory access begins to slow down. That is, the raw access speed of the individual chip becomes submerged by the circuit delays of the supporting logic as it grows more complex, as seen in figures 2-12 through 2-14. For supercomputers, meanwhile, technologists are aggressively reducing the processor logic delays to the point where a 10-nanosecond clock cycle is possible in the arithmetic logic of the computer. Therefore, increased memory capacity and increased CPU logic speed tend to move each of these components away from the other in speed. The memory gets slower and the processor gets faster, while, at the same time, parallelism in the processor (introduced to further improve performance beyond that which can be yielded by technology) further widens the gap between the memory and processor performance. The result in this case is a mismatch in performance which can be solved by arranging the memory so that more than one operand can be retrieved in an access.

An examination of the bus structure shows that only a single operand can be transferred on the bus per clock cycle, making it necessary to develop a different interconnect scheme for the supercomputer environment. The alternative is to retain the bus structure and develop a multiplexing scheme which can put more than one operand on the bus in a clock cycle. This implies the presence of circuitry (to handle the short burst of pulses) which is faster than the logic family used for the processor, particularly as the number of operands transmitted exceeds two. For example, if the fastest logic family can operate at a clock rate of 10 nanoseconds, multiplexing four operands on the bus during a single cycle requires circuitry which can clock and latch data every 2.5 nanoseconds.

The interesting question arises in this instance as to why such circuitry is not employed in the processor to speed up the CPU clock. Taking this argument to its rediculous conclusion, one can see then that the CPU would possess a clock cycle of 2.5 nanoseconds, thus requiring four operands to be transmitted at the rate of 0.6 nanoseconds each. This in turn would require the development of a circuit family capable of operating at this rate, which would lead to the family being used in the CPU logic, thus reducing its clock cycle, and so on.

M by 2N Memory Access Via Interchange

The bus structure thus being abandoned for the supercomputer main memory, alternative schemes are considered. One of the most common is the connection of two or more memory modules to the processor via a centralized interchange unit, as shown in figure 2-17. In this situation, the memory interchange unit receives an address from the processor. By examining one or more of the address bits (usually either the high- or low-order bit), the interchange then requests the appropriate memory module, transmitting the appropriate address (less the one or more address bits used by the interchange to determine which module is active). The memory interchange, not being bus organized, requires logic for fanout and fanin of data and control signals. The data trunks from

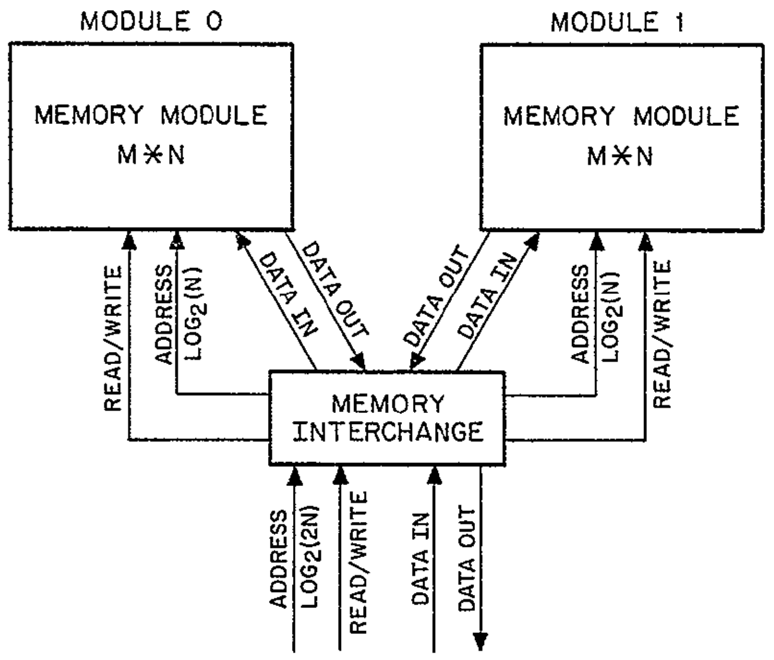


Figure 2-17. M by 2N Memory Access via Interchange

each module are merged within the interchange unit, and then fed to the processor. This implies not only fanin and fanout but selection logic, and perhaps some amount of buffering to be designed into the interchange unit. Depending on the method of accessing the attached modules, the interchange unit can become a complex logical entity.

Linear Select - No Banking

Several direct ways can be used to employ the interchange unit. In the simple example given in figure 2-17, all the data in memory module 0 (the left-hand block) could be addressed sequentially from address zero to address N-1 (words). Then module 1 (the right-hand block) could contain sequentially addressed words from address N to address 2N-1. In this case, only one operand would be delivered per clock cycle (normally). Figure 2-18 is a somewhat simplified version of figure 2-17 and shows another method of

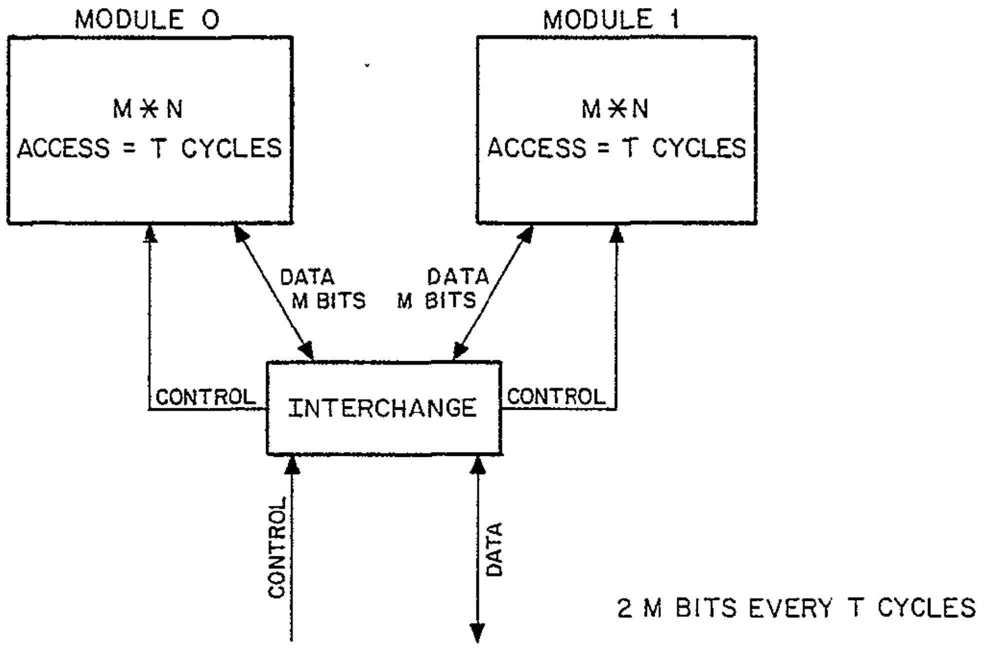


Figure 2-18. Linear Select - No Banking

access of the same physical modules. In this case, a single address is sent to the interchange unit from the processor. This address would actually be a reference to an operand pair. The interchange unit would then transmit the control and address signals to each module, the address bits being identical in both cases. After an access time T (the number of clock cycles needed to access the data in the module and return it to the interchange unit), both modules would deliver M bits (one operand is M bits) to the interchange unit. The interchange unit would then combine the two operands into a single 2M-bit wide stream for transmission to the processor. Thus, in one memory access time, T, 2M bits can be acquired. This scheme has been termed linear select since the operands are taken from adjacent addresses (N and N+1), and a uniform address is thus sent to both modules. In most memory systems of this type, the memory access time T is an integral number of CPU clock cycles. The terminology normally employed to describe this clock relationship is that the CPU clock cycle is called the minor cycle or fastest possible cycle, while the memory access time is called the major cycle. For example, in the STAR-100A computer with bipolar memory, the CPU clock cycle is 20 nanoseconds while the major cycle is 80 nanoseconds or 4 minor cycles.

In the following schemes, the portion of memory being accessed cannot receive another address or retrieve another piece of data until the completion of the access time T. Thus, the affected memory unit is busy for T minor cycles. For bipolar, static memory systems, the memory busy time is equal to the access time T. In the example in figure 2-18, the interchange unit and the memory modules will be busy for T clock cycles, while the CPU normally could make a new reference every clock cycle.

Linear Select - Two Banks

The memory could also be accessed as shown in figure 2-19. Here, instead of returning 2M bits every T cycles, the memory interchange is capable of receiving and responding to two different requests, usually appearing at consecutive minor cycles on the processor address lines. For example, module 0 might contain bank 0 which houses all even addresses (0, 2, 4, and so on) while module 1 contains bank 1 with all the odd addresses (1, 3, 5, and so on). Thus, on clock cycle 1 an address could be sent to the interchange unit. If this address is even, the appropriate control and address signals would be sent to module 0, and T cycles later the data would appear on the trunk (M bits wide) to the interchange unit. Meanwhile, during clock cycle 2, a different address could be sent to the interchange unit by the CPU. If this address is an odd address, it could then be sent to module 1 with the appropriate control signals. In this case the resulting data would arrive at the interchange unit one clock cycle after the data transmitted from the first request to module 0. Thus, in T+1 cycles, 2M bits would have been retrieved from memory. As long as the two addresses do not affect the same bank, this rate can be sustained. The major difference between this scheme and that of figure 2-18 is that the addresses need not be contiguous in the second method, as long as they do not conflict in their bank access. For example, if the second address received by the interchange unit turned out to be even, as was the first address, then the interchange unit cannot transmit it to module 0, which is busy for T cycles getting the first data requested. Thus, the system, in this instance (bank conflict), can yield only M bits every T cycles. However, for linear select, this scheme yields M bits every T/2 cycles (no bank conflict). In a multiple processor configuration, the memory bandwidth, the rate at which data can be

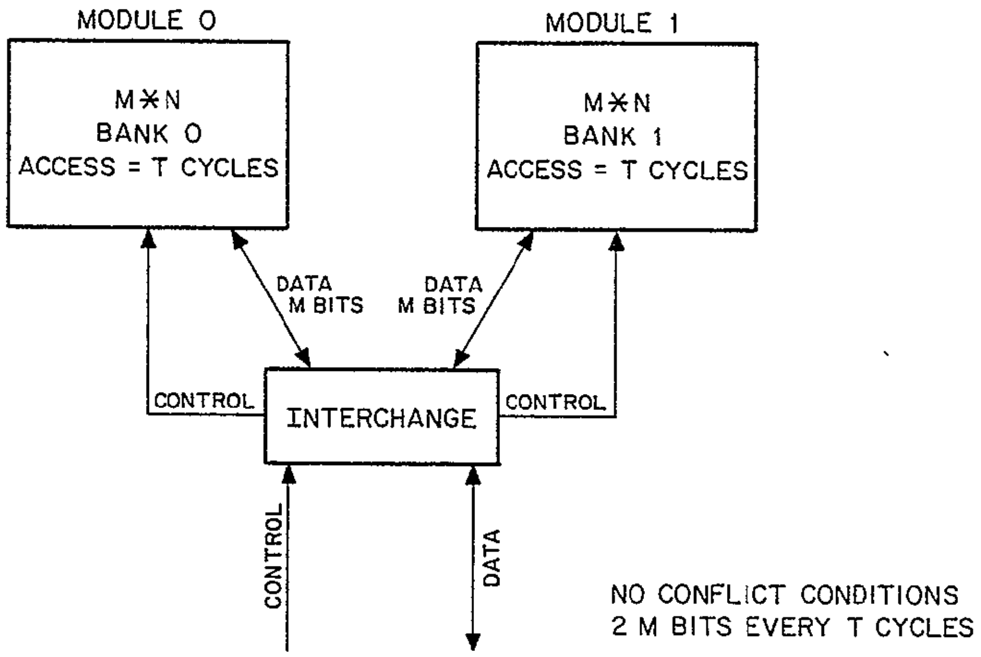


Figure 2-19. Linear Select - Two Banks

delivered from memory to the processors and later returned, becomes a significant factor on total system performance. Thus, concern arises about access time for random access, unstructured memory references (typically invoked by scalar or nonvector code), and bandwidth (for structured accesses) where parallel functional units are employed for improved performance.

Nonlinear Select

One way of improving the bandwidth from M bits every T cycles to 2M bits every T cycles for noncontiguous addressing is to provide separate address and data trunks for each of two M-bit paths. Thus, in a single minor cycle, two addresses would be sent to the interchange, which would determine if the two were in conflict (that is, referencing the same physical bank of memory). If the addresses are in two separate modules, the interchange would pass them on, using the appropriate addressing trunks. At time T, the data from each module would arrive at the interchange and a rate of 2M bits every T cycles would be achieved.

In such cases, several logical solutions must be devised to overcome the problems imposed by parallel address and data transmission, and the possibility of bank conflicts occurring.

Using the scheme given in figure 2-20, if both addresses sent in the same minor cycle were references to the same bank, the interchange unit must hold (buffer) one of them (after making a judicious choice of the candidate) and wait T cycles before transmitting it to the same module as the first address. In all probability, such a system would also provide a holding buffer for the first M bits of data, which arrived at the interchange from the memory at time T, until the second data from that same module arrived at time 2T. The two data quantities would then be sent to their respective data trunks. The data trunks are related to the address trunks and not the module addressed. This means that an address on control 1 trunk would result in data on data 1 trunk regardless of the module actually addressed.

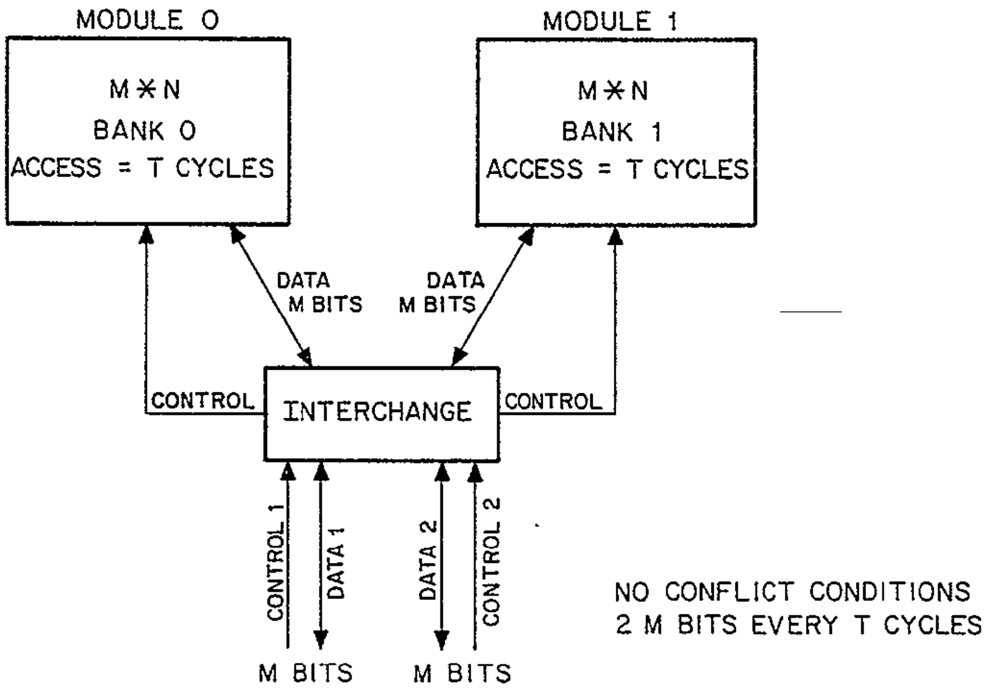


Figure 2-20. Nonlinear Select, Two Banks, Two Data Trunks

Linear Select - Four Banks

The concept of banking, which provides for a more frequent request rate from a slower memory by a faster processor, can be extended to a system of several banks. Although bank electronics yields a measurable overhead in circuit delay and cost to the memory system, the effect is negligible in both areas, representing less than a 5-percent penalty. As the number of banks increases, however, so too do the number of wired connections between the banks and the interchange unit. At some point, such wiring becomes prohibitively expensive to manufacture and maintain. Still, the desire to provide a sufficient number of banks is quite attractive for supercomputers. Consider the following reasons.

- Given a supercomputer with a three-address vector processor, whose arithmetic units can produce a new result every minor cycle and thus require a new set of operands every minor cycle, a balanced memory system would require a bandwidth of three operands per minor cycle (one write operand and two read operands).
- Assume that the memory access time is 4 minor cycles.
- A single bank system would then yield only 1/12 of the desired bandwidth.
- If the memory could be subdivided into four banks and all references thereto ensured to be sequential and nonconflicting, the resulting system would then be 1/3 the required bandwidth.
- Since powers of two are nicely atuned to binary-oriented computers, the desire is to increase the banking to the next power of two which would achieve the desired memory bandwidth (2^4 equals 16 banks).
- Assuming no conflicts and sequential, single-stream access, a 16-bank system can provide the three operands every minor cycle, while offering an extra operand possibility each minor cycle to cover the need for input/output and instruction fetching.

In some instances, the separate interconnect of 16 banks may be impractical due to the mechanical design and packaging system. Thus, several banks might be assigned to the same module. Figure 2-21 shows a system wherein each module contains two banks. In fact, the internal connection of these two banks within the module appears much as the schematic discussed in figure 2-16, where modules in this case become banks, and the common address, data, and control busses then become the module's interface to the interchange unit shown in figure 2-21.

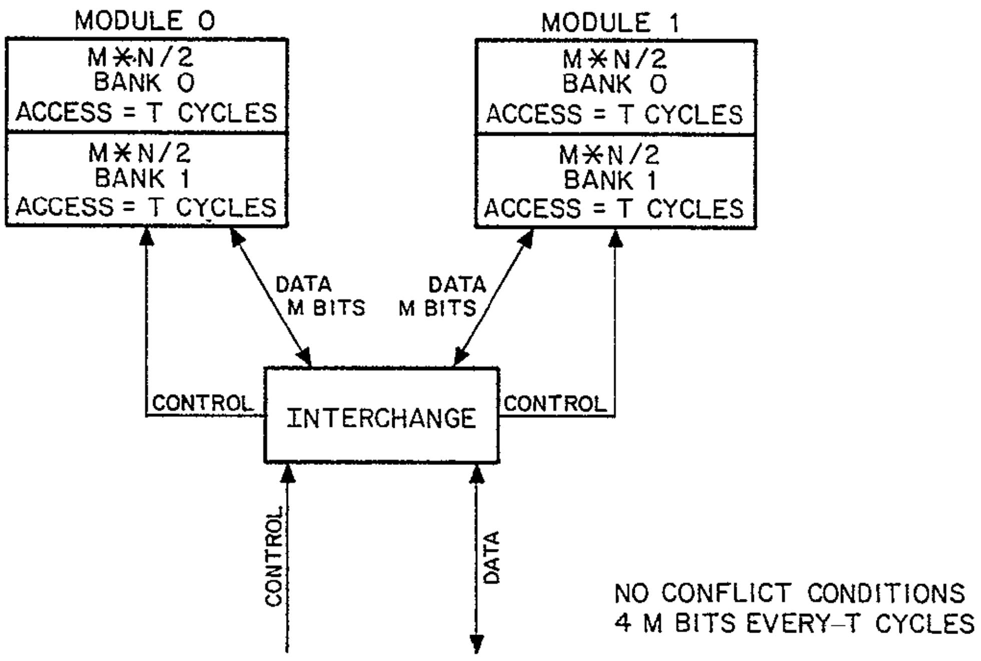
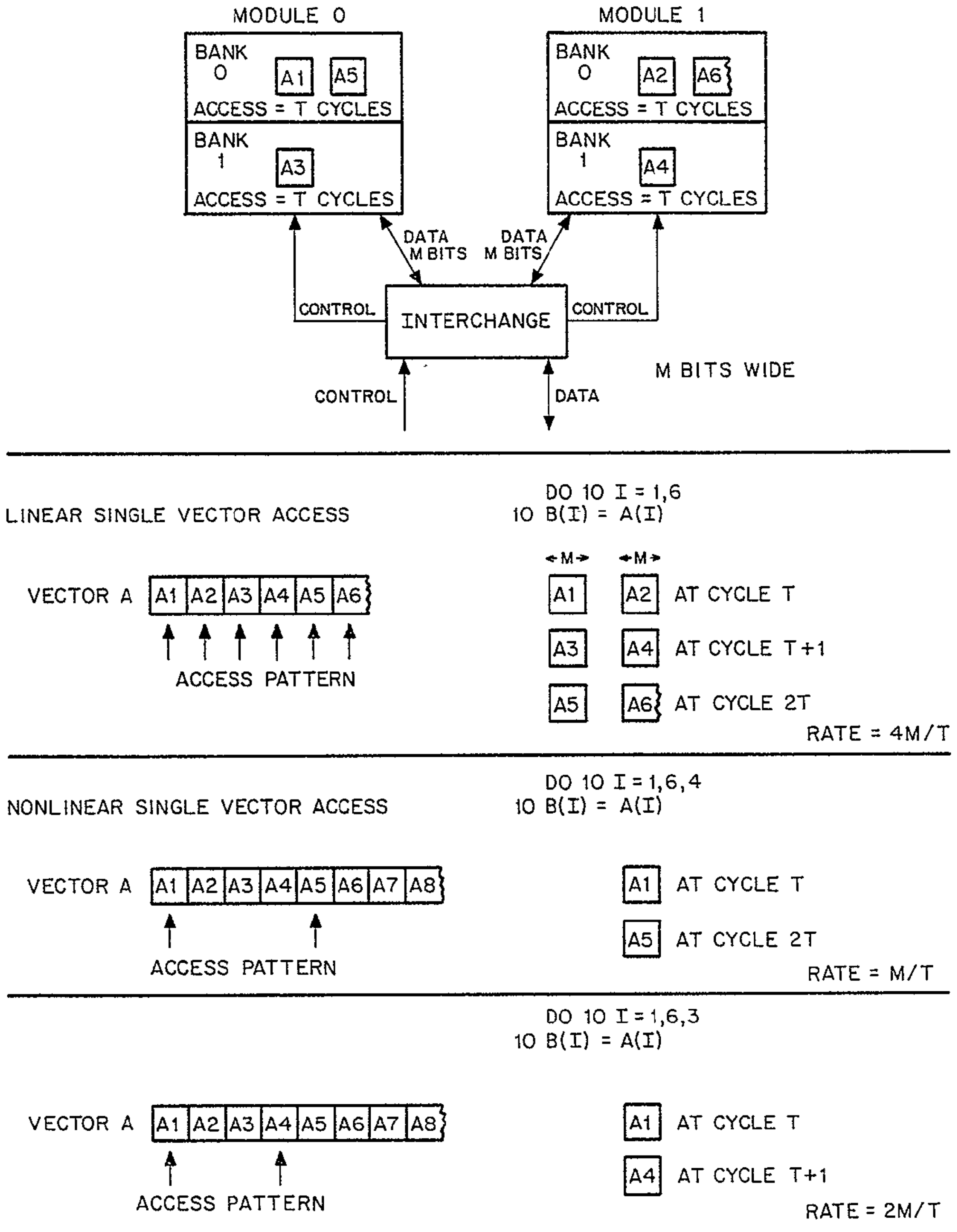


Figure 2-21. Linear Select, Four Banks, Two Modules

If the system in figure 2-21 can respond to four consecutive, nonconflicting memory requests at the rate of one per minor cycle, it can then achieve a data rate to the processor of 4M bits every T cycles. However, an interesting phenomenon occurs with this arrangement. In addition to the problem of bank busy, the system is now confronted with module busy. This means that a given module may not receive two addresses or transmit or receive new data in a given minor cycle. Thus, various forms of addressing can create interference in the rate of memory access. This effect is shown in figure 2-22.



```
   DO 10 I = 1,6
10 B(I) = A(I)
```

```
   DO 10 I = 1,6,4
10 B(I) = A(I)
```

```
   DO 10 I = 1,6,3
10 B(I) = A(I)
```

Figure 2-22. Memory Access Patterns

Memory Access Patterns

To discover the behavior of these memory schemes, it is useful to analyze the real methods in which they are utilized and the problems that arise from various computational techniques. Figure 2-22 shows examples of short vectors. A vector is an ordered set of numbers, stored sequentially in the memory of the computer. It is the orderliness of such vectors that permits the hardware architect to create parallel structures for increasing performance. This is in contrast to scalar (non-vector) programs, wherein the quantity and mode of computing is unknown, or random, and varies with each minor cycle of the CPU. In most cases, however, operations on vectors are uniform across all operands and normally involve several elements of the vector, thus ensuring a predictable amount of simultaneous operation.

The first example in figure 2-22 shows a vector A with elements A1, A2, A3, A4, A5, A6, and so on.

In a traditional banking scheme, the elements would be stored as shown with A1 beginning in bank 0 and A2 in bank 1. If the access method is linear select with banks from both module 0 and 1 capable of supplying data in the same minor cycle, elements A1 and A2 could then both appear on the data trunks to the processor at time T, A3 and A4 could appear at time T+1, and A5 and A6 could appear at time 2T. The rate is then 4M bits every T cycles.

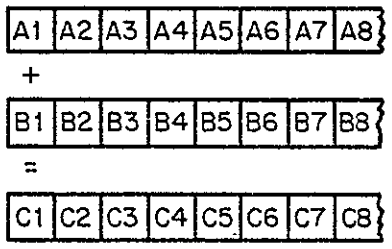
In some applications, the vector A is not always addressed sequentially. Typical of such situations is the sweeping of computations down the rows of a matrix instead of the columns. This access pattern occurs in some of the computations for the implicit and explicit Navier-Stokes solving codes under development by NASA. The nonsequential nature of referencing the vector A is illustrated in the second example in which the DO loop essentially retrieves every fourth element of A. With the vector stored as shown, this means that all references will be to bank 0, thus imposing a bank busy for every reference for M bits. This effectively reduces the bandwidth to 1/4 the potential of the memory system (and some fraction of the processor potential).

A slightly more well-behaved access pattern is given last in figure 2-22, wherein every third element is accessed. In this case, sequential references are made to alternating banks (and in this case modules also), thus providing two operands every T cycles or only 1/2 of the potential bandwidth of the memory. However, the desired achievement is a sustained data rate of 4M bits every minor cycle, if possible. In actual practice, this becomes more difficult in a three-address system which requires referencing and transmitting up to three streams. This growth in complexity is shown in figure 2-23 with storage allocation for three vectors. Obviously, if all three vectors begin in the same bank, there will be a delay in getting both A1 and B1 to the processor, as the memory will be busy for T cycles getting A1 first and then getting B1. In addition, if N is not 1 (and therefore references nonsequential storage), the rate of getting data from memory will be affected,as shown by the rates.

In the linear select method, operands A1 and A2 are normally stored in consecutive banks, so that one single memory reference in one clock cycle can retrieve two operands for one vector. On the next clock cycle, an attempt can then be made to reference two operands from the B stream. For the two-module scheme shown in figure 2-23, this method furnishes the arithmetic units with a pair of input operands every minor cycle, assuming that there is a one-operand holding register in each read trunk to hold the second operand fetched with each memory reference.

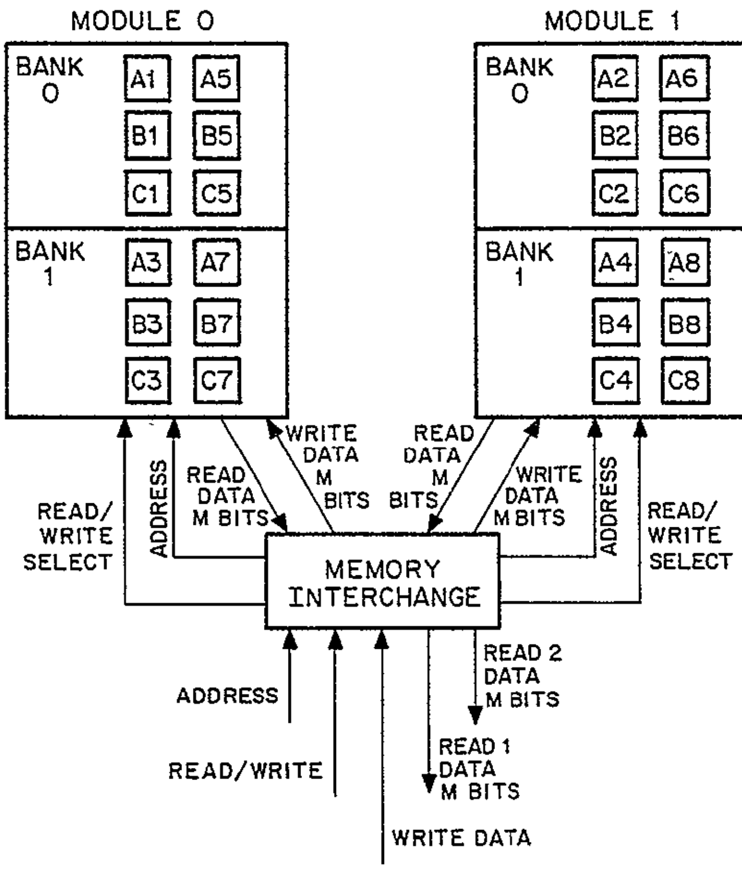
Obviously, two banks (or modules) are not sufficient to guarantee a sustained rate of two operand pairs per minor cycle, even when the data is being accessed sequentially. The number of banks necessary is a function of the access time T in minor cycles. This means that if T is four minor cycles and the goal is to provide an A operand and B operand every cycle (assuming a single arithmetic processor), eight banks of memory are required. Further, one of the read streams must be able to buffer up to four operands while waiting for the other read stream to catch up when its operands begin in the same banks.

VECTOR ADD C = A + B



```
   DO 10 I = 1,8,N
10 C(I+L) = A(I) + B(I+J)

   WHERE   1 ≤ N ≤ 8
           0 ≤ L < 4
           0 ≤ J < 4
```



READ RATES	N = 1 4M/T BITS PER CYCLE
	N = 2 2M/(T+1) BITS PER CYCLE
	N = 3 2M/(T+1) BITS PER CYCLE
	N = 4 M/T BITS PER CYCLE

Figure 2-23. Linear Select

For example, using figure 2-23, assume both A1 and B1 are stored in the same bank. Before the operation (C1 equals A1+B1) can be performed, both A1 and B1 must have been read and put on the trunks to the processor. In order to achieve the one per minor cycle sustained rate, a new A operand must be read every minor cycle (or a pair every other minor cycle) and a new B operand read at the same rate. This requires a wait of four minor cycles (access time for this example) before reading the first B operand. Since the linear select method implies that in a given minor cycle all banks are given the same address, both B and A cannot be addressed in the same minor cycle. Thus, the choice is to read operands in the following way:

T1 = A1,A2
T2 = A3,A4
T3 = A5,A6
T4 = A7,A8
T5 = B1,B2
T6 = B3,B4
T7 = B5,B6
T8 = B7,B8
T9 = A9,A10
and so on.

If there are eight banks instead of the four shown in figure 2-23 and if there is an eight-operand buffer for A, after eight minor cycles eight operand pairs will have been read and can be transmitted to the arithmetic unit while beginning to read the next set of eight pairs.

From this scheme the following characteristics of the linear select referencing method should be evident.

- Startup time

  In the example given (with both A and B operands beginning in the same bank), there is a delay of eight minor cycles before the first operand pair can be sent to the arithmetic unit.

- Sustained rate

  To maintain a sustained rate, some form of intelligent phasing of memory requests must be implemented to accommodate all cases of starting addresses from the full conflict (both A1 and B1 in the same bank) to no conflict (A1 and B1 stored four banks apart).

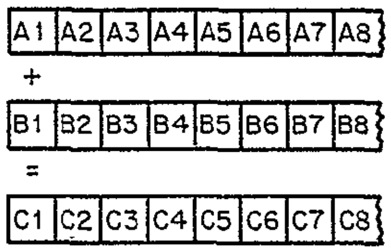
From a hardware standpoint, the linear select method requires a small amount of control logic and a large amount of buffering for the A, B, and C streams. The example discussed to this point has considered only the operand read operations; however, the third stream (results for vector C) must also be accommodated in the timing of memory requests. The further complications arising from considering the C stream, such as the need for more banks and logic to decide which streams need to be buffered and by how much, should be obvious.

Nonlinear Select, Separate Addressing

The alternative memory addressing scheme, which has been called nonlinear select, is shown in figure 2-24. In the extreme, this system provides that three separate addresses A, B, and C can be presented to the memory interchange every minor cycle. Simultaneous with the address strobing, the data can be transferred on three separate trunks to and from the interchange. In figure 2-24, only two address trunks are shown for separate requests to illustrate the principle of the method.

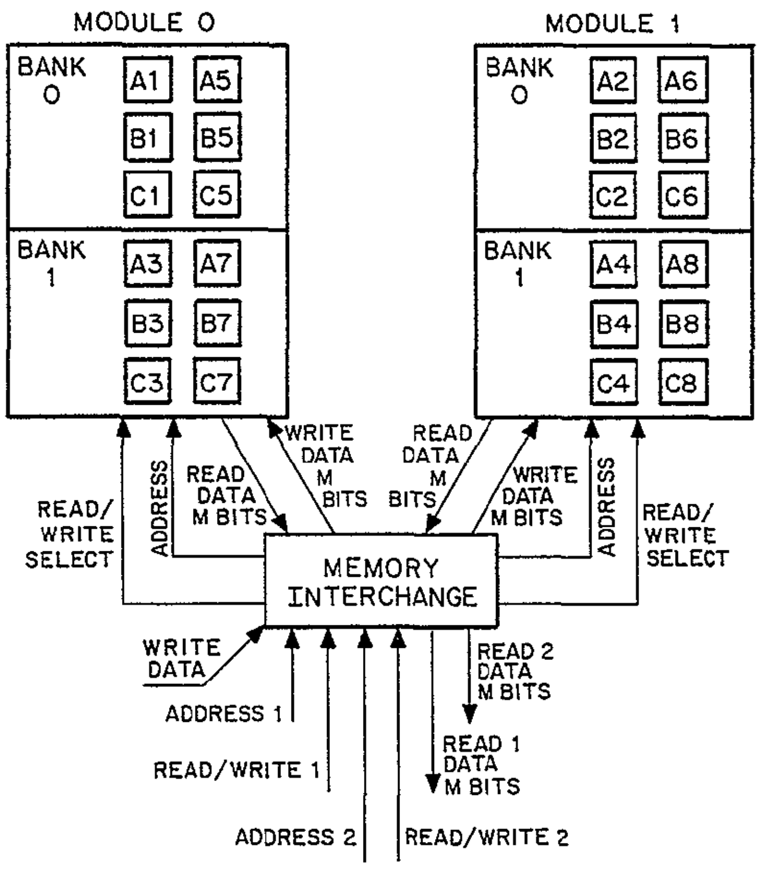
As in the linear select method, this approach can yield two operands per minor cycle on the read trunks to the arithmetic units. The major difference is that the two operands need not be sequentially stored as long as they are stored in different modules (since a single module can only accept a single request in a minor cycle). Thus, it is possible to access A1 and A2, A1 and A4, or even A1 and B2 in the same minor cycle. This access method exhibits the following comparable characteristics when measured against the linear select reference scheme.

VECTOR ADD C = A + B



```
   DO 10 I = 1,8,N
10 C(I+L) = A(I) + B(I+J)
```

WHERE $1 \leq N \leq 8$
$0 \leq L < 4$
$0 \leq J < 4$



READ RATES. N = 1 4M/T BITS PER CYCLE
N = 2 2M/(T+1) BITS PER CYCLE
N = 3 4M/T BITS PER CYCLE
N = 4 M/T BITS PER CYCLE

Figure 2-24. Nonlinear Select, Separate Addressing For Two Busses

- Startup time

  If A1 and B1 are both stored in the same bank, the startup time is the same as in the linear select method. If A1 and B1 are stored in different banks, the linear select scheme cannot issue the request for B1 until the minor cycle following the request for A1. In the nonlinear select system, if a single A operand is to be retrieved every cycle (in the linear select, two A operands were obtained with one request), then two different addresses, one for A1 and one for B1, can be sent to the memories; thus, the startup time could be one minor cycle less.

- Sustained rate

  For sequentially stored operands, the sustained rate is the same as the linear select method. However, for nonsequentially stored operands, the nonlinear select system can deliver higher operand rates. As an example, one clock cycle could reference A1 and A4, and the next cycle could reference A7 and A10, while in the linear select method, each operand would require a separate minor cycle, thus achieving half the rate of the nonlinear select method. The nonlinear select method throughput can degrade if the address increment N causes all references to be to the same bank (A1, A5). This reduces operand rates to one every T cycles (access or bank busy time). If the references are to the same module (A1, A3), the rate is one reference every minor cycle, the same as the linear select method.

One method employed to reduce the probability of conflict on references to elements of a vector A (nonsequential access) is to implement a prime number of banks. This reduces the possibility that in a given minor cycle any of the referenced operands are stored in the same bank (which obviously slows things down). Aside from an infinite number of banks, a prime number of banks seems to be the ideal choice, since array references for elements A1, A5, A9, and A13 could be stored in 23 banks and all referenced simultaneously.

While the nonlinear select method possesses some desirable characteristics in sustained bandwidth regardless of access pattern for a particular vector, other important tradeoffs should be considered.

- If it is to have the same sustained sequential access bandwidth as the linear select method, it must have all of the buffering and phasing control of that system.
- The additional address and data trunks require more hardware and interconnect than the linear select method.
- The management of conflict checking and addressing for nonpowers of two bank systems is complex and requires additional hardware. If multiplexing schemes are used to reduce trunk widths, the memory system usually must run at some multiple of the CPU clock, even for data transmission.
- The amount of nonsequential access for data in a given application should dictate whether one method is superior to another in performance as well as reliability and cost.

With the examples given it should be obvious that a more parallel machine, with a multiplicity of processors, would require more operands per minor cycle and thus wider trunks (or more trunks) to and from the memory interchange. It is in the management of these increased bandwidth problems, which take into account the number of banks as well as bank phasing to achieve sustained rates, that the memory interchange system can rapidly become so large and complex as to become a risk item in the hardware development unless certain simplifications are made in the methods of memory access.

SYSTEM BALANCE

Supercomputer systems must be conceived within a total system philosophy and implementation at the very outset. Failure to do so could result in a very high-performance, Indianapolis racing engine being tied to a Volkswagon transmission and differential. Such a mismatch either reduces the overall system performance to a fraction of the potential of the high-performance engine or overburdens the I/O to the point of frequent failure and system crashes. Either of these situations is intolerable in the supercomputer business, although models are in evidence today that mismatch the system components (the poor I/O capability of the ILLIAC IV is an example).

To deal with this problem, terms such as CPU and system balance are frequently used. CPU balance is achieved when there is sufficient random access memory available of sufficient bandwidth to permit calculations to continue uninterrupted for the totality of a job, while input and output operations are carried on concurrently. It is obvious that, given a specific case, a given CPU configuration can be in- or out-of-balance. Pursuing this concept of balance further:

1. Assume that for a given total job, all pertinent data will not fit into the real memory of the CPU. This means that after a certain number of calculations, the data must be evicted to be brought back into the real memory at a later time for additional processing.
2. The amount of real memory available will determine, therefore, how often data must be evicted from and returned to the main memory.
3. This data will generally be of three forms.
   a. Input data from a data base, consisting of parameters, collected or generated data, or data from previous processes.
   b. Intermediate or working data, local to the job-in-process and representing the overflow requirements of the particular job.
   c. Output data to be presented to the user, to be archived, or to be processed by a subsequent job.

4. It is possible then to place some upper and lower·limits on the concept of balance.

   a. If the real memory were infinite in volume, input and output data would still not be stored there because they represent the necessary contact with the outside world.

   b. If the speed of the CPU were infinite, total processing time would become the amount of time needed to pass over the input data stream and to spew the output stream to external media. Productivity is obviously determined by I/O bandwidth.

   c. If the real memory were quite small so that most of the intermediate calculations had to be swapped back-and-forth with external storage devices, the CPU would need to be no faster than the I/O bandwidth; otherwise, the system would be out-of-balance (I/O bound).

   d. If the I/O bandpass were to be too high, we would also be out-of-balance (CPU bound), with the I/O waiting for the CPU to complete calculations.

   e. For a single job, it is most desirable then that sufficient real memory be in place to contain all intermediate results, plus sufficient space for I/O buffers for moving input and output data concurrently, plus code space and whatever room the operating system requires.

   f. For a computational facility (attached to a front-end computer), it is desirable that the CPU be computation limited, not I/O limited, since the computational engine concept militates for the highest calculating technology achievable (practical and at reasonable cost).

   g. The requirement to use the computational facility interactively for program development and production run setup (as in common-place or current machines) affects the CPU balance only slightly if sufficient I/O bandpass is guaranteed so as to keep the CPU compute-bound on production runs.

   h. The amount of memory for working storage of intermediate results is itself affected by architecture. In a vector-oriented architecture, intermediate results appear as vectors of various lengths. Quite often one will find numerous vectors of length 65 kwords lurking around in the real memory for considerable time, while on a 7600, for example, such intermediate data is stored and retrieved one element at a time.

   i. There is no scientific tried and true method for arriving at a specific system balance for a machine, but one can examine the operating environments of interest (weather, energy, aerodynamics, and structures) and with a bit of experience predict what would be a suitable balance for a given architecture.

5. The amount of real memory available to a CPU is dictated by cost, reliability, and real estate. The amount made available to the user thence orients his thinking such that at almost all costs he constrains his problems to keep all intermediate results in the main memory. This affects decisions regarding the number of mesh points, cells, or time steps permissible in a given problem.

6. The amount of CPU processing power is constrained by technology and the ability to communicate at high data rates with the main memory (CPU/memory bandpass). Further considerations in sizing the CPU horsepower are the realistic size of main memory and the realistic I/O bandpass.

If a system consists of a computational processor (CPU and its associated memory) plus rotating mass storage devices (system memory) plus unit record devices, terminals, and archival equipment, it too can be examined for balance as was the CPU. The bandpass of the attached CPUs and the amount of system memory can be thought of as duals of the CPU processor bandwidth and CPU real memory. Thus, if all of an installation's working files (intermediate result data seen from the system standpoint) cannot reside on the high-speed mass storage units, it then must be staged on and off those devices from archival storage units (such as the mass storage tape file, or 6250-bpi tape units) while data to be input or output to the user may be communicated through card readers, printers, terminals, and removable media devices such as tape units and 844 type disk units.

If the rate at which the attached CPUs swallow and disgorge data exceeds the space and speed of the related system storage units, then the system is out-of-balance. Obviously if the peripheral subsystem is more than adequate to provide data transfer capabilities of the CPUs, then the system is out-of-balance, since the total system is compute bound. This condition, if not excessive (like that of the compute bound CPU), is the more desirable of the two out-of-balance conditions. If the excess becomes something like an order of magnitude, however, such an approach is obviously not going to be cost-effective.

I/O SYSTEM

Fundamentally the requirement is for a system of peripherals, interconnections, and controlware that matches the performance characteristics of the high-performance computational engine.

- The concept of a standard front-end with a special computational facility permits the use of standard peripherals and software for the normal workaday activities.
- Peripheral or peripheral station development can be limited to those components whose performance is required to support the supercomputer. This item is currently limited to on-line mass storage systems, probably rotating magnetic disks and backing storage.
- The I/O channel capacity should support a minimum channel configuration with expandability, at the customer's option, to meet the challenge of new peripherals.
- System interconnectability should permit multimainframe operation as well as multiple front-end operation.
- Most mass storage facilities should be accessible by all mainframes without passing the data through other processors or their attached memories.
- Existing stations or peripheral devices should be attachable without hardware or software change.
- Existing systems of software and hardware should be used to reduce the development time and money required.

MEMORY HIERARCHY

Circuit technologists have developed a range of storage media for computer systems. Each component is characterized by tradeoffs between speed, cost, power input, and dissipation requirements. The computer architect is therefore offered a variety of media to employ in each new machine. The simplest engineering approach would be to have a single, homogeneous, superfast, super-dense, low-powered memory system that could contain all problem variables. For the Navier-Stokes solver, the volume of program and temporary variables can easily exceed 32 million words (64 bits long). With today's technology, it would be impossible to build a reliable computing system with the speed circuits necessary for the fastest forms of the calculations. Thus the engineer, the architect, and the programmer collaborate in the establishment of a memory hierarchy, which can provide fast random access when absolutely essential, and high-volume, lower-speed access where large amount of data must be retained. Each level in the hierarchy is placed in the system to fit the task at hand. For example, the ultra high-speed, high-cost memory devices are used in small buffer memories where high-speed access is needed for random processing by functional units and other data streams such as I/O.

High-speed, high-power, moderate-cost memories are used where random access performance is a matter of concern but where a fair-sized volume of data must be stored.

Medium-speed, medium-power, medium-cost memories are used where a large amount of data must be stored for immediate, direct access.

Slow-speed, high-density, low-cost memory is utilized where random access is not important but where block transfers can be effectively used to move data.

Very slow speed devices with massive storage capacities usually employ magnetic recording techniques with rotating media. Here the access time is in milliseconds, with moderate transfer rates, usually moving large blocks of data (2^{19} bits or larger).

Very low-speed, very high-density, very low-cost memory systems are of the archival type, with storages in the 10^{12} bit range with access times in seconds and transfer rates around 1 Mbit/s.

The proportion and size of each level of memory in the hierarchy is dependent on the nature of the problem solution. In preliminary system design in section 5, a particular hierarchy is spelled out for a particular architectural approach to the NASF system. It is Control Data's opinion, however, that some form of memory hierarchy will appear in most other supercomputers of the 1980s, employing all of the levels of memory described above.

FRONT-ENDING

Control Data's experience in the utilization of the 7600 and STAR computers in operational environments is that such machines are best suited to a limited number of tasks (normally called number-crunching) to which their entire resources should be dedicated. This type of problem consists mainly of a high degree of floating-point computational load, massive data base processing and high-speed. mathematically oriented decision making. The resources required to develop the modest amount of software (both systems and applications) needed by such computational behemoths is extraordinary, even in the sophisticated programming world of the 1970's. The question then arises as to whether other general-purpose applications (including business-oriented ones) should be implemented on these large scale machines, along with the necessary system support software, instead of limiting the applications of the 7600/STAR class machine. The answer to this question lies in two facts.

- There exists a substantial number of commercially available, off-the-shelf computing systems, totally replete with software and applications packages, to perform the accepted general-purpose functions, with adequate performance and attractive costs.
- There are few highly specialized machines capable of dealing with the compute-bound number-crunching problem of today.

These facts, coupled with the development cost for general-purpose software for the computational engines, strongly suggest a system structure that totally limits the function of the 7600/STAR/NASF computing machine to the narrow range of high volume scientific computation. Thus, the 7600 and STAR mainframes of 1977 are imbedded in a larger system, consisting of I/O equipment as well as front-end processors. In a minimal system, a front-end processor need only serve as an I/O staging computer, handling communications trunk management, card, tape, and printing scheduling, and other installation functions. As the requirements for the system expand to include graphics and interactive data processing, the network can be upgraded to a more powerful machine possessing the exquisite software systems. This approach will obviously yield a stable, operational computing system faster than commencing development of new software for the computational engine.

This approach also allows an engineering decision to be made regarding the emphasis to be placed on portions of the performance of the computational cruncher. If the application of such a cruncher can be truly limited to a narrow class of problems, many general-purpose features (for example, to support time-sharing and BDP) can be ignored, and the development time and real estate resulting can be invested in better computational processing. Therefore. in the NASF, it has been obvious from the outset there would be a hierarchy of processors in the system. The first large number cruncher is referred to as the back-end machine, and the general-purpose computing facility is called the front-end machine. 'The front-end machine is literally the first one encountered when coming through a communications network, card reader, or graphics terminal to submit a job.

The front-end computer should have sufficient power to perform all installation housekeeping, file management, system scheduling, graphics subsystems data storage, retrieval, and reduction. In addition, the front-end machine should be sufficiently powerful to perform compilations for the back-end processor, loading, linking, and management of program debugging facilities. A machine of the capacity of the CDC CYBER 174 is indicated from the initial evaluation of the computational load in the network. It is quite possible, however, that the computational load may demand that another processor of the CDC CYBER 172 class be added to the network to handle interfacing to all the I/O and graphics subsystems that are anticipated.

The I/O subsystem in the form of the network trunk makes it possible to add new facilities to the installation as the system matures, with minimal impact on the software or general operating characteristics of the basic computational facility. Since a large part of the system's responsibility on a day to day basis is to assist in data preparation, data display, and management of remote communications, it appears essential at the outset that two front-end machines be a part of the NASF. This redundancy is needed by front-end systems to ensure that customer contact is not lost, nor computational jobs mislaid due to front-end hardware failure. Although every step will be taken to ensure the integrity of the back-end machine, an occasional loss of service is to be expected in a realistic environment. Such a loss need not be fatal to the system if sufficient redundancy resides in the front-end system to keep vital data from being lost. To achieve this redundancy, the sizing in this report is based on a front-end configuration of two CDC CYBER 173 computers rather than the CDC CYBER 174/CYBER 172 configuration mentioned above. In addition, all front-end peripheral devices have dual access to interface both front-end machines.

With the network scheme previously discussed, it should be pointed out that all of the processors in the network need not be compatible in data formats or instruction set. Thus, it would be possible to attach the most effective processor to the network to accomplish the task at hand. For example, a graphics front-end machine more suitable than the CDC CYBER family might be the most appropriate device to attach to the trunk for interactive graphics support. With the network proposed, such an approach is quite easy to accommodate with the hardware available but with some additional effort using the design automation software that will be in place.

FAULT TOLERANT COMPUTING

One of the major concerns in creating a computational center with unique resources for a particular application such as the NASF is the availability and reliability of the total system. Experience with the recent generations of supercomputers has made users of such systems acutely aware of pitfalls awaiting the implementors of aggressive, state-of-the-art computer systems. Two major pitfalls are system interruptions and undetected errors.

- System interruptions

  The most obvious form of problem confronting a nonhomogeneous user population is the frequency and duration of system interruptions. If a problem solution requires and heavily exercises massive amounts of memory (both main memory and rotating mass storage), and such memories contain error detection systems with a reasonable amount of honesty built in, the probability of system stoppages can be quite high, based on the available technologies and sheer component counts. When simple parity was added to main CPU memories in the early 1960's, the number of system stoppages increased, but the number of nonstoppage, mysterious occurrences decreased. System interruptions due to transient failures (which are the most frequent) in peripheral subsystems have been reduced substantially in the past 10 years through the employment of error detecting and correcting codes for disk and tape units. These correcting algorithms, coupled with the addition of smart controllers which handle functional retry of selected operations and operating systems incorporating backup and automatic recovery methods, have reduced visible (to the user) system interruptions due to peripheral subsystem deficiencies. Thus, two maturing technologies (circuit and system) have converged to provide disk storage subsystem (75×10^9 bits) availability of 99.8 to 99.9 percent.

- Undetected errors

  The most frightening of all possible occurrences is the possibility that a computer system will commit some grievous mistake, unknown to anyone, and thereby inject some spurious and potentially dangerous results into a crucial project (for example, the designing of a bridge). While a tradeoff analysis may show that component reliability is so high that an automatic correction scheme need not be employed to reduce system interruptions, the need for complete error detection can never be dispensed with.

There are costs attendant to the inclusion of error detection and correction schemes in any hardware system, of course. In disk systems, a certain percentage of the disk space is required to store the detection and correction patterns. The amount of space needed for these patterns depends largely on the most probable failure mechanisms (burst or single-bit error) resulting from the recording technology and the degree to which one desires to lower the probabilities of undetectable, if not uncorrectable, errors occurring. This could range from an elementary parity scheme with only 1-percent storage overhead to a totally redundant scheme using independent electronics and storage medium with a 100-percent storage overhead. In addition to the storage overhead is the increased cost and complexity of controllers and controlware to support the correction and detection methods.

In Control Data's judgement, the problem of undetected errors is the crucial argument surrounding the concept of fault tolerant computing. If one is to take the phrase literally, a large complex such as the NASF would have to continue computing, regardless of the type of severity of the fault (short of a complete power failure). In such a case, 100-percent redundancy would not be sufficient since two systems not yielding congruent answers, would only cause the entire complex to pause while the failing component was isolated. Thus, a configuration of three systems would be needed to instantaneously vote on the correct solution (mathematical, operating system, or I/O) and present that result to the user. For certain applications, such as the real-time computations in support of space missions, the need for tolerant computing is persuasive. In the case of the NASF, however, the cost, physical space requirements, and control complexity far outweigh the need for a totally uninterruptible operating environment. What is needed for the NASF is the achieving of realistic goals which ensure that, on any day of the week, a customer can get his work done on the NASF within a reasonable amount of time (for example, the 7 to 15 minutes given in the design objectives document). Therefore, if the predicted component failure rate is such that a system interruption due to such a failure occurs at the rate of no more than twice a month, and if the average time needed to isolate and repair the component is about 30 minutes, this requirement can be satisfied without a totally redundant system.

Error detection is essential for the NASF, however, and it is this aspect that must be addressed in any design of the NSS computational engine. Redundancy in the front-end equipment and standard failure mode recovery and correction techniques are available today for the peripheral subsystems. An NASF which produces results at the rate of one billion per second must have built-in safeguards to ensure that not a single undetected error can occur. This objective in practice is impossible without total redundancy (which was declared above to be impractical and too expensive), thus alternatives must be sought.

One approach that has been pursued is the inclusion of parity information in all arithmetic and data networks. An alternative is one of the several residue or modulus arithmetic schemes that have been developed. This alternative entails parallel operation on parity data as the actual data is being processed through networks, and then checks the parity (residue) of the arithmetic results with the resulting parity (or residue) of the parallel error checking network. This approach can guarantee the integrity of the data trunks as well as most arithmetic networks (the high-speed multiply algorithm is difficult to check and still maintain any kind of performance). However, this approach falls short in several crucial instances. For example, the total number of circuits needed for control, control fanout, and data selection can, in some high performance computers, amount to 25 percent to 50 percent of the total circuits in the computer. Thus, checking just the data trunks can lead to a false sense of security.

For example, in a twos complement floating-point addition, the occurrence of a single-bit, legal overflow due to the addition must result in a right-shift, post-normalization of the resulting operand. This operation is data dependent and only detectable by the actual add network, since information in the residue or parity data is insufficient to cause this to happen. Therefore, a single wire carrying this information must then control not only the arithmetic shift in the add unit, but must control the end-case calculation needed by the parity or residue system. A failure in this single wire could cause a wrong answer to be produced (loss of the overflow bit, the most significant bit) which would be undetected by the checking network.

Thus, some other scheme is recommended, based on a need to check both control and data. This seems to lead to the development of identical units, each containing all of its own control, which can check each other on a continuous basis. The cost of total redundancy being what it is, it seems infeasible for these identical units to always be checking each other; rather, some system must be devised to allow them to sample each other during appropriate idle moments.

This approach is not fault tolerant in the extreme but does offer a tradeoff between predicted reliabilities of component portions of the system and the need for an available, reliable computing system. For the NASF environment, that is what fault tolerant computing is really all about.

ARCHITECTURAL PROJECTIONS AND CONCLUSIONS

CPU

As stated several times in this report, Control Data feels that the main course of supercomputer development is necessarily going to be concentrated on parallel computational systems. Therefore, in the 1980's, it can easily be projected that versions of the MIMD and SIMD architectures will be developed for the various supercomputer markets. One thousand processor MIMD systems will be found in air traffic control and missile tracking systems, where intraprocessor control is minimal. Computational rates for some processors will reach and exceed the threshold of 1 gigaflop (1 billion floating-point operations per second). Peak rates for several of the architectural alternatives will reach the 1-gigaflop threshold, but the ability to achieve a sustained rate of 1 gigaflop will depend on the degree of customization that can be performed with the machine/software system for a particular application. The suitability of the MIMD schemes for specific application areas is definitely tied to very specific hardware and software organization. For example, in the case of the flexible processor, a high computation rate can be sustained if the problem being solved is photographic analysis of a large number of points. This was the task for which the machine was designed and the software structured. For other general-purpose floating-point computational tasks, however, the performance of the flexible processor is seriously degraded.

As the MIMD architectures must be tuned to the problem at hand, so too must the SIMD structures be customized to meet the 1-gigaflop rate of solution of the Navier-Stokes equations for the NASF. With the proper technology and machine structure, it is certainly possible to build a 1-gigaflop machine for the NASF, for operational employment in 1982.

It is the judgement of Control Data, however, that a SIMD machine is the best candidate for the NASF computational engine, from the point of view of buildability, reliability, and most of all programmability. The experiments to date, employing MIMD architectures for general-purpose problem solving, have shown that there is a good deal of work yet to be done in programming technology for multiple processor systems.

With all other attendant risks of this project being necessary (that is, new state-of-the-art technology, customized design, and short time frames for development), it seems desirable to take a traditional approach to the programmability problem using somewhat conventional languages (such as FORTRAN) with modest extensions. Compilers and object time systems for a SIMD machine are, in the judgement of Control Data, easier to get into production status than would be the comparable MIMD system.

To achieve parallelism in a SIMD architecture implies some need for a conceptual notion of parallel processing and some straightforward means of describing such processing to the computer. The notion of vectors is probably the most universally recognized means for concieving of parallel operations on data streams. Even the more sophisticated array processors, PEPE, STARAN and ILLIAC, use the concept of vectors when describing problem solutions. The question then of what type of vector processing is desired is of secondary importance. The CDC STAR-100, the Texas Instruments ASC, and the CRAY I each implement vector processing in radically different ways, but the fundamental property of achieving performance, through the foreknowledge that more than one element is going to be processed, is the common ground of all such machines. In section 5 of this report, an extension of the existing vector processing architectures is described which Control Data feels is the optimum match to the problem, cost, time scale, and reliability requirements for the NASF.

SYSTEM

While the computational engine for the NASF must be customized for the application to meet the established performance objectives, the remainder of the system can and should be built from relatively off-the-shelf components. This is due to the need to reduce the overall project risk (more customized components means more risk) and further, it is due to the fact that extant technologies for front-end subsystems appear to possess all the performance and capabilities that are desired.

Standard product systems such as the CDC CYBER line offer a range of compatible processors that can operate as front-end processors to the computational engine. To support the graphics systems and data preparation requirements of the NASF, a fairly powerful CDC CYBER class mainframe will be required. It is proposed that all system activities, permanent file managements, back-end scheduling, interactive access, graphics support, data preparation, and routing be performed by the front-end computer. The NASF computational engine will then access files only through the front-end, take the next task given to it by the front-end and let the front-end control all I/O operations, thus reducing the resident monitor system in the computational engine to about 16 tasks and about 8000 words of locked-down central memory, not to mention near-zero overhead.

The I/O system described in section 5 is similar in structure (network trunk) to that being offered on an increasingly broader scale by several vendors. Thus it can be expected that new equipment and subsystems can be interfaced to such a trunk scheme more readily than is now possible with the turnkey I/O systems normally available in the 1970's. The advantage to NASA in this situation is the ability to expand the NASF by adding new subsystems from a variety of vendors with a variety of alien formats and data structures, thus giving NASA access to the latest and best facilities, while preserving the basic structure of the original NASF.

Peripheral equipment now in existence or being planned for introduction as standard components before 1981 appears to be adequate to meet the initial system requirements of the NASF. In the long term, however, more dense, higher transfer rate mass storage subsystems appear to be essential to support the workload anticipated for the post-1984 period, when the facility can be expected to be at its peak in acceptance by the using community. The extant archival subsystems would also appear to be inadequate to meet the demands to be made in that time frame. Thus, development efforts in these two key areas may have to be encouraged, or even pushed by the NASF developers, since general-purpose requirements in this same time frame may not offer sufficient incentive to develop such aggressive peripheral equipments.

RISKS

There appear to be two significant risks involved in this project.

- The monumental amount of work to be performed and the high caliber of resources required to ensure success.
- The need to use the most aggressive technologies in several places in the computational engine ensures that the period from 1979 to 1981 will contain many anxious moments as kinks are encountered and conquered in the new technology.

Ancillary risks are:

- The final system performance may not meet its goals. Discovery of this fact would be delayed until actual production codes are engaged.
- The performance metrics chosen and carefully analyzed may not portray the final state of the Navier-Stokes solutions developed by Ames personnel. Machine features designed specifically for a given methodology may no longer be needed, while other features required by the final techniques may be missing.
- Cost and schedule overruns could get out of control and not become visible until crisis conditions arise.

SECTION 3

BENCHMARK DEFINITION/PROJECT MEASUREMENT CRITERIA

ANALYSIS OF FLOW MODEL PROGRAMS

Two forms of Navier-Stokes solution have been under development at Ames Research Center for some time. The programs are presently operational on the CDC 7600, with one code also heavily used on the ILLIAC IV. At this stage of algorithm development and hardware availability, it is practical to perform only two-dimensional analysis of aerodynamic configurations with these programs. Both programs have undergone revision and improvements since this study phase began, and in the case of one code, this has complicated the projections of its behavior into three-dimensional modeling.

The purpose of the study team in working with these programs has been twofold.

- To determine precisely the form and content of the computations required by the NASF, as an aid in the specification of particular architectural and design concepts for the NSS machine itself.
- To aid NASA in the identification and specification of a set of performance metrics which could be frozen in actual code and used to measure the final performance yielded by the detailed design phase and the actual hardware construction phase of the NASF project.

At the suggestion of Ames personnel, primary emphasis on the first of these purposes (design criteria for the computational hardware) has been directed towards one code, called the implicit code. The second code, called the explicit code, was used, essentially, to check the NSS architecture to ensure that the performance of the explicit code would be supported equally well (or in some instances, not penalized too severely). The reason for this placement of emphasis will be discussed later.

PROCEDURE

The process decided upon for analysis of these models and extrapolation to 3-D consists of:

1. Converting the original FORTRAN form of both codes to the STAR-100. This permits making sophisticated measurements of the program, since STAR possesses extensive internal instrumentation for examining both the computational and mathematical behavior of a set of programs.
2. Establishing a set of test run parameters, and known CDC 7600 solutions to ensure that the STAR results are correct.
3. Data gathering on the STAR-100, for the 2-D versions.
4. Vectorizing the codes for STAR in 2-D (to check answers) for determining to what extent a parallel solution approach can be simplified.
5. Data gathering on the vectorized version.
6. Adding 3-D operators and mesh points to the explicit code, nonvectorized.
7. Data gathering on the pseudo 3-D, nonvectorized code.
8. Vectorizing the pseudo 3-D model.
9. Data gathering on the vectorized model.
10. Developing a report describing the hardware requirements to meet specified performance standards for that particular set of pseudo 3-D codes.

At this time steps 1 through 4 have been completed, and it is Control Data's intention to complete the entire set before the end of 1977. The inability to pursue this effort to completion prior to the end of the study period is due in part to the investigator's diversion into making radical changes in the NSS structure as data evolved from earlier steps in this analysis. In part, also, the difficulty in creating a meaningful 3-D pseudo code was substantially underestimated.

OBSERVATIONS OF CODES

Some observations can be made about the form of solution using explicit or implicit codes. Both codes provide solutions of the Navier-Stokes systems of differential equations, at discrete time intervals, to determine at the end of that time interval the somewhat instantaneous (time-averaged) values for density, velocity vectors, and energy in small regions of space surrounding an airfoil in a moving stream of air.

Explicit Code

The explicit form of solution has some intriguing merits, not the least of which is that it is more easily visualized in terms of the actual physical space and behavior of the air around the airfoil as it proceeds. In essence, the differential equations can be solved by a simple integration scheme where, in a simple left-to-right scan, new values at each point are calculated from all the old values at that and surrounding points in the mesh. This scheme requires little additional space than that required to hold all the basic flow parameters and boundary conditions, thus reducing the need for memory (which is the most costly segment in the NSS). The process of moving a step at a time across the mesh in discrete intervals, calculating at each step, permits the introduction of special computations in regions of interest. For example, as the program sweeps across the mesh, if it encounters an area of violent mathematical behavior (probably mirroring extreme activity in an actual model), the program can pause to massage the data at that point, or group of points, to a greater extent than remaining points computed.

The objectionable aspect of the explicit form is that extremely tiny time steps must be taken in relationship to the physical intervals in the mesh. Thus, many computations must be done to yield pictures of the modeled flow that are of any value.

The actual explicit program analyzed was in a state of evolution as its developers experimented with methods of reducing running time on the CDC 7600. By introducing a variety of special-case processing, the total number of computations was reduced during this 6-month period by as much as a factor of seven. This improvement was not all free, however, as the special-case processing introduced a large number of conditional tests and branches to apply the processing. Tests and branches affect parallel processing adversely in many instances, since they interrupt the otherwise orderly flow of instructions in the processor while some decision is being made. To vectorize the explicit code in its present form for the NSS (to achieve parallelism), the scalar special casing has to be replaced by vector special casing. This is achieved through the use of Iverson (reference 1) techniques of selection (of the regions to be special-cased), the compression of all relevant data from these regions into working vectors, the processing of this smaller amount of data by vector operations, and then the merging of the results back into the larger data base. These operations are performed in the proposed NSS machine (refer to section 5) by the compress,mask,merge operators.

The explicit code as it now stands is an excellent representation of the mathematical characteristics of the solution. In particular, this approach to the solution uses what are called split operators. Each is an integrating operator which advances the solution through (a portion of) a time step. The operators are split according to spatial direction as well as type of effect, namely, a parabolic operator for diffusive effects and a hyperbolic operator for advection effects. (By contrast, the operators in the implicit code are split strictly according to spatial direction.) This approach is supported in the explicit code by having each operator coded as a separately called subroutine to make their characteristics visible to the mathematician as well as to make programming of the algorithms direct and readable. As a teaching tool, this form of the program is excellent (as was intended by its developers). As a means for maximizing performance of a computer (of any architecture), it is poor (also recognized by the developers).

Subroutine calls are expensive luxuries when performance is the dominating requirement. In the STAR-100 vectorized form of the code, all subroutine calls are eliminated within the major operators LI, LJ, LJH, and LJP. In this process, many calculations were removed from inner loops (where they appeared for convenience reasons) and other one-dimensional arrays were changed to two-dimensional arrays to permit the vectorization of their use with the flow variable arrays.

The flow through the model is clear in the current explicit program. However, estimating computation rates for the program is difficult because of the special casing that is done. The code then exhibits some highly data-dependent behavior. Some calculations manually estimated (from examination of the code) differed by 32 percent from the actual count of calculations taken from STAR-100 monitoring counters. This error is due to the large number of data accesses made for conditional testing purposes in exchange for a consequent reduction in floating-point computations.

It appears that the direction of development of this code, to enhance its use as an existing production program, is in further special casing, perhaps becoming quite extensive as other turbulence models are introduced. The more this effort is pursued, the further away from parallel processing schema the actual code becomes.

Implicit Code

The implicit form of the Navier-Stokes solution relies on the simultaneous solution for values at all points in the mesh in a given time step. Since this method then does not rely on just the old values of variables but on the simultaneous movement of all variables to new values, it induces improved mathematical stability. This inherent stability permits larger time steps to be taken in the solution and thus reduces the computational load.

In exchange for this reduction in computations, an expansion of memory space is required to hold temporary values as fairly large matrices are being inverted. This trade of space for time led to some of the key decisions regarding NSS architecture discussed in section 5.

Although using the explicit code was difficult for estimating performance or the effects of three-dimensional processing, it was an excellent vehicle for parallelization via vector processing, with long vectors being created for every operator along the direction of sweep and the orthogonal direction as well. The implicit code, on the other hand, permits estimation of computational load since there is no special casing in the heart of the calculations; all arithmetic is engaged in inverting the matrices for all points. The implicit code causes some problems in vectorization of a brute force nature since the recursion relationships inherent in the block tridiagonal inversion reduce the computation rate. By performing the inversion on several slices at one time, the full power of a parallel machine can be brought to bear on the problem. This is the approach taken by Ames developers who have created a slicing system for the ILLIAC IV version of the code.

Whereas the manually computed predictions for the explicit code varied widely from measured results, the implicit computations agree within 1 percent between measured data and estimates. This is a measure of the predictability of the code and is one of the reasons why it is an excellent vehicle for manually estimating the performance of candidate architectures.

Relationship of 3-D to 2-D

The extension of the existing 2-D explicit code was originally considered to be quite elementary. The addition of a Z-direction operator and the extension of all dimension statements seemed to be all that was required. However, the rather more asymmetric treatment of spatial directions in the explicit code (splitting by effect) made unclear exactly what the substance of the third-dimensional operator should be. The resulting effort was beyond the availability of the explicit code analyst. The conclusion is that a major restructuring of even the simple operators LI and LJ would be needed and the existing turbulence model could not be retained in its present form. The resulting decision was to pass the rewriting of the explicit code to the expertise of the NASA mathematicians who are involved with this problem already.

Instead, the remaining analytical effort was focused upon the extrapolation of the behavior of the implicit code because of the favorable characteristics mentioned previously. For that purpose, an assumption was made that a 100x100x100 mesh size would some day be of interest, although Ames instructions aimed at approximately 80x80x80 mesh points. Section 5 includes a discussion of some 3-D elements of the implicit code and their effect on computational loading of the NSS.

MEASUREMENT RESULTS

Utilizing the STAR-100, extensive measurement was made of the explicit code to determine its computational behavior when processing a sample run of the Garabedian-Korn airfoil in 2-D. Thus, focus was on the explicit code because of the special casing previously mentioned. Although detailed tables of the code behavior are available, many of the items in that table are not meaningful except as measures of the efficiency of the compiled object code. For the 7-minute run (on the CDC 7600) of the explicit code:

	Instruction Counts for One Cycle of:		
	Inviscid	Viscid	Total Run
All arithmetic (integer, address floating point)	1,440,494	5,168,547	940,754,013
Floating-point operations total	843,091	3,695,424	682,656,694
ADD	253,963	959,274	
SUB	182,671	782,248	
MULT	322,624	1,629,975	
DIV	79,629	309,949	
SQRT	4,204	13,978	
Subroutine calls	28,523	90,927	
Branches	174,510	401,373	

Time spent in subroutine LJ = 47.3 percent of total execution

Time spent in subroutine LI = 37.6 percent of total execution

Time spent in subroutine LJP = 1.5 percent of total execution

EVALUATION REMARKS

The numbers given above are consistent with similar numbers derived from earlier versions of the explicit code. Taking only the floating-point counts and counting square roots as one operation rather than the seven normally required for evaluation, it is found that the CDC 7600 7-minute run was operating at 2.24 million floating-point operations per second. One of the original objectives of the project was to find a machine design that would meet 100 times the 7600 performance. That would mean that the NSS need only perform 224 million operations per second. However, the best estimate for a 3-D extrapolation for the explicit code is that about 500 times the calculations presently done would be needed for a production explicit code on the NASF. Thus, 500 x 2.24 equals 1120 million floating-point operations per second, in excess of the 1-gigaflop rate derived independently by NASA.

The implicit code is running at 2.1 megaflops on the CDC 7600, and the data presented in section 5 shows that, again, about 500 times more calculations are required for the 3-D version. This would then make a 50-minute run for a 100x100x100 problem feasible if a 1- to 2-gigaflop processor could be built. The objective, however, is to run this code in the 7- to 15-minute time period established by NASA. Thus, some mathematical and programming work must be done to the implicit code to reduce the total computation requirement to about 1/5 that of the present algorithm. This effort is underway with optimistic projections being made by Ames personnel.

Other significant information that can be derived is that

- Subroutine calls at a cost of about 20 floating-point operations each are as expensive as all the adds and subtracts combined.
- The number of multiplies almost equals the number of adds and subtracts, thus equal emphasis should be placed on multiply and add/subtract functional units.
- A divide rate of 1/4 the add/multiply rate would keep the divides in balance with the adds and multiplies as far as rate is concerned.
- With square root function needed only about 1/100th as often as the add/subtract combination, little justification exists for providing a special square root unit as is done on the STAR-100.

- With branches requiring approximately six floating-point operation cycles each, the total time consumed in branching almost exceeds all arithmetic operations combined. Again, special casing must be done other than through branching.

BENCHMARK DEFINITION

The previous discussion has been aimed primarily at the derivation of data to aid the design and architecting of the NSS. The same information bears, however, on the very important task of establishing benchmarks or metrics that can be used throughout the remainder of this project.

Key items

Any benchmark must contain properly weighted measurements of several key areas of the known codes.

- The block tridiagonal inversion (for performance reasons)
- Formation of the Jacobian (for mathematical significance reasons)
- The entire algorithm in the explicit LI operator subroutine (to ensure that this key element is not ignored in the machine architecture)
- Evaluation of the parabolic and hyperbolic operators LJP and LJH
- A significant operation not truly represented by the existing codes is the generation and remapping of three-dimensional meshes for these codes

Parameters

The parameters for measuring any architectural candidates against these elements are the operating parameters expected for initial demonstration of the system. For the explicit evaluation, an actual simulation of one inviscid cycle and one viscid cycle each using matrices of 10x10x10 is possible and desirable, with manual extrapolations for a given number of chord lengths and a maximum mesh size of 100x100x100. It is believed that the worst-case performance estimates should always be based on 100x100x100 meshes. An identical airfoil configuration should be submitted to the implicit code, to make comparisons of the explicit and implicit approaches readily visible.

Ground rules	It is strongly suggested that the 2-D models be the frozen versions of the current explicit and implicit codes. Thus, all analysis to date can be brought to bear in detail on the simulated and actual runs of these baseline codes. The measurement criteria should remain the 384 time step solution for the Garabedian-Korn airfoil presently used for metric development and analysis.
Extrapolation	The 3-D models must have room for 100x100x100 meshes. The actual program for 3-D solutions, including mesh generation, should be used with one fixed configuration of airfoil, turbulence model, and mesh stretching common to both the explicit and implicit forms. Working programs must be available to the study teams within 3 months of beginning phase II of the NASF study. Without acceptable 3-D programs developed by Ames personnel by that time, it will be impossible to develop the detailed structural design of the NASF in time to meet end-date commitments for available hardware.

REFERENCES

The following is a list of all references that are called out in the text of this section.

Reference Number	Reference
3-1	Iverson, Kenneth E.: A Programming Language. John Wiley and Sons, Inc., 1962, Chapter 1.

SECTION 4

CDC DESIGN OBJECTIVES DOCUMENT

INTRODUCTION

During the last 10 to 12 years, Control Data has developed a series of standard documents used internally for initiation, measurement, and control of development projects. One of these standard documents is the Design Objectives document which comprises this section of the report, modified slightly to be consistent with report format. It is the statement of understanding which the project has of a desired development as specified by strategic plans which defines the need for the development. The information presented here represents target objectives or goals, as opposed to firm requirements and/or stated commitments. As the project progresses, differences between these objectives and requirements or expected accomplishments will be resolved into firm requirements, followed by specifications.

DEFINITION

The Numerical Aerodynamic Simulation Facility (NASF) is a computational system customized for aerodynamic research and engineering. The system must be capable of extremely high performance in the solution of time-averaged Navier-Stokes equations. The computational engine, Navier-Stokes Solver (NSS), will be optimized for numerical simulation of three-dimensional, viscous flow fields by utilizing custom elements wherever necessary, yet the system will be configured with standard hardware and software to the extent possible while maintaining performance objectives.

REFERENCES

PLANNING JUSTIFICATION

1. National Aeronautics and Space Administration, Ames Research Center Request for Proposal 2-26539 (AS) For Preliminary Study for a Computational Aerodynamic Design Facility, dated October 4, 1976.

2. CDC Proposal for Preliminary Study for a Computational Aerodynamic Design Facility, in response to RFP2-26539 (AS), dated November 12, 1976.

3. National Aeronautics and Space Administration, Ames Research Center Contract NAS2-9457 for a Preliminary Study for a Computational Aerodynamic Design Facility, dated February 1, 1977.

INTERDEPENDENT DO AND DR DOCUMENTS

No interdependent Design Objectives (DO) or Design Requirements (DR) are known at this time.

TECHNICAL

1. Planning Justification references 1 and 2 above.
2. Final Report on a Study for a Numerical Aerodynamic Simulation Facility (Note: this is the report of which this Design Objectives section is a part) dated September 1977.
3. F. Ron Bailey (Ames Research Center) letter of May 20, 1977 with NASF Wish List attached.
4. Explicit/Implicit code (MacCormack) received from NASA-Ames on May 6, 1977.
5. Implicit code (Lomax, Steger) received from NASA-Ames on March 22, 1977.
6. Garabedian-Korn airfoil definition, associated grid data, and input parameters for execution of the codes in 4 and 5 above. These were supplied by NASA-Ames for the explicit/implicit code with the code on May 6, 1977 and for the implicit code separately on magnetic tape, first in April 1977 with an updated version in June 1977. The latter shall be the applicable data.

STANDARDS

At this early phase (prior to detailed design considerations), it is difficult to identify specific standards, those which are applicable and those to be waived. The CDC Standards Checklist (including industry and joint standards) is included as appendix A of this section. Basically, the NASF design will comply with all applicable standards, possibly with waivers necessary for the NSS (because of its custom design) as follows:

1. Some documentation standards, however, Quotation for Special Equipment (QSE) standards would be followed.
2. Some circuit and packaging standards because of special requirements.
3. Design practices which could satisfy UL certification requirements will be followed, but formal UL certification will not be sought.
4. The intent of standards on environmental conditions, vibration, and shock will be followed, but physical characteristics, such as size and weight, may preclude standard methods of testing.

OBJECTIVES

FUNCTIONAL AND PERFORMANCE OBJECTIVES

1. Capability to solve the three-dimensional Reynolds averaged Navier-Stokes equations using both explicit/implicit and totally implicit, dimensionally split, finite difference methods.

2. Sufficient flexibility to handle a variety of:

   a. Boundary conditions including given values and derivatives of the independent variables, extrapolated values, and auxiliary relations for the pressure.

   b. External and internal geometric boundaries and their cooresponding finite difference grids.

   c. Turbulence models ranging from algebraic to seven differential equation descriptions.

3. The ability to compute solutions for up to one million grid points. This implies a data base range from 14 million words for:

   5 conservation variables at two time levels (10)
   1 turbulence variable
   3 grid coordinates

   to 40 million words for:

   5 conservation variables at two time levels (10)
   7 turbulence variables at two time levels (14)
   3 grid coordinates
   12 metrics (including time)
   1 Jacobian

4. The ability to obtain steady state solutions for one million grid points in 7 to 15 minutes CPU time for 3-D problems using algebraic turbulence models. At present this must be measured using 2-D explicit/implicit and implicit codes as performance metrics.

   Explicit/implicit code (MacCormack) status: 2-D airfoil steady state solution obtained in 7 minutes on CDC 7600 for 2100 grid points. Steady state reached after 13 chord lengths of travel by computing inviscid solution for seven chords and viscous solution for remaining six chords. Effective computing speed on 7600 is about 2 megaflops. Assuming twice the computational effort at each grid point for 3-D cases, this implies that to compute 13 chords in 10 minutes for one million grid points requires an effective computing speed of 1.4 gigaflops. Further efficiencies by 1980 can be expected that would reduce this requirement to some extent.

Implicit code (Lomax, Steger) status: 2-D airfoil steady state (13 chords traveled) obtained in 50 minutes on CDC 7600 for 2300 grid points. All calculations viscous. Effective computing speed on 7600 unknown. This code implies that an effective computing speed of 10 gigaflops is needed for a 3-D calculation over one million grid points. However, researchers working on the implicit code are confident that improvements in the treatment of boundary conditions and other strategies can improve the speed of the method by a factor of 10, which implies that at least a 1-gigaflop effective rate will be needed.

It is concluded that the minimum effective computing rate needed for the Navier-Stokes problem is 1 gigaflop. Tradeoffs between the ability to maintain this rate for the two approaches is desired.

5. A precision of 32-bit floating-point will be sufficient for some operations, but greater precision, 48-bit or 64-bit floating-point will be required.

6. Front-end processors will have state-of-the-art general-purpose computing capability for the early 1980's; however, they will be dedicated to driving the NSS and keeping it fed with jobs and data.

7. The back-end processor, or NSS, will be highly customized and specifically tuned for solution of the Navier-Stokes equations for numerical aerodynamic simulations.

8. A high level programming language consistent with ease of mapping solution methods on the machine, optimum machine performance, and the available language development time is to be delivered with the machine. Efficient programming of the machine should be as painless as possible, and ease of debugging is essential.

   Although flexibility is desired, the major issue is the ability to perform Navier-Stokes solutions for various flow problems, and the flexibility to do other flow problems, while desirable, is a secondary issue.

9. Programmability objective of the Navier-Stokes machine is to provide a FORTRAN-like high level language with extensions necessary for efficient problem mapping and the following features.

   a. A stable optimizing compiler

   b. Good compiler diagnostics

   c. Warning from the compiler of possible run-time inefficiencies

   d. Ability to give good run-time diagnostics and statistics

e. Vector length independence

f. Freedom from the need to do explicit mode vector manipulation

g. Ease in specifying data allocation

10. Front-end system functional objectives are in the following general categories.

a. Direct support of the NSS

1) Compiling of NSS programs

2) Linking of NSS programs

3) Scheduling of NSS jobs

4) NSS hardware diagnostics

5) Program debugging support

b. Direct support of facility users

1) Grid generation and modification

2) Body geometry generation and modification

3) Grid and body geometry display

4) Data reduction and display

5) Program debugging facilities

6) Normal interactive facilities

c. Direct support of facility hardware

1) Front-end file system

2) Remote user communications

3) Standard peripherals

4) Support of NSS data base

5) Link to NSS monitor

6) System diagnostics

11. NSS output objectives are:

    a. Snap Shots

        1) Integrated quantities such as drag, lift, and moments approximately every 15 to 30 seconds.

        2) Surface quantities such as pressure and skin friction; up to approximately 60,000 words output every 15 to 30 seconds.

        3) Flow quantities in the field such as pressure or Mach number require the heaviest output, and for 60-minute runs, would accumulate up to 50 million words.

    b. Restart Dumps

        Restart dumps would be taken at the end of many cases and would be taken about every 10 minutes for long jobs. The restart dump information would include:

3	grid coordinates
5	conservation variables
1 through 7	turbulence variables
9 through 15	words per grid point

or 9 through 15 million words for 1 million grid points.

    c. Debug Dumps

        Debug dumps would be on the same order of size as restart dumps.

    d. Formatted I/O

        Formatted I/O for at least a limited amount of data is required.

12. The NASF is intended for 24-hour-per-day, 7-day-per-week operation, given sufficient time for routine preventive maintenance.

13. The facility must be capable of producing a minimum of 120 hours of flow simulations per week while concurrently providing problem preparation and result analysis.

14. The useful operational lifetime of the facility is to be at least 10 years.

COMPATIBILITY OBJECTIVES

1. Front-end processors will be standard, general-purpose machines of the latest technology, and therefore, fully compatible with the product line chosen for selection.

2. Peripheral subsystems and devices, likewise, will be standard and compatible with product lines. In particular, interchangeable media such as disk packs, magnetic tapes, and punched cards will be compatible for interchangeability reasons.

3. In order to benefit from past and current efforts, thereby reducing development time and cost as well as parts, diagnostic, and maintenance cost, the NSS will be compatible with the STAR-100C to the extent possible without sacrificing aerodynamic simulation performance in the following respects.

   a. Circuit (semiconductor) technologies

   b. Memory devices (components and modules)

   c. Packaging and cooling

   d. Data formats

   e. Vector processing concepts

   f. Some functional units

ACQUISITION COST OBJECTIVES

The acquisition cost objective for the Numerical Aerodynamic Simulation Facility is 36 million dollars in 1982. This is to include all costs for acquiring the necessary hardware and software, but cost of a building (or housing) for the facility is specifically not included. Also not included in this cost objective are hardware and software development costs (covered in section 5) and operating and support costs (covered in section 6).

APPENDIX A
CDC STANDARDS CHECKLIST

Title	Industry or Joint Standards	CDC-PUB/ CDC-STD
Cateogry - CODES		
Code for Information Interchange	ANSI X3.4	
Code for Information Interchange	FIPS PUB 1	
7-Bit Coded Character Set for Information Processing Interchange	ISO 646	
Implementation of the Code for Information Interchange and Related Media Standards	FIPS PUB 7	
Code Extension Techniques for Use With the 7-Bit Coded Character Set of ASCII	ANSI X3.41	
Code Extension Techniques in 7 or 8 Bits	FIPS PUB 35	
Code Extension Techniques for Use With the 7-Bit Coded Character Set	ISO 2022	
Control Data Subset of ASCII		1.10.003
Subsets of the Standard Code for Information Interchange†	FIPS PUB 15	
Perforated Tape Code for Information Interchange	ANSI X3.6	
Perforated Tape Code for Information Interchange	FIPS PUB 2	
Representation of 6- and 7-Bit Coded Character Sets on Punched Tape	ISO 1113	
Hollerith Punched Card Code	ANSI X3.26	
Hollerith Punched Card Code	FIPS PUB 14	
Representation of the 7-Bit Coded Character Set on 12-Row Punched Cards	ISO 1679	
Representation of 8-Bit Patterns on 12-Row Punched Cards	ISO/R 2021	
Representation of Numbers in Packed Decimal Form		1.10.016

†The standard has selectable options.

Title	Industry or Joint Standards	CDC-PUB/ CDC-STD
Category - PROGRAMMING LANGUAGES		
COBOL	ANSI X3.23	
COBOL	FIPS PUB 21-1	
Programming Language COBOL	ISO/R 1989	
FORTRAN	ANSI X3.9	
Basic FORTRAN	ANSI X3.10	
Programming Language FORTRAN	ISO/R 1539	
ALGOL		1.86.003
Category - SPECIAL PURPOSE LANGUAGES		
APT	ANSI X3.37	
Industrial Computer System FORTRAN Procedures for Executive Functions and Process Input/Output	ISA S61.1	
Category - OPERATING SYSTEMS		
Magnetic Tape Labels for Information Interchange		1.87.002
Magnetic Tape Error Detection and Recovery		1.87.004
System Error Recovery for Rotating Mass Storage		1.87.005
Category - DATA REPRESENTATION		
Representation of Calendar Date and Ordinal Date for Information Interchange	ANSI X3.30	
Calendar Date	FIPS PUB 4	
Representation of Ordinal Dates	ISO 2711	
Guidelines for Describing Information Interchange Formats	FIPS PUB 20	
Representation of SI and Other Units in Systems With Limited Character Sets	ISO 2955	

Title	Industry or Joint Standards	CDC-PUB/ CDC-STD
Category - DATA COMMUNICATION		
Mode 4C Data Communication Control Procedure		1.10.020
Synchronous Signaling Rates	ANSI X3.1	
Synchronous High Speed Signaling Rates	ANSI X3.36	
Synchronous Signaling Rates Between Data Terminal and Data Communication Equipment	FIPS PUB 22	
Bit Sequencing of ASCII in Serial-by-Bit Data Transmission	ANSI X3.15	
Bit Sequencing of the Code for Information Interchange in Serial-by-Bit Data Transmission	FIPS PUB 16	
Character Structure and Character Parity Sense for Serial-by-Bit Data Communication in ASCII	ANSI X3.16	
Character Structure and Character Parity Sense for Serial-by-Bit Data Communication in the Code for Information Interchange	FIPS PUB 17	
Character Structure for Start/Stop and Synchronous Transmission	ISO 1177	
Character Structure and Character Parity Sense for Parallel-by-Bit Data Communication in ASCII	ANSI X3.25	
Character Structure and Character Parity Sense for Parallel-by-Bit Data Communication in the Code for Information Interchange	FIPS PUB 18	
Signal Quality at the Interface Between Data Processing Terminal Equipment and Synchronous Data Communication Equipment for Serial Data Transmission	ANSI X3.24	
Procedures for Using the Control Characters of ASCII in Specified Data Communication Links	ANSI X3.28	
Basic Mode Control Procedures for Data Communication Systems	ISO 1745	

Title	Industry or Joint Standards	CDC-PUB/ CDC-STD
Data Terminal and Data Communication Interchange Circuits - Assignment of Connector Pin Numbers	ISO 2110	
Connector Pin Allocations for Use With High-Speed Data Terminals	ISO 2593	
Basic Mode Control Procedures - Code Independent Information Transfer	ISO 2111	
Basic Mode Control Procedures - Complements	ISO 2628	
Basic Mode Control Procedures - Conversational Information Message Transfer	ISO 2629	
Use of Longitudinal Parity to Detect Errors in Information Messages	ISO 1155	
Determination of the Performance of Data Communication Systems	ANSI X3.44	
Category - KEYBOARDS		
Ten-Key Keyboards for Numeric Data Entry		1.10.004
Typewriter Keyboards	ANSI X4.7	
Alphanumeric Keyboard Arrangements for ASCII and OCR	ANSI X4.14	
Keyboard for International Information Processing Interchange	ISO 2530	
Keyboards for Countries Whose Languages Have Alphabetic Extenders - Guidelines for Harmonization	ISO 3243	
Function Key Symbols on Typewriters	ISO/R 1090	
Layout of Printing and Function Keys on Typewriters	ISO/R 1091	
Basic Arrangement for the Alphabetic Sections of Keyboards	ISO/R 2126	
Principles Governing the Positioning of Control Keys on Keyboards	ISO/R 3244	

Title	Industry or Joint Standards	CDC-PUB/ CDC-STD
Cateogry - DOCUMENTATION		
Hardware Configuration Management Standards Manual (Entire Contents Included)		1.01.000
Software Configuration Management Standards Manual (Entire Contents Included)		1.01.100
Preparation of Microcircuit Procurement and Acceptance Test Specifications		1.03.007
Terminology for Referencing American National Standards		1.10.010
Vocabulary for Information Processing		1.10.012
Reliability, Availability, and Maintainability Standards		1.12.000
Graphic Symbols for Electrical and Electronic Diagrams		1.41.101
Reference Designations for Electrical and Electronic Parts and Equipment		1.41.102
Graphic Symbols for Logic Diagrams		1.41.104
Logic Diagrams		1.41.108
Hardware and Software Product Support Manual Standards Manual (Entire Contents Included)		1.50.000
Flowchart Symbols and Usage		1.80.003
Microcircuit Handbook		15006100

Title	Industry or Joint Standards	CDC-PUB/ CDC-STD
Category - CHARACTER RECOGNITION		
Print Specifications for Magnetic Ink Character Recognition	ANSI X3.2	
Print Specifications for Magnetic Ink Character Recognition	ISO/R 1073	
Bank Check Specifications for Magnetic Ink Character Recognition	ANSI X3.3	
Coding of Character Sets for MICR and OCR	ISO 2033	
Character Set and Print Quality for Optical Character Recognition (OCR-A)	ANSI X3.17	
Character Set for Optical Character Recognition (OCR-B)	ANSI X3.49	
Optical Character Recognition Character Sets†	FIPS PUB 32	
Alphanumeric Character Sets for Optical Recognition†	ISO/R 1073	
Printing Specifications for Optical Character Recognition	ISO/R 1831	
Character Set for Handprinting	ANSI X3.45	
Character Set for Handprinting	FIPS PUB 33	
Specifications for Credit Cards	ANSI X4.13	
Optical Reader Subsystem Testing		1.88.001
Category - ROTATING MAGNETIC MEDIA		
Unrecorded Magnetic Six-Disk Pack (General, Physical, and Magnetic Characteristics)	ANSI X3.46	
Interchangeable Magnetic Six-Disk Pack - Physical and Magnetic Characteristics	ISO 2864	
Category - MAGNETIC CASSETTE/CARTRIDGE MEDIA		
Implementation of the 7-Bit Coded Character Set and its 7-Bit and 8-Bit Extensions on 3.81-mm Magnetic Tape Cassettes for Information Interchange	ISO 3275	

†The standard has selectable options.

Title	Industry or Joint Standards	CDC-PUB/ CDC-STD
Category - MAGNETIC TAPE MEDIA		
Recorded Magnetic Tape, 0.50 Inch, 9-Track, 800 CPI, NRZI		1.10.005
Recorded Magnetic Tape, 0.50 Inch, 9-Track, 1600 CPI, Phase Encoded		1.10.006
Unrecorded Magnetic Tape, 0.50 Inch		1.10.007
Recorded Magnetic Tape, 0.50 Inch, 7-Track, 200 CPI, NRZI		1.10.013
Category - PAPER CARD MEDIA		
General Purpose Paper Cards and Punched Hole Requirements		1.10.008
80-Column Card Files for Information Interchange		1.10.009
Category - PAPER TAPE MEDIA		
Eleven-Sixteenth-Inch Perforated Paper Tape for Interchange	ANSI X3.19	
One-Inch Perforated Paper Tape for Information Interchange	ANSI X3.18	
One-Inch Perforated Paper Tape for Information Interchange	FIPS PUB 26	
Dimensions of Punched Paper Tape for Information Interchange	ISO 1154	
Take-Up Reels for One-Inch Perforated Paper Tape for Information Interchange	ANSI X3.20	
Take-Up Reels for One-Inch Perforated Paper Tape for Information Interchange	FIPS PUB 27	
Specifications for Properties of Unpunched Oiled Paper Perforator Tape	ANSI X3.29	
Properties of Unpunched Paper Tape	ISO 1729	
Interchange Rolls of Perforated Paper Tape for Information Interchange	ANSI X3.34	
Data Interchange on Rolled Up Paper Tape	ISO 2195	

Title	Industry or Joint Standards	CDC-PUB/ CDC-STD
Category - MECHANICAL DESIGN		
Product Identification Emblems		1.20.006
Acoustical Noise†	31-04	1.20.007
Operating and Maintenance Meters		1.20.008
Industrial Design		1.20.010
Category - ELECTRICAL DESIGN		
General Design Standard for Electronic Power Supplies		1.30.001
Equipment Safety	35-01	1.30.003
Color Coding of Wires, Harnesses, and Cables	35-02	1.30.005
Cable Classification and Marking		1.30.008
Computer and Peripheral Equipment Design Requirements		1.30.011
EMC Performance Requirements	31-03 and 31-08	1.30.022
Digital Computer System Grounding		1.30.023
Analog Computer System Grounding		1.30.024
EMI Suppression and Certification		1.30.025

†The standard has selectable options.

Title	Industry or Joint Standards	CDC-PUB/ CDC-STD
Category - GENERAL DESIGN		
Component Selection		1.03.002
Component Qualification		1.03.003
Qualified Vendor List		1.03.006
Electronic Logic Packaging		1.03.006
Microcircuit Selection		1.03.010
Power Connector Common Parts List		15004300
Wire Common Parts List		14005500
Microcircuit Handbook Volume II		15006100
Transistor Common Parts List		15006600
Category - ENVIRONMENTAL		
Application Guidelines†		1.03.201
Temperature, Humidity, and Barometric Pressure†	31-01	1.03.202
Vibration and Shock		1.03.203
Air Cleanliness†	31-06	1.03.205
Illumination	31-07	1.03.206
Input Power and Grounding†		1.03.207
Physical Characteristic†	31-09	1.03.209

†The standard has selectable options.

These risks have been present for virtually any project ever begun but have in the last 10 years been very evident in the high-technology, high-risk ventures such as ILLIAC and STAR. The degree of nation-wide interest in yet another attempt at producing the world's greatest "whizz-bang" is going to create great pressures for the project managers and developers. Certainly every attempt should be made to reduce the risks in each category, and at the very least, expose those risks and their status at every step in the project.

The volume of work to be accomplished is definitely the most awesome thing encountered when taking the broad view of this project. Control Data's experience with production of finished designs using the latest high performance LSI technology has been that one LSI board, containing about 25,000 gates, requires from 16 to 28 man-months to complete to the point of release of final artwork. Since no more than two people can work in such a confined space at a time, this implies a lead time of from 8 to 14 months to complete the design of a single 25,000 gate entity. Since the proposed CDC system (section 5) possesses about 36 unique board types, this would mean that about 1000 man-months could be consumed at worst. If 2 full years were available for the design effort, over 40 men would be required to accomplish the task. In Control Data's experience with the 6600, 7600, and STAR, such a large number of designers with appropriate skills cannot be assembled together at one time, nor would the total group be completely effective. It has been found at Control Data that a design team of 12 to 16 people is the maximum that could be effectively used on the supercomputer projects. Thus, some means must be used to reduce the number of unique part and board types to meet the NASF schedule and probable available resources. The technique proposed by Control Data at the outset of this project consisted of using existing parts and existing designs, as much as possible, from machines possessing the qualities needed by the NASF engine. At first it appeared that much of the STAR-100C computer could be utilized as the basic building blocks of the NASF computational engine. However, as the metric analysis and cost estimates developed, it became obvious that some radical departures needed to be taken from the STAR machine line to meet the 1-gigaflop objective. In several areas useful employment of materials developed for STAR has been found.

- The STAR simulator software has evolved to a fairly mature state in features and performance. This system is more than adequate to support NASF machine development.
- Classes of array or part types have been developed that, when mapped onto a denser, higher speed technology, will be sufficient for most of the NASF design.
- Memory interchange design and memory packaging can be almost identical between the NASF and STAR machines.
- Floating-point formats and thus the basic floating-point unit design can be used for both machines.
- The scalar portion of the machine could be utilized if converted to a newer technology, thus reducing the design time in those areas.

By this method the number of unique board types needed for the NASF would be about 16, while the remaining 20 board types would essentially be converted directly (almost one-to-one) to a newer technology family (both circuit and packaging). This would reduce the design team requirements in the worst case to about 18 people, while optimistically about 10 to 12 people can do the job in 24 months.

In the software area, there is also a substantial amount of systems software that could be developed. The only way to reduce this risk is to almost totally eliminate specialized software that operates on the computational engine, while beginning early to develop the necessary system control software on the front-end computer.

This means that file management, access and transfers, scheduling, and I/O actions should all be controlled by the front-end machine and thus totally absent from the NASF engine. Further, after much study, Control Data agrees with Ames researchers that multiprogramming of the NASF engine complicates the software requirements and hardware for the computational facility. Thus, most support software (including perhaps compilers) should reside on the front-end machine if suitable optimization can be obtained there.

It is believed that the technology risk is a necessary evil of this project. As stated in the technology summary, the next generation LSI seems to be required for the development of the NASF. The experience in producing the first generation of high performance LSI for Control Data has been sobering, as each technological hurdle has seemed to be insurmountable. It is felt, however, that this first generation experience will pay off in the reduction of risk for the next high speed LSI development. The microscopic and macroscopic software and engineering tools are now in place at both the vendor and computer manufacturer to provide support that was just developing during the first generation exercise. Only the careful monitoring of the technology development and vendor capabilities by knowledgeable project personnel can keep this risk from overwhelming the NASF development.

The first and second ancillary risks are dealt with in part by the use of simulation and other analytical techniques throughout the design phase of the NASF. There is no reason why, at the completion of the total facility design, firm specifications for the final performance metrics cannot be in place. The basic kernels of these metrics should have been completely simulated by the software representation of the hardware design to guarantee that the hardware is functionally capable and that it meets the performance requirements.

Finally, the third ancillary concern relates to the manner and methodology of rigorous project management. It would appear to be in the best interests of Ames to have on its staff, or under contract, knowledgeable people in the computer field to aid NASA in the monitoring and control of the entire facility development. While, for instance, Control Data might be totally responsible for the development of the NASF engine and perhaps the I/O and front-end systems, other subsystems might be procured at Ames request from alternate vendors. Since the installation and software development for such systems might involve the combined efforts of vendors, NASF developers, and Ames personnel, it appears obvious that an overall supervising authority will be needed to reduce the risks of project interfaces failing.

In general, the entire risk assessment and reduction necessary for the success of the NASF depends on the developers' ability to deal with the engineering realities and to scale the work to be done sufficiently to ensure the buildability and integrity of the entire complex.

REFERENCES

The following is a list of all references that are called out in the text of this section.

Reference Number	Reference
2-1	Flynn, Michael J.: Some Computer Organizations and Their Effectiveness. IEEE Transactions on Computers, Vol. C-21, no. 9, September 1972, pp. 948-960.
2-2	Thurber, Kenneth J.; and Patton, Peter C.. The Future of Parallel Processing. IEEE Transactions on Computers, Vol. C-22, no. 12, December 1973, pp. 1140-1143.
2-3	Kuck, David J.; Muraoka, Yoichi; and Chen, Shyh-Ching: On the Number of Operations Simultaneously Executable in Fortran-Like Pograms and Their Resulting Speedup. IEEE Transactions on Computers, Vol. C-21, no. 12, December 1972, pp. 1293-1310.
2-4	Weissberger, Alan J.: Analysis of Multiple-Microprocessor System Architectures. Computer Design, Vol. 16, no. 6, June 1977, pp. 151-163.
2-5	Spetz, William L.: Microprocessor Networks. Computer, Vol. 10, no. 1, July 1977, pp. 64-70.
2-6	Akkoyunlu, Eralp; Bernstein, Arthur; and Shantz, Richard: Interprocess Communication Facilities for Network Operating Systems. Computer, Vol. 7, no. 6, June 1974, pp. 46-54.

Reference Number	Reference
2-7	Bhandarkar, Dileep P.: Analysis of Memory Interference in Multiprocessors. IEEE Transactions on Computers, Vol. C-24, no. 9, September 1975, pp. 897-908.
2-8	Juliussen, J. Egil; and Mowle, Frederic J.: Multiple Microprocessors with Common Main and Control Memories. IEEE Transactions on Computers, Vol. C-22, no. 11, November 1973, pp. 999-1007.
2-9	Iverson, Kenneth E.: A Programming Language. John Wiley and Sons, Inc., 1962, Chapter 1.

SECTION 5

PRELIMINARY SYSTEM DESIGN

This section constitutes the heart of this study report, the development of a total system design to meet the objectives of an operational NASF installation in 1982. The objectives call for:

- The complete solution of a flow field simulation using either the implicit or explicit form of the time-averaged Navier-Stokes equations for grid sizes up to 100x100x100 points in no more than 7 to 15 minutes actual elapsed time
- Concurrent provision of facilities for interactive and batch mode problem definition, setup, and debug including mesh generation, with guaranteed response times acceptable to users in interactive environments
- Concurrent provision of systems management, scheduling, accounting, and information storage and retrieval in support of the flow field simulations
- Capability for concurrent development of new computer programs and system software to improve the behavior and performance of the total NASF

To be able to describe in detail some of the system components, as well as to provide estimates of power and cooling requirements, floor space occupied, and costs, all specimen components with the exception of the Navier-Stokes Solver (NSS) have been drawn from Control Data's standard equipment offerings. Since such actual equipments will not be procured until the 1980 time-frame, the actual brand-names or equipment identifications can be expected to change as more advanced, cost-effective hardware becomes available in the next 3 years.

Details of the equipment configurations and requirements for support may be found in section 6 of this report. Although the majority of this section is devoted to a detailed description of the NSS, some of the salient features of the overall system should be addressed. However, since the flow field simulation model is the basis of this design, a brief tutorial on the model is presented first.

FLOW FIELD SIMULATION MODEL

The fundamental purpose in the creation of the NASF site is to provide aerodynamicists and airframe designers with the ability to simulate the flow of air around various aerodynamic configurations. The mathematical approach to implementing this computer simulation is to solve a system of partial differential equations that describe the physics of fluid flow in terms of conservation of momentum, energy, and mass within the fluid. If this system of equations, known as the Navier-Stokes equations, can be solved for a specific geometrical shape suspended in a specific fluid of interest, results from the model are developed which can closely approximate physical reality. The correlation between the simulation results and actual physical phenomena can be determined in many cases by direct measurement of parameters in wind tunnels, or in some cases, by sampling parameters from aircraft in flight. Once the validity of such models has been established, the simulation system can then be used to supplement wind tunnel experimentation in the design of aerospace vehicles.

The complete solution of the Navier-Stokes equations for three-dimensional bodies in an airstream with velocity greater than 1 mile per hour is not yet possible with direct methods by man or computer. This means that the full set of equations with all the requisite terms, including those for turbulence and dissipation (including viscosity effects) is, in the end, a monumental set of differential equations, which would consume incredible amounts of computing horsepower to reach solutions of interest to aircraft designers. Thus, the physicist and mathematician join forces to reduce the extensiveness and complexity of the equations by making judgements involving physical and mathematical approximations and compromises. One of the compromises that is introduced involves the treatment of turbulence. Since the turbulent velocity field can be regarded as a random vector field, the type of average involved in the Reynolds stresses is the ensemble average. Experimentally such averages are never encountered. The averages which are derived experimentally are mainly time averages. Hence, Maxwell's ergodic hypothesis, that is, the time average equals the ensemble average, is used to relate the Navier-Stokes equations in the abstract to a set of equations computationally viable and experimentally verifiable, the so-called time-averaged Navier-Stokes equations.

Among the several candidate computer solution approaches for the set of time-averaged Navier-Stokes equations, the implicit solution involving the solution of blocks of simultaneous equations has been of particular interest. The implicit solution methods characteristically exhibit better mathematical stability than some of the other proposed methods, while promising a potential reduction in computations needed to reach a solution. Thus, the analysis conducted by Steger, Lomax, and Pulliam at Ames laboratories has been directed toward extension of existing algorithms into three dimensions from two dimensions, while simultaneously developing more efficient programs for the CDC 7600 and the ILLIAC IV computers on site at Ames. Since the analysis conducted by Control Data and several architectural features of the proposed NSS are based on the implicit methods, the following presents a perhaps simplified introduction into the algorithms being used, their programming for conventional machines, such as the 7600, and their implementation on an NSS-type architecture.

The solution methodology involves the computer integration of the time-averaged Navier-Stokes equations for a particular geometry, a specific orientation of that geometry in the airstream, and a representative set of velocities, densities, and energies. The first step is to specify a geometry by some means. Usually this involves establishing a coordinate system within which the geometry can be described as a set of points with spatial meaning. Figure 5-1 shows a rough representation of a specimen airframe surrounded by a mathematician's imaginary box. A possible side view of the airframe in the box shows a wing cross-section in a two-dimensional plane. Here, a conventional X-Y grid has been chosen, which will be called the Cartesian coordinate system, on which to describe the wing geometry much as a draftsman designing the structure might draw and measure its elements. Thus, the rightmost point on the airfoil (the trailing edge) could then be given a set of coordinates (X=9, Y=4), and all other points along the surface of the airfoil could be given similar coordinates. Obviously, a more refined set of measures (in terms of accuracy) is used in an actual airframe specification. It is expected that geometrical descriptions such as these will be retained on a mass storage-oriented data base at the NASF, allowing the configuration of complex structures by assembling the descriptions of substructures through input data, perhaps from graphics terminals.

Once a Cartesian description of the structure can be defined, many human-oriented tasks can be undertaken, using graphics terminals, plotters, and the like to display the structure and allow further refinements and adjustments. With that task completed, the geometry can be transformed into another mathematical world where the properties of implicit solution can be applied. To accomplish this, a new coordinate system is created, in this case with directions related to the ordinals of interest in the final solution. Beginning in the wake (following the trailing edge of the wing in the two-dimensional cross-section), a starting point is chosen. From there one proceeds along the wake to the trailing edge, then along the upper surface of the wing, and finally back along the bottom surface of the wing out into the wake to the starting point. At regular intervals, points

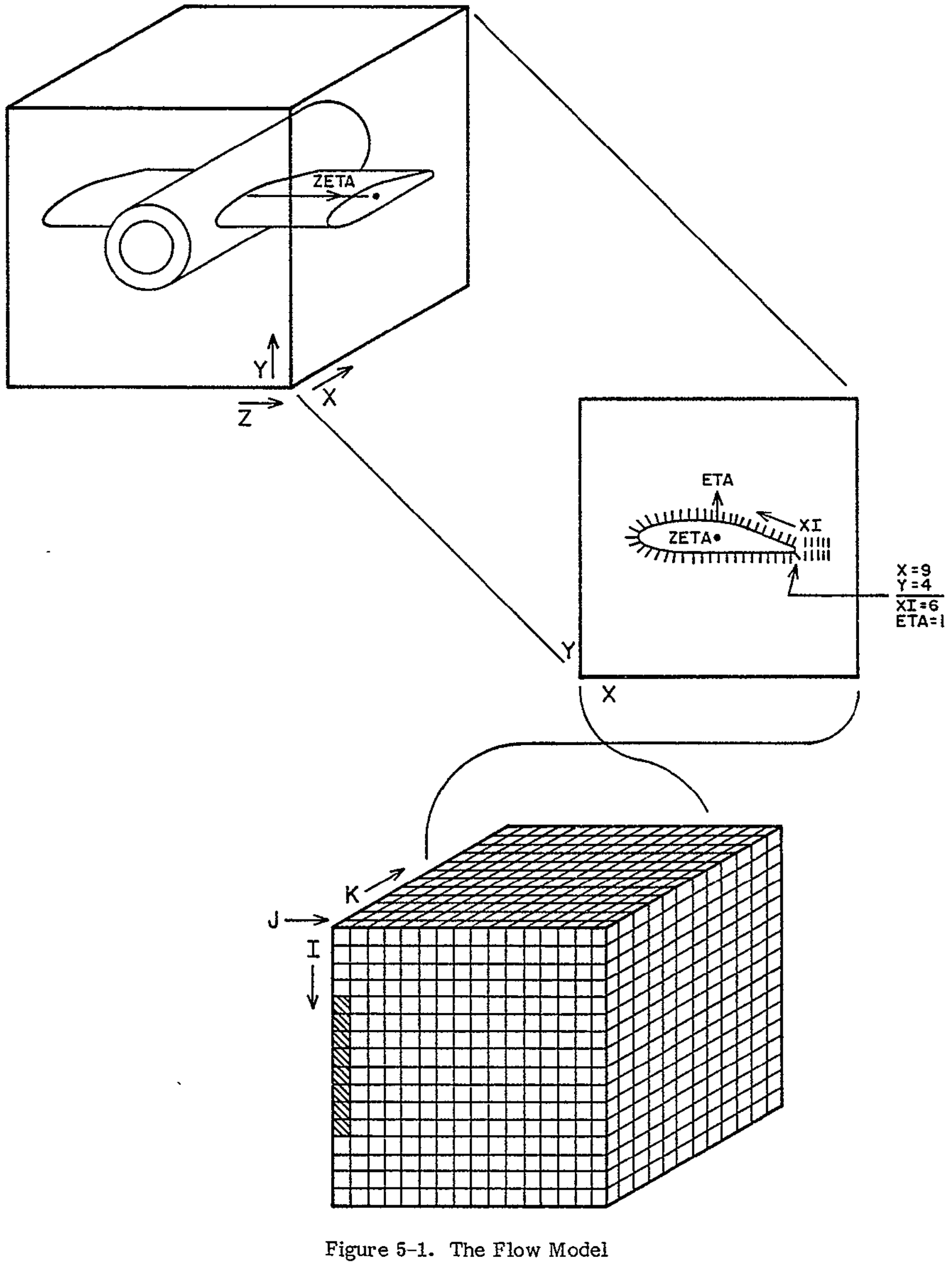


Figure 5-1. The Flow Model

are chosen (shown as hash marks in the figure) which can be related to the Cartesian coordinates. The direction of travel chosen for this path will be called the XI-direction. Movement outward from, or normal to, the path just described (from the wake, around the wing, and back to the wake) is in the ETA-direction. Thus, a point on the wing at X=9, Y=4 could be described in the new coordinate system as XI=6, ETA=1. Movement in the third dimension would be in the Z-direction for the Cartesian coordinate system and in the ZETA-direction for the new coordinate system. In this case, ZETA would relate to points (or planes) along the long axis of the wing. If lines are drawn in each of the ZETA-, ETA-, and XI-directions at regular intervals, a mesh is generated whose coordinates reflect the surface of the three-dimensional body. The conversion between the mesh and the corresponding Cartesian coordinates can easily be accomplished with a table of values, to whatever accuracy is desired for transforming the geometry at hand.

The next step, more difficult to visualize, is to unwrap the mesh just generated, and from it form a regular rectangle (for two-dimensional solutions) or rectangular solid (in the case of three-dimensional solutions) as shown at the bottom of figure 5-1. The need for this new form arises from the mathematical methods employed in the implicit solution, as well as the fact that computer solutions in some cases can be more efficient when the mesh is homogeneous and contiguous (without voids). The process of unwrapping the mesh is complex in terms of computation, and depending upon the complexity of geometry and the sophistication of the conformal mapping technique, the computer solution can easily consume many hours of 7600 time. The result is to create a matrix of values for every point in the new, uniform solid which provides the mapping function for the geometry and for the set of data representing the airstream, called flow variables. The flow variables describe the properties of density, energy, and velocity components in the X-, Y-, and Z-directions. Generally, the density is called RHO, the energy E, and the velocity components V_x, V_y, and V_z.

A small cube in the Cartesian space, then, has a unique set of these properties (RHO, E, V_x, V_y, and V_z). In the process of transforming the Cartesian system into the new coordinate system, a matrix of values is created for every point in the I, J, K solid. These values can be used arithmetically to convert the Cartesian form of the flow variables into the computational mesh form, or vice versa (of particular importance when creating displays of the results or inputting new data). This conversion matrix can be called a metric tensor which is used to modify vectors in one system to create vectors in a new system. The components of this matrix for a single point in the I, J, K solid are the partial derivatives of the newly chosen XI, ETA, ZETA coordinates with respect to their Cartesian X, Y, Z counterparts. Thus, at point I=1, J=1, K=1, the following matrix applies:

$$\begin{bmatrix} \frac{\partial XI}{\partial x} & \frac{\partial ETA}{\partial x} & \frac{\partial ZETA}{\partial x} \\ \frac{\partial XI}{\partial y} & \frac{\partial ETA}{\partial y} & \frac{\partial ZETA}{\partial y} \\ \frac{\partial XI}{\partial z} & \frac{\partial ETA}{\partial z} & \frac{\partial ZETA}{\partial z} \end{bmatrix} = XY\,(1,1,1)$$

While this matrix for the single point is called XY (1,1,1), in keeping with Ames usage in the implicit code the entire matrix of matrices will be called XY, there being a matrix of nine values for each point I, J, K in the transformed system. If one computes the determinant of the upper 3x3 submatrix, the resulting scalar value (called the Jacobian determinant) can be applied to result values such as velocities to produce the stretched magnitude of the velocities in the other coordinate system.

The culmination of the application of the transform method is to yield the solid shown at the bottom of figure 5-1, where points on the surface of the wing are now transformed into the points shown in shaded blocks at J=1, K=1 and I from 5 to 13. Note that all points along the surface of the airframe in the Cartesian space will be mapped onto the surface of the rectangular solid, and in particular, all such points will lie in the I, K plane at J=1. Thus, all points in the Cartesian grid adjacent to the surface points would map into the I, K plane at J=2, and so forth. The property of having the surface and all boundaries of the Cartesian system appear on the surface (or extremes) of the transformed system is not only necessary mathematically but also facilitates the computer handling of the boundary value adjustments and computations.

INITIAL CONDITIONS

Once the transformed coordinate system has been established, the initial flow field can be generated for each point in the I,J,K space. This is done quite simply in some cases (as in the two-dimensional implicit code supplied by Lomax and Pulliam) by generating the density directly into the mesh, utilizing the Mach Number of the impinging airstream and the angle of attack of the body. In more complex three-dimensional cases, considerations of yaw and Y-direction wind components could require a complex premodeling of the flow to generate a first guess at the field surrounding the vehicle. This field in the Cartesian system would then be transformed by the metric tensor into the I,J,K space to initiate computations. For the Ames version of the implicit code, the velocity vectors are converted to momentum by multiplying the transformed velocities by the density. Thus, the quantities appear in the solution as E-energy, RHO-density, RHOU-momentum in the XI-direction (transformed X-direction), RHOV-momentum in the ETA-direction (transformed Y-direction), and RHOW-momentum in the ZETA-direction (transformed Z-direction). It is the value of these variables that is recomputed for incremental time steps until they become unchanging or recognizably oscillate between a set of values. This condition of staticizing of the flow variables or their settling into a repetitive pattern is termed the steady state solution of the system of equations for that set of geometries, boundaries, and initial conditions. In general the aerodynamic engineer is interested in examining the steady state values while the mathematician who is developing and refining the programs wishes to analyze the results at interim time steps to determine the behavior of the solution process.

THE SOLUTION

The process of yielding the steady-state solution consists of numerically integrating the set of time-averaged Navier-Stokes equations until steady-state conditions are reached. At the simplest level, numerical integration consists of computing the value of a function at various points of an independent variable (for example, the time T) and summing the approximate area under the function curve. A gross approximation consists of choosing a time interval Dt (delta T), then computing the value of the mathematical function at time zero from the initial conditions and thence at time zero plus Dt, and multiplying the average of the two functions (at time zero and time zero plus Dt) by the value of Dt to yield the area constituting the first partial sum of the numeric integral. Therefore, the major problem is to solve for the function (as a solution of the system of differential equations) at each time step t + Dt. Examination of the problem will show that a new value for each of the flow variables at a new time step is dependent upon the values of the flow variables at that particular point and also upon the values of the flow variables at the surrounding points. For example, the density of the fluid at a particular point will change from time step to time step as the momentum quantities of all adjacent points, as well as the momentum for that particular point, cause a change in mass for that point.

A method that exhibits good mathematical stability (as mentioned earlier) for this solution is to solve for all points in the I,J,K mesh simultaneously using the implicit technique. Stated in the matrix form, this is

$$AxM = F$$

where A is the three-dimensional integration operator, M represents the new value of the matrix of flow variables at time t + Dt, and F is a function of the conditions at time step t, a function also of the boundary conditions and geometry as described by the metric tensor XY. Computationally, the algorithm is implemented by factoring the integration operator into its three spatial components A_{Xi}, A_{Eta}, and A_{Zeta}, thus making the matrix solution appear as:

$$A_{Xi}(A_{Eta}(A_{Zeta}M))=F$$

where A_{Xi} is the integral operator with respect to Xi, and so forth. This technique is called the split-operator method. As a result, the first step in the solution (after computing the initial starting values for F) is to compute the inverse function A_{Xi}^{-1} and thus yield $A_{Eta}(A_{Zeta}M)= A_{Xi}^{-1}F$.

The second step is:

$$Z_{Zeta}M= A_{Eta}^{-1} (A_{Xi}^{-1}F)$$

Finally:

$$M = A_{Zeta}^{-1} (A_{Eta}^{-1} (A_{Xi}^{-1}F)) \quad .$$

This yields M, the new value of the flow variables at the next time step. In these examples, A^{-1} signifies the inverse operator, and in the case of the numerical solution being investigated for the NASF, this implies an inverse matrix operation. In effect this solution approach permits the evaluation of incremental changes in the flow variables in a given direction, due only to the effects of adjoining flow variables in that direction, and independent of flow variables in any other direction. Referring again to figure 5-1, incremental change quantities for the leftmost column (J=1, K=1) can be computed without utilizing the values in adjacent columns in either the K- or J-directions. Once the changes are computed by column, they are then computed by row (left to right, horizontal sequence of points), and finally in the K-direction. Each separate solution provides its own contribution to the changes in the flow variables which, when combined together, give the total, incremental change in the flow field. When added to the previous flow variables, this yields a new set of values at the next time step.

It is the independence of the three separate sets of calculations to yield a mathematically sound result that excites the interest of parallel programmers and parallel machine architects. This simplified model shows that, since the computations in the I-direction (for the first split-operator A_{Xi}) can be performed on all columns simultaneously and independently, parallel computing elements can be utilized quite effectively, one per column. For the purpose of this report, the application of a split-operator to the incremental solution in a given direction (XI, ETA, ZETA) is called sweeps in those directions.

SOLUTION OUTLINE

The following description presents, in very simplified terms, the mathematics that are involved in a single time step computation.

At time t=0 with the initial values of the flow variables, all other quantities initialized, and the metric tensor precomputed, the process of breaking down

$$A_{Eta}(A_{Zeta}M) = A_{Xi}^{-1}F$$

begins by computing F, the right-hand side of the system. The major portion of these calculations involves the creation of a set of flux variables from the current state of the flow variables (in this case, their initial condition). Five flux variables (corresponding to the five flow variables) are generated, and the boundary conditions and geometry considerations (provided by the metric tensor XY) are applied. The approximate derivatives formed by differencing of the variables in this process must be smoothed, to eliminate the significant discontinuities (or noise) created by the differencing and ratioing of discrete values at each point in the mesh. Once the right-hand-side matrix is computed (with a set of five variables at each point), the left-hand-side solution can proceed.

The starting flow variables are used to derive block tridiagonal matrices, one tridiagonal matrix for each column (if proceeding in the XI-direction). The solution for a single column then becomes the solution of the system

$$TxC = FV$$

where T is the block tridiagonal matrix just formed, FV is a column on the right-hand matrix, and C represents the columnar contribution for that direction (XI) to the incremental change in the flow field. A classical solution for the block tridiagonal matrix thus yields its inverse, and when multiplied by the right-hand-side FV terms, produces a new right-hand-side column for that particular value of J or K.

It should be obvious that for this system, a parallel method of solving several columns of the tridiagonal matrices at a time could improve the overall computation speed for this algorithm.

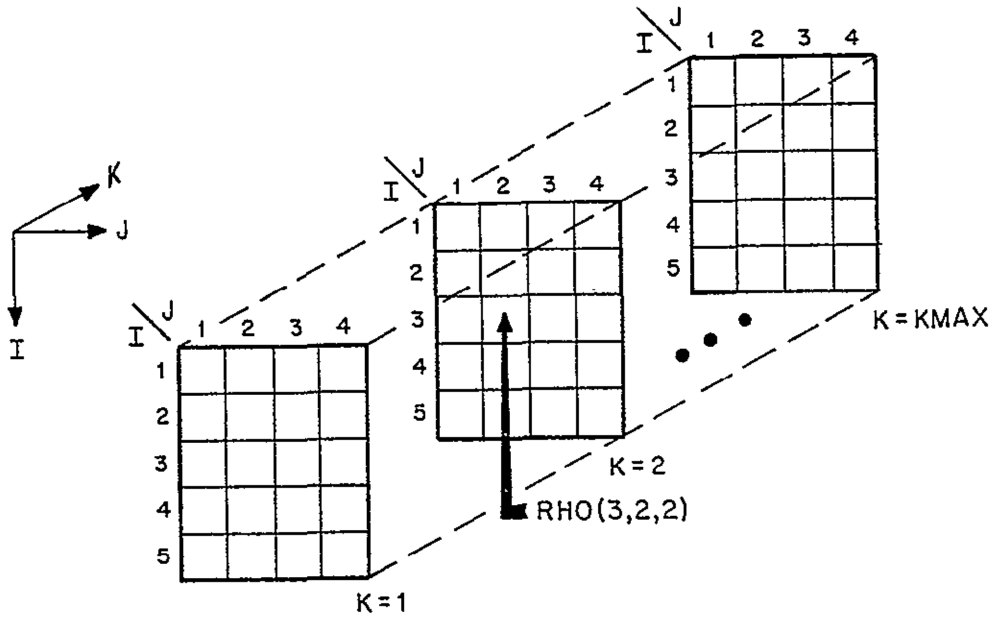
This is particularly true since the standard gaussian elimination form of solution (which is the most notably reliable method) is, in effect, quite recursive. That is, new values for the matrix are computed from previous new values, and the back substitution of all final values cannot proceed until the entire matrix has been scanned. Recursion is the anathema to speedy computing, and most certainly to parallel computation. Therefore, performing several identical calculations on independent columns simultaneously is the best method for achieving performance improvements in this environment. The split-operator process is then continued (for the ETA-direction) in the row-wise manner, forming block tridiagonal matrices for each row, and solving these matrices. Finally, the same technique is used in the ZETA-direction to yield the final incremental change factor. Once all changes have been determined, the result is added to the previous values of the flow field, to produce a new flow field for time t+1.

These results are checked for convergence, displays are produced of selected portions of the intermediate results, and the process is repeated until the problem becomes stable or reaches steady state.

STORAGE IMPLICATIONS

Up to this point no treatment has been given to the realities of computer programs that implement the algorithm just described. In fact, the allocation of the data variables to memory becomes of major concern, particularly when parallel machines are employed in their solution.

The following diagram shows the columnwise allocation of density-RHO(5,4,KMAX) and gives an example of how the flow field might be stored in actual computer memory.



From the diagram, it is seen that:

- RHO(I+1,J,K) is contiguous to RHO(I,J,K).
- RHO(I,J+1,K) is five memory cells distant from RHO(I,J,K).
- RHO(I,J,K+1) is 5x4=20 cells away from RHO(I,J,K).

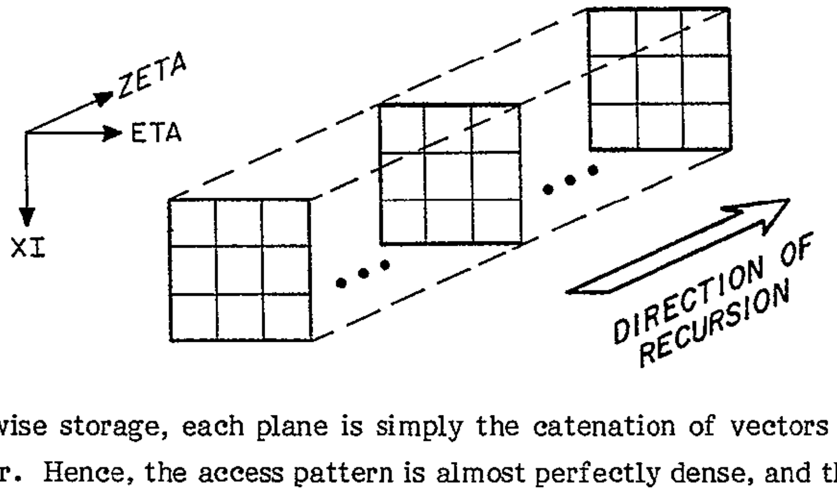
Each step of the solution process in a given direction depends directly on the result of the previous step in that direction.

For a parallel solution of all the systems of equations in the K-direction, the organization of the data array, as shown, is perfect since the I-J planes are each a series of I-vectors which constitute a long vector allowing maximum employment of parallelism. However, for a solution in the I-direction, the data organization requires transposition (gathering) to get J-K planes which are vectors. The J-direction is amendable to vector processing by transposing the algorithm's use of I and K. Therefore, when processing I-J planes, no mapping is required, while J-K planes require gathering of noncontiguous elements into vectors, and K-I planes require gathering of a series of vectors into a larger vector.

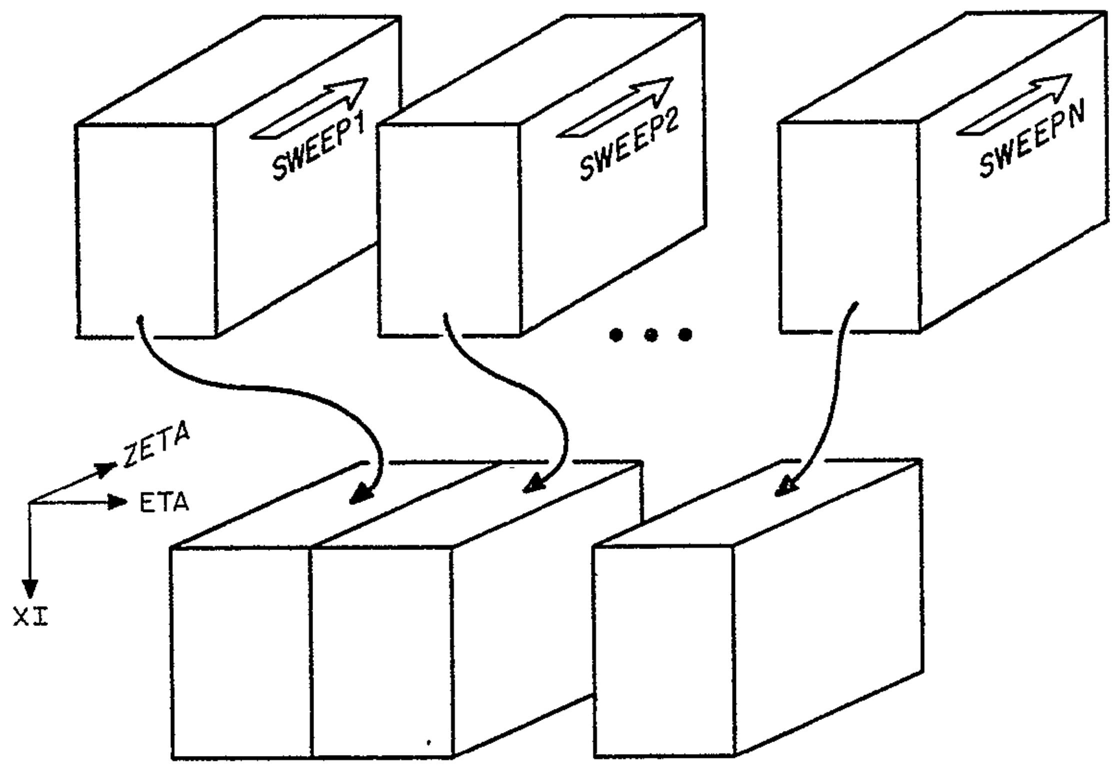
In a hierarchical memory system, as is proposed for the NSS, the allocation and accessing of memory becomes the primary programming consideration. Consider the example of formation of the matrix elements during construction of the block tridiagonal system of equations. Each step of the LU decomposition, forward elimination, and back substitution in the ETA-direction could be performed on all of the variables in the XI-direction at one time, that is, in parallel. In doing this, linear coordinated demands would be made for accession of data because the XI-direction is along the column direction of storage. With very large high speed memories, a choice exists: to either obtain very fast bulk access with an interelement stride of unity and moderately fast (1/4 or 1/8 of the rate) with nonunit strides, or throw away the excess bandwidth and design a system which works at the 1/4 rate in both modes. The path chosen is that of providing the extra bandwidth whenever possible, and through careful analysis and programming, minimizing the more costly noncontiguous vector accesses.

Thus, at the appropriate place in the sequence of calculations, mapping functions are invoked in parallel, which gather noncontiguous pencils into vectors (columnwise arrays). This is really just matrix transposition and clearly a way must be found to minimize its negative contribution in favor of the gains from processing arrays in coordinated fashion.

Consider the problem of accessing an array, for purposes of generating and solving systems of equations, in its third dimension. Assume that the entire data base and vector temporaries cannot be housed in the main memory and thus must be moved in blocks, back and forth between the high-speed main memory and a larger, lower speed backing store. (This memory configuration is described later in this section.)

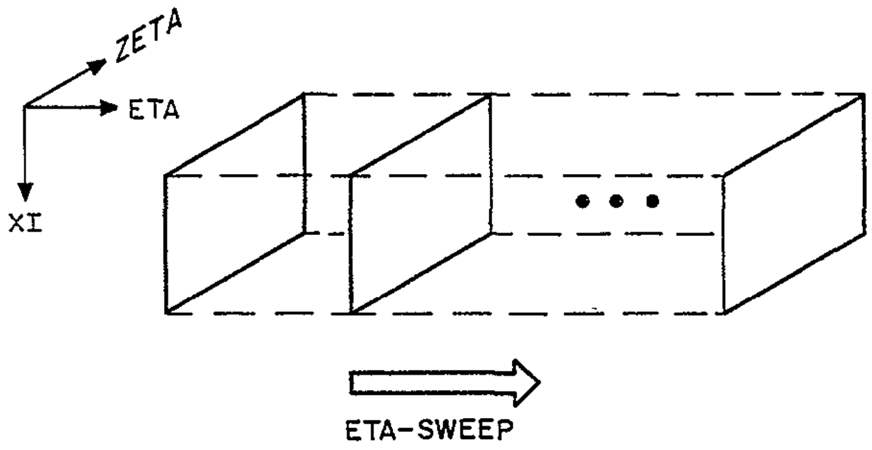


With columnwise storage, each plane is simply the catenation of vectors (columns) into one big vector. Hence, the access pattern is almost perfectly dense, and the calculations can be streamed at maximum rate with no remapping.

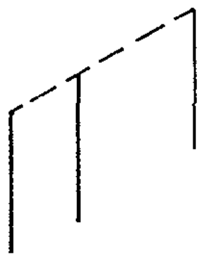


Slicing the XI-ETA planes, as is indicated above, should be done in a way that maximizes the density of reference to each backing store sector. (Note how each ZETA-sweep accesses a pencil of noncontiguous subarrays.)

When the recursive sweep is proceeding in the ETA-direction, the following occurs.

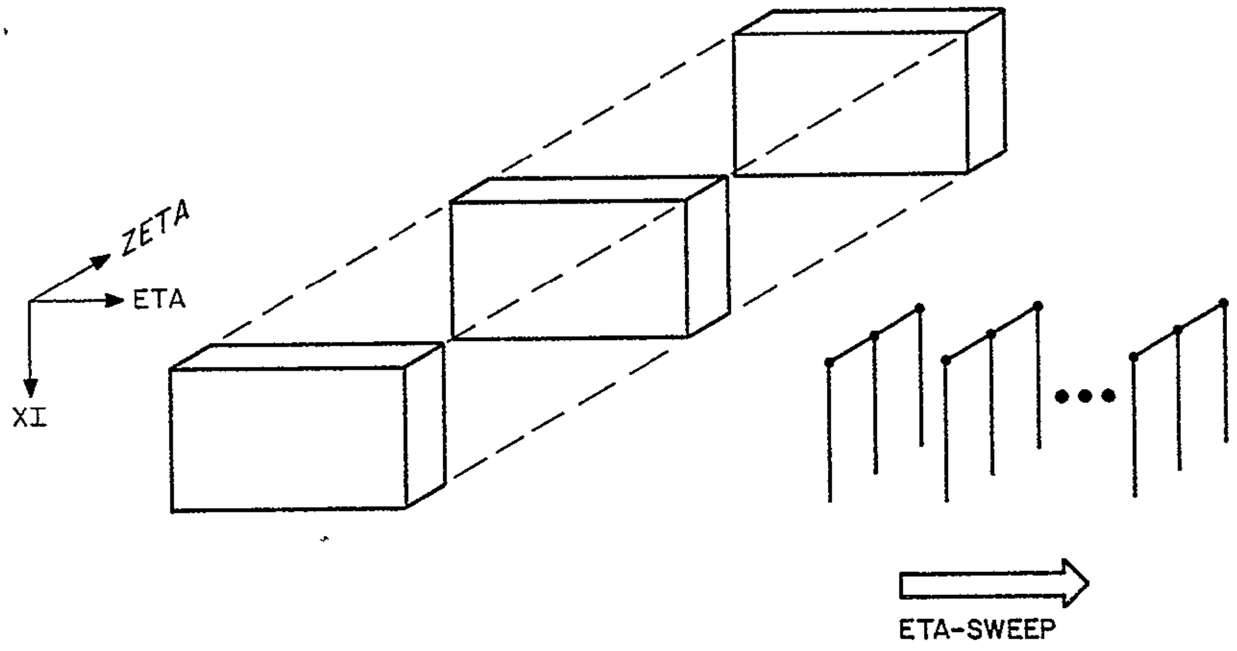


This operation might be sliced in terms of column vectors and one column processed at a time because a XI-ZETA plane is a noncontiguous sequence of vectors

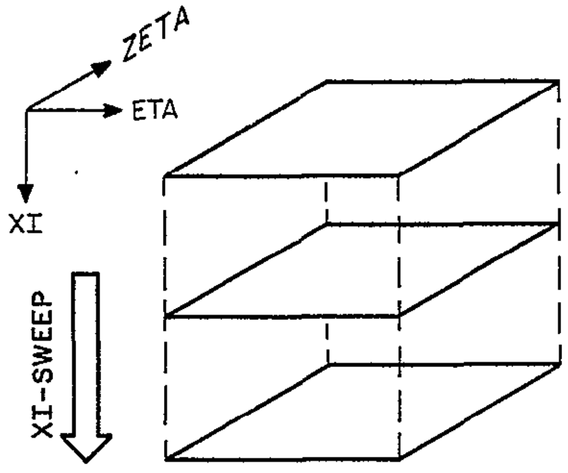


where each vector is of length equal to the first dimension, and the corresponding elements of each of the columns are separated by a stride equal to the product of the first two dimensions.

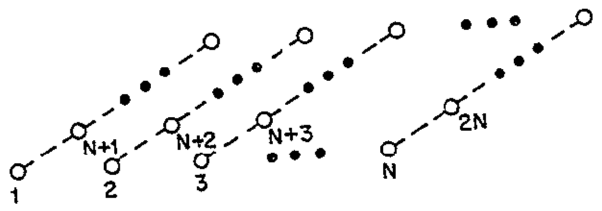
The mapping operation which is needed to aggregate a slice of the (or the whole) XI-ZETA plane into a vector is known as a "periodic gather" with groups. With judicious choice of array dimensions to avoid bank congruence, the periodic gather can proceed at a rate approaching that of a vector to vector copy (eight elements per minor cycle). The burden is that it does have to be done as an extra operation and it occupies the memory while other register-to-register operations proceed. The way one slices in this dimension is:



When the recursive sweep is to proceed in the XI-direction, the picture is:

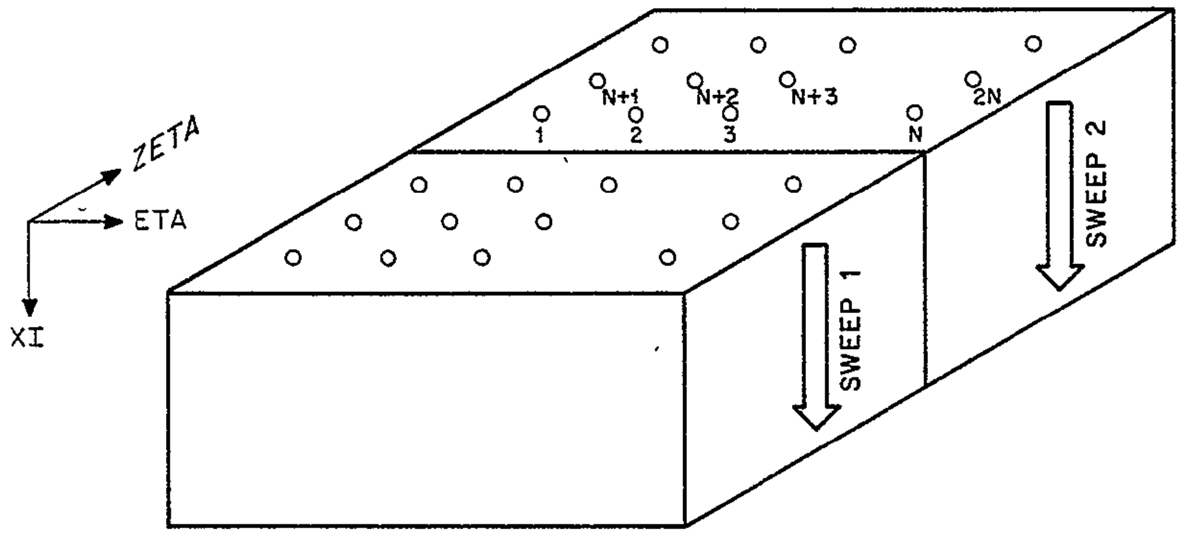


It can be immediately seen that the desired plane in which to work is a true pencil in that its elements are evenly spaced (they are uniformly one column length apart). No way exists to access this data except for:



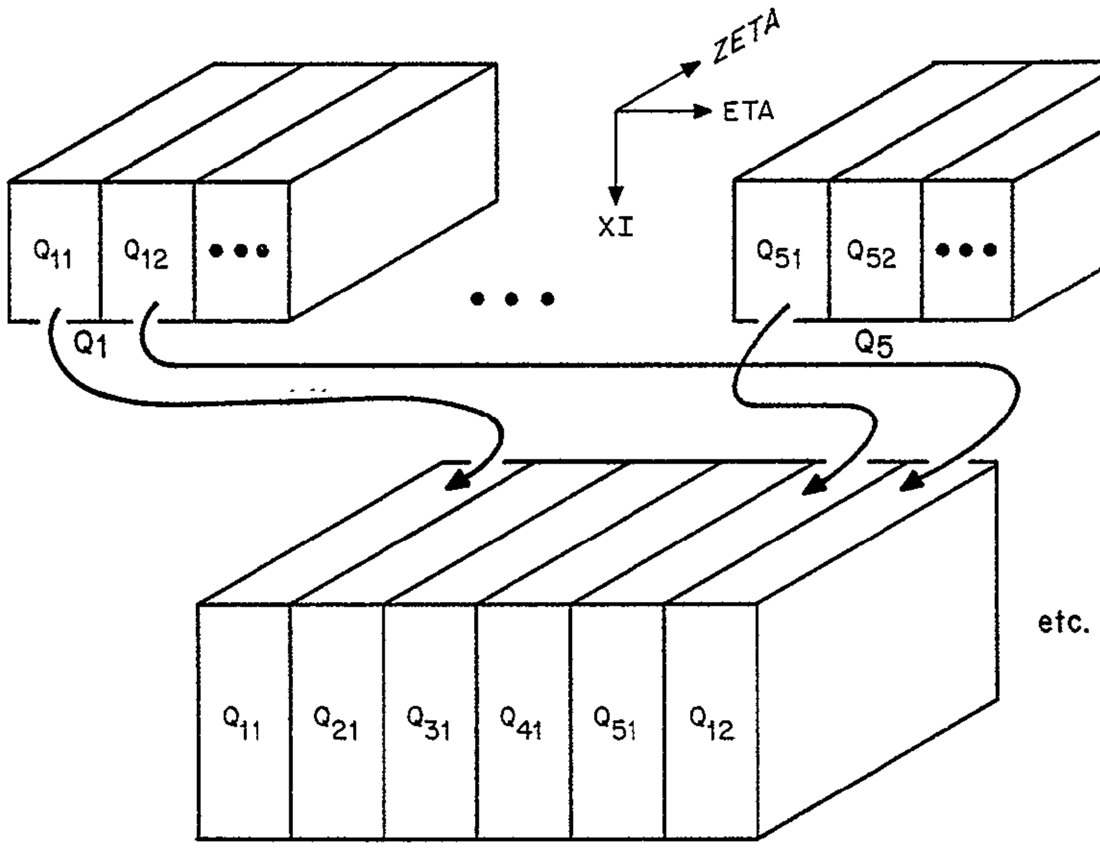
which again is a periodic gather.

Since the group length equals 1, only two results are produced per minor cycle. Fortunately, these sweeps have a very large number of operations per access to the variable arrays and all intermediate operations are done without mapping operations except for an occasional overlapped stream-rate copy. The slice for the XI-direction is like the ETA-direction slice and would normally be the same except for the mapping function.



The only other consideration which is offered in this regard is that significantly better backing store performance could be obtained if the following data reorganization was done.

(Q1, Q2, Q3, Q4, Q5) are (RHO, RHOU, RHOV, RHOW, E)



This type of data storage allocation is readily done at problem setup time. Arrays exist only as sets of descriptors (called descriptor arrays) which in turn describe the data. There is no rationale for the fixed-size, compiled-in arrays typical of FORTRAN programs. Vector programs should always generate their own data bases once attributes are known during the initialization of execution.

XI-ETA slices of the merged array from backing store yield ready to transpose or use slices of the essential variables in main store while accommodating the granularity of the backing store.

STORAGE ACCESS FOR THE FULL IMPLICIT CODE

The processes in the implicit code which comprise a typical cycle of the iteration are:

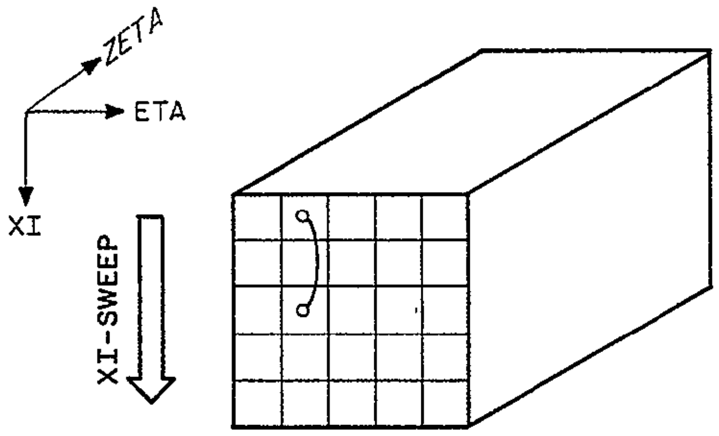
- Adjustment of the values at the edges of the mesh so that boundary conditions (BC) continue to be satisfied.

- Generation of the flux variables, differencing them, and smoothing the resulting approximate derivatives.

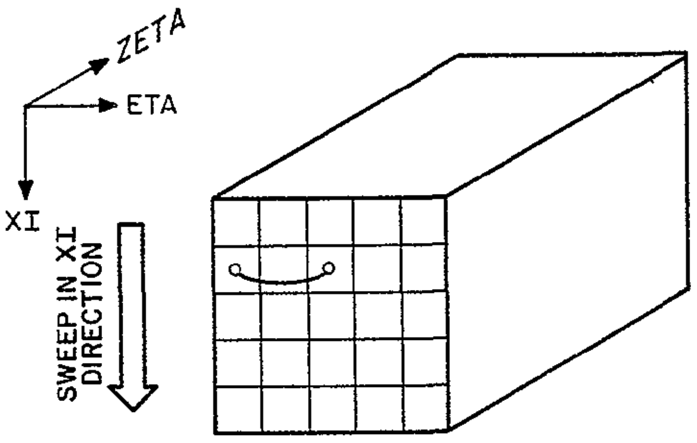
- Derivation and solution of tridiagnonal systems of equations with 4x4 matrix elements and update of the flow variables.

Consider a combination of horizontal and vertical-sliced techniques. The recursive nature of the solution of the system of equations will limit the width of the vector solution to at most the other array dimension when sweeping in a direction. This limitation is characteristic of two-dimensional codes and is removed by entering the third dimension as was seen in the discussion on access to 3-D arrays.

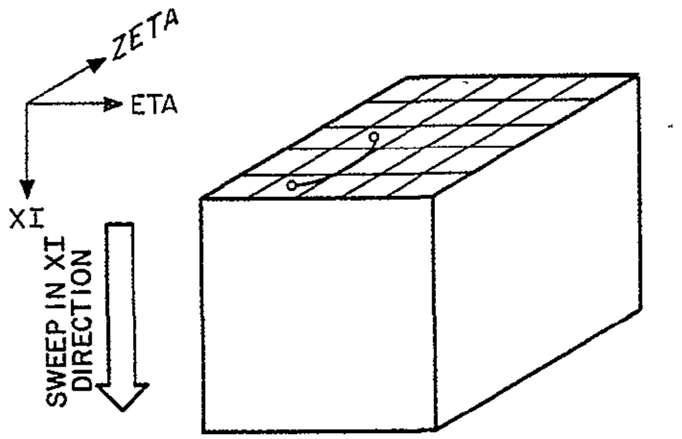
Spatial differencing is easily vectorizable. In particular, if it is in the XI, ETA, ZETA directions, the flow is always such that the columnwise direction is allowable as seen in:



OFFSET = 2 FOR CENTRAL DIFFERENCE



OFFSET = 2*JMAX



OFFSET = 2*JMAX*KMAX

Smoothing, in addition to being amendable to vectorization, has the advantage in that only one long product will be formed before a series of offset multiples by constants and adds. Thus, it is anticipated that formation of the right-hand side will yield a performance spike.

In addition to the order in which data is stored in memory for efficiency of access, the storage requirements that arise from the use of vector operations instead of scalar operations must be considered. A simple loop such as

```
      DO 10 I=1, 10000
   10 A(I) = (B(I)**2 + C(I)**2)**.5
```

when performed in scalar mode would require at best a single temporary register or memory location to hold the value of a specific B(I)**2 while the corresponding value of C(I)**2 is being computed. In the worst-case on some machines, temporary locations would be required for the squares of B(I) and C(I) as well as the partial result of the sum of the squares prior to the computation of the square root. When the loop is vectorized for a machine such as the STAR-100, two temporary vectors are required to hold the partial results of the squaring and sum, for every element of the vectors B and C. Thus, instead of two memory cells in the scalar case, the need arises for 20,000 words of memory to hold the temporary vectors. The amount of storage required can be reduced through several techniques, linked operations, intermediate buffers, and vector slicing. Linked operations, as are possible with the proposed NSS vector units, permit the computation of B**2 and C**2 and the sum to be computed with one pass through the vector unit. In the specific example given, the results of the summation could be stored temporarily in the final result vector A, thus obviating the need for any temporary vector storage.

Intermediate buffers in the vector units not only permit faster access to data by the vector units, as well as overlapped operation of the vector units, but they also provide temporary storage for at least 8000 elements that need not be assigned to main memory. Vector slicing processes a portion of the vectors at one time, rather than the entire vector. Thus, a vector of 10,000 elements might be processed as 10 subvectors of 1000 elements. In the example given here, without the use of buffers and linked operations, only 2000 words of temporary vector need be set aside, rather than the 20,000 words identified in the first instance.

Slicing is not without its costs, however, as the efficiency of the NSS machine (the amount of productive, floating-point computations being performed per microsecond, versus the theoretical peak rate of the machine) is affected by the overhead costs (usually associated with starting up the vector operation) and the vector lengths. For example, if it takes 16 cycles to start up a vector operation and the vector length is 16, the total execution time becomes 18 minor cycles (assuming eight results per cycle produced by the vector units). Since 16 results would be produced in 18 cycles, the average is 1.1 cycle per result, or worse than 1/8 of the processor's actual peak capability. On the other hand, if the vector length is 1024 elements, the total cycles to perform a simple vector function would be 16+(1024/8) or 144 cycles, which is about seven results per minor cycle or roughly 89 percent of the total capacity of the vector units.

Thus, it can be seen that the programmer (or compiler) must make some judgements about the tradeoffs between efficiency, memory storage demands, and program complexity when developing algorithms for the NSS. A combination of the various methods, then, is called for to make the best use of the architecture. Using the NSS description given later in this report and the compromises just discussed, it is possible to project the probable storage requirements for a three-dimensional version of the implicit flow simulation model.

The main variable arrays in the three-dimensional code will be

Q(100,100,100,5)	holding	(RHO,RHOU,RHOV,RHOW,E)
XYZ(100,100,100,9)	holding	$\begin{bmatrix} \text{XIX} & \text{ETAX} & \text{ZETAX} \\ \text{XIY} & \text{ETAY} & \text{ZETAY} \\ \text{XIZ} & \text{ETAZ} & \text{ZETAZ} \end{bmatrix}$
P(100,100,100)	holding	pressure
S(100,100,100,5)	used in advancing the state of the system	
X(100,100,100)		
Y(100,100,100)		
Z(100,100,100)		
XIT(100,100,100)	holding	XIT
ETT(100,100,100)	holding	ETAT
ZET(100,100,100)	holding	ZETAT

for a total of 26×10^6 words. Additionally, it is required to have a working area for A, B, C, D, DU, and F each of dimension (100,100,4,5,5) to hold temporary slices for another 6×10^6 words. Thus, already 32×10^6 words of problem memory have been consumed.

A 100x100 2-D model would accomplish 9,674,356 operations at an average rate of 825 Mflop on a 10-ns NSS, so each iteration would take approximately 11.7 ms. The 3-D 100x100x100 would consume at least $300 \times 9{,}674{,}356 = 2.9 \times 10^9$ operations and would take 3.5 seconds to accomplish 1 iteration at an 825-Mflop rate. The backing store could transfer 0.7×10^9 words ($3.5 * 12.8 * 10^9/64$) in that time.

In one step of iteration toward solution, each array would be read three times; hence, it would be necessary to read and write at most $4 \times 26 \times 10^6$ words $= 0.104 \times 10^9$ words. Adequate vector length to yield overall result rates in excess of one gigaflop would result if main memory temporaries consisting of four planes were used; that is:

Q(100,100,4,5)

XYZ(100,100,4,9)

P(100,100,4)

S(100,100,4,5)

```
X(100,100,4)
Y(100,100,4)
Z(100,100,4)
XIT(100,100,4)
ETT(100,100,4)
ZET(100,100,4)
```

in which case

> 26x40,000 = 1,040,000 words plus vector temporaries would be required. Thus, the very high transfer rate of the backing store would enable large problems to reside there, thereby decreasing the need for main memory.

Increased complexity of the turbulence model is likely to cause introduction of additional state and temporary vectors. The finite length of the pipeline scratchpad registers necessitates that certain vector temporaries reside in main store. Computational complexity, such as offset array usage, also causes certain quantities (such as first difference tables intermediate to second differences) to reside in main store. Features, such as air intakes, propulsion pods, flow fences, and landing gear, are likely to require more detail than a 100x100x100 mesh can encompass. (If only two times the mesh points were required, problem memory requirements would escalate to nearly 80 million words.)

Since the buffered transfer of data between backing store and main store occurs at a rate of 12.8 gigabits/s, a problem size of 128 million 64-bit words can be accommodated in the environment where the trunk system unloads a previous flow simulation and loads a subsequent task concurrent with execution of the current flow model. During periods of operation when monoprogramming is an acceptable mode of operation, 256 million word (512 million 32-bit words) simulations could be undertaken, thus providing a facility for extremely large model research and exploitations.

THE SYSTEM

Figure 5-2 gives a general overview of the system components and the interconnection scheme.

The major unique feature of the overall system is the method of interconnection and data transmission via the network trunk. The characteristics of this trunk are described later in this section.

Using a network trunk in this system provides flexible point-to-point transmission paths between all high bandwidth devices in the system such as disk, tape, front-end, and back-end processors. The minimum acceptable bandwidth of this trunk is 50 Mbits/s while the desirable bandwidth is 200 Mbits/s to match the transfer rate possible from the NSS and to four synchronized mass storage disks transmitting in parallel. Since each trunk can support 16 connections with relatively inexpensive transmission media (coax), and since a high degree of redundancy is wanted, four single-channel (each 40-Mbits/s bandwidth) trunks are given in the minimum configuration in figure 5-2. The minimum bandwidth is dictated by the transfer rate of the 819 disk (peak rate of 40 Mbits/s).

By attaching four disks on a single controller, which can be accessed from any of the four trunks, simultaneous transfers can be obtained between the front- and back-end processors and the high performance mass storage units. This makes possible a bidirectional data interchange operation between the NSS and the disks at the full 40-Mbit rate of the minimum channel (or at 80 Mbits/s if two disks are recorded in tandem). If a complete simulation run requires 128 million words of data base, a complete exchange of all data for a simulation run could be accomplished in 204.8 seconds (at 40 Mbits) or 102.4 seconds (at 80 Mbits), while the current job is in execution at least 420 seconds.

The size of the main mass storage system is roughly 150×10^9 bits. For a simulation run in which a data base of 128 million words is required, space would be available for 18 jobs awaiting swapping to the NSS or to be analyzed after being swapped back from the NSS. In actual fact, no more than six jobs will be queued by the front-end processors. Thus, the remaining space can be used to drain off intermediate arrays for presentation in printed or graphic form for the job in progress. From the outline developed in cooperation with Ames staff members, about 25 million words (1.6×10^9 bits) could be

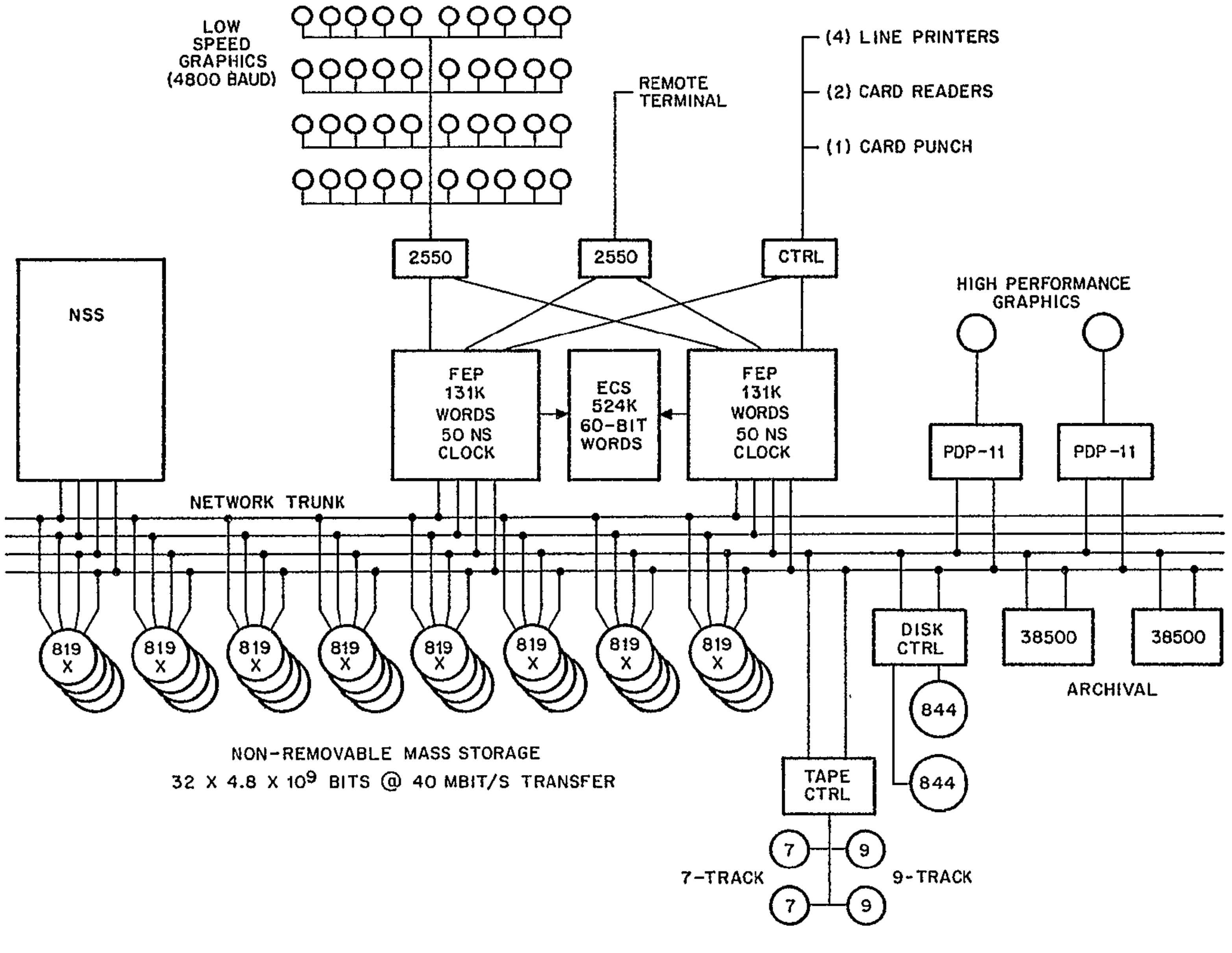


Figure 5-2. NASF System Interconnection

produced for restart dumps and intermediate analysis every 10 minutes. Thus, with six queued jobs, the high-speed mass storage system could absorb 61 loads of such data without purging data or staging data to slower speed devices. This would be sufficient to cover a 10-hour operating day with no additional I/O staging.

In reality however, many of the restart dumps will be purged from the system as soon as the data is not needed for error recovery or subsequent runs. Further, the front-end processor will perform data reduction on a substantial volume of the results and print or display the output. Once an interactive user has examined the information, he will often purge the unneeded information. Finally, the front-end processor will be asked to stage a portion (perhaps 25 percent) of the remaining intermediate results plus associated programs and data bases to the lower-speed, higher-density devices. The 38500 Mass Storage Subsystem provides about 1 trillion bits of storage which is sufficient for 640 10-minute run segments (of 25 million words) or 2560 compressed data segments (at the 25-percent rate). This would cover a total system activity of 10x1000 minutes of operation or roughly one week of operation before data would have to be purged from the archives or staged to removable media.

The removable media include disk packs and tapes to accommodate the variety of formats expected to be submitted by NASF users. They are placed on the trunk also so that high bandwidth staging can be accomplished from media to media without passing through the actual front-end processor. In this case, the front-end processor would control all transmissions but would not intercept the actual data blocks.

The dual front-end processor configuration is provided for redundancy and to meet the peak system requirements for support during data preparation and post-run analysis. The power required is expected to be about the current CDC CYBER 173 capability, which is roughly the same as a 6600 computer (with which more people are familiar at this time). The coordination of the two front-end processors is via a large amount of extended core storage (ECS) which is used to maintain certain front-end system tables and for data buffers. As can be seen from figure 5-2, the front-end processors (FEPs) control all low-speed peripherals such as printers, card readers, low-speed graphics terminals, and communications terminals directly. The data volume being passed and the bandwidth of the FEPs proposed are well matched in this instance. The higher performance graphics terminals, with high resolution, fast response, and sophisticated interactive features, will be driven by a graphics station subsystem (such as the PDP-11). In this case, data reduction of information to be displayed from the NSS will be performed by the FEP, but all graphics control and support services will be handled by the PDP-11 system. Therefore, since much data can flow directly from the storage media (819, 844, and so on), these higher performance devices are also placed directly on the trunk. This way they can receive control information and limited data from the FEP but can perform their own data base search and transfer directly with the storage devices attached to the trunk.

Later in this section, figure 5-2 will again be discussed in the context of a typical operational outline for a specific user/designer.

THE NAVIER-STOKES SOLVER

The main component of the NASF is, of course, the assemblage of computing equipment which performs the solution of the Navier-Stokes equations, namely the Navier-Stokes Solver (NSS). Its primary purpose is to provide production computation of the implicit or explicit Navier-Stokes solutions developed by or in coordination with NASA Ames researchers. Its secondary purpose is to perform the initial mesh generation function and the remeshing functions as they arise at the beginning of a problem preparation or during its operation. There shall be no other tasks assigned to this machine at any time, and thus no hardware included beyond that needed to optimize the solution of the flow field simulations. In order to reduce parts counts (for reliability and cost) and reduce complexity for development time and maintainability, it has been necessary to make the previous statement a hard and fast rule.

At the outset of this project, several methods of approach were examined for the creation of the NSS.

- To utilize existing designs, parts, and hardware wherever possible to reduce development risk
- To utilize existing or already developing software wherever possible to reduce development risk and cost
- To utilize existing conceptual models of parallel processing wherever possible to minimize the need for new architecture and to reduce the experimenting and probing that has accompanied most other efforts to convert serial algorithms to parallel forms

In the case of the Research and Advanced Design Laboratory of Control Data Corporation, researchers have been pursuing extensions to the STAR-100 structure and programming for several years. With the introduction of the upgraded STAR series machines (the STAR-100A and STAR-100C), the use of existing parts was quite tempting, particularly since a substantial amount of software is in place for these machines, a major component being the vectorizing FORTRAN compiler.

After initial cost/performance and reliability data had been studied, it was felt that STAR technology components could not meet the aggressive requirements of this project. Thus, late in this study, a reassesment was made of emerging technologies, both circuits and packaging, and of the overall architecture being planned to see what could be eliminated or restructured. First, it was determined that the degree of parallelism possible with the existing STAR technology (a maximum of four functional units) was not sufficient to meet the gigaflop processing requirements. Second, the 16-ns clock cycle achievable with this technology was inadequate. Third, preserving all the aspects of the STAR architecture to provide a basis for using all existing software was a costly imposition on a machine to be used in a restricted environment.

Therefore, a wholly new structure is defined using, where appropriate, STAR concepts of processing, and where possible, STAR designs for parts and subassemblies, such as a floating-point multiply unit. The following are some of the major features of the NSS.

32-/64-bit arithmetic	Studies of the preliminary performance metrics (implicit and explicit codes) have indicated that in many instances the data base and intermediate results can be stored and processed in 32-bit form. However, the formation of the Jacobians and solution of the block tridiagonal systems in the implicit portions of the explicit code, as well as in the implicit code, may require greater computational significance than can be assured by 32-bit arithmetic. The present experience on STAR with the 32-/64-bit arithmetic modes has proven the value of this approach in the solutions of large systems.
Massive, directly addressable, on-line memory system to contain whole problems	Solutions on 100x100x100 meshes consume between 32 and 40 million words of problem storage. Complexity arising from turbulence models and airframe features could increase storage requirements significantly. The backing-store will provide both capacity and bandwidth for 256-million-word flow simulations. Thus, the optional 32-million-word main memory may not be necessary.

Single instruction stream, multiple data stream (SIMD) control over all parallel processes

Of the various forms of problem programming that have been proposed, a vector form appears the most feasible to implement and for which to develop a FORTRAN compiler. The STAR scheme of distinct vector operations, drawn from Iverson's APL (reference 5-1), provides a sufficiently rich repertoire of functions and adequate vector lengths (65,000 elements) to be employed as the instruction formats for the NSS. The major drawback of the STAR-100 system, in that all vectors must be stored linearly in real memory, had to be removed to meet the problem solution methodologies of the NASF.

The fastest possible functional units available in 1981

To achieve the parts counts and interconnect counts considered necessary to make the NSS manufacturable, the main processor clock rate must be as aggressive as possible to reduce the number of parallel units (and thus parts and interconnects) to a minimum. Technology currently under development is expected to produce a circuit family capable of providing networks operating at 10 ns or better. The design objective, therefore, for the NSS is to clock the entire NSS ensemble at a minor cycle of 10 nanoseconds. As has happened in the past, however, constraints on packaging, transmission lines, and reliable circuitry may require some reductions in speed. Based on existing computers using advanced technology, such as the STAR family, a clock rate of 15 nanoseconds appears to be the worst speed achievable in the timeframe of the NSS production.

Independent scalar processor

The main instruction stream is interpreted by the scalar unit which performs all scalar code and passes interpreted vector operations to the vector units. The scalar processor is fully capable of operating in parallel with any vector operation. When in debugging or maintenance mode, the entire ensemble can be operated sequentially so that no overlap occurs.

Programmer or compiler responsibility for hazards

The parallel operation of several units in the NSS can create situations in which one unit needs an operand being modified by another. These occurrences are labeled hazards and can occasionally result in incorrect answers being produced. The memory map and scalar units include hardware which prohibits the occurrence of hazards from delivering out-of-sequence (and thus erroneous) results between the two. However, the programmer should be aware of one situation and thereby use proper coding techniques. If the programmer initiates a swap operation which does not involve an integral number of transfer units (8192 words), the operation of the swap unit in parallel with the

	memory map unit is still allowed, and no hardware exists to guarantee that a programmer is not trying to modify a vector which he has already started to exchange to the backing storage device. There are compiler-level statements which will guarantee that this situation cannot arise, but a zealous machine-language programmer could invoke an erroneous overlap of operations.
All vector divide operations	These operations would be performed as the reciprocal function 1/N, thus permitting full streaming rate of that function. Integer divide is provided by the scalar unit for integer and address arithmetic.

OVERVIEW OF THE NSS HARDWARE

Figure 5-3 depicts the major components of the NSS and the interconnections of the data paths. The basic NSS machine consists of a main, high-speed memory of up to 8 million 64-bit words. The main memory is linked to the backing store memory which is designed to hold two complete Navier-Stokes problems, plus working storage area for I/O buffering. To meet the existing objectives established by the explicit and implicit performance metrics now in hand, a backing storage of 256 million 64-bit words is required.

The dashed lines indicate the possibility of the presence of up to 32 million words of medium-speed access memory, if future requirements arise for introducing that level in the hierarchy. For that purpose the memory addressing scheme allows for the inclusion of this additional level of memory in the NSS.

All network trunk input and output are performed to and from the backing store. No I/O operations except backing-store swaps are performed with the main memory. The memory system is connected to 11 independent functional units through an additional unit, the memory interchange.

- The scalar unit (SCU) which performs all vector setup and scalar operations
- Eight identical vector functional units (VFUs) which perform all arithmetic operations on vectors
- The memory mapping unit (MMU) which performs array transpositions and other data reformatting operations

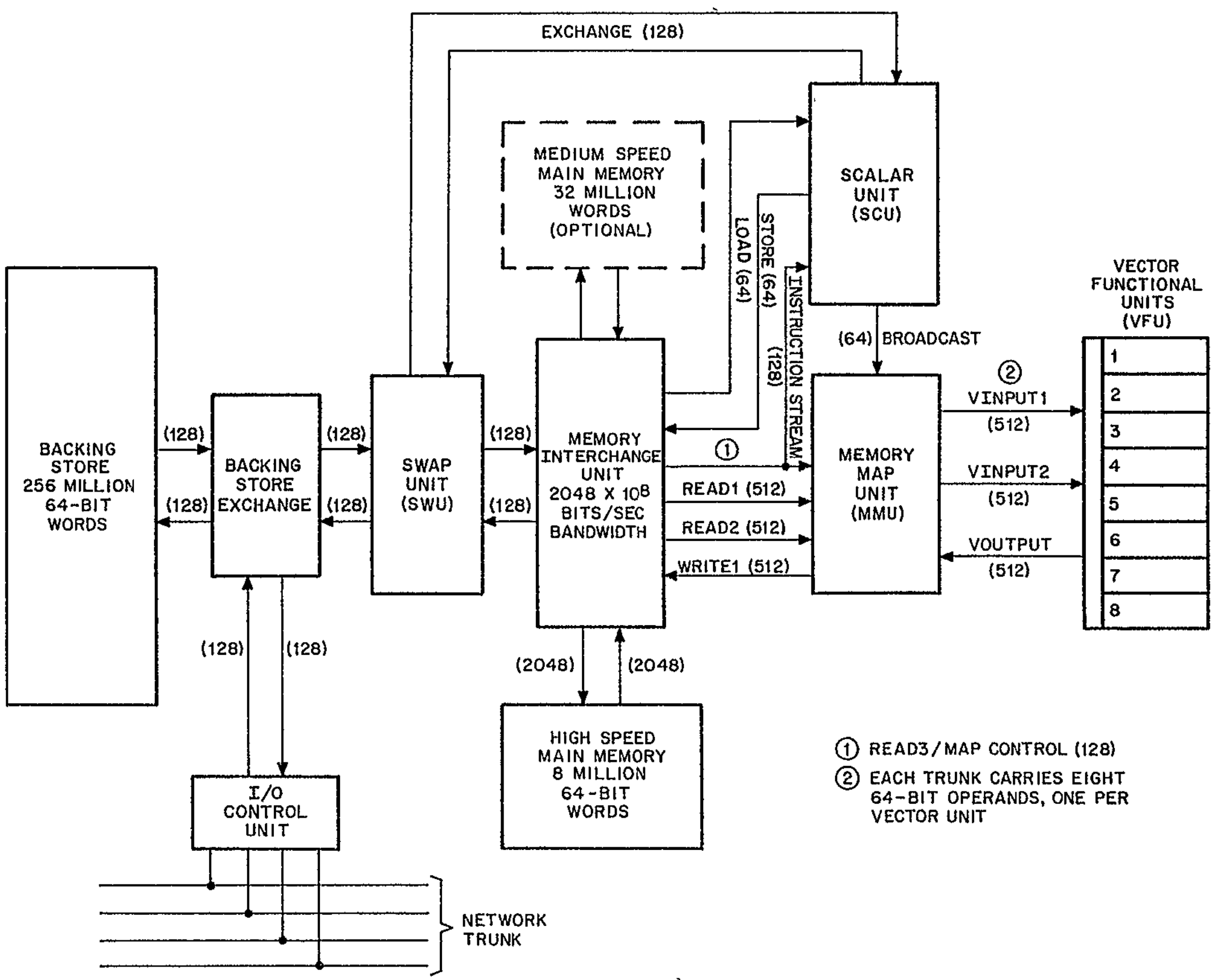


Figure 5-3. Major NSS Components and Data Paths

- The swap unit (SWU) which interfaces the backing store and can execute a sequence of block transfers with the main memory simultaneous with vector operations

- The memory interchange unit which connects all NSS components to the main memory

All nonshared data trunks shown contain 64 data bits plus 14 bits (7 bits for each 32 data bits) of single-error-correction, double-error-detection (SECDED) information. The data on these trunks can be viewed as single 64-bit items or pairs of 32-bit items. The shared trunk called READ3 in figure 5-3 transmits 128 bits plus SECDED to the instruction stack in the scalar unit or to the control section of the memory mapping unit. The scalar, memory map, swap, and vector functional units can all operate on data independently and simultaneously although when data is transmitted to the VFUs from main memory or from the VFUs to memory, the mapping unit must be used for the transmissions. Data from one VFU must be returned to memory before it can be used by another VFU. Data from memory is spread laterally across the eight vector units, with 64 bits from each trunk (READ1,READ2) going to a different unit. Thus, the first element of vector A would go to VFU(1), the second element of vector A to VFU(2), and so on. There is no direct connection between VFUs for data.

THE UNITS

The Scalar Unit

Figure 5-4 shows a block diagram of the basic elements of the scalar unit. In the figure, the presence of SECDED is indicated on all trunks leading into and out of the scalar unit. In addition, the following scalar unit buffer memories contain two parity bits for each 64-bit word: monitor stack, job instruction stack, monitor register file, and job register file. All functional control for all units is performed by microcode, which contains one parity bit per word (no matter what the varied word length is). All data trunks within the scalar unit, with the exception of the high-speed multiply unit, contain a parity prediction network. The high-speed multiply unit has one additional stage of logic which differs from the comparable unit on the STAR-100A/C. This stage performs a parity recalculation, since the multiply unit itself cannot pass proper parity prediction through itself. All parity is checked on every store into the register file, and every transfer of data into the load/store unit for storing to memory. All error codes, failing element location, and faulting main or microcode memory addresses are available to be sampled by the maintenance station function, through the system I/O unit.

This unit is the main instruction decode and control for the entire NSS ensemble. It is similar in structure and design to the STAR-100C scalar unit, since that device represents the fastest possible scalar processor (at 10 ns) that can be built with the extant technology. The major differences are the elimination of some floating-point end-cases to make the unit compatible with the vector units, the inclusion of an advance branch feature to speed up scalar branching, and the exclusion of a number of instructions to simplify the unit and make room for the monitor register file and logic.

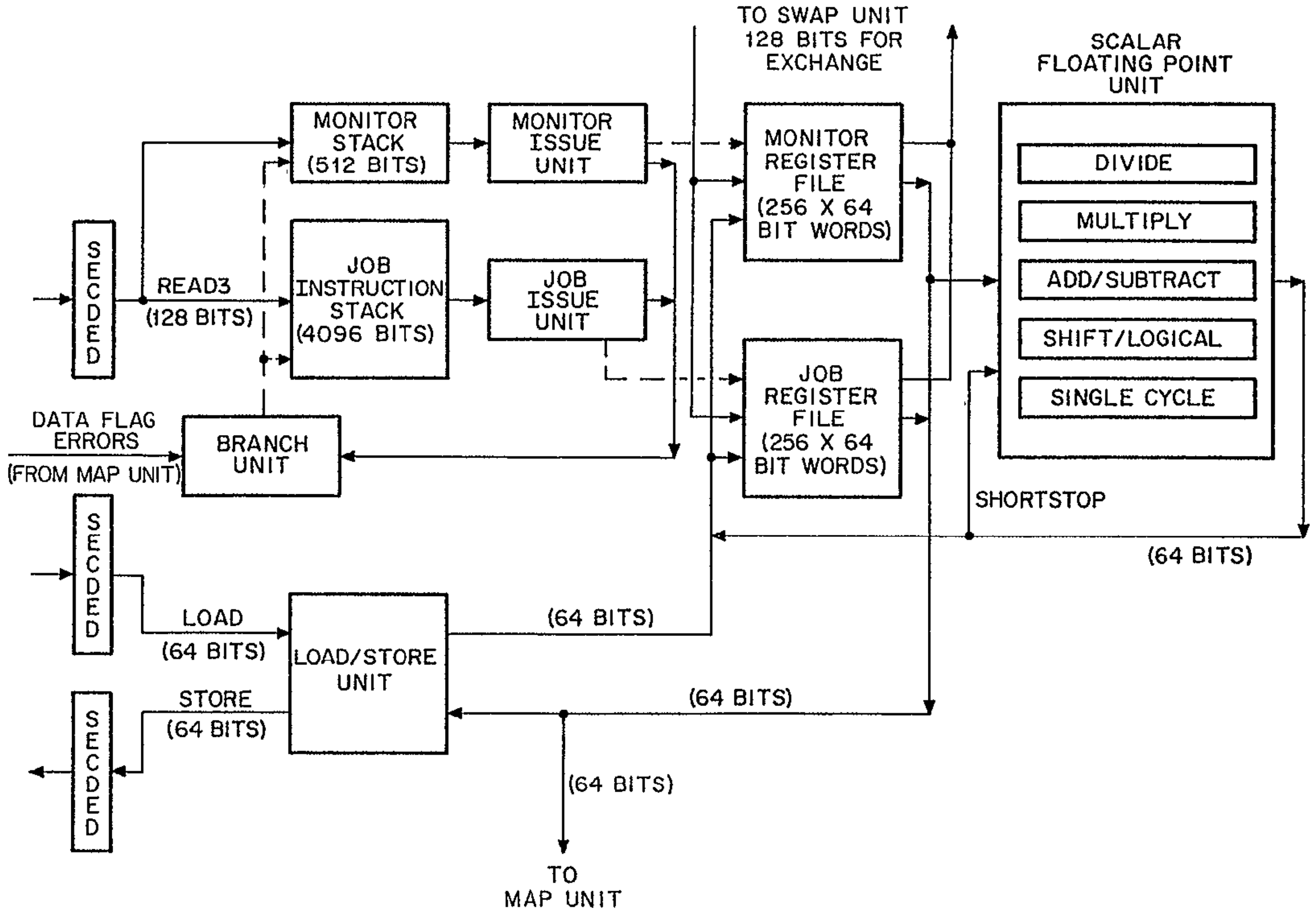


Figure 5-4. NSS Scalar Unit

Scalar Unit Components

Several scalar unit components are shown in figure 5-4.

READ3	A 128-bit trunk capable of supplying data every minor cycle to either the job or monitor instruction buffers.
Load-store trunk	Two 64-bit paths providing data connection to main memory. Each path is capable of transmitting one 64-bit quantity every minor cycle.
SECDED networks	All data transmitted to and from the scalar unit and main memory carries 7 bits of SECDED for each 32 bits of data along with it. At the termination or beginning of each transmitting/receiving trunk in the scalar unit, a network is provided to check and correct the data paths. At the output of the network, the corrected data is available, as well as a single parity bit for each 32-bit quantity which is carried through the remainder of the scalar paths, register files, and functional units. Not shown in figure 5-4 are the parity checkers and generators in the functional units nor the error flag paths from the error networks to the maintenance control channels.
Job instruction stack	This buffer memory consists of 4096 bits plus 128 parity bits of high-speed storage. A 32-bit or 64-bit instruction can be read out of the buffer every minor cycle, while 128 bits of instruction stream can be written into the buffer every minor cycle, simultaneous with the read operation. All instructions in the NSS are 32 bits in length except ten 64-bit instructions which are needed to perform special FORTRAN branching, as well as the loading of address and constant data into registers.
Monitor instruction stack	The low processing performance monitor instruction stream requires only a small instruction buffer. Since most data is read from memory in a 512-bit block (for efficiency reasons), a 512-bit buffer (plus parity) is provided to hold instructions awaiting execution by the monitor issue unit.
Load/store unit	The scalar unit communication with memory is via the load/store unit, which performs 64- or 32-bit transfers to and from main memory and which can sustain memory requests at a rate of one every minor cycle (or one every four minor cycles if the banks are busy). To keep the instruction issue rate at a high level, even if a requested bank is busy, the load/store unit can queue load and store operations (six loads, three stores, or any combination thereof). With the exceptions of the register file SWAP

	instruction and end of job exchange operations (where the register file is stored in main memory), the load/-store unit provides all data transfers between main memory and the job or monitor register files. Note that the load/store unit is shared by the job and monitor execution units, with priority going to the job unit.
Branch unit	This unit performs all branches for the job and monitor execution units. It creates all requests for instruction data via the READ3 data trunk, while providing all control over the instruction stacks of the job and monitor systems. The branch unit can perform in-stack (instruction data already present in the instruction buffers) branches for the job unit in six to nine minor cycles (depending on the complexity of the branch). All branches in the monitor unit are treated as out-of-stack branches.
Monitor issue unit	This elementary unit performs one instruction at a time, waiting for the operation to be completed by whatever functional unit is invoked. It can perform all instructions that the job instruction unit can perform plus several privileged instructions limited to monitor control of the I/O and backing store operations. The monitor issue unit advances to the next instruction only when the job issue unit is unable to issue an instruction in a given minor cycle, due to load/store, register file, or functional unit conflicts. Thus, it appears that the scalar functional units (branch, load/-store, and floating point) are time-multiplexed between the monitor and job execution units with the job execution unit always getting the first use of the functional units.
Job issue unit	This unit is similar to the unit found in the STAR computers. It provides for a sustained rate of one instruction per minor cycle, with some provision for sustaining that rate even when functional units are busy. This unit stages instructions into a pipeline where conflicts with previous instruction usage of the functional units and register file are checked and resolved.
Job register file	This very high-speed memory unit contains 256 64-bit words plus 512 parity bits (one for each 32-bit quantity). It is capable of delivering two 64-bit operands and receiving one 64-bit operand every minor cycle. It is also capable of simultaneous read and write operations of 128-bit items during exchange and swap operations.

Monitor register file

This file is identical in structure to the job register file. It is loaded by a half-exchange during initial start of the NSS; it can be stored into main memory by the swap instruction or by the maintenance station function during system debugging and maintenance. The monitor system is capable of stopping the job mode processor and storing its register file to main memory preparatory to swapping the current job for a new one.

Scalar floating-point unit

This unit contains five arithmetic units.

- Divide unit

  This unit performs an iterative divide 2 bits per minor cycle, in either 32- or 64-bit mode. The divide unit is not pipelined and thus remains busy during the entire divide operation (about 28 cycles for 32-bit mode and 52 cycles for the 64-bit format). Therefore, only one divide operation can be in progress at a time with one more divide queued up at the front end of the divide unit waiting to start. The divide unit shares logic with the add/subtract unit and the multiply unit for pre- and post-normalize operations during floating-point calculations. The results from the iterative divide will not be identical to the results yielded by the vector units in several end-cases, particularly integer division.

  A special floating-point divide function is provided, which utilizes the basic divide unit, the multiply unit, and add units to form a result which is identical to that produced by the vector pipelines. This function is primarily used for diagnostic purposes, since for normalized floating-point data, both the normal scalar and vector divides provide equivalent answers.

- Multiply unit

  This unit is a four-segment pipeline capable of accepting a new pair of input operands every minor cycle and producing the corresponding results at the rate of one per minor cycle. The multiply unit requires four minor cycles to produce a 48-bit coefficient result. The multiply unit shares the front- and back-end pre- and post-normalize networks with the add/subtract unit, as does the divide unit.

- Add/subtract unit

  This unit is fully pipelined, like the multiply unit, producing a new result every minor cycle. This unit operates on 32- or 64-bit data and can provide the necessary conversion between the two formats. All address arithmetic performed in scalar code is done in the add/subtract unit.

- Shift/logical unit

  This unit provides a pipeline function for shift, logical, and insert/extract operations to be used for aligning and reformatting data to be used by the NSS.

- Single cycle unit

  This unit delivers a new result every minor cycle. It is used for instructions which carry all or part of the operands within their structure (such as load register R with the data contained in the rightmost 16 bits of the instruction).

The floating-point unit provides a shortstop path which can circulate one or two result operands to the beginning of the floating-point pipelines for immediate use rather than requiring that the results be stored and read from the register files. Note that only data in the register files can be transferred to the vector and mapping units. Likewise, only the scalar register files can receive results or status data transmitted to the scalar unit from the swapping unit, mapping unit, or vector units.

The Vector Unit

Along with the memory system and its bandwidth, the vector unit is the key to the NSS performance capability. As shown in figure 5-5, this unit consists of a buffer memory; two multiply units; two add units; a divide algorithm; and appropriate parity, SECDED, checking, and selection circuits. The key features of the vector units are:

- The pipeline ability of all entities to produce a new result every minor cycle
- The uniform design and construction of each add and multiply unit regardless of position in the system
- The uniform ability of each unit to produce either 32- or 64-bit results at each and any cycle
- The ability of the coincidence check to compare automatically the output of each idle add or multiply unit with that of its twin (add/add or multiply/multiply) every minor cycle when one or the other is idle

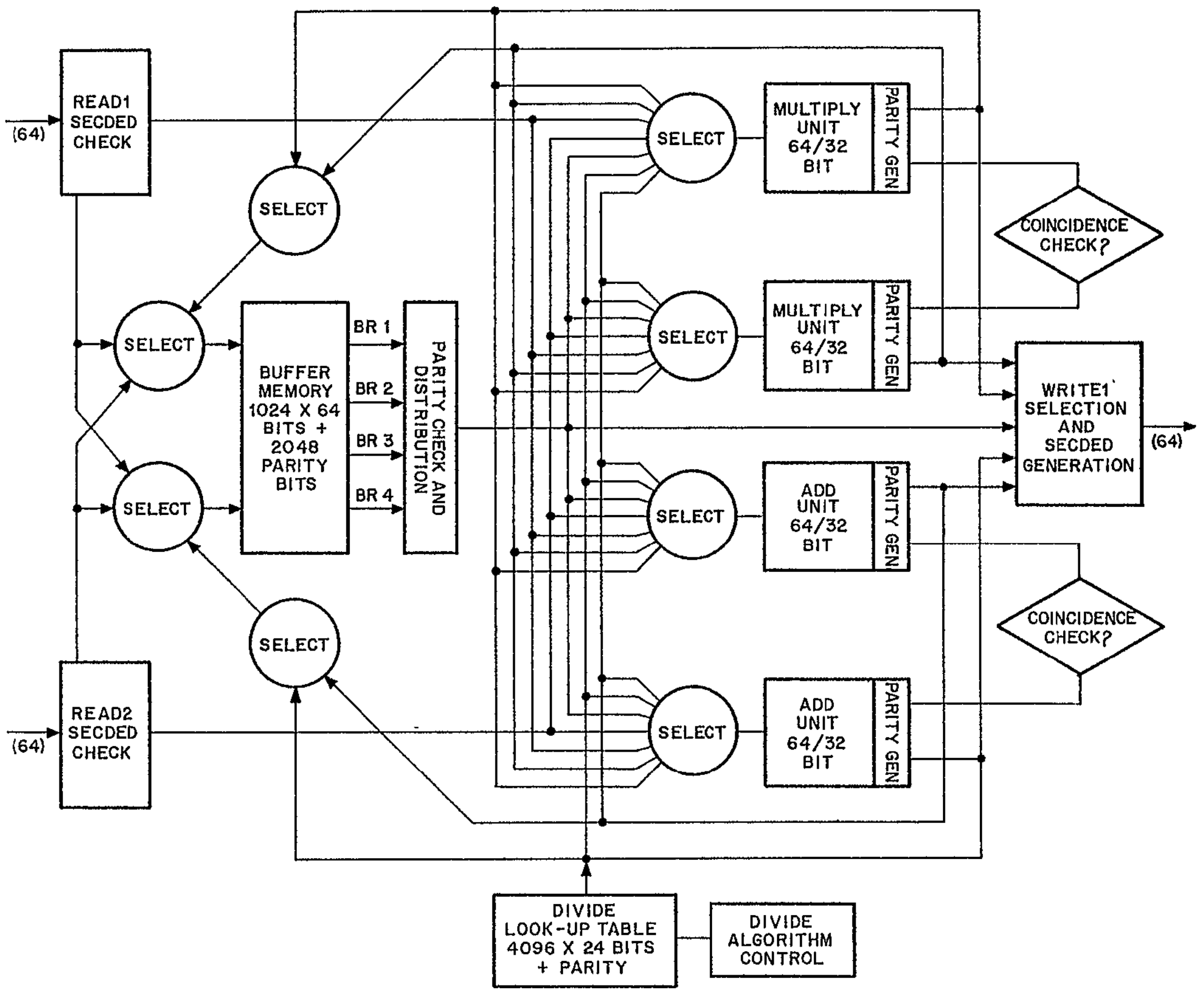


Figure 5-5. One Vector Unit

- The interconnection of any unit to any other and the main memory through READ1, READ2, and WRITE1 data ports
- The production of divide results by a reciprocal approximation algorithm, using all of the functional elements to achieve a higher rate of result production
- The local control of all data streams within the vector unit by the vector control system. Data can move from the buffer to arithmetic components and back concurrently with the movement of data to and from main memory.

The following are the internal features of the vector unit.

- One main feature is the very high performance buffer memory consisting of 1024 64-bit words (plus 2 bits of parity each). Each buffer can deliver four separate read operands while accepting two unique write operands every minor cycle. Figure 5-5 shows that this arrangement permits the vector unit to operate for some period of time without communication with main memory.
- Duplicate multiply and duplicate add units permit the programmer to invoke a chain of four arithmetic operations to occur at peak rates or any subcombination of add and multiply operations.
- A single stream can be distributed to both inputs of an arithmetic unit to form A^2 or 2A.
- When running at full rate, with all arithmetic units performing useful work (as in (A*B)+(C*D)-B) where A*B are main memory contained vectors and C*D are in the buffer memory), eight vector units in tandem can produce 32 64-bit results every minor cycle or 64 32-bit results per minor cycle for corresponding computation rates (peak) of 3.2 and 6.4 gigaflops (10^9 floating-point operations per second). In reality, many computations will be able to invoke the use of only one or two units (usually a linked multiply/add combination), thus allowing the paired arithmetic unit to automatically check results of the other arithmetic unit.
- The multiply units and add units are fully pipelined systems, each unit capable of producing one 64-bit result or two 32-bit results every minor cycle, with four minor cycles required for a result to emerge from the pipeline.
- Vector division is accomplished by the use of a reciprocal algorithm which requires all four arithmetic units to be utilized to produce either the 64- or 32-bit results. The divide operation is pipelined, but since four arithmetic operations are required after an initial starting quotient is produced from a read-only look-up table, only one minor cycle is required to produce a result. Thus, eight pipelines can produce eight results per minor cycle. However, because all arithmetic operations are required during the divide time, no chaining or other operations can proceed within the vector units.

The Map Unit

The map unit controls all data transmissions between main memory and the vector units and performs certain memory-to-memory functions which are nonarithmetic in nature.

READD3

This trunk is shared with the scalar unit instruction stream. It provides a 128-bit trunk on which streams of bits can be used to control certain vector operations. Two key features of vector processing that have been proved valid on the Control Data STAR machines have been the concepts of control vector and order vector, which are taken from Kenneth Iverson's A Programming Language (reference 5-1).

Control vector

Quite often in solutions of equations with complex boundary conditions, the values of certain variables at the boundaries must remain fixed during parts of the calculation while their neighbors are changed. When the boundaries are at the head and tail of vectors, these values can remain unmodified easily by proper choice of starting and ending addresses of the vector operations. When, as is often the case, the boundary conditions exist on all sides of a three-dimensional geometry, the corresponding boundary values become embedded within the vectors to be processed. To improve the sustained rate of vector operations, it is desirable to keep from fragmenting operations into a series of short vectors (one of the methods for skipping over the boundary values). One way to accomplish this is to inhibit the storing of newly calculated values on top of the boundary values by use of a vector of bits, one for each value to be stored. If the corresponding bit is a one, the value is stored; if the bit is a zero, the value is calculated but not stored. The bit string can be generated by a set of scalar commands or through a set of vector conditional test commands, and is called a control vector.

Order vector

Many problem solutions require special handling of regions of data. For example, in the explicit code used as a preliminary performance metric, the amount and kind of processing for a given region somewhat depends on the flow behavior in that region. Therefore, vectors can be extracted and compressed from the larger matrixes and processed more efficiently. The functions provided by Iverson (reference 5-1) to accomplish this are:

- Compress

  A bit vector of ones and zeros removes all elements in a data vector corresponding to one bits and stores them in another data vector.

- Mask

  A bit vector of ones and zeros extracts all elements from vector A corresponding to one bits in the bit vector, extracts all elements from vector B corresponding to zero bits in the bit vector, and stores the extracted elements in the order encountered into a result vector.

- Merge

  A bit vector of ones and zeros merges all elements of vectors A and B together, taking A elements when the bit vector is a one and B elements when it is a zero, and stores them into a result vector.

The bit vector controlling these operations can be produced by the scalar unit, or from data-dependent vector conditional operations performed in the vector units. This bit vector is called an order or ordering vector.

Figure 5-6 shows the units which handle the control and order vectors.

Bit string addressing unit

This unit permits addressing bit strings on other than word boundaries (for example, to begin at some other point than the starting element of a vector), and can supply 16 bits per minor cycle to the control vector processor or the compress, mask, merge network.

Control vector processor

This processor synchronizes the bit vector with results coming from the vector units or the internal scatter/gather network. The time delay required for transmission of data to and from the vector units implies a moderate degree of buffer storage for the bit stream to keep it synchronized with the data stream. The control vector processor transmits a series of write enables to the memory system, one for every 32 bits to be stored. These enables control the actual storing of data at the memory unit.

Compress/mask/merge network

This system combines the READ1 and READ2 data as directed by the order vector provided by the bit string addressing unit. Data emerging from this network can be transmitted directly back to main memory or can be transmitted to the vector units via either the READ1 or READ2 paths.

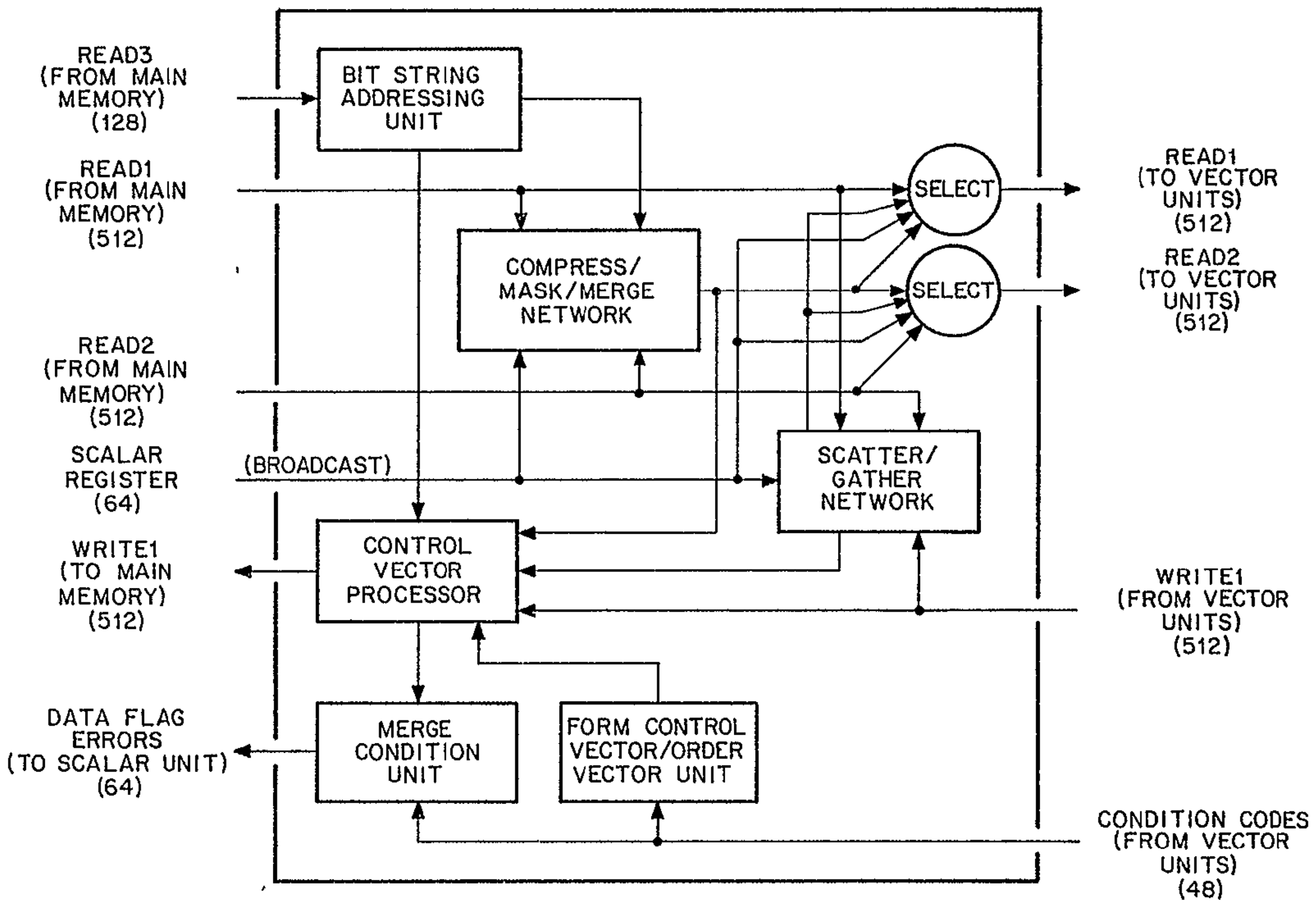

READ3 (FROM MAIN MEMORY) (128)
READ1 (FROM MAIN MEMORY) (512)
READ2 (FROM MAIN MEMORY) (512)
SCALAR REGISTER (64)
(BROADCAST)
WRITE1 (TO MAIN MEMORY) (512)
DATA FLAG ERRORS (TO SCALAR UNIT) (64)
BIT STRING ADDRESSING UNIT
COMPRESS/ MASK/MERGE NETWORK
SELECT
SELECT
SCATTER/ GATHER NETWORK
CONTROL VECTOR PROCESSOR
MERGE CONDITION UNIT
FORM CONTROL VECTOR/ORDER VECTOR UNIT
READ1 (TO VECTOR UNITS) (512)
READ2 (TO VECTOR UNITS) (512)
WRITE1 (FROM VECTOR UNITS) (512)
CONDITION CODES (FROM VECTOR UNITS) (48)

Figure 5-6. Map Unit

Merge conditions unit

This unit is required to make sure that the error or condition flag register (called the data flag register) is set only on the right conditions. For example, the calculation of a value which is not to be stored in memory (because it would overlay the boundary values) could create an overflow, or overflow condition, which is not significant since the data is not valid. Thus, the condition data flag register should not be set for that error case. This is accomplished by inhibiting (or masking) the data flag condition codes with the corresponding control vector bit.

Form control vector/ order vector unit

This device can sample the bits (called condition bits) returned from the floating-point vector units. The overflow/underflow bits and the condition met bit are of interest. This latter bit is set (one per result from each pipeline, 8 for 64-bit results and 16 for 32-bit results) when a given condition is to be sampled by the vector units such as A greater than B, A equal to B, and so on. These bits can be formed into a result bit vector and stored in main memory. During this time the bit addressing unit is needed to control the bit address and trunk manipulations necessary.

Scatter/gather network

Perhaps the most troublesome aspect of the development of an efficient processor for the three-dimensional Navier-Stokes solution mechanism has been the effect on processing bandwidth caused by the need to address array data along some dimension where data cannot be accessed in a sequential vector manner (later in this section a detailed discussion is provided on accession of data from three-dimensional arrays). The approach taken in this proposed NSS machine is to adopt the STAR concept of ultra-high bandwidth memory trunks and processing power to match that bandwidth. The only way in which this bandwidth can be effectively produced and utilized is to structure the data in some constrained way so that a single memory request can yield more than one operand. Such a sequential scheme harmonizes with the concept of vector processing very well except in the case of the odd dimension. Whereas in the structure described thus far eight 64-bit operands can be retrieved per stream from main memory and eight results can be returned to main memory each minor cycle, the memory system will support no more than two random (nonsequential) requests each minor cycle. Thus, a sweep across a plane, the elements of which are noncontiguous, can reduce processing bandwidth by a factor of 4, best case (no conflicts) to 32, worst case. A solution to this problem is to provide a mechanism for accomplishing the nonsequential access concurrent with other calculations. To this end the scatter/gather network is provided in the mapping unit.

To aid in understanding the scatter/gather network, the following description is provided for the GATHER and

SCATTER instructions. The GATHER instruction forms an indexed list of result elements in vector field C by transferring elements from addresses in vector field B as indexed by the item counts in the A-vector field. The rightmost 48 bits (no half-word option) of each element of vector A contains an item count. The instruction adds the first item count in vector A to the base address of vector B. The element at the new address is transferred to result vector C. The instruction then adds the item count from the next element of vector A to the base address of vector B. The resulting address indexes the second element of vector B. Before the addition of each item count (index) to the base address, the index is left-shifted five places (32-bit operands) or six places (64-bit operands) to form the half-word or full-word address, respectively. This process continues until vector A is exhausted.

The SCATTER instruction adds the item count (rightmost 48 bits) of the first element of vector field A to the base address in register C to form the address of the first element of result vector field C. The instruction then transmits the first element of vector field B to the computed address in C. Similarly, the instruction forms the address of the second element of vector field C by adding the item count in the second element of vector field A to the base address in register C. The second element of vector field B is then transmitted to the computed address in the result vector field C. As with the GATHER instruction, before the addition of each item count (index) to the base address, the index is left-shifted five places (32-bit operands) or six places (64-bit operands) to form the half-word or full-word address, respectively. The instruction continues in this manner until the A vector field length is exhausted.

For both the GATHER instruction and the SCATTER instruction, elements of vector field A are 64 bits while the elements of vectors B and C are 64 bits or 32 bits as specified optionally by the instruction.

Normally the GATHER and SCATTER instructions transmit single elements as previously described. With an alternate option, a group of elements is transmitted from vector B to vector C for each element of vector A. The group length is specified in the upper 16 bits of register B for GATHER or register C for SCATTER. All groups are of equal length.

In addition to using main memory, a further option permits all elements of vector B for GATHER or C for SCATTER to reside in the register file within the range of bit addresses 0 through 3FCO. If all the addresses are not contained in the register file, the instruction is undefined.

For an additional option of SCATTER, vector B is broadcast. This operation takes one element from the register file and uses it for all elements of the B vector.

Figure 5-7 shows an example of a GATHER instruction with assumed register content and vector fields. The first item count is read from address 4000. This value indexes the B vector base address by five half-words after the left-shift of five. Thus, the instruction transfers the first B vector element from address 70A0 to the C vector element address 9000. Six B vector elements are transferred to the C vector.

Note: For this example, the GATHER instruction specifies the following.

32-bit operands
Vector B resides in central memory
Single element
Register 03 points to indexes
Register 04 contains base address
Register 06 points to destination

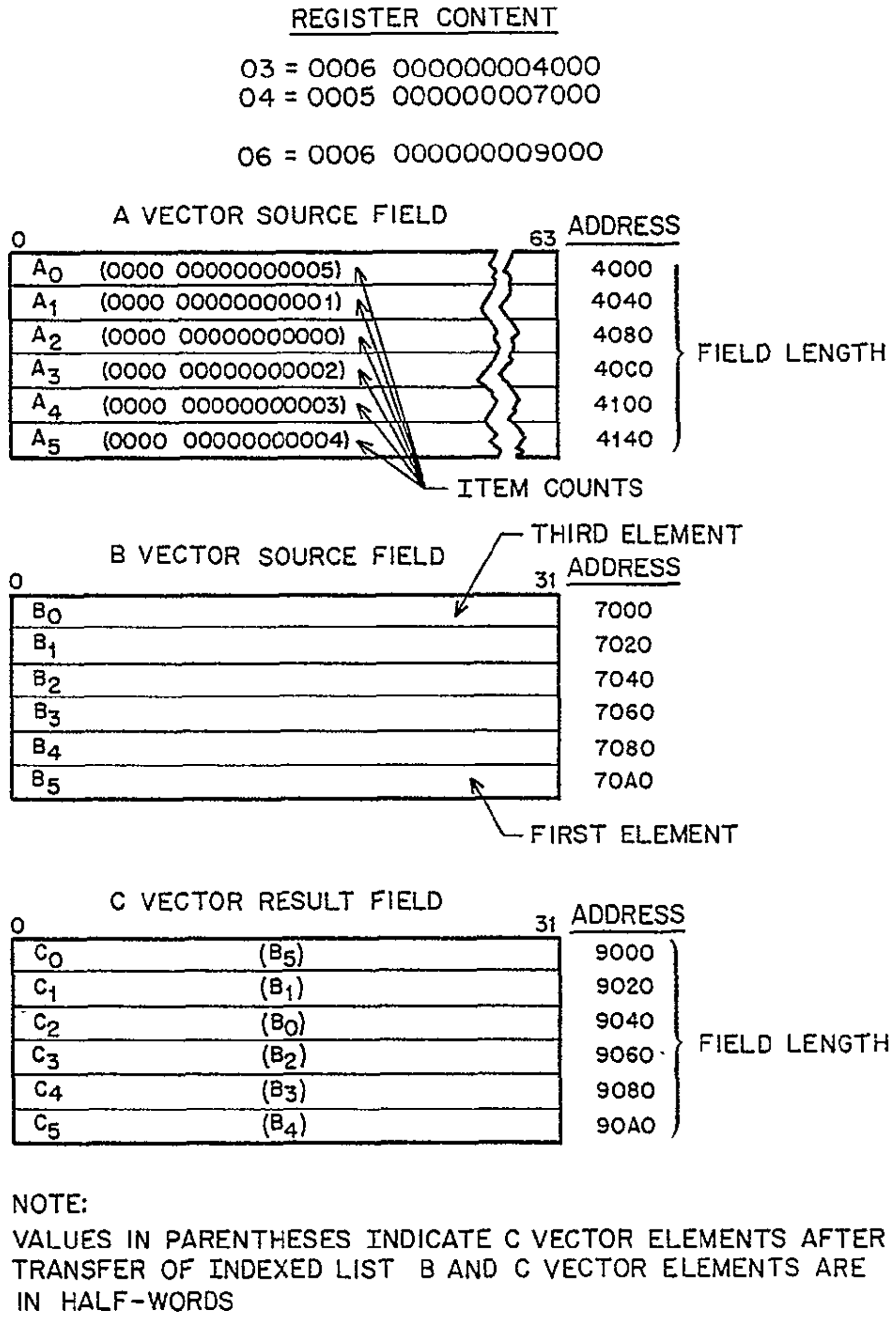


Figure 5-7. Example of GATHER Instruction

The scatter/gather network can perform the following functions.

- Gather-read word (element)

  From a list of indexes provided via READ1, the addresses of data in main memory are computed and this data is read from READ2, merging the sequential results into a stream which can be transmitted to the vector units via either READ1 or READ2 or transmitted back to main memory via WRITE1.

- Gather-read group

  Operates the same as the previous operation but more than one word is transmitted sequentially from the starting address computed.

- Scatter-write word (element)

  From a list of indexes provided via READ1, the addresses into which data from READ2 is to be stored are computed.

- Scatter-write group

  Same as scatter-write word, except that more than one word is transmitted to the starting address computed from the index quantity.

In the above four cases, the index list from memory can be absent and a fixed increment substituted, as in the case of sweeping across the rows of a two-dimensional array. All result data can be sent to the vector units and/or main memory.˙ Since this unit can operate in parallel with the vector units when they are performing buffer-to-buffer operations only and not accessing memory, transpose, restructure, and nonsequential access operations can be overlapped to improve the apparent performance of the system in this area.

The Swap Unit

Figure 5-8 gives a block diagram of the swap unit. This device performs the following functions.

- Exchanges operations under the control of the monitor, for the change from one job to another.
- Swaps job mode registers with main memory. This function allows the programmer to transmit from 2 to 256 registers to any main memory location, while simultaneously fetching a different number of registers (starting with the

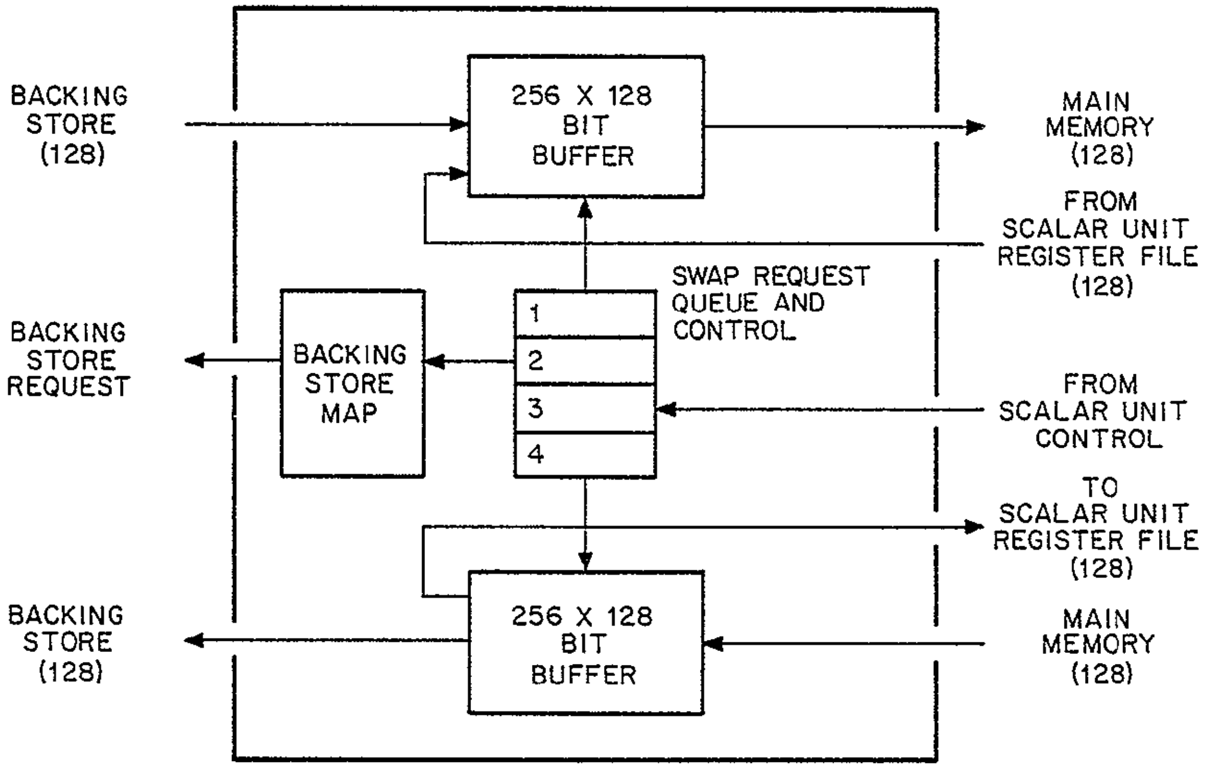


Figure 5-8. Swap Unit

same register number) from main memory. Both the source and destination main memory locations can be identical, but no other form of overlap of the main memory data areas is permitted. This operation proceeds at the rate of 128 bits transmitted in each direction every minor cycle; however, memory access priority is given to the memory map unit, and thus these streams may be interrupted in case of bank conflicts.

- Swaps blocks of main memory with the backing store memory. This operation moves data from the backing store into main memory at the rate of 128 bits each way every minor cycle. The source and destination addresses in the backing store system may be different or identical, while the address for the swap in the main memory identifies only one starting location.

To overcome the synchronization problems inherent in competing with the vector units for attention, two 32-kbit buffers are provided to sustain the backing store transmission rate in spite of memory conflicts. In addition, up to four swap requests of any kind can be stacked in the swap unit. This permits the release of the scalar unit for other work (as well as the attached vector units) while various working storage swaps are queued and executed.

The Backing Store Map

Although there is no virtual memory system in the NSS, the monitor can control the allocation of addresses in the backing store by toggling the leftmost 8 bits of the absolute address in the backing store map unit. Thus, any job can be prohibited from using half of the backing store, while the monitor uses that area to prepare the next job for execution.

At the same time, all jobs can be coded and compiled to reference the backing store as if they had access beginning with the first physical address. Monitor can then set the backing store map table to cause backing store address zero references to actually be backing store address 128 million. The map table is not associative but provides a monitor-controlled conversion of the uppermost bits of the address.

Memory Interchange Unit

The connection of all NSS components to the main memory is via the memory interchange, which takes the data from the individual modules, reads this data at 2048 bits per minor cycle, buffers this data, and transmits the appropriate quantities on one of several trunks. The READ1 trunk supplies 512 bits of data plus SECDED to the mapping and vector units. The READ2 trunk supplies 512 bits of data plus SECDED to the vector and mapping units. The READ3 trunk supplies 128 bits as often as every minor cycle to the instruction streams of the job and/or monitor execution units. The backing store trunk supplies 128 bits every minor cycle to the backing store. The WRITE1 trunk receives, buffers, and stores up to 128 bits per minor cycle. The backing store WRITE trunk receives and buffers up to 128 bits per minor cycle. The memory interchange also receives the write enables generated by the control vector processor in the mapping unit or the two write enables generated on STORE operations by the scalar unit to control the actual bank requests in each module. Write enables are based on 32-bit size items, and modules can be controlled as 64- or 32-bit elements.

NSS MEMORY

The memory system for the Navier-Stokes Solver is based on a directly addressed, hierarchical memory structure. Each level in the hierarchy is different from the next in its access speed, capacity, cost, and power requirements.

High-speed buffers	The high-speed buffers are made up of very fast ECL RAM chips and are found in the input/output interface and the scalar and vector functional units of the CPU. The first of these, the scalar unit register file, provides storage for 256 64-bit words, randomly addressable. This register file can sustain a continuous access rate of two read and one write operations in a single 10-ns cycle. The second buffer system is used in the input/output interface and contains 256 64-bit words with the ability to perform one read and one write in the same 10-ns minor cycle. These two buffer types are based on the STAR register file which contains 256 64-bit words, also addressable as 32-bit quantities for half-word instructions or half-word I/O and as 128-bit quantities when performing exchange operations at job startup and shutdown, or when transferring data between the memory interchange unit and the I/O system. The third high-speed buffer appears in each of the vector functional units. Each vector buffer unit contains 1024 64-bit words of randomly addressable memory. Each unit is capable of sustaining four read operations and two write operations, each a 10-ns minor cycle.
High-speed main memory	The high-speed main memory is also built out of high-

speed ECL RAM chips. This memory can contain from 2 million to 8 million words, each 64 bits. The read or write access time of this memory will be 40 ns per bank. The memory system contains 256 banks, each possessing from 16,384 to 65,536 by 32 bits. The memory is physically organized into 64 modules, each having four banks. Each module can accept a new request to a nonbusy bank every minor cycle. Bank-busy time is the same as memory access, four minor cycles. Each module is capable of delivering 32 bits each minor cycle to the memory interface. Thus, when delivering streams (or vectors of data) to the vector functional units, the memory bandwidth can reach a peak of 2048 bits every minor cycle (64 modules at 32 bits each), for a peak rate of 204.8 Gbits (10^9 bits) per second.

Optional medium-speed main memory

This memory provides a high-density storage device, at a reasonable access time, with the bandwidth to match the high-speed memory. It is built from an ECL-compatible RAM chip. The medium-speed memory is organized into 32 modules, each of which can hold from 262,144 to 1,048,576 64-bit words. A module is organized into 32 banks, each bank containing up to 32,768 64-bit words of memory. The access time of this memory system is 300 ns or 30 minor cycles per bank. Each module can accept a new request every minor cycle if the accessed bank is not busy. As in the high-speed memory, bank busy is the same as access time. Each module can deliver 64 data bits every minor cycle, thus matching the bandwidth of the high-speed main memory.

In both of the main memory systems, additional bits of SECDED information are stored and transmitted when memory is accessed. Also, a single 64-bit word or double 128-bit word can be accessed without causing all other banks to become busy. For example, under normal streaming conditions, all modules (of either the high-speed or medium-speed memory system) are triggered by a memory request, thus delivering 2048 bits and making the corresponding banks busy for four minor cycles. When scalar (or random access) data is required, only the affected 64-bit bank and module are made busy.

Backing storage

This memory system provides for backing storage, swapping storage of very high density. It will most likely be built of CCD memory components, with a capacity of from 32 to 256 million 64-bit words. This memory is a block transfer memory only, with the size of the block established by the operating system for the duration of a particular run. Minimum block size is 8 kwords, while the maximum block size is 65 kwords. A block-oriented error correction code is utilized in this memory to reduce the large overhead found in normal SECDED systems.

The access time for the backing storage is 100 μs to the desired block, with block transfer beginning immediately even though it does not begin with the actual first address of data to be transferred. This means that most blocks are transferred from the first address available until the end of the block is reached, and then the transfer continues with the first word of the block in a circular manner until the full block is transferred. All transfers begin on a 2048-bit boundary at both the backing storage system and the main memory system. This secondary memory system is capable of transferring 128 bits (for either input or output) every minor cycle for a peak bandwidth of 12.8 Gbits/s. The system is capable of bidirectional transfer simultaneously, thus yielding 6.4 Gbits each way. This is the same as 0.1 Gwords/s, making it possible to exchange the entire contents of main memory (maximum of 41,943,040 words) in 419 ms. A single swap of a 65,536-word block to and from main memory would require 655 μs.

Memory Interconnection

Figure 5-3 shows the interconnection relationship between the main memory system and the backing store. All input/output to peripheral subsystems of the front-end are performed to and from the backing store only. These transfers can only occur when the backing storage is not performing swaps with the main memory subsystem. Thus no method is provided to perform I/O operations directly with the main memory system. The two performance levels of main memory share the same trunks to the vector units, the scalar unit, and the backing storage media. The control system in the CPU permits simultaneous transfers of data with the backing store while operating the vector and scalar units in full streaming mode, as long as there is no conflict in the data regions being accessed.

Memory Addressing

In the descriptions of the physical characteristics of the memory systems, a range of capacity for each subsystem is given. This is to permit sizing of the facility to match the production problems without excessive memory size. In order to accomplish this with a simple memory address translation system, the hierarchy must be visible to the software system so that appropriate data can be stored in the proper level of memory.

To accomplish this, an adaptation of the STAR-100 memory addressing scheme has been chosen.

The following address definitions are expressed in hexadecimal notation (base 16).

- All addresses are 48-bit addresses. In essence, the programmer can address any single bit in memory; however, in this implementation, only control and order vectors will truly be accessible by bit. However, the concept of a uniform addressing scheme (whether it be bit, byte, or word) throughout the architecture has proven to be useful as well as consistent. Bit addresses up to, but not including, 000000100000 are reserved for monitor and system functions.

- Bit addresses from 000000100000 to 000000180000 directly reference the high-speed buffer memories in the vector units.

- Bit addresses from 000001000000 to 000021000000 directly reference the high-speed memory portion of the system (or whatever part exists in the current configuration).

- Bit addresses from 000100000000 to 000180000000 will directly address the optional medium-speed portion of the memory system.

- Bit addresses from 001000000000 to 001400000000 directly address the backing store.

- Addresses from 001400000000 upward are either illegal or are used by the operating system to trigger I/O actions or other system communication functions.

- All of a particular address space need not actually be present in the machine; however, the addresses reserved for that region will still be associated with that region of the memory. Any reference to the missing area will be treated as an illegal instruction and abort the job.

- Vectors or streams may not begin in one memory system and end in another. A single vector operation such as A=B+C can have each of its streams in a different portion of the main memory hierarchy (high-speed buffers, high-speed main memory, or medium-speed memory); however, the entire stream (A, B, or C) must begin and end in the same area of the memory hierarchy.

- Virtual memory as known in the STAR-100 does not exist. Each job has full use of the memory system, beginning with absolute address zero and continuing through the addressing structure. References to addresses in the second-level storage by a job program can only be by instructions that invoke block transfers of the minimum size specified by the operating system. In all cases of this type, one of the addresses must be a region in the main memory while the other addresses may be in the backing storage device.

Figure 5-9 shows a block diagram of the I/O system with only one I/O path shown. The I/O channel unit operates on a ready-resume basis (on 512-bit sword increments) with the CPU I/O unit. Transmission line lengths must assure a 32-bit (data word) transfer every 160 ns (in one direction) to ensure a channel average of 200 Mbits/s. This requires a physical placement of the I/O channel unit within 30 m (100 ft) of the CPU chassis, if data transmissions were to be asynchronous and on a 32-bit basis. However, it appears most convenient to drive the interconnects between the I/O unit and the I/O channel unit directly with F100 compatible levels, so that a direct connection may be made to LSI chip pins. The maximum allowable distance under these conditions (determined by RADL) is 10 m (32 ft).

The network trunk is a bit-serial line, which can sustain 50 Mbit/s transfers (with today's coax and modem technologies) for distances up to about 450 m (1500 ft) (depending on the number of connections made to that line). This permits the distribution of peripheral and multimainframe hardware across a broad geography to reduce centralized computer congestion (refer to appendix A at the end of this section).

Figure 5-10 shows a typical configuration organized around the proposed I/O scheme. Each of the serial channels connected to the I/O channel unit is capable of supporting up to 16 drops (connected units) while maintaining a 50-Mbit bandwidth. Where 819 disks are concerned, however, the data traffic for each controller is expected to be quite high (near saturation for some applications). Thus one of the lines must be available almost full time for each 819 subsystem.

The option of adding drops to a single line permits multiple attachments for redundancy (each 819 subsystem has at least two independent trunk connections) and for access by other mainframes.

In this scheme, the mass storage peripherals (844 disks in this case) are attached to the front-end via the trunk, rather than directly to the front-end itself. This permits data staging to occur for both the back-end and front-end systems without going through the front-end processor and its consequent bottlenecks for data transmission.

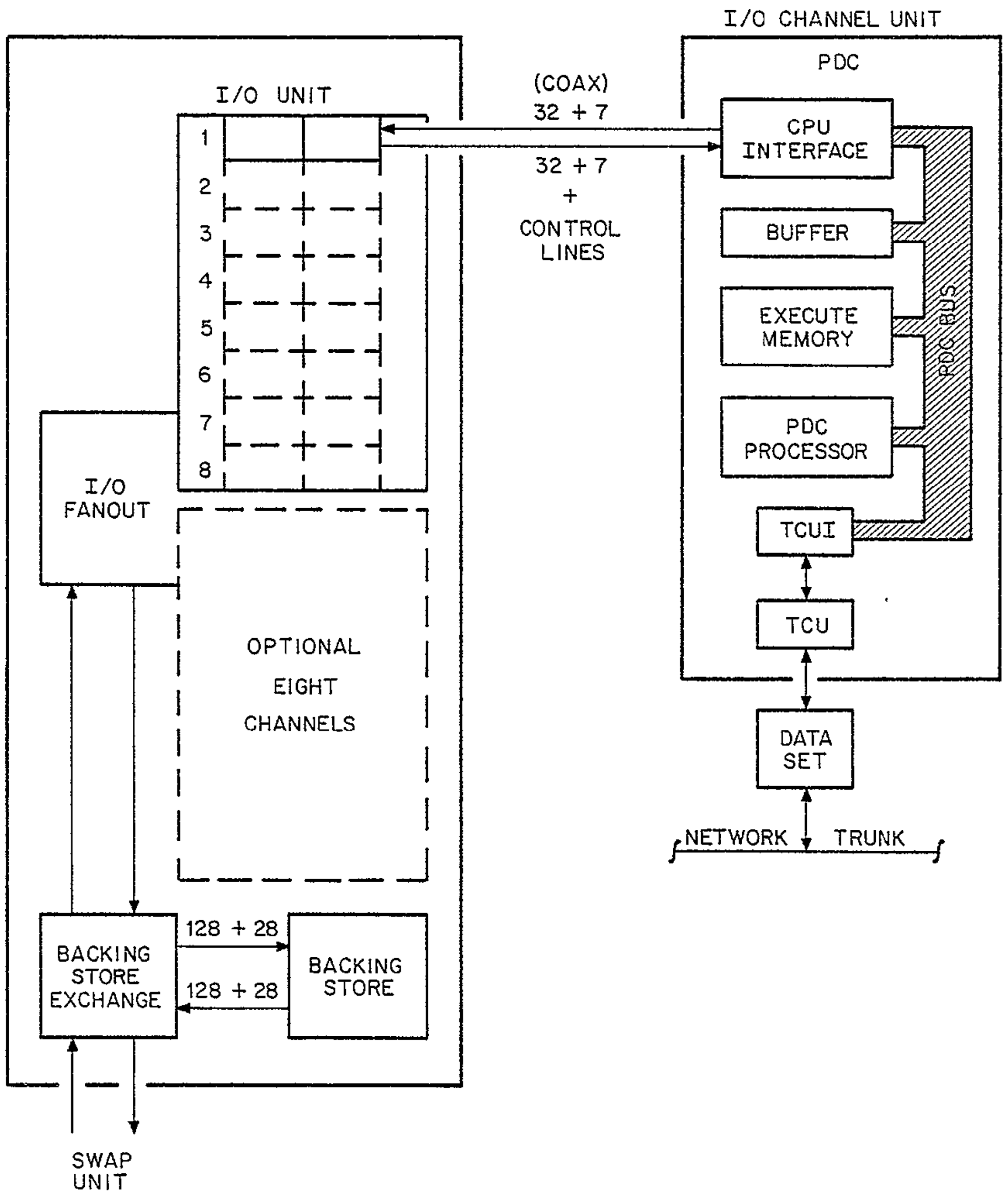


Figure 5-9. I/O Block Diagram

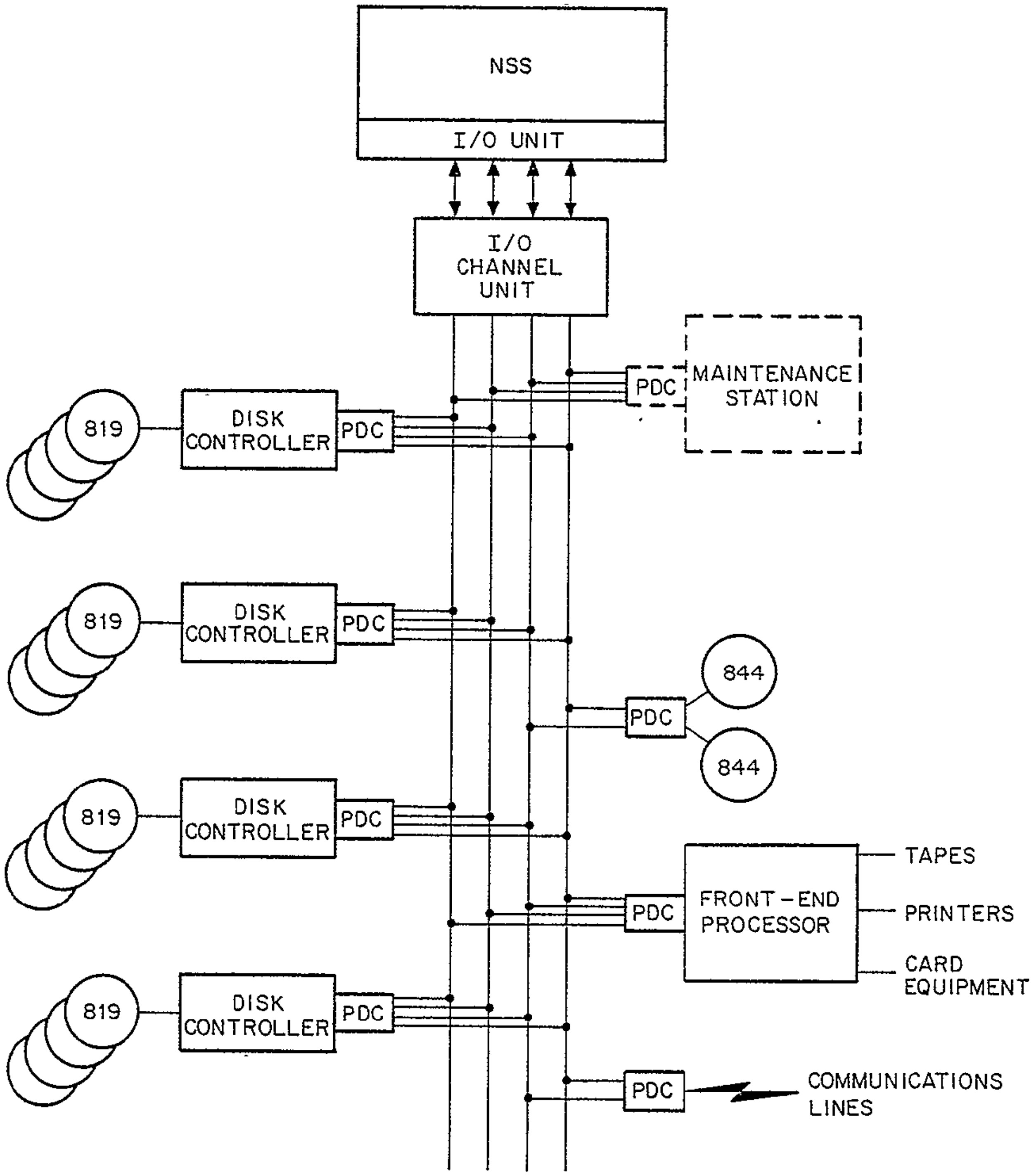


Figure 5-10. Proposed I/O Scheme

The maintenance station appears in dotted lines. Instead of a separate set of maintenance lines being required, the maintenance data can be transmitted by the I/O unit to any destination address given permission to be the maintenance station. This reduces the danger of having the NSS dependent upon a single entity such as the maintenance station in order to stay alive.

Since any attached, intelligent entity can perform maintenance station functions (if the software exists), it would be possible to eliminate that dedicated station (and its costs) by transplacing its functions to other stations in the system or even a front-end processor.

With the flexibility in attachment shown for the proposed I/O scheme, it can be seen that additional mainframes and peripherals can be readily attached to the system. Further, these functions will require only four or five channels. This constitutes what will probably become the minimal I/O subsystem and directs the form of packaging for the I/O channel unit (probably four PDC units in a single chassis).

Definition of Terms in Figure 5-9:

I/O Unit	The portion of hardware that provides the I/O function, the interface between the backing store exchange, and the I/O channel unit.
Backing Store	A high-speed, fast-access, block transfer device for paging to and from memory.
Backing Store Exchange	A 128-bit, bidirectional connection between the backing store and the swap unit or I/O channels. This exchange is synchronous with the CPU, with the CPU clock being sent to the backing store.
Network Trunk	The basic communications mechanism between all elements of the system. The trunk consists of a single line, bit-serial transmission system using a CDC-developed SDLC-type protocol for the message (data block) envelope (refer to appendix A).
Peripheral Device Controller (PDC)	The basic building block of the network trunk system. The PDC contains buffering, a microprocessor for control, bit-serial trunk interfaces, and custom-design device interfaces.
I/O Channel Unit	The connection between the CPU I/O unit and the network trunk. There may be from 1 to 16 I/O channel units (IOCU). These units operate up to rates of 100 Mbits/s.

INTERFACE SPECIFICATIONS

ELECTRONIC

At this stage of the system design it is impossible to give a complete engineering specification of many parts of the system, particularly the electrical components. There are some general areas that can be discussed however. The primary logic family will be emitter coupled logic (ECL) with all memory except for the backing store being built with bipolar parts having input/output levels compatible with the ECL external gates. Transmission of signals will be via 75-ohm foil paths or high-velocity coax for NSS signals and I/O trunks. Other electrical specifications will be related to the off-the-shelf equipment which is chosen for the front-end and peripherals. Although the circuit technology for the NSS must of necessity represent the most advanced state-of-the-art, all interfaces will be designed to conform to the interfaces of a standard family of parts such as the MECL 10K or the F100K families.

LOGICAL

In Control Data designs using ECL, a negative-going pulse is considered a logical one, and a pulse becoming less negative, reaching a threshold close to but less than 0 volt, is considered a logical zero. This has been described as negative or inverted logic. All interfaces within the NSS would conform to this standard, while all external interfaces would meet standards required by the peripheral and front-end systems.

At a higher level, the NSS system is broken down into logical units, each of which has a clearly defined and narrow interface to other components. For example, the swap unit contains interfaces for six 128-bit data trunks (plus SECDED), plus a 72-bit control interface that describes the function to be performed, the addresses to be accessed, and the length. The unit returns one status line to the scalar processor. There are no other interfaces. Each unit marked as an entity in figure 5-2 has such constrained interfaces. Further specification of logical interfaces must be in the form of block-model interface specifications which are part of the projected next phase of this study.

USER

The primary user interface to the NSS will be through the software and hardware supporting the graphics and correspondence terminal subsystems. The application codes will provide parameter formats and input data checking consistent with the operating environment established at Ames Research Center during the development of the NASF.

A selected group of NSS users will deal with the creation and checkout of the operational codes. The secondary user interface will therefore be the compiler language and debugging tools offered for access to the NSS. The choice of programming language is complicated by several issues.

- The different kinds of programming languages that are available today for scientific programming, such as FORTRAN, ALGOL, PL/1, and even PASCAL. These intermediate-level languages are supplemented by various extensions created by vendors and users for their own particular operating environment.
- The evolution of a set of problem-oriented languages which are uniquely created for specific computational environments. An example of such a language is DYANA, created at General Motors, which is used by engineers to describe dynamics problems (such as an automobile suspension system) in a language with syntax nouns and verbs familiar to that discipline.
- Current, traditional, accepted practice that has created countless computer programs, procedures, and documentation creates an enormous inertia that opposes changes in approach.
- The degree of hardware concurrency apparent in the NSS presents a challenge for the programmer to control the extensive parallelism.
- The state-of-the-art in compiler development is just now yielding modest results in the optimum utilization of concurrent resources through automatic recognition of parallelism and generation of optimized object code streams.

Many persuasive arguments exist in favor of abandoning current practice (particularly that which involves FORTRAN) and adopting a new language and programming system for a future machine such as the NSS. The ability to begin anew and avoid the pitfalls and poor programming practices of the past is enticing. However, the current generation of programmers and engineer-analysts has become deeply rooted in a predominantly FORTRAN language environment. An incredible inventory of card decks, tapes, and disks exists containing FORTRAN programs, many of which have an ancestry that is untraceable. Therefore, the actual content of some algorithms is buried in the mass of

FORTRAN code that has evolved over many years. Further, many modus operandi have been established for diagramming and flowcharting narrative documentation. Even conversation with nonprogramming personnel has had a dominant effect on most system developments in the United States. Words like dimension, common, real, and integer have become common in the vocabulary of engineers and designers who are not themselves directly involved in programming. They are convenient words, however, and are used in discussions among mathematicians, analysts, and programmers who are attempting to obtain results from some large-scale computer.

The fact that the American Standards Committee has invested an enormous effort in the development of a new FORTRAN standard, with concomitant investments on the part of user and manufacturer alike, is indicative that FORTRAN will continue to evolve and enjoy the full support of vendor and programmer. Since FORTRAN apparently will not vanish, by fiat or default, it can be expected to continue as the mainstream programming language for the 1980's (at the very least).

For these reasons, it is recommended that the NSS programming language be the FORTRAN language as described in the forthcoming standard to be published this fall (1977) by the American Standards Committee. It is further recommended that this language be extended to permit the description and manipulation of vectors explicitly, rather than relying on the implicit recognition of vector operations in the sometimes complex DO loops.

While the style and form of the basic standards are unarguable, the method of extension to fit the NSS could become a complex study project itself. It is suggested that, as a minimum, dimensioned arrays be described in terms of vector processing in one of two ways.

1. One way is to use familiar FORTRAN constructs with slight extensions, whereby the programmer can refer to a portion of an array as a vector of length L by using the array name, followed by the subscripts needed to define the starting point of the vector, followed by a semicolon, and then by the length L, which can be either a variable or a constant.

   DIMENSION A(100,100,100)

A(I,J,K;300)=1	Beginning at the element A(I,J,K), process 300 vector elements inserting a one into each.

2. Since such references require a lot of writing and are cumbersome, a second way (or form) is needed. The form used on the STAR-100 is called descriptor. This permits the use of a simple variable name to indicate a vector reference, thus uncluttering the program. The descriptor attribute is assigned to a variable by use of the DESCRIPTOR declaration. The intrinsic characteristics of the variable defined as a descriptor are the same as for any FORTRAN variable (that is, integer, real, or complex) and can be changed by the use of the corresponding FORTRAN REAL, INTEGER type statements.

   Before use in the program, the actual vector description must be set into the descriptor variable. Note that the vector description will remain unchanged after the first definition and thus may be reused any number of times until changed by a subsequent ASSIGN statement. The values of subscripts and length used in the ASSIGN statement may, as in the previous example be variables or constants. Thus:

```
    DIMENSION A(100,100,100)

    DESCRIPTOR X

    ASSIGN X, A(I,J,K;300)

        X=1
```

   performs the same operation as above.

While the first reflects the direct operation on the A array, the second is more succinct and can be meaningful when the plane or strip beginning at A(I,J,K) is a distinct entity such as shock boundary.

Although the syntax shown is derived from STAR-100 FORTRAN (reference 5-2), the specific form is not important but the function should be retained. So too should subarray references be supported.

```
A(I;30,J,K)    and     A(*,J,K)
```

The first example defines a subarray of the array A beginning at I and consisting of a subplane of 30 elements in the I-direction at the subscripts J,K. In the second case, the asterisk in any subscript position in the expression denotes all subscript values for that dimension. In this instance, it means all I, or the full plane at J,K in the array.

Some form of describing the creation of the control vector (for boundary condition control) should be retained. The normal conditional expressions on FORTRAN can be extended to make this possible. For example, in standard FORTRAN one could state

```
LOGICAL Z
```

and later in the program

```
Z = A.GT.B
```

where the TRUE or FALSE value is set into the scalar logical variable Z depending on the results of the conditional test. The extension to vectors then implies that a bit string Z can be defined to contain more than one TRUE or FALSE value, and the results of each vector comparison then sets the corresponding bit in the Z-vector to TRUE or FALSE. Thus:

BIT Z (300)	
Z(1;300) = A (I,J,K;300).EQ.3.14	Creates a bit vector where bits are set to one whenever the comparison is true for the corresponding element of A and set to zero when the comparison is false.

Finally, some means must be available to inform the compiler of the allocation of data relative to the memory hierarchy. Thus,

```
LEVEL1      A(10,100)
LEVEL2      B(20,30)
LEVEL3      C(100,100,100)
```

would allocate array A to the high-speed buffers in the vector unit. Such an allocation prohibits A from appearing in a common statement, an I/O statement, or from being specified as a passed parameter to a subprogram. LEVEL2 data such as the array B would reside in the high-speed main memory (or the optional medium-speed memory) and would have all the benefits and rights of a normal FORTRAN variable array. LEVEL3 data would reside on the backing store system and be subject to the same restrictions as LEVEL1 data.

With these few extensions, the FORTRAN language and related mathematical and I/O support routines would appear more than adequate for the NASF. The resulting language can be easily compiled for optimum utilization of the NSS parallel hardware. In Evaluation Against Benchmark later in this section, some of these notational conventions have been used in the description of the implicit benchmark applied to the NSS.

The lowest level of user interface would of course be the machine language level. It is obvious that a need will exist to produce some machine code for the NSS for highly optimized object-time FORTRAN library routines, as well as for the limited monitor code necessary to operate the main hardware complex. The vehicle for this type of programming in the past has been assembly language, which maps one input statement for one machine-language instruction. Computer system development has matured to the point where a language compiler system based on PASCAL can be invoked, quite successfully, in the production of systems software, including high-utilization mathematical subroutines. Thus, even machine specific code can be generated with a higher-level language. In this effort, however, the programmer must be aware of the architecture of the NSS and the behavior of certain sequences of instructions. Specifically, the programmer will have to deal with:

- Scalar instructions of the normal form (loads, stores, arithmetic, test-and-branches, loop control, and error flag sampling)
- Vector mapping instructions which control transmission of data to and from the vector units, control vector processing, and the map unit functions of compress, mask merge, and scatter/gather (for transpose and nonsequential access of vector data)
- Vector arithmetic instructions
- Swapping instructions
- System control instructions (executed by monitor only) such as exchange and I/O start-stop

PROGRAMMING CONSIDERATIONS

Most of the performance requirements of the NSS revolve around the speed of computation in the heart of the Navier-Stokes solutions, assuming that input/output times are not significant. The applications programmer, as well as the compiler developer, must be as aware of the details of machine structure and control as possible to ensure the optimum match of problem solution and computational hardware. The data flow in a typical problem would appear somewhat like this:

1. The front-end computer would compile any needed application programs and load and link these programs together, storing the result on high-speed mass storage devices.

2. Under control of input parameters, the front-end would then open all required data files, assemble maps for the location of the data base, and flag the NSS monitor that a job is ready and queued.

3. The NSS monitor would enter this job request in its queue and at the appropriate time, load the backing store from mass storage with the entire job (instructions plus data base). Meanwhile another job would have been executing, utilizing the other half of the backing store.

4. When the job in progress terminates, its entire instruction stream and all data in main memory are swapped back to its portion of the backing store. This will require, at most, 80 milliseconds during which time no other computations can be done. Simultaneous with the swap out of the existing job, the new job is swapped into main memory from the backing store. All of the instructions, scalar operands, constants, address lists, and starting slice of the data base are brought in with this initial load.

5. The job is started into execution by the monitor which exchanges into the scalar unit the starting register file and invisible package (which contains real-time clock information and program address).

6. As the job executes, the scalar unit will manage the flow of the program, issuing instructions to the vector units, the mapping unit, the swap unit, the branch unit, the load/store unit, and the scalar floating-point unit.

7. At some point in the computation, new slices of data base must be brought into main memory, while updated slices are returned to the backing store. The program will issue direct swaps for this data without the intervention of the monitor.

   Example:

   ```
   LEVEL 2 A(100,100,100)
   LEVEL 3 B(100,100,100),C(100,100,100)
   C(1,1,1;1000000)=A(1,1,1;1000000) $  A(1,1,1;1000000)=B(1,1,1;1000000).
   ```

   Since swapping takes considerable time (relative to the computation speed of the NSS), the programmer should consider that other computations can be performed while awaiting the data from region B to be swapped-in. This is no different than the technique of double buffered or overlapped I/O that is commonplace in computing. If the program needs the swapped data before it can continue, the statement:

   ```
   10  IF(.NOT.(A(*,*,*)))    GO TO 10
   ```

   could restrain further processing until the array is fully swapped and in place in the main memory.

Machines of the pipeline variety discussed in this system design possess a negative quality which should be addressed. This aspect is variously called startup, overhead, or priming time. It consists of several components.

- Instruction decode and vector address formation
- Starting the vector unit
- Memory request and access time
- Request and data transmission time to and from the memory
- Pipeline fill or prime time
- Data transmission to memory time

The time required to transmit the memory requests to the memory module and to transmit the resulting data from the modules to the functional units is the greatest part of the startup time. The very act of configuring the NSS with a large amount of main memory creates this problem. The physical space required to house the memory, the additional space needed for cooling, and maintenance access dictate that the memory will not be physically intimate with the functional units to which it must send data. The distances traversed can require up to 80 nanoseconds or eight minor cycles. Obviously, each minor cycle used for overhead reduces the peak rate (of 3.2 gigaflops) for short vectors to a value significantly less than 1 gigaflop.

Two solutions present themselves to the NSS designer. One involves the introduction of stacking hardware so that up to three vector operations for execution can be prepared while one is actually being executed by the mapping unit. The second solution is to provide the high-speed buffers in close contact with the corresponding vector functional unit. In the case of the high-speed buffers, the memory request, access time, request transmission time, and data transmission to memory time are reduced to a total of two minor cycles. This feature coupled with the stacking facility of the vector control system permits buffer to buffer operations to operate with an overhead of two to six minor cycles.

The concurrent nature of the tandem vector units with the vector mapping unit permits additional manual optimization to be performed by the programmer. In particular, it is advantageous to end with several short vectors in the buffer registers over which several computations can be performed. This leaves the memory trunks free to be utilized by the map unit to perform transpose, compress, merge, and gather/scatter operations. To the extent that local operations can be sustained in the vector units while the mapping unit is busy, all of the overhead of organizing data for efficient linear vector operations can be buried. It is felt that for both the explicit and implicit codes, such overlap is possible, and consequently, the odd modes (nonsequential access) of memory access needed for the 3-D forms will not penalize the NSS architecture.

There is a degree of concurrency within the vector units themselves. At best, one can utilize the fact that any stream read from the buffers can provide both input operands to any arithmetic element. Thus, the operations A*A,B*B,C+D could be proceeding simultaneously, with one result A*A being chained with the result B*B through the remaining ADD/SUB unit to form A^2+B^2. With the interconnections available, both result streams could be returned to the buffer memory in the same minor cycle, while the remaining buffer read stream could be sent to memory. Obviously, a careful analysis of the NSS structure is necessary to achieve the optimum performance possible. Note, however, that for every minor cycle when both ADD or both MULTIPLY units are producing useful results, they are not able to check one another. Thus, as the programmer achieves higher and higher concurrency within the vector units, confidence in the integrity of results from those units diminishes.

The elimination of input/output instructions for the programmer means a major reduction in system overhead encountered in the NSS CPU. If at all possible, the programmer should be discouraged from performing formatted type I/O in the NSS but should rely on the front-end system for such activities. FORTRAN ENCODE/DECODE statements will remain available, however, for those cases when data must be formatted in the NSS before transfer to the backing store for later transmission to I/O devices.

The NSS provides the ability to repeat single elements, scalar values, or portions of vectors.

Termination conditions

The termination of any input vector before the result vector is complete creates the need for defining what the remaining input elements will be. The options are:

- Fill the remaining input items with machine zero (in the case of ADD operations) or with a normalized floating-point one (in the case of multiply/divide).
- Fill the remaining input items with an illegal data item (such as the indefinite operand used on the STAR-100).
- Fill the remaining vector elements with a scalar quantity taken from the scalar unit register file.
- Fill the remaining element of the vector by beginning at the first element of the vector again and continuing until the output vector is filled.

Repeat conditions

An element, or group of elements, may be repeated several times according to a repeat count.

- Repeat each element N times.
- Repeat a column of L elements N times before proceeding to the next column.

These operations are permitted on any read stream from the buffer memory to the arithmetic units and also are permitted in the MAP unit on any input stream. These functions, plus the scatter/gather and compress,mask,merge operations, provide the core of the transpose and ZETA-direction sweep required of the Navier-Stokes programs.

As the NSS design is refined in later stages of the NASF project, and detailed timing relationships become known, as well as the behavior of the compiled object code, a more comprehensive set of programming guidelines will be required.

FUNCTIONAL DESCRIPTION

SYSTEM BALANCE

In this report, the concept of system balance has been used frequently. In a pragmatic sense, this concept, if followed, ensures that the overall system achieves the best economies in dollars and parts counts for the performance desired. Balance can be viewed from the perspective of system capacity or from the bandwidths needed to meet the performance throughout the entire system. In the discussion of the overall system design earlier in this section, data volumes needed at main memory, backing store, mass storage, and archival storage were quantified. Within the boundaries of the outline envisaged today for the use of the NASF, these items seem to balance the capacity requirements.

System bandwidth requirements up to, but not including, the NSS itself are easy to determine. If a production job in the NSS requires 7 minutes to complete, then during this time the mass storage system must be able to perform a complete swap of one job for another into the backing store. Allowance must be made for conflicts in the use of the backing store bandwidth and trunk requests for loading and unloading the same I/O device with data transmitted between peripherals and the front-end/graphics subsystems.

This means that 128 million words (assuming 1/2 the backing store is swapped) needs to be exchanged in that 7-minute period. A total of 256 million words is thus exchanged (128 million each way), which is approximately $16x10^9$ bits. The transfer rate of one network trunk, shown in figure 5-2, is $40x10^6$ bits per second which would require 400 seconds or 6.7 minutes to complete the exchange. Since the I/O port capability of the NSS I/O unit is $1.6x10^9$ bits per second, it is apparent that an additional $40x10^6$ bit trunk is easily accommodated. Thus, with no more than two trunks active (for performance purposes) and with two additional trunks available (for redundancy and low-volume data traffic), the network is sufficient to handle the expected load.

The front-end systems and graphics support systems can operate on these trunks but must process a substantially lower volume of data or they will become engulfed with data disgorged by the NSS. This fraction (about 20 percent) is expected to require no more front-end bandwidth than will be available from projected off-the-shelf equipment.

The NSS presents its own picture of system balance. The choice of 12.8 Mbits for the channel between the backing store and the main memory is based on the estimates of calculations that can be performed on a given piece of the data base once it is brought into main memory. With 35 to 50 computations being performed per element transferred on this path from the backing store, the vector units can be kept quite occupied (above 85 percent) while actually paging data into and out of the main memory system.

The main memory is matched to the bandwidth rates of the vector pipelines (1536 bits per minor cycle required) with sufficient excess bandwidth to support the instruction streams, control vectors (128 bits per minor cycle maximum rate), and the swapping store channel (128 bits per minor cycle maximum rate). The excess bandwidth (256 bits per minor cycle) is needed to take up the slack due to occasions when memory conflicts or trunk conflicts arise.

The main memory capacity, discussed earlier in Flow Field Simulation Model, is based on containing sufficient working storage to act as buffers for the swapped data base elements, temporary vectors and constants, and local variables needed to keep the vector units running at close to their peak rates. The peak rate of 3.2 gigaflops occurs only when all ADD and MULTIPLY units are running, which arises only a small percentage of the projected time. The cases where three of the units (multiply, multiply, and add) can be kept busy arise quite often in the probable three-dimensional codes. This configuration yields 2.4 gigaflops and will probably arise during 20 percent of the total execution time of the problem. It is expected that the vector units will be operating for 85 percent of the total problem time at rates from 0.8 to 2.4 gigaflops. In fully 64-bit mode, the sustained rate should be above 1 gigaflop.

SYSTEM INTEGRITY

The need for a high confidence in the results produced by the NASF and the requirement that the system availability attain an average of 98 percent in order to meet customer demands dictates extraordinary measures to provide system integrity, at least in relationship to what has been acceptable for supercomputational facilities in the past. Several commonplace methods of providing such integrity can be seen in figure 5-2. These include multiple, redundant components and the multiple data paths to provide for alternative means of keeping the system alive in the event of single-point failures. The most difficult to diagram is the extent to which system software must have embedded within it the ability to recognize and respond to crisis failure conditions automatically. The need for automatic recognition becomes greater as the number of facilities users grows, and with that number, the volume of data being transferred within and in exchange with the facility becomes enormous. The time required to alert a user for action in most superscale systems currently is too long to prevent major portions of the system from losing data, programs, and integrity. The development of such redundant, fail-safe software is the major focal point of most system developers in the 1970's. Extant systems demonstrating the production status of such software abound, and multimainframe, multitrunk configurations are available to measure degrees of system integrity and for the development of specialized features that might be needed to support the NASF. A major subject for future project concern is the monitoring of candidate off-the-shelf systems' reliability and integrity preserving software.

Within the NSS itself, integrity of operation is crucial but creates some dilemmas for the developer who must maintain the fine line of compromise between hardware and parts count costs, desired performance goals, and reliability features. Within the NSS, there are three classes of networks which can fail, detected or undetected. These networks are memory, data path (including add and shift networks), and control. The undetected failures can be catastrophic, since result data may be used to arrive at some major, erroneous design conclusions. Although no design would ever be pursued, let alone built and flown, on the basis of a single, perhaps spurious set of results, nevertheless some viable design alternative might be discarded because a first-look simulation yielded unsatisfying results with costly consequences.

Thus, the issue of integrity has been of paramount importance to Ames researchers and has rightfully been assigned first priority in their considerations.

Conclusions on Integrity

After examining the alternatives available to provide integrity within the NSS, the following conclusions have been formulated.

- Except for a completely identical machine being included as part of the NASF configuration, assuring 100-percent reliability of operation and answers is impossible. This being to costly to consider, an alternative is a level of reliability that will ensure correct answers and 98-percent system availability.

- The reduction of the probability of failures occurring (particularly parts failures) is directly dependent upon a reduction of parts counts, and the establishment of stable operating parameters through judicious packaging and imaginative preventive maintenance techniques.

- Parts counts at the required performance level indicate that high-speed LSI is essential to achieve a manufacturable commodity. Stable temperatures at the expected power dissipations must come from liquid cooling (to reduce junction temperatures, thus reducing failure rates) as well as reducing thermal shocks which occur during power cycling of the system.

- The use of microdiagnostics, separate maintenance stations (programmable devices for locating failures), and built-in sampling points for maintenance purposes will permit more extensive preventive analysis and repair to be performed during regularly scheduled sessions on the NSS.

- The greatest expected source of failure will be in the memory systems because of the volume of parts required, their density, and the power dissipation expected within them. It is calculated that the mean time to failure for parts in these memories is less than the practical maintenance interval. Therefore it is absolutely necessary that both the backing store and main memory systems possess the SECDED facility, which permits the NSS to continue operation even when a memory component fails.

- Wherever possible, the SECDED data, 7 bits per 32 data bits, should be carried as far along all data trunks as possible, thus ensuring the integrity of the trunks.

- All data paths where SECDED is not practical (such as shift networks) should carry a parity bit throughout. On the NSS, this means one parity bit for every 32 bits of data.

- The most difficult failures to detect are in the control networks. Thus, simply carrying some form of parity or redundancy check through an add unit is not sufficient to guarantee that the unit is producing reliable results. A simple failure in a fanout or normalize count can create havoc with the results while producing correct parity at the output of the unit and thus a false sense of security.

- The need for a very high-speed multiply unit precludes the ability to carry parity (or any other redundant data scheme) through the multiply system.

- Alternative error checking systems such as modulus arithmetic can check algorithms but cannot handle the end-cases rampant in any computer arithmetic system known. For example, the need to right-shift a normalized result in two's complement addition when the uppermost bit overflows is difficult to implement in a simple, inexpensive modulus scheme. The amount of hardware to accomplish a full checking system would be equal to or greater than the unit being checked.

- With eight vector units operating, some form of arithmetic checking is mandatory. Thus, the divide unit of the original design was discarded and replaced by additional multiply and add units. The divide can be accomplished at a slower rate with these units, but when a given unit is idle in this design it can be used for checking its twin unit.

- This feature coupled with full parity or SECDED in all buffers and trunks covers most of the hardware that produces high bandwidth results. The confidence in the results of such an ensemble should be close to 99 percent.

On-Line Maintenance

As on the STAR family of computers, a large number of CPU signals, registers, and internal controls are available to be sampled or strobed by the maintenance station (MS) function. This function can be performed by any processor, attached to the network trunk, which has been given permission through the setting of a set of locked switches establishing that processor's identity as the maintenance station.

While the NSS is in operation, certain diagnostic modes can be established for periods of code execution, time-sliced with the actual running code. This permits a continuous monitoring of NSS function integrity while concurrently running codes. These diagnostic modes are reserved for the maintenance station alone. In addition, there are over 200 special bits of data transmitted each way between the maintenance station and the NSS for controlling and monitoring conditions in the CPU. In particular, all parity errors or SECDED failures are transmitted to the MS for recording in a master error log, and further action, by either the front-end processor or the operator.

Control lines from the maintenance station allow the operator to degrade certain portions of the machine, such as a reduction in memory by 1/2, so that a failing half can be repaired by customer engineers while the NSS continues to operate on a production problem. The same thing can be done to the pipelines, reducing them to 4,2, or 1 pipeline with an attendant reduction in performance, but still allowing the production job to continue.

With the maintenance monitors and the controls available to the maintenance station, a number of automatic recovery procedures can be developed for actual operational codes. In addition, complete error histories and actions taken by both the hardware and the operator/customer engineer are kept in a permanent file system for review by both user and vendor. With proper security precaustions via passwords and keys, vendor's engineering personnel can log onto the maintenance station processor via a remote terminal and perform analysis of the system behavior. In addition, when permitted by a special password from the customer's site, the remote engineer can actually participate in the maintenance session with the on-site engineers.

Expected Failure Rates

Table 5-1 presents some estimated parts counts and projected failure rates for major portions of the NSS. In addition, statistics from four CDC CYBER 173 sites show a hardware system interruption rate of 1.05 per system per month. These rates support an expected high availability for the proposed system.

TABLE 5-1. PROJECTED FAILURE RATES FOR NSS

Function	Estimated Parts Count	Basic Fail Rate Failures/MHr [1]	With SECDED Failures/MHr [2]
Main memory	150,000	15,000	150
Backing store	80,000	8,000	80
Vector units	10,000	1,000	900
Control/scalar	1,500	150	135
I/O channels	1,500	150	140
Mapping unit	1,000	100	80
Swapping unit	300	30	27
	244,300	24,430	1,512
		= 17.8 Failures/ Month	= 1.1 Failures/ Month [3]

[1] Basic failure rate is assumed to be 0.1 failures per million hours per component.

[2] SECDED is expected to improve the memory error rate by a factor of 100. The worst-case improvement is a factor of 10. Others have used up to a factor of 1000 with good memory maintenance.

[3] This failure rate can perhaps be projected to a system interruption rate of two to three per month.

The expected failure rate for the NSS as given above may seem very small for a machine of its capability. This result is due to two main factors.

- SECDED

  Control Data has machines in the field, in customer installations, that have never experienced a failure due to memory (time about 1 year)..

- The vastly reduced component count due to LSI

  The history of component failure in Control Data equipment has shown that if junction temperature is kept in a moderate range, the failure rate for a complex computer system is directly proportional to the number of connections in a system. The number of connections is the total number of all connections. For example, a discrete transistor is seven connections: three to the board, three lead connections to the silicon, and one header connection silicon to package. The level of integration will vastly reduce this connection count.

EVALUATION AGAINST BENCHMARK

In order to illustrate some of the numerous ways in which architectural parameters influence performance, programming style, intermediate storage swell, and so on, the following describes alternative code formulations for each of several functional areas of the two-dimensional implicit code. In each case, the actual expansion of the program will be written in terms of clusters of array operations which execute in parallel and overlapped sequential modes. Certain of these clusters (refer to Programming Considerations) will complete 24 64-bit operations or 48 32-bit floating-point operations in each machine cycle (for a machine cycle of 10×10^{-9} seconds, that results in a peak result rate of 4.8 billion operations per second or 4.8 gigaflops). This rate is based on the probability of utilizing three of the four arithmetic elements in each pipeline for prolonged periods of the code execution. The theoretical peak rate in 32-bit mode using all elements is, of course, 6.4 gigaflops.

This discussion will be limited to those aspects of programming the NSS which are appropriate to the split-operator, implicit algorithm which employs a central difference scheme. Such a program is an incomplete exercise for a processor with the full range of Iverson operators (reference 5-1) compress, mask, merge, gather, and scatter, which will be exercised extensively by data-dependent branching (McCormack explicit code) and data rearrangement and conformation (Montreal Spectral General Circulation Model). During 1976, Control Data expended a great deal of effort in studying the impact of spectral weather models on the architecture of a vector machine. In addition to vectorizing, the finite Fourier transforms and time integration processes, the transformations between spectral coordinates and grid space involving associated Legendre polynomials (with complex coefficients) were vectorized since they consumed a sizeable percentage of computer time. This experience led to inclusion of pipeline control capabilities such as the repetition of an exhausted stream (e.g., repetition of a single row of a pre-multiplier matrix to produce products with all columns of a post-multiplicand matrix, thereby eliminating numerous short-vector startups) and ability to enable recursive linkage in the pipelines. The integration of Iverson operations in the mapping unit with vector arithmetic and linked/iterative/recursive feedback operations in the pipelines fits solution methods such as SOR, LSOR, SIP, and direct solvers extremely well. In a subsequent, more detailed study, it is anticipated that vector programs derived from production codes will be presented to illustrate the general applicability of the NSS processor architecture.

Each subprogram representation is accompanied by estimates of execution time based on the pipeline model, N=P+L/M. N represents the number of cycles required to execute a cluster of array operations, P represents the number of cycles for start-up of the pipeline mechanism, and L/M represents the number of cycles consumed in processing (L operand sets at a rate of M outputs per minor cycle).

The essence of a vector processor is the way in which the arithmetic operations acquire operands and dispose of results. Hardware which enables simultaneous acquisition of eight pairs of operands and disposition of eight results in each of several consecutive clock cycles is possible only if each of the streams consists of contiguous array elements. Such stream references are called vectors since they are characterized by a starting point, direction, and length.

If one defines a two-dimensional array with

```
DIMENSION A(4,3)
```

a FORTRAN compiler will allocate the column elements to contiguous memory cells as represented in figure 5-11. A loop which advances the first subscript would reference a series of contiguous elements. For example

```
    D0 1  I=1,4
  1 B(1)  =  A(I,2)
```

would be a vector reference to the second column of A as denoted by the marks next to elements 5,6,7, and 8 in figure 5-12.

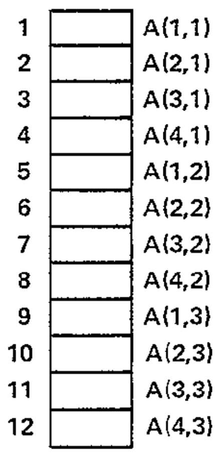


Figure 5-11. FORTRAN allocates 12 contiguous cells columnwise in response to DIMENSION A(4,3)

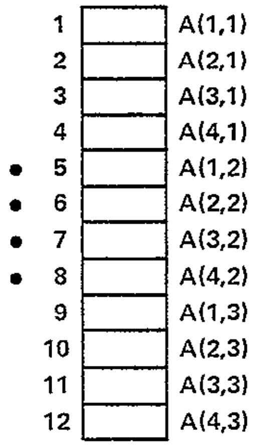


Figure 5-12. A contiguous, i.e., vector, reference to A by

```
      DO 1  I=1,4
    1 B(I)  = A(1,2)
```

A loop which references a row of A is exemplified by

```
      DO 2  J=1,3
    2 B(J)  =  A(2,J)
```

In figure 5-13 it is seen that A(2,3) is four cells distant from A(2,2) and eight cells distant from A(2,1), i.e., the indexing function is periodic with a period of 4 (the first dimension of the array).

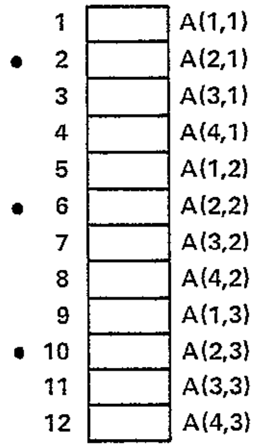


Figure 5-13. A discontiguous, i.e., pencil, reference to A by

```
      DO 2  J=1,3
    2 B(J)  =  A(2,J)
```

The following notation is introduced for the vector reference of loop 1

B(1;4) = A(1,2;4)

where B and A should be thought of as descriptors for their respective arrays, i.e., quantities which describe where the first element of an array is to be found and how many elements comprise the array. The parenthetic suffix to the descriptor provides a vehicle to describe a subarray which is a vector commencing with element 1,2 of A and extending to include element 4,2. The semicolon is a syntactic artifact which discrimates a vector reference from a scalar reference to a single element such as

```
A(2,3)
```

The quantity appearing after the semicolon is the length of the vector reference.

Another syntactic representation for

```
   DO 1  I=1, 4,1
1  B(I)  =  A(1,2)
```

is

```
B(1: 4: 1)  =  A(1: 4: 1,2)
```

where the colons are used to discriminate the three DO parameters: N1, N2, N3 with N1:N2:N3 implying that an initial value of N1 is to be used and an increment of N3 is to be added to form each successor subscript component until N2 is exceeded. For ease of visual identification colon notation will be employed for non-contiguous periodic references; hence,

```
B(1;3)  =  A(2,1:3)
```

would be written in lieu of

```
   DO 2  J=1,3
2  B(J)  =  A(2,J)
```

where 1:3 is understood to mean 1:3:1 just as J=1,3 means J=1,3,1. The vector patois refers to loop 2 and its equivalents as periodic gathers.

The presentation that follows will trace the flow of calculations through a complete step of the iteration process which leads to generation of the asymptotic solution to the problem. Throughout, a tutorial approach will be attempted in that unusual aspects of the FORTRAN-like programming language and unraveling of the data structure will be explained prior to their occurrence.

In this discussion, several parameters will be mentioned. Their relationships are:

LS	LJ	LL	KMAX	JMAX
100	60	30	50	100
256	80	40	64	128
800	100	50	80	200

LS usually indicates the stored length of a vector (length of slice), and LJ and LL indicate the number of actual operations performed in the J- and K-directions.

Figure 5-14 gives the FORTRAN form of STEP, while figures 5-15 and 5-16 diagrammatically illustrate the loop structures present in the implicit code. Figure 5-15 shows that STEP calls RHS, which in turn calls BC. The vertical bracket extending from CALL RHS to the bottom of the figure indicates the scope of RHS (Right-Hand Side calculation). When a bracket has a J next to it, it signifies a DO loop which operates on the first dimension of arrays, while a K-bracket denotes a loop which spans the second dimension of higher dimensioned arrays. The first calculations are accomplished in subroutine BC where the various boundary conditions are imposed. BC consists of five simple loops (three on J and two on K), which "patch-up" the edges of the problem variable arrays. BC accomplishes only a small number of floating-point operations; hence, the reader might ask "Why bother vectorizing it?" The answer is that on any parallel processor scheme, the imposition of boundary conditions may consume an inordinate amount of computing time and, hence, the burden is on the proposer of the architecture to prove that the "tail does not wag the dog".

Loop 81 of BC is displayed in figure 5-17. DO parameters J1 and J2 are, respectively, the subscripts for the top downwind edge and bottom downwind edge. Thus, LJ = J2 -J1 + 1 is the number of points in the warped cylindrical grid along the surface of the airfoil.

```
  Line
 Number         Source Statement

   1            SUBROUTINE STEP
   2            CALL RHS
   3            RESID = 0.
   4            DO 15 K = 1,4
   5            DO 15 K = 2,KHM
   6            DO 15 J = 2,JM
   7      15    RESID = RESID+S(J,K,N)**2
   8            RESID = SQRT(RESID/((JM-1)*(KHM-1)))
   9            DO 10 K = 2,KM
  10            CALL FILTRX(K,1,JMAX)
  11            CALL BTRI(2,JM)
  12            DO 11 J = 2,JM
  13            DO 11 N = 1,4
  14      11    S(J,K,N) = F(J,N)
  15      10    CONTINUE
  16            DO 20 J = 2,JM
  17            CALL FILTRY(J,1,KMAX)
  18            CALL BTRI(2,KM)
  19            DO 21 K = 2,KM
  20            DO 21 N = 1,4
  21      21    Q(J,K,N) = F(K,N)+Q(J,K,N)
  22      20    CONTINUE
```

Figure 5-14. FORTRAN Form of STEP

```
STEP
┌call    RHS
│        ┌call    BC
│        │        J ┌ loop 81
│        │          │                 scale ρu, ρv
│        │        K ├ loop 10
│        │          │                 upstream reflected (NF.EQ.O)
│        │        K ├ loop 20
│        │          │                 downstream reflected (NB.EQ.O)
│        │        J ├ loop 31
│        │          │                 VISCOUS on lower surface
│        │        J ├ loop 40
│        │          │                 permeable b.c. in wake   ↑ J = 2:(JTAIL1-1)
│        │          │                                          ↓ J = JM: (JTAIL2-1):(-1)
│        ├loop      └19
│        │        J ┌ loop 10
│   K    │          └ call FLUXVE generate 4-tuple of flux values at (J,K)
│        │          ┌ call DIFFER  central difference - column direction
│        │          │          J ┌loop 10
│        │          │            └
│        ├call      └ SMOOTX      fourth order smoothing - column direction
│        │        K ┌ loop 10
│        │          │          J ┌loop 42
│        │          └            └
│        ├loop 29
│        │        K ┌ loop 20
│   J    │          └ call FLUXVE generate 4-tuple of flux values at (J,K)
│        │          ┌ call DIFFER  central difference - row direction
│        │          │          K ┌loop 10
│        │          └            └
│        ├call SMOOTY             fourth order smoothing - row direction
│        │        J ┌ loop 50
│        │          │          K ┌loop 52
└        └          └            └
```

STEP cont'd in next figure

Figure 5-15. Simplified Loop Structure Representation of Right-Hand Side Calculations (Imposition of Boundary Conditions, Evaluation of Flux Variables, Central Differencing, and Fourth-Order Smoothing)

STEP (cont'd)

```
K [ loop 15                            form L2 residual
        J [ loop 15

K [ loop 10
  [ call FILTRX
        J [ loop 10
          [ call AMATRX                evaluate 4 x 4 matrix element (J,K)
        J [ loop 20                    form A,B,C,F matrices
  [ call BTRI
            call LUDEC                 LU decomposition
        J [ loop 13                    forward elimination solving (A, B, C)X=F
          [ call LUDEC
        J [ loop 21                    back substitution

J [ loop 20
  [ call FILTRY
        K [ loop 10
          [ call AMATRX                evaluate 4 x 4 matrix element (J,K)
        K [ loop 20                    form A,B,C,F matrices
          [ call VISMAT                algebraic turbulence (INVIS.GT.O)
                K [ loop 10
                K [ loop 20            form turbulence coefficients U, DU
                K [ loop 30            skew A,B,C,
  [ call BTRI
            call LUDEC                 LU decomposition
        K [ loop 13                    forward elimination
          [ call LUDEC
        K [ loop 21                    back substitution
```

Figure 5-16. Simplified Representation of Loop Structure for Assembling the Block Tridiagonal Matrices (A,B,C) and Solving (A,B,C) X = F for All Columns and All Rows

Line Number		Source Statement
1		DO 81 J = J1,J2
2		Q(J,1,2) = Q(J,1,2)*(1. - SCAL)
3	81	Q(J,1,3) = Q(J,1,3)*(1. - SCAL)

Figure 5-17. Scalar FORTRAN: Loop 81 BC Scales RHO*U and RHO*V

In figure 5-18, the corresponding vector code is displayed. Line 1 merely evaluates the vector length. Line 2 evaluates the scalar quantity: S = 1. - SCAL. Lines 3 and 4 are vector multiplies. Since the map unit supports a single result stream to memory, only one multiplier will be in use, and the duplex unit will check it. In table 5-2, the timing factors indicate that each pipeline generates one result per minor cycle. Hence, the peak rate is eight results per minor cycle. Startup and rundown will include the time it takes to prime (four minor cycles), and empty (four minor cycles) plus the time it takes to get data from memory to the functional units (12 minor cycles). For scalar instruction sequences, the estimate is based on STAR-100A instruction sequencing and overlap. The 8-cycle time required to execute line 2 includes the scalar code to generate the descriptors for line 3. Since the scalar unit will execute in parallel, it is assumed that descriptor setup for line 4 is covered by the vector multiply of line 3. Total execution cycles for loop 81 (table 5-2) are:

N64 = 50 + 2*LJ/8	64-bit mode
N32 = 50 + 2*LJ/16	32-bit mode

The count of floating operations is:

NOP = 4*LJ

Table 5-3 gives specimen times for several values of LJ.

Line Number	Source Statement
1	LJ = J2-J1+1
2	S = 1. - SCAL
3	Q(J1,1,2;LJ) = Q(J1,1,2;LJ)*S
4	Q(J1,1,3;LJ) = Q(J1,1,3;LJ)*S

Figure 5-18. Vector FORTRAN: Loop 81 BC

Table 5-2. NSS Timing: Loop 81 BC 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Scalar	2
2	Scalar	Scalar	8
3	1/8	12+1*8	20+LJ/8
4	1/8	12+1*8	20+LJ/8
		Total	50+LJ/4

Table 5-3. Specimen Times for NSS: Loop 81 BC 64-Bit Mode

LJ	Corresponding Value of LS	Operations (4*LJ)	Cycles (50+LJ/4)	Results Per Microsecond (10 ns Clock)
60	100	240	65	369
80	250	320	70	457
100	500	400	75	533

In figure 5-19, the scalar code for loop 20 appears (identical to that for loop 10). The loop next is reordered and K-pencils result. Vectorization will consist of periodic gathers on lines 3,4,5,6, and 7 of figure 5-20 and periodic scatters (copying a column into a row) on lines 8,9,10,12. On lines 4,5,6,7,8,9, and 10, a dollar sign separates two statements which execute in parallel, e.g., the divide on line 4 is hidden by the periodic gather on that line. DYNAMIC ARRAY's are descriptors which are assigned automatically to dynamic space when they are used. Lengths are determined from the source expression when they are first assigned and used as destinations. In the absence of other declarations, they are assigned to the register files in the pipelines. Table 5-4 illustrates that the result rate for the periodic gather/scatter is two per minor cycle.

The reader will observe that "unrolling" the N-loop results in code that is gather/scatter - driven with only line 11 employing a linked multiply/add which did not hide behind a gather or scatter. The count of floating operations for loop 10 or 20 is:

NOP= 15*KMAX

Total cycles for execution are:

N64 = 147 + 9*KMAX/2 + KMAX/8
N32 = 147 + 9*KMAX/2 + KMAX/16

The scalar (figure 5-21) and vector (figure 5-22) versions of loop 31 from BC do not use constructs which are different. It can be observed in figure 5-22, that each and every line of vector code (except 1 and 2 which are scalar) uses both multipliers. This portion of BC vectorizes with high utilization of the multiple functional units in the pipelines and, hence, resembles the block tridiagonal solution in utilization of hardware capability. The number of floating operations is:

NOP= 23*LJ

The number of cycles required is (table 5-5):

N64 = 380 + 11*LJ/8
N32 = 380 + 11*LJ/16

Specimen times are given in table 5-6.

```
 Line
Number          Source Statement

  1             J = JMAX
  2             DO 20 K = 1,KMAX
  3             DO 21 N = 1,3
  4      21     Q(J,K,N) = Q(J-1,K,N)*P(J-1,K)/P(J,K)
  5             PINFGI = 1./(GAMMA*GAMI*P(J,K))
  6      20     Q(J,K,4) + PINFGI + .5*(Q(J,K,2)**2+Q(J,K,3)**2)/Q(J,K,1)
```

Figure 5-19. Scalar FORTRAN: Loop 20 BC
(Loop 10 is identical except J=1)

```
 Line
Number          Source Statement

  0             DYNAMIC ARRAY T1,T2,T3,T4,T5,T6, PINFGI
  1             J = JMAX
  2             G = 1./(GAMMA*GAMI)
  3             T1 = P (J,1:KMAX)
  4             T2 = P (J-1,1:KMAX)  $  PINFGI = G/T1
  5             T2 = T2/T1  $  T3 = Q(J-1,1:KMAX,1)
  6             T3 = T3*T2  $  T4 = Q(J-1,1:KMAX,2)
  7             T4 = T4*T2  $  T5 = Q(J-1,1:KMAX,3)
  8             T5 = T5*T2  $  Q(J,1:KMAX,1) = T3
  9             T6 = T4**2+T5**2  $  Q(J,1:KMAX,2) = T4
 10             T6 = T6/T3  $  Q(J,1:KMAX,3) = T5
 11             T6 = PINFGI + .5*T6
 12             Q(J,1:KMAX,4) = T6
```

Figure 5-20. Vector FORTRAN: Loops 10 and 20 BC

Table 5-4. NSS Timing: Loop 20 (10) BC, 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Scalar	1
2	Scalar	Scalar	10
3	1/2	12	12+KMAX/2
4	1/2	12	12+KMAX/2
5	1/2	12	12+KMAX/2
6	1/2	12	12+KMAX/2
7	1/2	12	12+KMAX/2
8	1/2	12	12+KMAX/2
9	1/2	12	12+KMAX/2
10	1/2	12	12+KMAX/2
11	1/8	12+2*8	28+KMAX/8
12	1/2	12	12+KMAX/2
			147 + 9*KMAX/2+KMAX/8

KMAX	Corresponding Value of LS	Operations (15*KMAX)	Cycles (147+9*KMAX/2 +KMAX/8)	Results per Microsecond (10 ns Clock)
50	100	750	378	198
65	256	975	448	218
80	800	1200	517	232

Line Number	Source Statement
1	K = 1
2	DO 31 J = J1,J2
3	P2 = P(J,2)*GAMI*(Q(J,2,4)-.5*(Q(J,2,2)**2+Q(J,2,3)**2)/Q(J,2,1))
4	P1 = P2
5	DIVJ = 1./P(J,1)
6	RHO = Q(J,K,1)
7	Q(J,K,1) = Q(J,2,1)*P(J,2)*DIVJ
8	Q(J,K,2) = Q(J,K,2)*Q(J,K,1)/RHO
9	Q(J,K,3) = Q(J,K,3)*Q(J,K,1)/RHO
10	Q(J,K,4) = DIVJ*P1/GAMI+.5*(Q(J,K,2)**2+Q(J,K,3)**2)/Q(J,K,1)
11	31 CONTINUE

Figure 5-21. Scalar FORTRAN· Loop 31 BC
Viscous on Lower Surface

Line Number	Source Statement
0	DYNAMIC ARRAY P2,DIVJ,RHOR
1	K = 1
2	LJ = J2-J1+1
3	P2 = Q(J1,2,2;LJ)**2+Q(J1,2,3;LJ)**2
4	P2 = P2/Q(J1,2,1;LJ)
5	P2 = P(J1,2;LJ)*(Q(J1,2,4,LJ)-.5*P2)
6	DIVJ = 1./P(J1,1;LJ)
7	RHOR = 1./Q(J1,K,1;LJ)
8	Q(J1,K,1;LJ) = Q(J1,2,1;LJ)*P(J1,2;LJ)*DIVJ
9	Q(J1,K,2;LJ) = Q(J1,K,2;LJ)*Q(J1,K,1;LJ)*RHOR
10	Q(J1,K,3;LJ) = Q(J1,K,3;LJ)*Q(J1,K,1;LJ)*RHOR
11	RHOR = Q(J1,K,2,LJ)**2+Q(J1,K,3;LJ)**2
12	RHOR = RHOR/Q(J1,K,1;LJ)
13	Q(J1,K,4;LJ) = DIVJ*P2+.5*RHOR

Figure 5-22. Vector FORTRAN: Loop 31 BC

Table 5-5. NSS Timing: Loop 31 BC, 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Scalar	1
2	Scalar	Scalar	6
3	1/8	12+2*8	28+LJ/8
4	1/8	12+4*8	44+LJ/8
5	1/8	12+3*8	36+LJ/8
6	1/8	12+4*8	44+LJ/8
7	1/8	12+4*8	44+LJ/8
8	1/8	12+2*8	28+LJ/8
9	1/8	12+2*8	28+LJ/8
10	1/8	12+2*8	28+LJ/8
11	1/8	12+2*8	28+LJ/8
12	1/8	12+4*8	44+LJ/8
13	1/8	12+2*8	28+LJ/8
		Total	380+11*LJ/8

Table 5-6. Specimen Times for NSS: Loop 31 BC, 64-Bit Mode

LJ	Corresponding Value of LS	Operations (23*LJ)	Cycles $(380 + \frac{11}{8}LJ)$	Results Per Microsecond (10 ns Clock)
60	100	1380	463	298
80	250	1840	490	376
100	800	2300	516	446

Figures 5-23 and 5-24 represent the source code for scalar and vector versions of loop 40 of BC. This loop has two interesting facets; it evaluates a square root, and it is a split loop in which the first half of the loop runs forward (increment = +1), the second half runs backward (increment = -1), and the two join in the middle.

The square root operation and timing problem are resolved by expanding a rational approximation in-line and performing one corrective iteration. Thus, it can be seen in lines 8-17 how effective the pipeline structure is in doing mathematical functions. For simplicity's sake, range testing and choice of coefficients has not been embedded in the code, which turns out to be overlapped scalar code and would cause, at most, a delay of 10 cycles for issue of a series of load instructions by the scalar unit. (Range testing for vector mathematical functions is beyond the scope of this report.) The split loop causes the vectorizer to identify those backward operations which can run as well forward. To minimize periodic gather/scatter with period = -1, ordinary vectors were used up to the point where forward and backward sweeps were combined to produce the necessary interaction across the permeable boundary in the wake. In line 37, one sees a periodic scatter with period = -1 which accomplishes the reflection across the permeable boundary. In line 38, that reflected data is reused from memory as a vector stream.

```
NOP=  4*38*(JMAX-LJ)/2
   =  76*(JMAX-LJ)
```

From table 5-7, the number of cycles is:

```
N64 =  1880 + 61*LL/8
N32 =  1880 + 61*LL/16
where LL = (JMAX-LJ)/2
```

Table 5-8 provides specimen times for Loop 40 of BC.

```
 Line
Number          Source Statement

  1             DO 40 M = 1,4
  2             DO 40 J = 2,JS
  3             I = JMAX+1-J
  4             ST = ((X(J,2)-X(J,1))**2+(Y(J,2)-Y(J,1))**2)/
              1      ((X(J,3)-X(J,2))**2+(Y(J,3)-Y(J,2))**2)
  5             SB = ((X(I,2)-X(I,1))**2+(Y(I,2)-Y(I,1))**2)/
              1      ((X(I,3)-X(I,2))**2+(Y(I,3)-Y(I,2))**2)
  6             QSAV = .5*(Q(J,2,M)*P(J,2)
              1         -SQRT(ST)*(Q(J,3,M)*P(J,3)-Q(J,2,M)*P(J,2))
              2         +Q(I,2,M)*P(I,2)
              3         -SQRT(SB)*(Q(I,3,M)*P(I,3)-Q(I,2,M)*P(I,2)))
  7             Q(J,1,M) = QSAV/P(J,1)
  8     40      Q(I,1,M) = QSAV/P(I,1)
```

Figure 5-23. Scalar FORTRAN: Loop 40 BC
Permeable Boundary in Wake

```
  Line
 Number        Source Statement

   0           DYNAMIC ARRAY ST,SB,QSAV,T1,T2,T3,T4
   1           LL = JS-2+1
   2           IB  = JMAX+1-JS
   3           T1 =  X(2,2;LL)  $  ST = (T1-X(2,1;LL))**2
   4           T2 =  Y(2,2;LL)  $  ST = ST+(T2-Y(2,1;LL))**2
   5           T1 =  (X(2,3;LL)-T1)**2  $  T2 = (Y(2,3;LL)-T2)**2
   6           QSAV = T1+T2
   7           ST =  ST/QSAV
   8           T1 =  ST+C1           Lines 8-17 constitute an in-line
   9           T2 =  C2/T1           expansion of a rational approximation
                                     for SQRT (ST).
  10           T1 =  ST+C3
  11           T1 =  C4/T1
  12           ST =  C5-T1-T2
  13           T1 =  ST+C6
  14           T2 =  C7/T1
  15           T1 =  ST+C8
  16           T1 =  C9/T1  $  T3 = X(IB,2;LL)
  17           ST =  C10-T1-T2  $  T4 = Y(IB,2;LL)
  18           T1 =  (T3-X(IB,1;LL))**2  $  T2 = (T4-Y(IB,1;LL))**2
  19           T3 =  (X(IB,3;LL)-T3)**2  $  T4 = (Y(IB,3;LL)-T4)**2
  20           SB =  T1+T2  $  QSAV = T3+T4
  21           SB =  SB/QSAV
  22           T1 =  SB+C1           Lines 22-31 are an in-line
  23           T2 =  C2/T1           expansion for SB = SQRT (SB).
  24           T1 =  SB+C3
  25           T1 =  C4/T1
  26           SB =  C5-T1-T2
  27           T1 =  SB+C6
  28           T2 =  C7/T1  $  T3 = P(2,3;LL)
  29           T1 =  SB+C8  $  T4 = P(2,2;LL)
  30           T1 =  C9/T1  $  T5 = P(IB,3;LL)
  31           SB =  C10-T1-T2  $  T6 = P(IB,2;LL)
```

Continued

```
 Line
Number          Source Statement

  32            DO 40 M = 1,4
  33            T1 =  Q(2,3,M;LL)*T3-Q(2,2,M;LL)*T4
  34            T1 =  Q(2,2,M;LL)*P(2,2;LL)-ST*T1
  35            T2 =  Q(IB,3,M;LL)*T5-Q(IB,2,M;LL)*T6
  36            T2 =  Q(IB,2,M;LL)*P(IB, 2;LL)-SB*T2
  37            Q(JM:IB:-1,1,M) = T2
  38            QSAV = .5*(T1+Q(IB,1,M;LL))
  39            Q(2,1,M;LL) = QSAV/P(2,1;LL)
  40      40    Q(IB,1,M;LL) = QSAV/P(IB,1;LL)
```

Figure 5-24. Vector FORTRAN: Loop 40 BC

Table 5-7. NSS Timing: Loop 40 BC, 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Scalar	1
2	Scalar	Scalar	6 (incl. 4 packs)
3	1/8	12+2*8	28+LL/8
4	1/8	12+3*8	36+LL/8
5	1/8	12+2*8	28+LL/8
6	1/8	12+1*8	20+LL/8
7	1/8	12+4*8	44+LL/8
8	1/8	12+1*8	20+LL/8
9	1/8	12+4*8	44+LL/8
10	1/8	12+1*8	20+LL/8
11	1/8	12+4*8	44+LL/8
12	1/8	12+2*8	28+LL/8
13	1/8	12+1*8	20+LL/8
14	1/8	12+4*8	44+LL/8
15	1/8	12+1*8	20+LL/8
16	1/8	12+4*8	44+LL/8
17	1/8	12+2*8	28+LL/8
18	1/8	12+2*8	28+LL/8
19	1/8	12+2*8	28+LL/8
20	1/8	12+1*8	20+LL/8
21	1/8	12+4*8	44+LL/8
22	1/8	12+1*8	20+LL/8
23	1/8	12+4*8	44+LL/8
24	1/8	12+1*8	20+LL/8
25	1/8	12+4*8	44+LL/8
26	1/8	12+2*8	28+LL/8
27	1/8	12+1*8	20+LL/8
28	1/8	12+4*8	44+LL/8
29	1/8	12+1*8	20+LL/8
30	1/8	12+4*8	44+LL/8
31	1/8	12+2*8	28+LL/8

Table 5-7. Continued

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown		Cycles Required to Execute This Line
32	Scalar	Scalar		13
33	1/8	12+2*8		28+LL/8
34	1/8	12+2*8		28+LL/8
35	1/8	12+2*8		28+LL/8
36	1/8	12+2*8	4x	28+LL/8
37	1/8	12		12+LL/8
38	1/8	12+2*8		28+LL/8
39	1/8	12+4*8		44+LL/8
40	1/8	12+4*8		44+LL/8
		Total		1880+61*LL/8

Table 5-8. Specimen Times for NSS: Loop 40 BC, 64-Bit Mode

LL	Corresponding Value of LS	Operations (38*4*LL)	Cycles (1880+61*LL/8)	Results Per Microsecond (10 ns Clock)
30	100	4560	2109	216
40	256	6080	2185	278
50	500	7600	2262	336

In addition to the five component loops, a small number of overhead scalar calculations are required, as well as prologue/epilogue time. The totals for BC are:

Loop	NOP	N64	N32
81	4*LJ	50+2*LJ/8	50+2*LJ/16
10	15*KMAX	147+9*KMAX/2+KMAX/8	147+9*KMAX/2+KMAX/16
20	15*KMAX	147+9*KMAX/2+KMAX/8	147+9*KMAX/2+KMAX/16
31	23*LJ	380+11*LJ/8	380+11*LJ/16
40	76*(JMAX-LJ)	1880+61*(JMAX-LJ)/16	1880+61*(JMAX-LJ)/32
Overhead	11	84	84
Total	11+30*KMAX +76*JMAX -49*LJ	2688+74*KMAX/8 +61+JMAX/16 -35*LJ/16	2688+146*KMAX/16 +61*JMAX/32 -35*LJ/32

In the final analysis, the number of floating operations accomplished and the number of complete cycles consumed by the BC process will be brought into perspective. BC contributes very few floating-point results and consumes very few computation cycles relative to the aggregate for the STEP process. It does, however, require a great deal of skill and effort in vectorization. Some specimen times for BC (64-bit mode) are:

LJ	KMAX	JMAX	Total Operations	Total Cycles	Results Per Microsecond (10 ns Clock)
60	50	100	6,171	3400	182
80	65	128	7,769	3602	216
100	80	200	12,711	3972	320

Up to this point, only simple loops have been encountered. The reader should now refer to figure 5-15, where it will be observed that loop 19 in RHS will be the next focus of attention in the vectorization process. The following problems will be encountered.

- Loop 10 involves repetitive calls to a subprogram FLUXVE in a J-loop inside a K-loop. Potentially, the KMAX*JMAX subroutine prologue/epilogue executions could rival the multitude of subscript index-function evaluations in non-productive use of machine cycles.

- Loop 19 also involves a J-loop within subroutine DIFFER which is called within the K-loop.

The vectorization process will entail "flattening" the loop structure so that FLUXVE and DIFFER can be replaced by long-vector procedures which can best be characterized by the almost total absence of consumption of machine cycles for evaluation of index-functions and subroutine prologue/epilogue execution. Loop 19 forms the four flux components at each point in the mesh and differentiates them in the XI-(J-)direction. Then SMOOTX applies fourth-order smoothing to the results of the differencing. Most of the J-loops span from 2 to JMAX-1; hence, a "structural control vector" might be needed to suppress storage of unwanted results in the borders of receiving arrays. In fact, the J-vectors will be allowed to run from 1 to JMAX, and as a final step, either a control vector will be imposed on result storage or a border restoration loop will be executed.

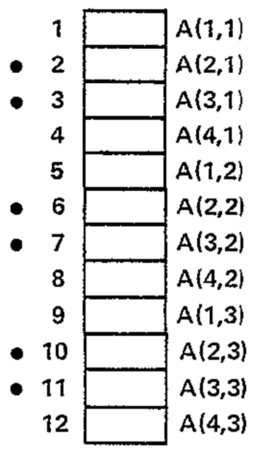
To provide familiarity with the control vector concept, consider the following example.

```
  DIMENSION A(4,3)
  DO 3 K=1,3
  DO 3 J=2,3
3 A(J,K) = 10.
```

In figure 5-25a, a doubly-periodic pattern of reference is specified. The loop structure could also be written as:

```
   DO 30 K=1,3
   DO 30 J=1,4
   IF  (J.GE.2 .AND. J.LE.3) A(J,K)=10.
30 CONTINUE
```

Suppose the logic of the one-branch IF were "in memory", i.e., a logical array of 1's and 0's were created which "controlled" the storage of values in the result array, (see figure 5-25b).



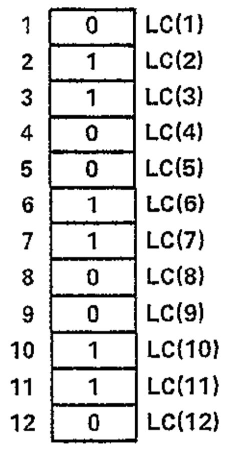


Figure 5-25a. The Nest

```
      DO 3  K = 1,3
      DO 3  J = 2,3
    3 A(J,K) = 10.
```

Accomplishes a Doubly-Periodic Pattern of Access to the Array

Figure 5-25b. Logical Control Vector which describes the Access Pattern for Figure 5-25a

Then, loop 30 could be expressed by the equivalent statement

```
IF (LC(1;12))A(1,1;12)=10
```

or its equivalent

```
IF (LC(2;10)) A(2,1;10)=10
```

where the second statement reflects that there is no reason to attempt storage of elements (1,1) and (4,3) since the control will be nonpermissive. The rationale for converting a loop nest into a single long-vector operation is elimination of startup and rundown times:

```
      DO 3 K=1,3
      DO 3 J=2,3
    3 A(J,K)=10
```

could be written as

```
      DO 31 K=1,3
   31 A(2,K;2)=10
```

or as

```
      IF (LC(2;10)) A(2,1;10)=10
```

But, loop 31 would have an execution time of

3*(20+CEILING(2/8))= 63 cycles

while the vector statement would require only

(20+CEILING(10/8)) = 22 cycles

where CEILING ( ) is the least integer greater than or equal to the argument. As the length of the inner loop increases, the return for implementing the long-vector structure is proportionately less; witness the following.

```
      DIMENSION A (100,50)
      DO 40 K=1,50
      DO 40 J=2,99
   40 A(J,K)=10
```

where

```
      DO 41 K=1,50
   41 A(2,K;98)=10
```

consumes 50*(20+13)=1650, while

```
   IF (LC(2;4998)) A(2,1;4998)=10
```

consumes (20+625) = 645.

An appropriate choice of algorithm will usually be a hybrid of short- and long-vector technique, i.e., choice of vector length short enough so that all vector temporaries will fit in pipeline scratchpads (level 1) but long enough (spanning several columns, if necessary) to ameliorate the impact of startup time. The reader should note the reduction of demand for streams from memory resulting from use of the pipeline scratchpads and the corresponding ability to employ the multiple functional units to advantage. He is also advised to note the frequency with which all four (the number available) read-ports of the pipeline scratchpads are in use.

Figure 5-26 displays the scalar FORTRAN for the loop 19 nest and the invocation of SMOOTX extracted from RHS. Additional complexity is introduced by presence of loops 12 and 14 on N=1:4 and hidden loops in FLUXVE and SMOOTX which operate over the 4-tuple of flow variables. Figure 5-27 shows sliced vector code which evaluates and differentiates the flux vector NJ columns at a time. For the sake of correspondence to

```
 Line
Number         Source Statement

  1            DO 19  K = 2,KM
  2            DO 10  J = 1,JMAX
  3            R1  =  XY(J,K,1)
  4            R2  =  XY(J,K,2)
  5            R3  =  XIT(J,K)
  6            CALL FLUXVE(J,K,R1,R2,R3,0.)
  7            DO 12  N = 1,4
  8      12    F(J,N)  =  FV(N)
  9      10    CONTINUE
 10            RO  =  -.5*DT
 11            CALL DIFFER(F,FV,FD,RO,2,JM)
 12            DO 14  J = 2,JM
 13            DO 14  N = 1,4
 14      14    S(J,K,N)  =  F(J,N)
 15      19    CONTINUE
 16            CALL SMOOTX
```

Figure 5-26. Scalar FORTRAN: Loop 19 RHS (form flux vector, difference in column direction, and apply fourth-order smoothing in column direction)

```
Source Statement

      DYNAMIC ARRAY FV(4)
      DESCRIPTOR R1,R2,R3
      ASSIGN FV(1:4) TO LEVEL 1 WITH LENGTH LS
      DO 19  K = 2,KMM,NJ
      ASSIGN R1,XY(1,K,1;LS)
      ASSIGN R2,XY(1,K,2;LS)
      ASSIGN R3,XIT(1,K;LS)
      CALL VFLUXVE(K,R1,R2,R3,0.)
      RO = -.5*DT
      DO 19 N = 1,4
      CALL VDIFFER (S(1,K,N;LS),FV(N),LC,RO)
19    CONTINUE
      CALL VSMOOTX
```

NOTES

1. LC is a level-2 bit-array which has been preset to a repetitive pattern of bits which inhibits result storage in rows 1 and JMAX of the S-array.

2. VFLUXVE and VDIFFER are assumed to be expanded in-line symbolically by the compiler.

3. RHS is sliced NJ columns at a time, where length of slice LS = NJ*JMAX and number of slices NS*NJ = KMAX-2.

Figure 5-27. Vector FORTRAN: Loop 19 RHS

the original source code, the fourth-order smoothing of SMOOTX has not been folded into the slice loop as would ordinarily be done. Lines numbered 5, 6, and 7 of figure 5-27 are examples of an extended assignment statement which replaces the value (address and length) of the descriptor on the left with the value of the descriptor appearing on the right. The reader should be aware that execution of

```
R1 = XY(1,K,1;LS)
```

specifies the movement of LS values from the XY-array to the area described by R1 while execution of

```
ASSIGN R1, XY(1,K,1;LS)
```

causes R1 to describe the LS values without moving them. In line 3 of figure 5-27, another variant of the ASSIGN statement will be observed. Execution of ASSIGN FV(1:4) to LEVEL 1 with length LS results in allocation of four areas of length LS in the pipeline scratchpad (level 1). The resulting array of descriptors is then passed to VFLUXVE, which generates its results in those areas of the scratchpad. In DO 19 N=1,4, the arrays described by FV(N), N=1:4 are differentiated into the appropriate subarray of S. Storage of results into S will be controlled with a bit vector, LC, which is permissive (corresponding value is 1) for those values corresponding to J=2:JM and nonpermissive (value is 0) for J=1 and J=JMAX.

Tables 5-9 and 5-10 summarize the performance of the components of loop 19.

```
NOP  =  (KMAX-3)*(28*JMAX-4)+(KMAX-2)(63*JMAX-240)

N64  =  863+NS*(388+13*LS/8)
           +31*JMAX*(KMAX-2)/8

N32  =  863+NS*(388+13*LS/16)
           +31*JMAX*(KMAX-2)/16
```

In figure 5-28, the scalar FORTRAN for FLUXVE can be seen. The vectorization given in figure 5-29 was done on the premise that the vector temporaries would all reside in the pipeline scratchpad; hence, a high degree of overlapped calculation would be possible. Several observations are in order.

- Usually, descriptor setup will be covered by a preceding vector instruction. This assumes that the FORTRAN compiler acquires the scalars and descriptions that it will need in infrequent gulps. That is facilitated by the large, general register set (256 64-bit registers), the broadband load/store unit (one per 10 ns cycle), and the swap unit (two per 10 ns cycle).

Table 5-9. NSS Timing: Loop 19 RHS, 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown		Cycles Required to Execute This Line
3	Scalar	Scalar		8
4	Scalar	Scalar		1
5	Scalar	Scalar		3
6	Scalar	Scalar		3
7	Scalar	Scalar		4
8	9/8	276	Executed NS Times	276+9*LS/8
9	Scalar	Covered	Executed NS Times	0
10	Scalar	Covered	Executed NS Times	0
11	1/8	12+2*8	Executed NS Times	4(28+LS/8)
12	Scalar	Covered		0
13	31/8	862		844+31*JMAX*(KMAX-2)/8
	Total	863+NS(388+13*LS/8)+31*JMAX*(KMAX-2)/8		

Table 5-10. Specimen Times for NSS: Loop 19 RHS, 64-Bit Mode

JMAX	KMAX	NS	LS	Operations (KMAX-3)* (28 JMAX-4) +(KMAX-2)* (63 JMAX-240)	Cycles 871+NS* (388+13LS/8) +31*JMAX* (KMAX-2)/8	Results Per Microsecond (10 ns Clock)
100	50	16	300	422,292	33,479	1261
128	64	31	256	703,468	56,547	1244
200	80	13	1200	1,394,972	91,715	1521

Note: NS is the number of slices of length LS processed.

```
 Line
Number     Source Statement

           SUBROUTINE FLUXVE(J,K,R1,R2,R3,R4)
           – COMMON AND LEVEL DECLARATIONS –
   1       RR  =  1./Q(J,K,1)
   2       U  =  Q(J,K,2)*RR
   3       V  =  Q(J,K,3)*RR
   4       RO =  R3+R4
   5       QS  =  RO-R1*U+R2*V
   6       PP  =  GAMI*(Q(J,K,4)-.5*Q(J,K,1)*(U**2+V**2))
   7       FV(1)  =  Q(J,K,1)*QS
   8       FV(2)  =  Q(J,K,2)*QS+R1*PP
   9       FV(3)  =  Q(J,K,3)*QS+R2*PP
  10       FV(4)  =  QS*(Q(J,K,4)+PP)-PP*RO
           RETURN
           END
```

Figure 5-28. Scalar FORTRAN: FLUXVE

```
 Line
Number     Source Statement

           SUBROUTINE VFLUXVE(R1,R2,R3,R4)
           COMMON/DESCR/FV(4),Q1,Q2,Q3,Q4,R1,R2,R3,R4
           DYNAMIC ARRAY RR,U,V,RO,T1,QS,T2,PP
           ASSIGN Q1,Q(1,K,1;LS)
           ASSIGN Q2,Q(1,K,2;LS)  $  ASSIGN Q3,Q(1,K,3;LS)
           ASSIGN Q4,Q(1,K,4;LS)
   1       RR  =  1./Q1
   2       U  =  Q2*RR  $  V = Q3*RR
   3       RO = R3+R4  $  T1 = U*U+V*V
   4       QS  =  RO+R1*U+R2*V
   5       T2  =  Q4-.5*Q1*T1
   6       PP  =  GAMI*T2  $  FV(1) = Q1*QS
   7       FV(2)  =  Q2*QS+R1*PP
   8       FV(3)  =  Q3*QS+R2*PP
   9       FV(4)  =  QS*(Q4+PP)-PP*RO
           RETURN
           END
```

Figure 5-29. Vector FORTRAN: VFLUXVE

- The statement in line 3 uses two memory streams (R3 and R4) and two buffer read ports (U and V). It utilizes both multipliers, both adders, and both buffer write ports.

- The statement in line 4 employs two memory streams(R1,R2), three buffer read ports (RO,U,V), two multipliers, two adders, and one buffer write port.

The overall rate presented in table 5-11 shows that only 9/8 cycle is required per result. Thus, the procedure has an asymptotic result rate of 26/(9/8) or 2311 Mflops in 64-bit mode. In 32-bit mode, the asymptotic rate would be 4622 Mflops. The length at which one-half asymptotic rate would be measured is also significant. It is 254 in 64-bit mode and 507 in 32-bit mode. Both of these lengths can be readily accommodated for the allocation of all the required vector temporaries to the buffers. The overall operation count is:

```
NOP = 26*LS
```

The cycle counts are:

```
N64 = 276+9*LS/8
N32 = 276+9*LS/16
```

Specimen times are shown in table 5-12.

The self-imposed restriction that a whole number of vertical slices be accomplished in one horizontal pass over a loop structure is arbitrary. The following table suggests that dimensions be chosen more carefully or the restriction be removed. LS has been chosen such that LS=NJ*JMAX and NS*NJ=KMAX-2 were integrally satisfied. The factors of KMAX-2 for typical dimensions are:

KMAX	KMAX-2	Factors
50	48	2,3,4,6,8,12,16,24
64	62	2,31
80	78	2,3,6,13,26,39

Table 5-11. NSS Timing: VFLUXVE, 64-Bit Mode

Line Number	Cycler Per Result Generated	Cycles Required for Startup and Rundown	Cycles Required to Execute This Line
1	1/8	12+4*8	44+LS/8
2	1/8	12+1*8	20+LS/8
3	1/8	12+2*8	28+LS/8
4	1/8	12+3*8	36+LS/8
5	1/8	12+3*8	36+LS/8
6	1/8	12+1*8	20+LS/8
7	1/8	12+2*8	28+LS/8
8	1/8	12+2*8	28+LS/8
9	1/8	12+3*8	36+LS/8
		Total	276+9*LS/8

Table 5-12. Specimen Times for NSS. VFLUXVE, 64-Bit Mode

LS	Operations (26*LS)	Cycles (276+9*LS/81)	Results Per Microsecond (10 ns Clock)
100	2,600	389	668
256	6,656	564	1180
800	20,800	1176	1769

Within the constraints, one choice for each of three pairs of dimensions is tabulated as follows:

JMAX	KMAX	NJ	NS	LS	Operations 26*LS	Cycles 276+9LS/8	Rate
100	50	8	6	800	20,800	1176	1769
128	64	2	31	256	6,656	564	1180
200	80	6	13	1200	31,200	1626	1919

In loop 29 of RHS, a long vector version of FLUXVE is presented for comparison with the sliced version presented here. The reader should contrast the two methodologies after he has studied both.

Differencing is a trivial operation in vector form. Unfortunately, in scalar form great effort is usually expended in avoiding use of newly generated values (figure 5-30). The corresponding vector code is merely (figure 5-31)

```
S = (FV(3;) - FV(1;))*RO
```

where S and FV are descriptors. Invariably, a full array is generated by differencing another array. Thus, a controlled long-vector approach should be used.

In the context of loop 19 of RHS, four differences are called for:

```
      L = LS-2
      DO 99 N=1,4
   99 S(2,2,N;L)=RO*(FV(N)(3;L)-FV(N)(1;L))
```

so

```
      NOP = 4(JMAX-2)(KMAX-2)
```

and

```
      N64 = 4*(28+(LS-2)/8)
      N32 = 4*(28+(LS-2)/16)
```

Timing and rates for DIFFER are presented in tables 5-13 and 5-14.

```
 Line
Number           Source Statement

                 SUBROUTINE DIFFER(F,FV,FD,RO,I1,I2)
   1             DO  12  N = 1,4
   2       12    FD(N)  =  F(I1-1,N)
   3             DO  10  I = I1,I2
   4             IP  =  I+1
   5             DO  10  N = 1,4
   6             FV(N) = (F(IP,N)-FD(N))*RO
   7             FD(N) = F(I,N)
   8       10    F(I,N) = FV(N)
```

Figure 5-30. Scalar FORTRAN: DIFFER

```
 Line
Number           Source Statement

                 SUBROUTINE VDIFFER (S, FV, LC, RO)
                 BIT LC
                 DESCRIPTOR S, FV, LC
   1             IF (LC S = (FV(3;)-FV(1;))*RO
                 RETURN
                 END
```

Figure 5-31. Vector FORTRAN: VDIFFER

Table 5-13. NSS Timing: VDIFFER, 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown	Cycles Required to Execute This Line
1	1/8	12+2*8	4(28+(LS-2)/8)

Table 5-14. Specimen Times for NSS: VDIFFER, 64-Bit Mode

LS	Operations (2*LS)	Cycles 4(28+(LS-2)/8)	Results Per Microsecond (10 ns Clock)
100	200	161	124
256	512	239	214
800	1600	511	313

SMOOTX (figure 5-32) is also a trivial exercise in vectorization. Only one level of code is present; therefore, there are no hidden DO loops, and the only variable appearing both right and left of the replacement symbol is S(J,K,N). Hence, the calculation is not recursive. The only trick in the vector coding of figures 5-33 is to recognize that one full-length product of Z and Q will produce all such products once Z has been promoted from one to two dimensions.

The number of operations is:

$$\begin{aligned} \mathrm{NOP} &= (\mathrm{KMAX}-2)(3*\mathrm{JMAX}+4*(\mathrm{JMAX}-4)(15)) \\ &= (\mathrm{KMAX}-2)(63*\mathrm{JMAX}-240) \end{aligned}$$

The execution cycle counts may be gleaned from tables 5-15 and 5-16.

$$\begin{aligned} \mathrm{N64} &= 862+31*\mathrm{JMAX}*(\mathrm{KMAX}-2)/8 \\ \mathrm{N32} &= 862+31*\mathrm{JMAX}*(\mathrm{KMAX}-2)/16 \end{aligned}$$

Then, the asymptotic rate, is 3252.

```
 Line
Number           Source Statement

                 SUBROUTINE SMOOTX
  1              DO  40  K = 2,KM
  2              DO  41  J = 1,JMAX
  3      41      Z(J) = XY(J,K,1)*XY(J,K,4)-XY(J,K,2)*XY(J,K,3)
  4              DO  42  N = 1,4
  5              DO  42  J = 3,JMM
  6      42      S(J,K,N) = S(J,K,N)-SMU*(Z(J-2)*Q(J-2,K,N)+Z(J+2)*Q(J+2,K,N)
  7            1 +6*Z(J)*Q(J,K,N)-4.*Z(J+1)*Q(J+1,K,N)-4.*Z(J-1)*Q(J-1,K,N))/Z(J)
  8      40      CONTINUE
```

Figure 5-32. Scalar FORTRAN: SMOOTX

```
 Line
Number           Source Statement

                 SUBROUTINE VSMOOTX
                 DYNAMIC ARRAY  Z,ZQ,T1
  1              L = JMAX*(KMAX-2)
  2              Z = XY(1,2,1;L)*XY(1,2,4;L)
  3              T1 = XY(1,2,2;L)*XY(1,2,3;L)
  4              Z = Z-T1
  5              DO  42  N = 1,4
  6              ZQ = Z*Q(1,1,N;L)
  7              T1 = ZQ(1;)-4*ZQ(2,)
  8              T1 = T1+6*ZQ(3;)
  9              T1 = T1-4*ZQ(4;)
 10              T1 = T1+ZQ(5;)
 11              T1 = T1/Z
 12      42      S(1,1,N;L) = S(1,1,N;L)-SMU*T1
                 RETURN
                 END
```

Figure 5-33. Vector FORTRAN: VSMOOTX

Table 5-15. NSS Timing: VSMOOTX, 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Scalar	10
2	1/8	12+1*8	20+L/8
3	1/8	12+1*8	20+L/8
4	1/8	12+1*8	20+L/8
5	Scalar	Scalar	8
6	1/8	12+1*8	20+L/8
7	1/8	12+2*8	28+L/8
8	1/8	12+2*8	28+L/8
9	1/8	12+2*8	4x- 28+L/8
10	1/8	12+1*8	20+L/8
11	1/8	12+4*8	44+L/8
12	1/8	12+2*8	28+L/8
		Total	862+31*L/8

Table 5-16. Specimen Times for NSS: VSMOOTX, 64-Bit Mode

JMAX	KMAX	Operations (KMAX-2)* (63JMAX-240)	L JMAX(KMAX-2)	Cycles 862+31L/8	Results Per Microsecond (10 ns Clock)
100	50	290,880	4,800	19,462	1495
128	64	485,088	7,936	31,614	1534
200	80	964,080	15,600	61,312	1572

Figure 5-27 embodies the vectorization of loop 29. The reader should note that the long-vector nature of this implementation results in each statement being at most a pseudo-dyad in complexity (that is, only two memory operand streams). In figure 5-35, the reader can observe the long-vector VSMOOTY where he might observe that lines 10 and 16 could be coalesced into a single vector subtract with a little effort. The operation count for loop 29 is:

NOP = (JMAX-2) (99*KMAX-224)

The execution cycle counts are:

N64 = 1991 + JMAX*(63*KMAX-41)/8
N32 = 1991 + JMAX*(63*KMAX-41)/16

```
            Source Statement

            SUBROUTINE SMOOTY
            DO 50 J = 2,JM
            DO 51 K = 1,KMAX
   51       Z(K) = XY(J,K,1)*XY(J,K,4)-XY(J,K,2)*XY(J,K,3)
            DO 52 N = 1,4
            K = KM
            S(J,K,N) = S(J,K,N)-SMU*.5*(-Z(K-1)*Q(J,K-1,N)
          1              +2*Z(K)*Q(J,K,N)-Z(K+1)*Q(J,K+1,N))/Z(K)
            DO 52 K = 3,KMM
   52       S(J,K,N) = S(J,K,N)-SMU*(Z(K+2)*Q(J,K+2,N)+Z(K-2)*Q(J,K-2,N)
          1 +6*Z(K)*Q(J,K,N)-4*Z(K-1)*Q(J,K-1,N)-4*Z(K+1)**Q(J,K+1,N))/Z(K)
   50       CONTINUE
```

Figure 5-34. Scalar FORTRAN: SMOOTY

```
 Line
Number          Source Statement

                SUBROUTINE VSMOOTY
                DYNAMIC ARRAY Z,ZQ,T1,T2
  1             L = JMAX*KMAX
  2             Z = XY(1,1,1;L)*XY(1,1,4;L)
  3             ZQ = XY(1,1,2;L)*XY(1,1,3;L)
  4             Z = X-ZQ  $  SMUH = .5*SMU
  5             DO  52  N = 1,4
  6             ZQ = Z*Q(1,1,N;L) $ L = JMAX*(KMAX-4)-2
  7             T1 = 2*ZQ(1,KM;JMM)-ZQ(1, KM-1;2*JMM)
  8             T1 = SMUH*(T1-ZQ(1,KM+1;JMM))
  9             T1 = T1/Z(1,KM;JMM)
 10             S(1,KM,N;JMM) = S(1, KM, N;JMM)-T1
 11             T1 = ZQ(1, 5;L)+ZQ(1, 1;L)
 12             T2 = 4*(ZQ(1,2;L)+ZQ(1, 4;L))
 13             T1 = T1+T2
 14             T1 = T1+6*ZQ(1,3;L)
 15             T1 = T1/Z(1,3;L)
 16      52     S(2,3,N;L) = S(2,3,N;L)-SMU*T1
                RETURN
                END
```

Figure 5-35. Vector FORTRAN· VSMOOTY

Table 5-17. NSS Timing: VSMOOTY, 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown		Cycles Required to Execute This Line
1	Scalar	Scalar		11
2	1/8	12+1*8		20+JMAX*KMAX/8
3	1/8	12+1*8		20+JMAX*KMAX/8
4	1/8	12+1*8		20+JMAX*KMAX/8
5	Scalar	Scalar		7
6	1/8	12+1*8	Executed Four Times	20+JMAX*KMAX/8
7	1/8	12+2*8		28+JMAX/8
8	1/8	12+2*8		28+JMAX/8
9	1/8	12+4*8		44+JMAX/8
10	1/8	12+1*8		20+JMAX*(KMAX-4)/8
11	1/8	12+1*8		20+JMAX*(KMAX-4)/8
12	1/8	12+2*8		28+JMAX*(KMAX-4)/8
13	1/8	12+1*8		20+JMAX*(KMAX-4)/8
14	1/8	12+2*8		28+JMAX*(KMAX-4)/8
15	1/8	12+4*8		44+JMAX*(KMAX-4)/8
16	1/8	12+2*8		28+JMAX*(KMAX-4)/8
		Total		1310+JMAX*(35*KMAX-25)/8

Table 5-18. Specimen Times for NSS: VSMOOTY, 64-Bit Mode

JMAX	KMAX	Operations (JMAX-2)* (63KMAX-196)	Cycles 1310+JMAX* (35KMAX-25)/8	Results Per Microsecond (10 ns Clock)
100	50	289,492	22,873	1266
128	64	483,336	36,750	1315
200	80	959,112	70,685	1357

```
 Line
Number         Source Statement

  1            DO 29 J = 2,JM
  2            DO 20 K = 1,KMAX
  3            R1 = XY(J,K,3)
  4            R2 = XY(J,K,4)
  5            R4 = ETT(J,K)
  6            CALL FLUXVE(J,K,R1,R2,0.,R4)
  7            DO 22 N = 1,4
  8      22    F(K,N) = FV(N)
  9      20    CONTINUE
 10            RO = -.5*DT
 11            CALL DIFFER(F,FV,FD,RO,2,KM)
 12            DO 24 K = 2,KM
 13            DO 24 N = 1,4
 14      24    S(J,K,N) = F(K,N)+S(J,K,N)
 15      29    CONTINUE
 16            CALL SMOOTY
```

Figure 5-36. Scalar FORTRAN: Loop 29

```
 Line
Number        Source Statement

  0           DYNAMIC ARRAY (FV(6)
              L1 has the value JMAX*KMAX
              Long vector FLUXVE expanded in-line.
  1           FV(1) = Q(1,1,2;L1)/Q(1,1,1;L1)
  2           FV(2) = Q(1,1,3;L1)/Q(1,1,1;L1)
  3           FV(3) = XY(1,1,3;L1)*FV(1)
  4           FV(4) = XY(1,1,4;L1)*FV(2)
  5           FV(3) = FV(3)+FV(4)
  6           FV(3) = ETT(1,1;L1)+FV(3)
  7           FV(4) = FV(1)**2+FV(1)**2
  8           FV(4) = .5*Q(1,1,1;L1)*FV(4)
  9           FV(4) = GAMI*(Q(1,1,4;L1)-FV(4))
 10           FV(1) = Q(1,1,1;L1)*FV(3)
 11           FV(2) = XY(1,1,3;L1)*FV(4)
 12           FV(5) = Q(1,1,2)*FV(3)
 13           FV(2) = FV(5) + FV(2)
 14           FV(5) = XY(1,1,4,L1)*FV(4)
 15           FV(6) = Q(1,1,4;L1)+FV(4)
 16           FV(6) = FV(3)*FV(6)
 17           FV(3) = Q(1,1,3)*FV(3)
 18           FV(3) = FV(3)+FV(5)
 19           FV(5) = FV(4)*ETT(1,1;L1)
 20           FV(4) = FV(6)-FV(5)
 21           RO = -.5*DT  $  L2 = JMAX*(KMAX-2)
 22           DO 24 N = 1,4
 23           FV(5)(1,2;L2) = (FV(N)(1,3;)-FV(N))*RO
 24     24    S(1,2,N;L2) = FV(5)(1,2;)+S(1,2,N;L2)
 25           CALL VSMOOTY
```

Figure 5-37. Vector FORTRAN: Loop 29

Table 5-19. NSS Timing: Loop 29, 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles for Startup and Rundown		Cycles Required to Execute This Line
1	1/8	12+4*8		44+JMAX*KMAX/8
2	1/8	12+4*8		44+JMAX*KMAX/8
3	1/8	12+1*8		20+JMAX*KMAX/8
4	1/8	12+1*8		20+JMAX*KMAX/8
5	1/8	12+1*8		20+JMAX*KMAX/8
6	1/8	12+1*8		20+JMAX*KMAX/8
7	1/8	12+2*8		28+JMAX*KMAX/8
8	1/8	12+2*8		28+JMAX*KMAX/8
9	1/8	12+2*8		28+JMAX*KMAX/8
10	1/8	12+1*8		20+JMAX*KMAX/8
11	1/8	12+1*8		20+JMAX*KMAX/8
12	1/8	12+1*8		20+JMAX*KMAX/8
13	1/8	12+1*8		20+JMAX*KMAX/8
14	1/8	12+1*8		20+JMAX*KMAX/8
15	1/8	12+1*8		20+JMAX*KMAX/8
16	1/8	12+1*8		20+JMAX*KMAX/8
17	1/8	12+1*8		20+JMAX*KMAX/8
18	1/8	12+1*8		20+JMAX*KMAX/8
19	1/8	12+1*8		20+JMAX*KMAX/8
20	1/8	12+1*8		20+JMAX*KMAX/8
21	Scalar	Scalar		1
22	Scalar	Scalar		16
23	1/8	12+2*8	Executed 4 times	28+JMAX(KMAX-2)/8
24	1/8	12+1*8		20+JMAX(KMAX-2)/8
25	JMAX(35KMAX-25)/8	1310		1310+JMAX(35KMAX-25)/8
		Total		1991+JMAX(63KMAX-41)/8

Table 5-20. Specimen Times for NSS: Loop 29, 64-Bit Mode

JMAX	KMAX	Operations (JMAX-2)* (99*KMAX-224)	Cycles 1991+JMAX* (63*KMAX-41)/18	Results Per Microsecond (10 ns Clock)
100	50	463,148	40,854	1134
128	64	770,112	65,847	1170
200	80	1,523,808	126,966	1200

Table 5-21. Total Right-Hand Side Timing: 64-Bit Mode

Grand totals then for the entire RHS (including BC) calculations are:

	NOP	N64	N32
BC	11 + 30*KMAX + 76*JMAX - 49*LJ	2688 + 74*KMAX/8 + 61*JMAX/16 - 35*LJ/16	2688 +146*KMAX/16 + 61*JMAX/32 - 35*LJ/32
Loop 19 RHS	(KMAX-2)* (89*JMAX-230)	863 + NS * (388 + 13*LS/8) + 31*JMAX*(KMAX-2)/8	863 + NS * (388 + 13*LS/16 + 31*JMAX*(KMAX-2)/16
Loop 29 RHS	(JMAX-2)* (99*KMAX-224)	1991 + JMAX*(63*KMAX-41)/8	1991 + JMAX*(63*KMAX-41)/16

JMAX	KMAX	LS	NS	LJ	NOP	N64	R64	N32	R32
100	50	800	6	30	886,949	73,911	1200	41,116	2157
128	64	256	31	40	1,471,855	161,986	909	72,107	2041
200	80	1200	13	50	2,909,429	222,755	1306	117,031	2486

Implicit Code: Left-Hand Side (LHS)

The vectorization techniques and Vector FORTRAN constructs introduced in the discussion of the Right-Hand Side (RHS) computations appear in similar form in the vectorization of the Left-Hand Side calculations. Thus, the following commentary will be limited to highlighting those additional computer features or techniques not yet covered. To begin with, the elementary loop 15 performs a simple vector sum of the square of the elements in the S-array. Figure 5-39 gives the possible vector form of this loop. The descriptor SD is assigned the full vector length of the total S-matrix. At the same time, the compiler is instructed in line 3 of figure 5-39 to associate the control vector LC with this same descriptor. This means that in all subsequent uses of the descriptor SD the control vector LC (which inhibits the use of elements on the boundaries) will be employed on all results. The FORTRAN extension, VSUM, causes the generation of an NSS instruction which performs a vector sum in the pipelines at the rate of eight elements per minor cycle, thence the generation of another NSS instruction which collapses the eight partial sums being carried in each vector unit (a partial sum will reside in each segment of each adder unit within the vector unit during the first phase of this instruction) into a single partial sum for each vector unit. A further sequence is then generated by the compiler to swap these eight partial sums (one per vector unit) directly to the scalar register file where a series of scalar instructions form the final residue. Tables 5-22 and 5-23 present timing information.

Figures 5-40 and 5-41 give the scalar form of the main loops in the Left-Hand Side code. The first, loop 10, invokes the K-direction sweep, while loop 20 invokes the J-direction sweep. Note that the K-direction sweep accesses data in a noncontiguous manner, requiring the use of the gather feature of the NSS swap unit, to collect the spread-out elements (along the rows of the matrix) into vectors which can be processed at the higher peak rate of the vector units. Both the loop 10 and loop 20 constructs disappear when the contained routines FILTRX, FILTRY, BTRI are fully vectorized.

The scalar version of FILTRX is given in figure 5-42. It is this routine which forms the tridiagonal matrix, where each element is itself a block of 4x4 elements. This is accomplished by generating the matrix A of 4x4 blocks to form the top band (above the diagonal) of the tridiagonal array, then the diagonal strip (matrix B) is computed and finally the strip below the diagonal (matrix C) is formed. At the same time the corresponding F (forcing function) matrix is extracted. The resulting A, B, C and F matrices are then ready for the 'tridiagonal solver' — BTRI.

```
 Line
Number        Source Statement

  1           DO 15 N = 1,4
  2           DO 15 K = 2,KHM
  3           DO 15 J = 2,JM
  4     15    RESID = RESID+S(J,K,N)**2

  Note:     KHM = (KMAX-1)/2
            JM = JMAX-1
```

Figure 5-38. Scalar FORTRAN: Loop 15, Left-Hand Side

```
 Line
Number        Source Statement

              DESCRIPTOR SD, LC
  1           LV = JMAX*((KMAX-1)/2)
  2           DO 15 N = 1,4
  3           ASSIGN SD,S(1,1,N;LV).CTRL.LC
  4     15    RESID = .VSUM.SD**2

  Note:     LC is a predefined bit vector with JMAX-1 bits set to ones.
```

Figure 5-39. Vector FORTRAN: Loop 15, Left-Hand Side

Table 5-22. NSS Timing: Loop 15, Left-Hand Side, 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Scalar	6
2	Scalar	Scalar	1
3	Scalar	Scalar	3
4a partial sums	1/8	12+2*8	28+LJ/8
4b final sum	Scalar	Scalar	12
		Total	50+LJ/8

Table 5-23. Specimen Times for NSS: Loop 15, Left-Hand Side, 64-Bit Mode

LV	Operations 2*LV	Cycles 50+LJ/8	Results Per Microsecond (10 ns Clock)
2400	4,800	350	1371
4064	8,128	558	1456
7860	15,720	1032	1523

```
 Line
Number          Source Statement

   1            DO 10 K = 2,KM
   2            CALL FILTRX(K,1,JMAX)
   3            CALL BTRI(2,JM)
   4            DO 11 J = 2,JM
   5            DO 11 N = 1,4
   6      11    S(J,K,N) = F(J,N)
   7      10    CONTINUE
```

Figure 5-40. Scalar FORTRAN: Loop 10, LHS

```
 Line
Number          Source Statement

   1            DO 20 J = 2,JM
   2            CALL FILTRY(J,1,KMAX)
   3            CALL BTRI(2,KM)
   4            DO 21 K = 2,KM
   5            DO 21 N = 1,4
   6      21    Q(J,K,N) = F(K,N)+Q(J,K,N)
   7      20    CONTINUE
```

Figure 5-41. Scalar FORTRAN: Loop 20, LHS

Figure 5-43 gives the vectorized version of FILTRX. The use of DYNAMIC ARRAY should be familiar by now, however, note that A, B, C, DD and F are all arrays of descriptors, each 4x4 elements long. That means that the compiler will generate and manage sixteen descriptors for each of the matrices A, B, C, DD and F. Note that the use of the variable name A in this code refers in actual fact to the array of descriptors called A. Thus in line 15, A(N,N) refers to the N,N descriptor, which in fact will describe a vector (or slice or pencil) of elements of the mathematical entity "A-matrix".

To further confuse the issue however, the artifice of a vector descriptor modifier has been employed following a vector descriptor. The expression A(N,N)(1,2;KMAX) then means take the N,N descriptor from the A-array of descriptors, compute a new starting address offset from that initial address by the indices 1,2, and assign the vector length KMAX to the resulting modified descriptor. Note that this action does not change the N,N element in the A-array, but only the temporary one formed at this point (line 15) in the execution of the code. This scheme allows definition of a method for describing subarrays of dynamically assigned and modified vectors at run time.

Understanding this construct one can then turn attention to line 8 of figure 5-43 where the last complex form used in this code arises.

This dynamically assigns to the descriptor set DD a set of pointers to subarrays within the mathematical array D. Note that in this case the descriptor set related to D is called DD while the actual array is called D (this usage appears in lines 15 and 17 of figure 5-43). The notation DD(1:4,1:4) found in line 8 denoted elements 1 to 4 in the J-direction and elements 1 to 4 in the K-direction, and is a commonplace vector extension in modern day FORTRAN compilers. After execution of line 8 the first element of the descriptor array DD, DD(1,1) would contain a descriptor pointing to D(2,J,1,1;KMM); the element DD(2,1) would contain a descriptor pointing to D(2,J,2,1;KMM) . . . and so on. The effect of this statement in this subprogram is to create a descriptor pointing to all elements in the exact same position in the 4x4 blocks in the tridiagonal being generated.

```
 Line
Number          Source Statement

                SUBROUTINE FILTRX(K,J1,J2)
                 -COMMON DECLARATIONS-

  1             JA = J1+1
  2             JB = J2-1
  3             DO 10 J = J1,J2
  4             R1 = XY(J,K,1)*HD
  5             R2 = XY(J,K,2)*HD
  6             CALL AMATRX(E,J,K,R1,R2)
  7             DO 11 N = 1,4
  8             DO 12 M = 1,4
  9       12    D(J,N,M) = E(N,M)
 10       11    D(J,N,N) = D(J,N,N)+HD*XIT(J,K)
 11       10    CONTINUE
 12             DO 20 J = JA,JB
 13             DO 21 N = 1,4
 14             DO 22 M = 1,4
 15             A(J,N,M) = -D(J-1,N,M)
 16             B(J,N,M) = 0.
 17       22    C(J,N,M) = D(J+1,N,M)
 18             F(J,N) = S(J,K,N)
 19             A(J,N,N) = A(J,N,N)+ALPX(J)
 20             C(J,N,N) = C(J,N,N)+ALPX(J)
 21       21    B(J,N,N) = BETX(J)
 22       20    CONTINUE
 23             RETURN
```

Figure 5-42. Scalar FORTRAN: FILTRX

```
 Line
Number         Source Statement

               SUBROUTINE VFILTRX
               DYNAMIC ARRAY R1,R2,Q1,Q2,Q3,Q4,RR,U,V,XIT1,A(4,4),B(4,4)
          1          C(4,4),DD(4,4),F(4)
               — COMMON DECLARATIONS—

  1            DO 10 J = J1,J2
  2            R1 = XY(J,2:KM,1)
  3            R2 = XY(J,2:KM,2)  $  R1 = R1*HD
  4            Q1 = Q(J,2:KM,1)  $  R2 = R2*HD
  5            Q2 = Q(J,2:KM,2)  $  RR = 1./Q1
  6            Q3 = Q(J,2:KM,3)  $  U=Q2*RR
  7            Q4 = Q(J,2:KM,4)  $  V = Q3*RR
  8            ASSIGN DD(1:4,1:4),D(2,J,1:4,1:4;KMM)
  9            CALL VAMATRX (DD,R1,R2,Q1,Q2,Q3,Q4,RR,U,V)
 10            XIT1 = XIT(J,2:KM)
 11            DO 10 N = 1,4
 12      10    DD(N,N) = DD(N,N)+HD*XIT1
 13            DO 30 N=1,4
 14            DO 20 M = 1,4
 15            A(N,M)(1,2;KMAX) = -D(N,M)
 16            B(N,M)(1,2;KMAX) = 0.
 17      20    C(N,M)(1,2;KMAX) = D(N,M)(1,3;*)
 18            DO 25 J = J1,J2
 19      25    F(1,J,N;KMAX) = S(J,1:KMAX,N)
 20            DO 30 J = JA,JB
 21            A(N,N)(1,J;KMAX) = A(N,N)(1,J;KMAX)+ALPX(J)
 22            C(N,N)(1,J;KMAX) = C(N,N)(1,J;KMAX)+ALPX(J)
 23      30    B(N,N)(1,J;KMAX) = BETX(J)
```

Figure 5-43. Vector FORTRAN: VFILTRX, Left-Hand Side

As was mentioned previously, the K-direction sweep, which invokes FILTRX, is along a direction necessitating the gathering of data from non-sequential locations in physical memory in the NSS. The architecture of the NSS permits and encourages the use of the vector buffers to perform calculations whilst the Map Unit is performing the slower gather operation. In line 2 of figure 5-43 is the first of such gather operations wherein elements are extracted from the matrix XY along the non-sequential direction of storage. In line 3, R2 is formed in a similar manner, however, useful floating-point computations can be performed on the previously gathered vector R1 which now resides in the vector buffers. This is shown by including two distinct statements on the same line, separated by the dollar sign to indicate simultaneous operation. This scheme is primarily for illustration purposes, since it is expected that a mature, vectorizing FORTRAN compiler will be able to detect the potential for such overlap and generate the instructions without being explicitly instructed by the programmer as shown here in lines 3 through 7.

Table 5-24 shows the effect of this overlap in lines 3 through 7. The timing formulas differ depending on the length KMM (or KM), since the startup and shutdown of the gather and the vector arithmetic, plus the throughput rates of the Map Unit and the Vector Unit, create a series of tradeoff considerations. This is particularly obvious in line 5 where the startup time of the divide algorithm (which requires all four arithmetic elements in a vector unit) covers more cycles of the 'paired' gather operation.

Table 5-25 gives the specimen times for VFILTRX. As can be seen there and by comparison with the corresponding data for VFILTRY (table 5-29), the cost of the gather operations substantially reduces the effective arithmetic rate for this segment of code. Where VFILTRY (which processes sequentially stored data can achieve a rate of 1181 operations per microsecond, VFILTRX (which must sweep non-sequential elements) is limited to 245 operations per microsecond. This degradation is not of great concern in the two-dimensional case since, as will be seen later (tables 5-40, 5-41), VFILTRX constitutes only 6% of the total operations of a single time step, and imposes only 17% of the total machine cycles.

AMATRX is vectorized in a very straightforward manner, as in previous examples. The vectorized version (figure 5-45) would be better expanded as inline code in the FILTRX and FILTRY programs rather than accepting the overhead of subroutine calls to this frequently used subroutine. The times for AMATRX are shown in tables 5-26 and 5-27 and are also included in the timings for FILTRX (tables 5-24 and 5-25) and FILTRY (tables 5-28 and 5-29).

Table 5-24. NSS Timing: VFILTRX

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown		Cycles Required to Execute This Line
1	Scalar	Scalar		5
2	1/2	12		12+KMM/8
3 (1)	1/2,1/8	12,12+1*8		12+KMM/2,20+KMM/8
4 (1)	1/2,1/8	12,12+1*8		12+KMM/2,20+KMM/8
5 (2)	1/2,1/8	12,12+4*8		12+KMM,44+KMM/8
6 (1)	1/2,1/8	12,12+1*8	Executed JMAX Times	12+KMM/2,20+KMM/8
7 (1)	1/2,1/8	12,12+1*8		12+KMM/2,20+KMM/8
8	Scalar	Covered		0
9	27/8	676		676+27*KMM/8
10	1/2	12		12+KMM/2
11	Scalar	Covered		0
12	1/8	12+2*8		4(28+KMM/8)
13	Scalar	Covered		0
14	Scalar	Covered		0
15	1/8	12+1*8		16(20+JMM*KMAX/8)
16	1/8	12		16(12+JMM*KMAX/8)
17	1/8	12+1*8		16(20+JMM*KMAX/8)
18	1/2	12		4*JMAX*(12+KMAX/2)
19	Scalar	Covered		0
20	Scalar	Covered		0
21	1/8	12+1*8	Executed 4*JM Times	20+KMAX/8
22	1/8	12+1*8		20+KMAX/8
23	1/8	12		12+KMAX/8
	Total	421+JMAX*(1146+81*KMAX/8)+KMAX*(6*JMAX-15)		

Notes:

(1) For KMM.GE.22 use left-hand formula

(2) For KMM.GE.86 use left-hand formula

Table 5-25. Specimen Times for NSS: VFILTRX

JMAX	KMAX	Operations 75*JMAX*(KMAX-2)+24*(JMAX-2)*KMAX	Cycles 421+JMAX*(1146+81*KMAX/8)+KMAX*(6*JMAX-15)	Results Per Microsecond (10 ns Clock)
100	50	477,600	194,896	245
128	64	788,736	278,245	283
200	80	1,550,160	486,421	319

```
 Line
Number      Source Statement

            SUBROUTINE AMATRX(A,J,K,R1,R2)
             —COMMON DECLARATIONS—

   1        GAMI = GAMMA-1.
   2        RR = 1./Q(J,K,1)
   3        U = Q(J,K,2)*RR
   4        V = Q(J,K,3)*RR
   5        UT = U*U+V*V
   6        C1 = GAMI*UT*.5
   7        C2 = Q(J,K,4)*RR*GAMMA
   8        A(1,1) = 0.
   9        A(1,2) = R1
  10        A(1,3) = R2
  11        A(1,4) = 0.
  12        A(2,1) = (-U*U+C1)*R1-U*V*R2
  13        A(2,2) = -(GAMMA-3.0)*U*R1+V*R2
  14        A(2,3) = -GAMI*V*R1+U*R2
  15        A(2,4) = GAMI*R1
  16        A(3,1) = -U*V*R1+(-V*V+C1)*R2
  17        A(3,2) = V*R1-GAMI*U*R2
  18        A(3,3) = U*R1+(3.0-GAMMA)*V*R2
  19        A(3,4) = GAMI*R2
  20        QS = U*R1+V*R2
  21        A(4,1) = (-C2+C1*2.)*QS
  22        A(4,2) = (C2-C1)*R1-GAMI*U*QS
  23        A(4,3) = (C2-C1)*R2-GAMI*V*QS
  24        A(4,4) = GAMMA*QS
```

Figure 5-44. Scalar FORTRAN: AMATRX

```
 Line
Number          Source Statement

                SUBROUTINE VAMATRX(D,R1,R2,Q1,Q2,Q3,Q4,RR,U,V)
                DYNAMIC ARRAY D(4,4),R1,R2,Q1,Q2,Q3,Q4,RR,U,V,UT,C1,C2,QS,T1,T2

   1            UT = U*U+V*V
   2            C1 = (.5*GAMI)*UT
   3            C2 = (Q4*RR*GAMMA)
   4            D(1,1) = 0
   5            D(1,2) = R1
   6            D(1,3) = R2
   7            D(1,4) = 0
   8            UT = R1*(C1-U*U)
   9            D(2,1) = UT-U*U*R2
  10            UT = (GAMMA-3.)*U*R1
  11            D(2,2) = V*R2-UT
  12            UT = GAMI*V*R1
  13            D(2,3) = U*R2-UT
  14            D(2,4) = GAMI*R1
  15            UT = R2*(C1-V*V)
  16            D(3,1) = UT-U*V*R1
  17            UT = U*R2
  18            D(3,2) = V*R1-GAMI*UT
  19            UT = V*R2
  20            D(3,3) = U*R1-GAMI*UT
  21            D(3,4) = GAMI*R2
  22            QS = U*R1+V*R2
  23            D(4,1) = QS*(2*C1-C2)
  24            T1 = U*QS  $  T2 = V*QS
  25            D(4,4) = GAMMA*QS  $  UT = C2-C1
  26            D(4,2) = UT*R1-GAMI*T1
  27            D(4,3) = UT*R2-GAMI*T2
  28            RETURN
```

Figure 5-45. Vector FORTRAN: VAMATRX

Table 5-26. NSS Timing: VAMATRX, 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles for Startup and Rundown	Cycles Required to Execute This Line
1	1/8	12+2*8	28+LB/8
2	1/8	12+1*8	20+LB/8
3	1/8	12+2*8	28+LB/8
4	1/8	12	12+LB/8
5	1/8	12	12+LB/8
6	1/8	12	12+LB/8
7	1/8	12	12+LB/8
8	1/8	12+3*8	36+LB/8
9	1/8	12+3*8	36+LB/8
10	1/8	12+2*8	28+LB/8
11	1/8	12+2*8	28+LB/8
12	1/8	12+2*8	28+LB/8
13	1/8	12+2*8	28+LB/8
14	1/8	12+1*8	20+LB/8
15	1/8	12+3*8	36+LB/8
16	1/8	12+3*8	36+LB/8
17	1/8	12+1*8	20+LB/8
18	1/8	12+2*8	28+LB/8
19	1/8	12+1*8	20+LB/8
20	1/8	12+2*8	28+LB/8
21	1/8	12+1*8	20+LB/8
22	1/8	12+2*8	28+LB/8
23	1/8	12+3*8	36+LB/8
24	1/8	12+1*8	20+LB/8
25	1/8	12+1*8	20+LB/8
26	1/8	12+2*8	28+LB/8
27	1/8	12+2*8	28+LB/8
		Total	676+27*LB/8

Table 5-27. Specimen Times for NSS: VAMATRX, 64-Bit Mode

LB	Operations (67*KMAX*JMAX)	Cycles Required For Startup and Rundown (676+27*LB/8)	Results Per Microsecond (10 ns Clock)
50	3,350	845	396
64	4,288	892	481
80	5,896	946	623
100	6,700	1014	661
128	8,576	1108	774
200	13,400	1351	992

```
 Line
Number          Source Statement

                SUBROUTINE FILTRY
                DIMENSION E(4,4)
  1             KA = K1+1
  2             KB = K2-1
  3             DO 10 K = K1,K2
  4             R1 = XY(J,K,3)*HD
  5             R2 = XY(J,K,4)*HD
  6             CALL AMATRX(E,J,K,R1,R2)
  7             DO 11 N = 1,4
  8             DO 12 M = 1,4
  9       12    D(K,N,M) = E(N,M)
 10       11    D(K,N,N) = D(K,N,N)+HD*ETT(J,K)
 11       10    CONTINUE
 12             DO 20 K = KA,KB
 13             DO 21 N = 1,4
 14             DO 22 M =1,4
 15             A(K,N,M) = -D(K-1,N,M)
 16             B(K,N,M) = 0.
 17       22    C(K,N,M) = D(K+1,N,M)
 18             F(K,N) = S(J,K,N)
 19             A(K,N,N) = A(K,N,N)+ALPY(K)
 20             B(K,N,N) = B(K,N,N)+BETY(K)
 21       21    C(K,N,N) = C(K,N,N)+ALPY(K)
 22       20    CONTINUE
 23             IF(INVIS.GT.0) CALL VISMAT(J)
 24             RETURN
```

Figure 5-46. Scalar FORTRAN: FILTRY

```
 Line
Number            Source Statement

                  SUBROUTINE VFILTRY
                  DYNAMIC ARRAY R1,R2,Q1,Q2,Q3,Q4,RR,U,V,XIT1,A(4,4),B(4,4)
             1             C(4,4),DD(4,4),F(4)
                   —COMMON DECLARATIONS—

   1              DO 10 K=1,KMAX
   2              RR = 1./Q(2,1,1;JMM)
   3              U = Q(2,1,2;JMM)*RR  $  V = Q(2,1,3;JMM)*RR
   4              ASSIGN Q4,Q(2,1,4;JMM)
   5              R1 = XY(2,1,3;JMM)*HD  $  R2 = XY(2,1,4;JMM)*HD
   6              CALL VAMATRX(DD,R1,R2,Q1,Q2,Q3,Q4,RR,U,V)
   7              DO 10 N = 1,4
   8         10   D(N,N)(2,K;JMM) = D(N,N)(2,K;JMM)+HD*ETT(2,K,JMM)
   9              DO 21 K = 2,KM
  10              DO 21 N = 1,4
  11              DO 22 M = 1,4
  12              A(N,M)(2,K;JMM) = -D(N,M)(2,K-1;JMM)
  13              B(N,M)(2,K;JMM) = 0.
  14         22   C(N,M)(2,K;JMM) = D(N,M)(2,K+1;JMM)
  15              F(N)(2,K;JMM) = S(N)(2,K;JMM)
  16              A(N,N)(2,K;JMM) = A(N,N)(2,K;JMM)+ALPY(K)
  17              B(N,N)(2,K;JMM) = B(N,N)(2,K;JMM)+BETY(K)
  18         21   C(N,N)(2,K;JMM) = C(N,N)(2,K;JMM)+ALPY(K)
  19              IF (INVIS.GT.0) CALL VVISMAT
  20              RETURN
```

Figure 5-47. Vector FORTRAN: VFILTRY

Table 5-28. NSS Timing: VFILTRY

Line Number	Cycles Per Result Generated	Cycles for Startup and Rundown		Cycles Required to Execute This Line
1	Scalar	Uncovered (pass 1 only)		6
2	1/8	12+4*8		44+JMM/8
3	1/8	12+1*8		20+KMM/8
4	Scalar	Covered		0
5	1/8	12+1*8	Executed KMAX Times	20+JMM/8
6	VAMATRX	VAMATRX		676+27*KMM/8
7	Scalar	Covered		0
8	1/8	12+2*8		4+(28+JMM/8)
9	Scalar	Covered		0
10	Scalar	Covered		0
11	Scalar	Covered		0
12	1/8	12+1*8		16+(20+JMM/8)
13	1/8	12		16(12+JMM/8)
14	1/8	12		16(12+JMM/8)
15	1/8	12	Executed KMM Times	4(12+JMM/8)
16	1/8	12+1*8		4(20+JMM/8)
17	1/8	12+1*8		4(20+JMM/8)
18	1/8	12+1*8		4(20+JMM/8)
19	VVISMAT	VVISMAT		1493+50*JMM/8+KMM*(2040+86*JMM/8)
		Total		1493+2040*KMM+(50+8*KMM)*JMM/8

Table 5-29. Specimen Times for NSS: VFILTRY

JMAX	KMAX	Operations JMM* (257*KMAX-228)	Cycles 1493+2040*KMM+ (50+8*KMM)*JMM/8	Results Per Microsecond (10 ns Clock)
100	50	1,236,956	104,730	1181
128	64	2,043,720	136,573	1496
200	80	4,025,736	177,295	2271

The codes for FILTRY and its vectorization VFILTRY (figures 5-46 and 5-47) are included at this point in the narrative so that they can be contrasted with the VFILTRX just discussed. Again, the vectorization is straightforward and the timing formula and specimen timing for this routine can be found in tables 5-28 and 5-29.

The highlight here, as has been stated, is the achieving of a floating-point operation rate in excess of 1000 per microsecond when the data is accessed in the sequential direction in storage. Note also that VFILTRY performs more calculations per time step than does VFILTRX by almost a factor of three. This is due primarily to the computation of the viscous effects by the VISMAT call. However it should be obvious that a method of storing slices of the matrices can be arrived at directly from the FORTRAN code, so that the most efficient processing (i.e., along sequential storage streams) is sustained for over 70% of the execution time.

The scalar form of BTRI is shown in Figure 5-48. The vector solution is broken down into three portions, primarily for readability, so that each computational phase can be better exposed.

Figure 5-49 and table 5-30 illustrate the vectorization of the startup phase of the tridiagonal solution. In the implementation given here, the algorithm for solving the tridiagonal has been copied from the scalar code, but vectorized in a straightforward manner. It should be pointed out that numerous efforts are underway to develop more efficient tridiagonal solvers for STAR and ILLIAC type machines. Given a better understanding of the numerical behavior of the solutions of interest to the NASF, perhaps a more efficient vectorization could be obtained. As can be seen by the timing formula, the two-dimensional version of BTRI suffers from the cost of vector startup compared to the length of the vectors processed, which are quite short. In the three-dimensional version of the code we can expect the vectors in BTRI to be quite a bit longer (one full plane's worth, in fact), and thus the computation rate will be much higher.

Figures 5-50, 5-51 and tables 5-31 and 5-32 demonstrate the forward sweep and back substitution vectorization and timing for BTRI. Again, the vectorization is straightforward, with most choices revolving around the tradeoffs between vector lengths, temporary storage requirements, and vector startup time. The slicing system given here, has evolved slowly from the original scalar code, and may be further improved in later phases of the study.

Line Number   Source Statement

```
      SUBROUTINE BTRI(IL,IU)
      IS = IL+1
      I = IL
      DO 11 N = 1,4
      DO 11 M = 1,4
   11 G(N,M) = B(I,N,M)
      CALL LUDEC(G)
      D1 = V1*F(I,1)
      D2 = V2*(F(I,2)-L21*D1)
      D3 = V3*(F(I,3)-L31*D1-L32*D2)
      D4 = V4*(F(I,4)-L41*D1-L42*D2-L43*D3)
      F(I,4) = D4
      F(I,3) = D3-U34*D4
      F(I,2) = D2-U24*D4-U23*F(I,3)
      F(I,1) = D1-U14*D4-U13*F(I,3)-U12*F(I,2)
      DO 12 M = 1,4
      D1 = V1*C(I,1,M)
      D2 = V2*(C(I,2,M)-L21*D1)
      D3 = V3*(C(I,3,M)-L31*D1-L32*D2)
      D4 = V4*(C(I,4,M)-L41*D1-L42*D2-L43*D3)
      B(I,4,M) = D4
      B(I,3,M) = D3-U34*D4
      B(I,2,M) = D2-U24*D4-U23*B(I,3,M)
      B(I,1,M) = D1-U14*D4-U13*B(I,3,M)-U12*B(I,2,M)
   12 CONTINUE
      DO 13 I = IS,IU
      IR = I-1
      DO 14 N = 1,4
      F(I,N) = F(I,N)-A(I,N,1)*F(IR,1)-A(I,N,2)*F(IR,2)
     1     -A(I,N,3)*F(IR,3)-A(I,N,4)*F(IR,4)
      DO 14 M = 1,4
```

Continued

Source Statement

```
          G(N,M) = B(I,N,M)-A(I,N,1)*B(IR,1,M)-A(I,N,2)*B(IR,2,M)
     1        -A(I,N,3)*B(IR,3,M)-A(I,N,4)*B(IR,4,M)
    14    CONTINUE
          CALL LUDEC(G)
          D1 = V1*F(I,1)
          D2 = V2*(F(I,1)-L21*D1)
          D3 = V3*(F(I,3)-L31*D1-L32*D2)
          D4 = V4*(F(I,4)-L41*D1-L42*D2-L43*D3)
          F(I,4) = D4
          F(I,3) = D3-U34*D4
          F(I,2) = D2-U24*D4-U23*F(I,3)
          F(I,1) = D1-U14*D4-U13*F(1,3)-U12*F(1,2)
          IF(I-IU)16,13,13
    16    CONTINUE
          DO 15 M = 1,4
          D1 = V1*C(I,1,M)
          D2 = V2*(C(I,2,M)-L21*D1)
          D3 = V3*(C(I,3,M)-L31*D1-L32*D2)
          D4 = V4*(C(I,4,M)-L41*D1-L42*D2-L43*D3)
          B(I,4,M) = D4
          B(I,3,M) = D3-U34*D4
          B(I,2,M) = D2-U24*D4-U23*B(I,3,M)
          B(I,1,M) = D1-U14*D4-U13*B(I,3,M)-U12*B(I,2,M)
    15    CONTINUE
    13    CONTINUE
          IT = IL+IU
          DO 21 II = IS,IU
          I = IT-II
          IP = I+1
          DO 22 N = 1,4
```

Continued

```
 Line
Number          Source Statement

  61            F(I,N) = F(I,N)-B(I,N,1)*F(IP,1)-B(I,N,2)*F(IP,2)
  62         1       -B(I,N,3)*F(IP,3)-B(I,N,4)*F(IP,4)
  63        22  CONTINUE
  64        21  CONTINUE
  65            RETURN
```

Figure 5-48. Scalar FORTRAN: BTRI

```
 Line
Number        Source Statement

   1          IS = IL+1  $  I = IL
   2          CALL VLUDEC(B)
   3          D1 = V1*F(I,1)  $  D2 = F(I,2)-L21*D1
   4          D2 = V2*D2  $  D3 = L32*D2
   5          D3 = V3*(F(I,3)-L31*D1-D3)
   6          D4 = L42*D2+L43*D3
   7          D4 = V4*(F(I,4)-L41*D1-D4)
   8          F(I,3) = D3-U34*D4
   9          F(I,2) = D2-U24*D4-U23*F(I,3)
  10          T1 = U13*F(I,3)+U12*F(I,2)  $  F(I,4) = D4
  11          F(I,1) = D1-U14*D4-T1
  12          DO 12 M = 1,4
  13          D1 = V1*C(I,1,M)  $  D3 = L31*D1
  14          D2 = V2*(C(I,2,M)-L21*D1)
  15          D3 = V3*(C(I,3,M)-D3-L32*D2)
  16          D4 = L42*D2+L43*D3
  17          D4 = V4*(C(I,4,M)-L41*D1-D4)
  18          B(I,3,M) = D3-U34*D4
  19          B(I,2,M) = D2-U24*D4-U23*B(I,3,M)
  20          T1 = U13*B(I,3,M)+U12*B(I,2,M)  $ B(I,4,M) = D4
  21          B(I,1,M) = D1-U14*D4-T1
  22       12 CONTINUE
```

Figure 5-49. Vector FORTRAN: SUBROUTINE VBTRI (Startup Solution Segment)

Table 5-30. NSS Timing: SUBROUTINE VBTRI (Startup Solution Segment) 64-Bit Mode

Line Number	Cycles Per Result Generated	Cycles for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Not covered	5
2	VLUDEC	VLUDEC	564+17*NT/8
3	1/8	12+2*8	28+NT/8
4	1/8	12+1*8	20+NT/8
5	1/8	12+3*8	36+NT/8
6	1/8	12+2*8	28+NT/8
7	1/8	12+3*8	36+NT/8
8	1/8	12+2*8	28+NT/8
9	1/8	12+3*8	36+NT/8
10	1/8	12+2*8	28+NT/8
11	1/8	12+2*8	28+NT/8
12	Scalar	Covered	0
13	1/8	12+1*8	4*(20+NT/8)
14	1/8	12+3*8	4*(36+NT/8)
15	1/8	12+3*8	4*(36+NT/8)
16	1/8	12+2*8	4*(28+NT/8)
17	1/8	12+3*8	4*(36+NT/8)
18	1/8	12+2*8	4*(28+NT/8)
19	1/8	12+3*8	4*(36+NT/8)
20	1/8	12+2*8	4*(28+NT/8)
21	1/8	12+2*8	4*(28+NT/8)
22	Scalar	Covered	0
		Total	1941+62 NT/8

```
 Line
Number         Source Statement

   1           DO 13 I = IS,IU
   2           IR = I-1
   3           DO 14 N = 1,4
   4           T1 = A(I,N,1)*F(IR,1)
   5           T1 = T1+A(I,N,2)*F(IR,2)
   6           T1 = T1+A(I,N,3)*F(IR,3)
   7           T1 = T1+A(I,N,4)*F(IR,4)
   8           F(I,N) = F(I,N)-T1
   9           DO 14 M = 1,4
  10           T1 = A(I,N,1)*B(IR,1,M)
  11           T1 = T1+A(I,N,2)*B(IR,2,M)
  12           T1 = T1+A(I,N,3)*B(IR,3,M)
  13           T1 = T1+A(I,N,4)*B(IR,4,M)
  14      14   G(N,M) = B(I,N,M)-T1
  15           CALL VLUDEC(G)
  16           D1 = V1*F(I,1)
  17           D2 = V2*(F(I,2)-L21*D1)
  18           D3 = L31*D1+L32*D2
  19           D3 = V3*(F(I,3)-D3)
  20           D4 = L42*D2+L43*D3
  21           D4 = V4*(F(I,4)-L41*D1-D4)
  22           F(I,3) = D3-U34*D4
  23           F(I,2) = D2-U24*D4-U23*F(I,3)
  24           T1 = U13*F(I,3)+U12*F(I,2)  $  F(I,4) = D4
  25           F(I,1) = D1-U14*D4-T1
  26           IF (I-IU)16,13,13
  27      16   CONTINUE
  28           DO 15 M = 1,4
  29           D1 = V1*C(I,1,M)  $  D3 = L31*D1
  30           D2 = V2*(C(I,2,M)-L21*D1)
```

Continued

```
 Line
Number         Source Statement

  31           D3 = V3*(C(I,3,M)-D3-L32*D2)
  32           D4 = L42*D2+L43*D3
  33           B(I,4,M) = V4*(C(I,4,M)-L41*D1-D4)
  34           B(I,3,M) = D3-U34*B(I,4,M)
  35           B(I,2,M) = D2-U24*B(I,4,M)-U23*B(I,3,M)
  36           T1 = U13*B(I,3,M)+U12*B(I,2,M)
  37           B(I,1,M) = D1-U14*B(I,4,M)-T1
  38      15   CONTINUE
  39      13   CONTINUE
```

Figure 5-50. Vector FORTRAN: VBTRI (Forward Sweep Segment)

Table 5-31. NSS Timing: VBTRI (Forward Sweep Segment)

Line Number	Cycles Per Result Generated	Cycles for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Covered	0
2	Scalar	Covered	0
3	Scalar	Covered	0
4	1/8	12+1*8	4*(IU-IL)*(20+NT/8)
5	1/8	12+2*8	4*(IU-IL)*(28+NT/8)
6	1/8	12+2*8	4*(IU-IL)*(28+NT/8)
7	1/8	12+2*8	4*(IU-IL)*(28+NT/8)
8	1/8	12+1*8	4*(IU-IL)*(20+NT/8)
9	Scalar	Covered	0
10	1/8	12+1*8	16*(IU-IL)*(20+NT/8)
11	1/8	12+2*8	16*(IU-IL)*(28+NT/8)
12	1/8	12+2*8	16*(IU-IL)*(28+NT/8)
13	1/8	12+2*8	16*(IU-IL)*(28+NT/8)
14	1/8	12+1*8	16*(IU-IL)*(20+NT/8)
15	VLUDEC	VLUDEC	(IU-IL)*(564+17*NT/8)
16	1/8	12+1*8	(IU-IL)*(20+NT/8)
17	1/8	12+3*8	(IU-IL)*(36+NT/8)
18	1/8	12+2*8	(IU-IL)*(28+NT/8)
19	1/8	12+2*8	(IU-IL)*(28+NT/8)
20	1/8	12+2*8	(IU-IL)*(28+NT/8)
21	1/8	12+3*8	(IU-IL)*(36+NT/8)
22	1/8	12+2*8	(IU-IL)*(28+NT/8)
23	1/8	12+3*8	(IU-IL)*(36+NT/8)
24	1/8	12+2*8	(IU-IL)*(28+NT/8)
25	1/8	12+2*8	(IU-IL)*(28+NT/8)
26	Scalar	Covered	0
27	Scalar	Covered	0
28	Scalar	Covered	0
29	1/8	12+1*8	4*(IU-IS)*20+NT/8
30	1/8	12+3*8	4*(IU-IS)*36-NT/8

Table 5-31. NSS Timing: VBTRI (Forward Sweep Segment) Cont.

Line Number	Cycles Per Result Generated	Cycles for Startup and Rundown	Cycles Required to Execute This Line
31	1/8	12+3*8	4*(IU-IS)*36+NT/8
32	1/8	12+2*8	4*(IU-IS)*28+NT/8
33	1/8	12+3*8	4*(IU-IS)*36+NT/8
34	1/8	12+2*8	4*(IU-IS)*28*NT/8
35	1/8	12+3*8	4*(IU-IS)*36+NT/8
36	1/8	12+2*8	4*(IU-IS)*28+NT/8
37	1/8	12+2*8	4*(IU-IS)*28*NT/8
38	Scalar	Covered	0
39	Scalar	Covered	0
		Total	(IU-IL)*(4444+163*NT/8)-(1104+36*NT/8)

Note: IS = IL +1

```
 Line
Number          Source Statement

  1             IT = IL+IU
  2             DO 21 II = IS,IU
  3             I = IT-II
  4             IT = I+1
  5             DO 22 N = 1,4
  6             T1 = B(I,N,1)*F(IP,1)
  7             T1 = T1+B(I,N,2)*F(IP,2)
  8             T1 = T1+B(I,N,3)*F(IP,3)
  9             T1 = T1+B(I,N,4)*F(IP,4)
 10             F(I,N) = F(I,N)-T1
 11        22   CONTINUE
 12        21   CONTINUE
```

Figure 5-51. Vector FORTRAN: VBTRI (Back Substitution Segment)

Table 5-32. NSS Timing: VBTRI (Back Substitution Segment)

Line Number	Cycles Per Result Generated	Cycles for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Covered	0
2	Scalar	Covered	0
3	Scalar	Covered	0
4	Scalar	Covered	0
5	Scalar	Covered	0
6	1/8	12+1*8	4*(IU-IL)*(20+NT/8)
7	1/8	12+2*8	4*(IU-IL)*(28+NT/8)
8	1/8	12+2*8	4*(IU-IL)*(28+NT/8)
9	1/8	12+2*8	4*(IU-IL)*(28+NT/8)
10	1/8	12+1*8	4*(IU-IL)*(20+NT/8)
11	Scalar	Covered	0
12	Scalar	Covered	0
		Total	(IU-IL)*(496+20*NT/8)

Figures 5-52 and 5-53 and tables 5-33 and 5-34 give the scalar and vector versions of the L-U decomposition algorithm embodied in the subprogram LUDEC. Here again the expectation is that it would be more efficient to expand LUDEC in-line in each of the calls in BTRI, to minimize overhead. The additional code space required to do this would be insignificant. In the vector version of LUDEC the choice made was to force the assignment of the decomposed variables V, U, and L to the vector buffers. Since these variables are used immediately by BTRI, this permits a high degree of utilization of the buffers and the multiple arithmetic units. Table 5-35 gives specimen times for the combined BTRI, LUDEC sequence, and it can be seen that even in the two-dimensional case, a judiscious choice of slicing parameters can yield almost a one-gigaflop computation rate. This is considered to be the worst case the NSS will encounter in reduction of the overall rate due to vector lengths and the transpose requirements for the non-sequential access sweep.

The final major segment of computation is VISMAT, which is shown in figures 5-54 through 5-59. Again the program is subdivided into loop 10, loop 20 and loop 30 for readability (primarily to fit on a single page). The viscous computations are also vectorized in a straightforward manner. The contribution to the overall calculations are given in tables 5-36 through 5-39 and also are combined in the timings given for VBTRI, the assumption being that the time step being estimated includes viscous effects.

Tables 5-40, 5-41 and 5-42 give the summations of the Left-Hand Side calculations and the combined timing factors for a single time step. Note that they are given in terms of 64-bit arithmetic only. Notice in table 5-42 the effect of mesh size and allocation on the computation rates. This is obviously due to the amount of computing that can be avoided in the non-sequential direction by a judicious choice of mesh parameters. The 661-megaflop rate is considered a worst case number for the Navier-Stokes solution in two dimensions. If all the computations were done in 32-bit mode however, the rates improve to the point shown in table 5-42, and exceed the minimum performance objective of 1 gigaflop for the NSS.

Based on a limited examination of the recently developed three-dimensional code and the work completed in this section, it is felt that with a proper mixture of 32-bit computations (when accuracy and stability permit) and long vectors representing planes in the three dimensions, a sustained result rate in excess of 3 gigaflops is achievable with a 10 nanosecond clock NSS. This projection will be examined in detail in subsequent phases of this project.

```
 Line
Number       Source Statement

  1          L11 = A(1,1)
  2          V1 = 1./L11
  3          U12 = V1*A(1,2)
  4          U13 = V1*A(1,3)
  5          U14 = V1*A(1,4)
  6          L21 = A(2,1)
  7          L22 = A(2,2)-L21*U12
  8          V2 = 1./L22
  9          U23 = (A(2,3)-L21*U13)*V2
 10          U24 = (A(2,4)-L21*U14)*V2
 11          L31 = A(3,1)
 12          L32 = A(3,2)-L31*U12
 13          L33 = A(3,3)-L31*U13-L32*U23
 14          V3 = 1./L33
 15          U34 = (A(3,4)-L31*U14-L32*U24)*V3
 16          L41 = A(4,1)
 17          L42 = A(4,2)-L41*U12
 18          L43 = A(4,3)-L41*U13-L42*U23
 19          L44 = A(4,4)-L41*U14-L42*U24-L43*U34
 20          V4 = 1./L44
 21          RETURN
```

Figure 5-52. Scalar FORTRAN: LUDEC

Line Number   Source Statement

```
      SUBROUTINE VLUDEC(A)
      IMPLICIT REAL L
      COMMON/LUDEC/V1,U12,U13,U14,L21,V2,U23,U24,L31,L32,V3,U34
     1          L41,L42,L43,V4
      LEVEL 1 ,V1,U12,U13,U14,L21,V2,U23,U24,L31,L32,V3,U34,L41
     1          L42,L43,V4
      DYNAMIC ARRAY V1,U12,U13,U14,L21,V2,U23,U24,L31,L32,V3
     1          U34,L41,L42,L43,V4
      DYNAMIC ARRAY A(4,4)
      EQUIVALENCE (L11,V1),(L22,V2),(L33,V3),(L44,V4)
      L11 = A(1,1)  $  V1 = 1./L11
      U12 = V1*A(1,2)  $  U13 = V1*A(1,3)
      U14 = V1*A(1,4)  $  L21 = A(2,1)
      L22 = A(2,2)-L21*U12
      V2 = 1./L22
      U23 = V2*(A(2,3)-L21*U13)
      U24 = V2*(A(2,4)-L21*U14)
      L31 = A(3,1)  $  L32 = A(3,1)-L31*U12
      L33 = A(3,3)-L31*U13-L32*U23
      V3 = 1./L33
      L34 = A(3,4)-L31*U14-L32*U24
      L34 = V3*L34  $  L41 = A(4,1)
      L42 = A(4,2)-L41*U12
      L43 = A(4,3)-L41*U13-L42*U23
      L44 = L42*U24+L43*U34
      L44 = (A(4,4)-L44)-L41*U14
      V4 = 1./L44
      RETURN
```

Figure 5-53. Vector FORTRAN: VLUDEC(A)

Table 5-33. NSS Timing: VLUDEC

Line Number	Cycles Per Result Generated	Cycles for Startup and Rundown	Cycles Required to Execute This Line
1	1/8	12+4*8	44+LB/8
2	1/8	12+1*8	20+LB/8
3	1/8	12+1*8	20+LB/8
4	1/8	12+2*8	28+LB/8
5	1/8	12+4*8	44+LB/8
6	1/8	12+3*8	36+LB/8
7	1/8	12+3*8	36+LB/8
8	1/8	12+2*8	28+LB/8
9	1/8	12+3*8	36+LB/8
10	1/8	12+4*8	44+LB/8
11	1/8	12+3*8	36+LB/8
12	1/8	12+1*8	20+LB/8
13	1/8	12+2*8	28+LB/8
14	1/8	12+3*8	36+LB/8
15	1/8	12+2*8	28+LB/8
16	1/8	12+2*8	28+LB/8
17	1/8	12+4*8	44+LB/8
		Total	564+17*LB/8

Table 5-34. Specimen Times for NSS: VLUDEC

LB (KMM or JMM Depending on Direction of Sweep)	Operations 38*LB	Cycles 564+17*LB/8	Results Per Microsecond (10 ns Clock)
48	1824	666	274
62	2356	696	339
78	2964	730	406
98	3724	773	482
126	4788	832	575
198	7524	985	764

Table 5-35. Specimen Times for NSS: VBTRI and VLUDEC

JMAX	KMAX	IU-IL	NT	Operations	Cycles	Results Per Microsecond (10 ns Cycle)
100	50	98	48	2,164,320	451,921	479
100	50	48	98	2,174,620	266,746	815
128	64	126	62	3,590,668	611,818	587
128	64	62	126	3,603,852	374,730	962
200	80	198	78	7,089,420	1,014,984	698
200	80	78	198	7,114,140	572,604	1242

Notes: Total operations: (VBTRI,VLUDEC) = (206+458*(IU-IL))*NT
Number of cycles: 3069+100*NT/8+(IU-IL)*(3716+143*NT/8)

NT = LB equals number of tridiagonal systems solved

```
 Line
Number          Source Statement

                SUBROUTINE VISMAT (J)
                DATA PR,PRTR,FRT/.72,1.,1.33 . . . 3/
                HRE = HS/RE
                FRT = 4./3.
                GPR = GAMMA/PR
  1             DO 10 K = 1,KMAX
  2             R1 = 1./Q(J,K,1)
  3             EXS = XY(J,K,3)**2
  4             EYS = XY(J,K,4)**2
  5             EXY = XY(J,K,3)*XY(J,K,4)*.33 . . . 3
  6             TURM = TURMU(J,K)
  7             VNU = FMU(K)+TURM
  8             GKAP = FMU(K)+PRTR*TURM
  9             DJAC = HRE/P(J,K)
 10             VNUJAC = VNU*DJAC
 11             C1(K) = VNUJAC*(FRT*EXS+EYS)
 12             C2(K) = VNUJAC*EXY
 13             C3(K) = VNUJAC*(EXS+FRT*EYS)
 14             C4(K) = GPR*DJAC*(EXS+EYS)*GKAP
 15             RR(K) = R1
 16             U(K) = R1*Q(J,K,2)
 17             V(K) = R1*Q(J,K,3)
 18             TC(K) = Q(J,K,4)*R1-(U(K)**2+V(K)**2)
           10   CONTINUE
```

Figure 5-54. Scalar FORTRAN: VISMAT, Loop 10

```
 Line
Number          Source Statement

  1             DO  20  K = 2,KMAX
  2             KR = K-1
  3             CC1 = C1(KR)+C1(K)
  4             CC2 = C2(KR)+C2(K)
  5             CC3 = C3(KR)+C3(K)
  6             CC4 = C4(KR)+C4(K)
  7             D(K,2,1) = - (CC1*U(KR)+CC2*V(KR))*RR(KR)
  8             DU(K,2,1) = - (CC1*U(K)+CC2*V(K))*RR(K)
  9             D(K,2,2) = CC1*RR(KR)
 10             DU(K,2,2) = CC1*RR(K)
 11             D(K,2,3) = CC2*RR(KR)
 12             DU(K,2,3) = CC2*RR(K)
 13             D(K,2,4) = 0.
 14             DU(K,2,4) = 0.
 15             D(K,3,1) = -(CC2*U(KR)+CC3*V(KR))*RR(KR)
 16             DU(K,3,1) = -(CC2*U(K)+CC3*V(K))*RR(K)
 17             D(K,3,2) = CC2*RR(KR)
 18             DU(K,3,2) = CC2*RR(K)
 19             D(K,3,3) = CC3*RR(KR)
 20             DU(K,3,3) = CC3*RR(K)
 21             D(K,3,4) = 0.
 22             DU(K,3,4) = 0.
 23             D(K,4,1) = -(CC4*TC(KR)+CC1*U(KR)**2+2.*CC2*U(KR)*V(KR)+
 24          1       CC3*V(KR)**2)*RR(KR)
 25             DU(K,4,1) = -(CC4*TC(K)+CC1*U(K)**2+2.*CC2*U(K)*V(K)+
 26          1       CC3*V(K)**2)*RR(K)
 27             D(K,4,2) = -CC4*U(KR)*RR(KR)-D(K,2,1)
 28             DU(K,4,2) = -CC4*U(K)*RR(K)-DU(K,2,1)
 29             D(K,4,3) = -CC4*V(KR)*RR(KR)-D(K,3,1)
 30             DU(K,4,3) = -CC4*V(K)*RR(K)-DU(K,3,1)
 31             D(K,4,4) = CC4*RR(KR)
 32             DU(K,4,4) = CC4*RR(K)
 33       20    CONTINUE
```

Figure 5-55. Scalar FORTRAN: VISMAT, Loop 20

```
 Line
Number         Source Statement

  1            DO  30  K = 2,KM
  2            KP = K+1
  3            DO 31 N = 2,4
  4            DO 31 M = 1,4
  5            A(K,N,M) = A(K,N,M)-D(K,N,M)
  6            B(K,N,M) = B(K,N,M)+D(KP,N,M)
  7            B(K,N,M) = B(K,N,M)+DU(K,N,M)
  8            C(K,N,M) = C(K,N,M)-DU(KP,N,M)
  9      31    CONTINUE
 10      30    CONTINUE
```

Figure 5-56. Scalar FORTRAN: VISMAT, Loop 30

```
 Line
Number          Source Statement

                VVISMAT
                 -DECLARATIONS-
  1             DO 10 K = 1,KMAX
  2             ASSIGN R1,RR(K)
  3             R1 = 1./Q(2:JM,K,1)
  4             EXS = XY(2:JM,K,3)**2 $ EYS = XY(2:JM,K,4)**2
  5             EXY = XY(2:JM,K,3)*XY(2:JM,K,4)*.333
  6             TURM = TURMU(2:JM,K) $ VNU = FMU(K)+TURM
           1         $   GKAP = FMU(K)+PRTR*TURM $ T1 = GPR*GKAP
  7             DJAC = HRE/P(2:JM,K)
  8             VNUJAC = VNU*DJAC $ C2(K) = VNUJAC*EXY
  9             C1(K) = VNUJAC*(FRT*EXS+EYS)
 10             C3(K) = VNUJAC*(EXS+FRT*EYS)
 11             C4(K) = T1*DJAC*(EXS+EYS)
 12             U(K) = R1*Q(2:JM,K,2) $ V(K) = R1*Q(2:JM,K,3)
 13             T1 = U(K)**2+V(K)**2
 14             TC(K) = Q(2:JM,K,4)*R1-T1
 15      10     CONTINUE
```

Figure 5-57. Vector FORTRAN: VVISMAT, Loop 10

```
 Line
Number         Source Statement

  1            DO 20  K = 2,KMAX
  2            KR = K-1
  3            CC1 = C1(KR)+C1(K)  $  CC2 = C2(KR)+C2(K)
  4            CC3 = C3(KR)+C3(K)  $  CC4 = C4(KR)+C4(K)
  5            T1 = CC1*RR(KR)  $  D(K,2,2) = T1
  6            T2 = CC2*RR(KR)  $  D(K,2,3) = T2
  7            T3 = CC1*RR(K)  $  DU(K,2,2) = T3
  8            T4 = CC2*RR(K)  $  DU(K,2,3) = T4
  9            D(K,2,1) = -(T1*U(KR)+T2*V(KR))
 10            DU(K,2,1) = -(T3*U(K)+T4*V(K))
 11            T5 = CC4*RR(KR)  $  D(K,3,2) = T2
 12            T6 = CC3*RR(KR)  $  (D(K,3,3) = T6
 13            T7 = CC4*RR(K)  $  DU(K,3,2) = T4
 14            T8 = CC3*RR(K)  $  DU(K,3,3) = T8
 15            D(K,3,1) = - (T2*U(KR)+T6*V(KR))
 16            DU(K,3,1) = - (T4*U(K)+T8*V(K))
 17            ASSIGN T9,D(K,4,1)
 18            T9 = T1*U(KR)**2  $  D(K,2,4) = 0.
 19            T9 = -T9-T6*V(KR)**2  $  DU(K,2,4) = 0.
 20            ASSIGN T10,DU(K,4,1)
 21            T10 = T2*U(KR)*V(KR)  $  D(K,3,4) = 0.
 22            D(K,4,1) = T9-T5*TC(KR)-2.*T10
 23            ASSIGN T9,D(K,4,2)
 24            T9 = T4*U(K)*V(K)  $  D(K,4,4) = T5
 25            T10 = T8*V(K)**2  $  DU(K,4,4) = T7
 26            T9 = T10+2.*T9+T7*TC(K)  $  DU(K,3,4) = 0.
 27            DU(K,4,1) = T9+T3*U(K)**2
 28            D(K,4,2) = T5*U(KR)-D(K,2,1)
 29            DU(K,4,2) = T7*U(K)-DU(K,2,1)
 30            D(K,4,3) = T5*V(KR)-D(K,3,1)
 31      20    DU(K,4,3) = T7*V(K)-DU(K,3,1)
```

Figure 5-58. Vector FORTRAN: VVISMAT, Loop 20

```
 Line
Number           Source Statement

  1              DO 30 K = 2,KM
  2              KP = K+1
  3              DO 31 N = 2,4
  4              DO 31 M = 1,4
  5              A(K,N,M) = A(K,N,M)-D(K,N,M)
  6              B(K,N,M) = B(K,N,M)+D(KP,N,M)
  7              B(K,N,M) = B(K,N,M)+DU(K,N,M)
  8              C(K,N,M) = C(K,N,M)-DU(KP,N,M)
  9      31      CONTINUE
 10      30      CONTINUE
```

Figure 5-59. Vector FORTRAN: VVISMAT, Loop 30

Table 5-36. NSS Timing: VVISMAT, Loop 10

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Not Covered On First Trip Only	1
2	Scalar	Not Covered On First Trip Only	5+8
3	1/8	12+4*8	KMAX*(44+JMM/8)
4	1/8	12+1*8	KMAX*(20+JMM/8)
5	1/8	12+2*8	KMAX*(28+JMM/8)
6	1/8	12+2*8	KMAX*(28+JMM/8)
7	1/8	12+4*8	KMAX*(44+JMM/8)
8	1/8	12+1*8	KMAX*(20+JMM/8)
9	1/8	12+3*8	KMAX*(36+JMM/8)
10	1/8	12+3*8	KMAX*(36+JMM/8)
11	1/8	12+3*8	KMAX*(36+JMM/8)
12	1/8	12+1*8	KMAX*(20+JMM/8)
13	1/8	12+2*8	KMAX*(28+JMM/8)
14	1/8	12+2*8	KMAX*(28+JMM/8)
15	Scalar	Covered	0
		64-Bit Total	12+KMAX(368+12*JMM/8)
		32-Bit Total	12+KMAX(368+12*JMM/16)

Table 5-37. NSS Timing: VVISMAT, Loop 20

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Scalar	1
2	Scalar	Scalar	1+5
3	1/8	12+1*8	KM*(20+JMM/8)
4	1/8	12+1*8	KM*(20+JMM/8)
5	1/8	12+1*8	KM*(20+JMM/8)
6	1/8	12+1*8	KM*(20+JMM/8)
7	1/8	12+1*8	KM*(20+JMM/8)
8	1/8	12+1*8	KM*(20+JMM/8)
9	1/8	12+3*8	KM*(36+JMM/8)
10	1/8	12+3*8	KM*(36+JMM/8)
11	1/8	12+1*8	KM*(20+JMM/8)
12	1/8	12+1*8	KM*(20+JMM/8)
13	1/8	12+1*8	KM*(20+JMM/8)
14	1/8	12+1*8	KM*(20+JMM/8)
15	1/8	12+3*8	KM*(36+JMM/8)
16	1/8	12+3*8	KM*(36+JMM/8)
17	Scalar	Covered	0
18	1/8	12+2*8	KM*(28+JMM/8)
19	1/8	12+3*8	KM*(36+JMM/8)
20	Scalar	Covered	0
21	1/8	12+2*8	KM*(28+JMM/8)
22	1/8	12+3*8	KM*(36+JMM/8)
23	Scalar	Covered	0
24	1/8	12+2*8	KM*(28+JMM/8)
25	1/8	12+2*8	KM*(28+JMM/8)
26	1/8	12+3*8	KM*(36+JMM/8)
27	1/8	12+3*8	KM*(36+JMM/8)
28	1/8	12+2*8	KM*(28+JMM/8)
29	1/8	12+2*8	KM*(28+JMM/8)
30	1/8	12+2*8	KM*(28+JMM/8)
31	1/8	12+2*8	KM*(28+JMM/8)
		64-Bit Total	7+KM*(712+26*JMM/8)
		32-Bit Total	7+KM*(712+26*JMM/16)

Table 5-38. NSS Timing: VVISMAT, Loop 30

Line Number	Cycles Per Result Generated	Cycles Required for Startup and Rundown	Cycles Required to Execute This Line
1	Scalar	Scalar	1
2	Scalar	Scalar	1
3	Scalar	Scalar	1
4	Scalar	Scalar	1+22
5	1/8	12+1*8	KMM*12(20+JMM/8)
6	1/8	12+1*8	KMM*12(20+JMM/8)
7	1/8	12+1*8	KMM*12(20+JMM/8)
8	1/8	12+1*8	KMM*12(20+JMM/8)
9	Scalar	Scalar	0
10	Scalar	Scalar	0
		64-Bit Total	26+48*KMM(20+JMM/8)
		32-Bit Total	26+48*KMM(20+JMM/16)

Table 5-39. Specimen Times for NSS: VVISMAT

JMAX	KMAX	Operations OP	Cycles 64	Cycles 32	Rate 64	Rate 32
100	50	727,944	150,594	125,004	483	582
128	64	1,204,056	212,740	170,357	566	707
200	80	2,373,624	327,874	244,302	724	972

Notes

OP = (JMAX-2)(152(KMAX-2)+132)

CY64 = 1493+50(JMAX-2)/8+(KMAX-2)(2040+86(JMAX-2)/8)

CY32 = 1493+50(JMAX-2)/16+(KMAX-2)(2040+86(JMAX-2)/16)

Table 5-40. Total Operation Count, 1 Time-Step, 64-Bit Mode

JMAXxKMAX	100x50	128x64	200x80
LOOP 10 VFILTRX	477,600	788,736	1,550,160
LOOP 10 VBTRI	2,174,620	3,603,852	7,114,140
LOOP 20 VFILTRY	1,236,956	2,043,720	4,025,736
LOOP 20 VBTRI	2,164,320	3,590,668	7,089,420
Total LHS	6,053,496	10,026,976	19,779,456
Total RHS	1,473,394	2,468,440	4,886,674
Total	7,526,890	12,495,416	24,666,130

Table 5-41. Total Cycle Count, 1 Time-Step, 64-Bit Mode

JMAXxKMAX	100x50	128x64	200x80
LOOP 10 VFILTRX	194,896	278,245	486,421
LOOP 10 VBTRI	451,921	611,818	1,014,984
LOOP 20 VFILTRY	104,730	136,573	177,295
LOOP 20 VBTRI	266,746	374,730	572,604
Total LHS	1,018,293	1,401,366	2,251,304
Total RHS	120,346	195,055	354,718
Total	1,138,639	1,596,421	2,606,022

Table 5-42. Rates - In Results Per Microsecond (10 ns Clock), 64-Bit Mode

JMAXxKMAX	100x50	128x64	200x80
LHS Rate	594	716	879
RHS Rate	1224	1266	1378
Overall Rate	661	783	946
		32-Bit Mode	
LHS Rate	741	1534	1755
RHS Rate	2157	2040	2486
Overall Rate	957	1745	1918

RISK ANALYSIS

A major concern of any knowledgeable computer-oriented person when hearing that another behemoth is about to be designed and developed is the probability of it being successful. In previous sections, the potential risks of such a project have been discussed from one direction. It might be well to review, however, these discussions in summary form.

HARDWARE RISKS

Various elements of hardware risks should be noted.

- Front-end machines

  Off-the-shelf equipment currently available is adequate to the task; thus, risks are near zero.

- Network trunk

  Examples of the designated trunk, at the required bandwidths, are in operation today; although they are undergoing some redesign to incorporate improvements, there is no substantial change anticipated in architecture or technology during the NASF development period. Therefore, the risks are minimal.

- Peripheral subsystems

  Tapes, disks, printers, card readers, and archival storage media. The system design that has been presented here utilizes existing peripheral equipment for sizing, performance, and costing purposes. The elements identified in this area are adequate to the task at hand; thus, the risks of availability are near zero. However, the project should be prepared to utilize new, standard equipment in the 1980 to 1985 time-frame.

- Graphics subsystems

  This technological area is perhaps the most unstable of the technologies. During the past 3 years, many new developments in terminals and supporting hardware have emerged. It is Control Data's opinion that no graphics subsystem now in existence provides the features, performance, cost basis, and reliability that is required for the NASF. Thus, this element of the system must await graphics developments that have just begun, with the attendant moderate risk that an optimum facility will not be available (due to particular graphics products faltering during creation, manufacture, or use). At this point a risk factor of 0.2 must be assigned to this item, although existing systems could act as interim terminals until better equipment is available.

- The NSS mainframe

  This device is, of course, the key element of the entire NASF. Without it, there is no need for the remaining front-end and graphics systems. In order for the facility to be successful, this computational element must meet the performance and reliability goals established (refer to Design Objectives Document, section 4). The issues which must be addressed in this area are:

  Manufacturability. The performance requirements must not demand physical spacings, thermal paths, power, and mechanical designs which stress the limits of manufacturing processes, either in terms of technical factors such as manufacturing tolerances or in terms of human factors (for example, a device is buildable or repairable only by humans, and because of design flaws, a human can be expected to introduce ancillary failures in the process of building 50 percent of the time). For high performance systems, with their attendant power and parts densities, manufacturability is a major problem and should invoke a risk factor for the NASF of at least 0.1.

  Technology. The ability of the NSS proposed in this report to meet the performance and reliability goals rests on the evolution of a second-generation, high-performance LSI system. This evolution requires 2 years to yield confidence in silicon processing and packaging concepts. Therefore, a long gap lies ahead, during which only extant technology is available. At this point the risk of such a technology coming to fruition with the required performance is perhaps 0.5. However, the NSS could withstand as much as 50-percent degradation in those performance goals for the LSI family and still produce solutions in 15 minutes given by Ames personnel as an upper limit. Thus, one could make rough estimates of risk to performance as follows:

Run Time	Risk	Methodology
30 minutes	.00	STAR-100C 4-pipe, standard product
15 minutes	.10	NSS 8-pipe, existing technology
13 minutes	.25	NSS 8-pipe, new mechanical package
10 minutes	.35	NSS 8-pipe, double-density chips
8 minutes	.40	NSS 8-pipe, double-density, 20-percent performance improvement
5 minutes	.60	NSS 8-pipe, double-density, 400-ps gate

Design. The design of the NSS could impose two risks on the success of the project. One risk is the parts count required to implement the design (thus impacting reliability), and the other risk is the possibility that the design made to match an existing performance metric will not meet the demands of the actual production codes. The first risk can be reduced to zero by fiat, by prohibiting any design techniques that exceed a given reliability figure (as stated in the Design Objectives Document). The second risk is more difficult to assess. Because of its presence, however, an attempt has been made to overextrapolate the existing metrics to the expected three-dimensional form. Based on this the estimated resultant risk is 0.2

Overall it is expected that the risk of failing to build, operate, and maintain a hardware facility that meets the NASF objectives is well below 0.5.

SOFTWARE RISKS

Certain elements of software risk should be noted.

- Compiler

  The risks involved in this effort are:

  The specified language is difficult to use and inhibits optimum usage of the facility.

  The specified language results in a compiler too complex to ensure a stable operating environment.

  The resources needed to compile, file, load, and execute the resultant language absorb too much of those available in the front-end system.

  These risks are well known from many years of development, and with proper project management should have no impact on the NASF development.

- Operating system

  By using the entire software available with off-the-shelf front-end gear, it should be possible to limit (but not quite eliminate) the development of a customized operating system. The NSS described in this proposal operates in a very straightforward manner, with one job running at a time and no I/O from the running job. Such factors must simplify the NSS operating system. The largest risk then appears to be in the development of software to tie front-end, back-end, and supporting subsystems together in a manner which makes the facility efficient and highly user-oriented. The risk of this not being achieved in the first implementation (not to mention the most highly visible phase) is believed to be quite high at 0.3.

SCHEDULE RISKS

If a firm end-date for having a completely operational facility is set for mid-1982, the amount of work and elapsed time required to develop the installation makes this effort risky if the start of the final development phase is delayed much beyond the first quarter of 1979. Even with this starting time, the risk of meeting the mid-1982 date is about 0.5. The greatest element of risk will be the software development. To provide the large number of facilities required, even with the assistance of already existing software, considerable time must elapse before a suitable, stable system can be offered to paying customers. The earliest possible start of the software development (and/or experimentation) will increase the probability of providing production services in mid-1982.

Earlier efforts on software production and checkout can be aided through the use of remote, time-sharing lease of computer resources from vendor computer centers housing machines targeted as the front-end processor. Further, early releases of software without all of the specified features could allow for broad exposure and more extensive exercise of critical components. Both aspects lend to the stability of the software system.

CONCLUSION

The conclusion is that the hardware system can be assembled with a moderate risk of not achieving its performance and reliability goals. The probability of meeting the stated schedule at this time appears to be a 50-50 proposition.

The NSS processor contains no design features which prohibit its use in any other general-purpose scientific environment. However, it should be expected that the NSS will not yield its optimum performance levels when operating in the general-purpose environment due to tradeoffs on buffer sizes and overlap made specifically for the flow model simulation codes. The NSS should process general-purpose codes at speeds equal to or greater than any other general-purpose computer of the super class in 1982.

NASF SCHEDULE

The following NASF development schedule gives a recommended schedule for the development of the complete NASF. No actual dates are given; all times are relative to a fixed end-date, which will be dictated somewhat by the time required for the various procurement cycles, final negotiated figures for configuration, and software development. Time intervals are shown in months, and major time intervals are in 6-month increments stated in terms of the final date (M-Month) minus so many months.

The major milestones are:

1. Complete phase I study

   This milestone marks the point at which both the contractor and NASA personnel are in agreement on the conclusions, recommendations, and data derived from the initial study phase for the NASF.

2. Complete phase II study

   Work products from phase II efforts are called out in the following separate milestones

   7,12,15,19,22,25,29,33,37, and 41

   This phase will result also in the construction of detailed block diagrams and a block model on which any selected portions of the flow simulation code can be executed for at least one cycle. The output of this simulation will be actual result data to check the integrity of the design and detailed timing of all data flow and control.

3. Complete detailed design of the NSS hardware

   Using the detailed block model and its attendant simulation, the entire NSS will be designed, LSI arrays will be defined and laid out, and the total ensemble will be simulated with the same selected metrics used in phase II. The results of this effort will be a set of magnetic tapes which contain the LSI array designs and LSI board routing information which can be directly applied by vendors to produce the actual parts. In addition, full mechanical, power, and cooling blueprints will be produced for the full machine.

4. Complete NSS assembly

   This phase involves the ordering, manufacture, and assembly of the component pieces of the NSS. To provide reasonable probabilities of meeting schedule dates, it is recommended that only 32 million words of backing store and 2 million words of main memory be assembled by this date to commence final checkout. The additional components may be assembled and added onto the NSS over the 6 months following this milestone.

5. Complete NSS checkout (with diagnostics) .

   At this point all diagnostics will have been run with a range of operating voltage and clock margins (to be identified in phase II). In addition, the performance metric codes will have been executed successfully and reliably for some continuous operating period to verify performance and availability of the standalone NSS. At this point, the memory system will be completely populated and exercised.

6. NSS system integration

   This milestone identifies the time when the entire NASF, NSS, peripherals, front-end, and graphics processors are integrated and a total user application run through the system. For this purpose, a system metric will be defined during phase II.

7. Complete front-end specification

   A detailed specification of the functional, physical, and performance requirements of the front-end system will be the work product of this phase. The format and content of this document must be sufficient to permit the procurement of such a system from the specifications to proceed immediately. Further, whatever subset functions can be identified for initial and subsequent phased deliveries of components must be described in separate documents.

8. Front-end services available

   One method of obtaining the use of the front-end system for software development and configuration experiments is to utilize the data center capability of a given vendor. Thus, through remote terminals and batch entry stations, NASA can begin their critical software developments earlier than a normal front-end procurement would allow.

9. Deliver basic front-end system

   At this point, the NASF system development needs the marriage of specialized peripherals and graphics capabilities not normally furnished by standard vendor-owned data center operations. A basic front-end processor must then be delivered to NASA, along with a minimum set of network, graphics, and mass storage components. It is possible that a unique configuration of these hardware items might be assembled at the vendor's site for the exclusive use of NASA developers, who could continue to operate from remote terminals.

10. Complete the front-end system

   An entire NASF front-end system must be assembled with all peripherals and graphics subsystems intact to continue software development. At this point, at least one of each type of equipment must be attached and operational. Additional equipment of each type can be phased in during the remaining year before system integration.

11. Front-end system integration

    A total integration of front-end and NSS as described in milestone 6.

12. Complete peripheral specifications

    As in milestone 7, a detailed specification of the peripheral subsystem will be provided so that programming and procurement may begin at that time.

13. Deliver minimum peripheral system

    As identified in phase II, a basic set of peripherals to support the front-end software development, as well as checkout and verification of the NSS, must be available at this time.

14. Deliver and integrate full peripheral configuration

    All equipment given in the specification, in their full configurations, will be integrated at this time.

15. Complete network specification

    As in milestones 7 and 12, a detailed specification will be produced to permit the initiation of programming and procurement of the network trunk subsystem.

16. Debug network available

    Since the network trunk concept is a significant departure from the input/output schemes that have been traditionally used in facilities like the NASF, certain programming and system considerations will have to be examined and tested in a real environment. It is essential and possible to assemble a minimum trunk system and check out the related software before the front-end processor is needed and before the NSS can be interfaced.

17. Deliver basic network

    Since the I/O coupling of peripherals, graphics, and front-ends is going to be via the trunk network system, a minimum version must be available to connect and support the basic front-end, minimum peripherals, and basic graphics systems specified in phase II.

18. Deliver and integrate full network

    This milestone corresponds to total system integration milestone for all components.

19. Complete graphics specification

    As in milestones 7, 12, and 15, a detailed specification for graphics hardware and software will be produced in phase II to permit initiation of programming and procurement.

20. Basic graphics terminal system availability

With the availability of a basic front-end, peripherals, and network system, at least one low performance and one high performance graphics subsystem should be delivered to commence system checkout.

21. Deliver and integrate full graphics subsystem

This milestone identifies total system integration of full configurations of terminals and graphics processors.

22. User language specification available

Although preliminary specifications of the user language will have evolved throughout phase I and II of this study, a final, approved specification of the applications programming language will be available at this time so that the broad user community may begin programming of the production, application codes.

23. Complete preliminary applications programming

At this point the preliminary programming of user codes can be submitted to the early versions of the compiler (which will run on the front-end processor). Hand-compiled segments can also be run through the simulator to verify performance and to aid in the development of optimum code sequences.

24. Complete checkout of applications programs

Those programs identified during phase II should be fully operational.

25. Complete user language compiler specification

A detailed specification !internal maintenance specification (IMS) in Control Data Standards" will be produced, identifying all functional characteristics of the compiler, and in particular, the code sequences to be generated for constructs of particular interest to the flow simulation programmers.

26. User compiler available for checkout

A preliminary version of the user language compiler will be available at this time. Emphasis will be on reliable code generation perhaps at the expense of some less significant features.

27. Complete user language compiler

A final version of the user language compiler will be frozen at this point to permit an extended period of exercise by programmers. Compiler stability is a crucial part of the success of the NASF in the near term.

28. Integrate compiler

This is the system integration milestone for all compiler components.

29. Complete monitor language specification

Despite its small size and limited functional requirements, the NSS monitor system will require some form of programming language. The need for minimal investment and quick implementation will drive the generation of a specification which will be produced at the end of phase II.

30. Monitor programming complete

Using the language specification and a breakdown of functional modules required by phase II studies, the monitor for the NSS will be programmed and documented by this time.

31. Complete diagnostic monitor checkout

Once the NSS machine is assembled and diagnsotics are operational, the monitor system can be added, and higher-level system diagnostics run with it (for example, on-line and off-line monitor and job mode operation).

32. Complete monitor checkout

The monitor is not complete until the integrated system is checked out.

33. Complete monitor compiler specification

As with the user language compiler specification, a detailed document will be produced as a conclusion to phase II giving the structure and functional behavior of the monitor language compiler, including key code constructs that must be dealt with, and the mapping between source language and machine code.

34. Preliminary monitor compiler available

An initial release of the monitor compiler will be available to monitor programmers for testing of coding and verifying compiler features.

35. Complete final monitor compiler

This version of the compiler must be frozen at this point so that all other interfacing parts of the system can count on its behavior being fixed during the remainder of system checkout.

36. Complete monitor compiler integration

This is in step with the total system integration. In fact, the monitor compiler will need to be fully operational before this time; however, the availability of the total system will perhaps change the means of using the compiler (even to use of interactive graphics, perhaps) and the location of data base and file system may change the execution characteristics of the compiler at this time.

37. Diagnostics specification

Phase II will yield a detailed set of specifications for diagnostics for the NSS and the network trunk adaptation scheme since these are the nonstandard portions of the NASF. These diagnostics will involve microcode and machine code routines to perform reliability verification, fault detection, and fault isolation.

38. Complete diagnostic programming

Using the monitor language system, the diagnostics will be programmed for the NSS. Using the front-end system, the network diagnostics will be programmed in the micro and macro code systems already available.

39. Complete diagnostics checkout

All diagnostic programs are fully operational at this point.

40. Diagnsotic system integration

On-line maintenance aids and diagnostics will be integrated and available at this time.

41. Complete front-end software specifications

A limited number of software functions will need to be programmed for the front-end systems. A specification of these functions, as well as standard off-the-shelf facilities, will be detailed in a specification which will be used to direct the procurement and contractual arrangements necessary to acquire the use of the standard versions. In addition, the specification will be used to direct the production of the unique entities needed for the NASF.

42. Complete front-end programming

Those system functions which must be produced specifically for the NASF will be programmed in one of the standard languages available on the front-end systems.

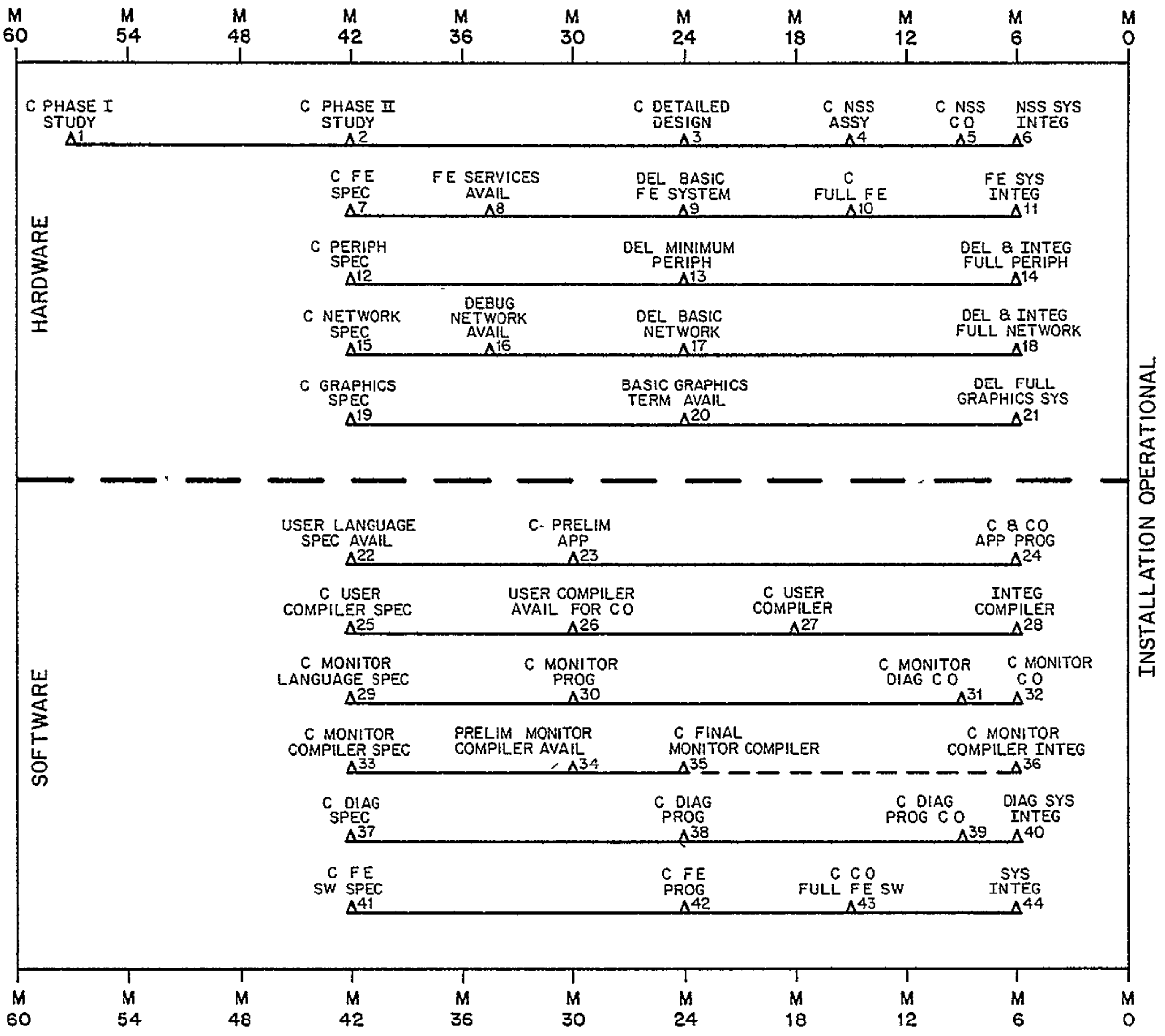
43. Complete checkout full front-end software

The unique NASF programs and the standard programs must be checked out in the total front-end environment with attached networks, peripherals, and graphics subsystems.

44. System integration of front-end software

The complete front-end software will be integrated with the complete system.

As in any large program, the schedules given above are somewhat general. More details will be provided in subsequent phases of the program.


M 60
M 54
M 48
M 42
M 36
M 30
M 24
M 18
M 12
M 6
M 0
HARDWARE
SOFTWARE
INSTALLATION OPERATIONAL
C PHASE I STUDY 1
C PHASE II STUDY 2
C DETAILED DESIGN 3
C NSS ASSY 4
C NSS C O 5
NSS SYS INTEG 6
C F E SPEC 7
F E SERVICES AVAIL 8
DEL BASIC F E SYSTEM 9
C FULL F E 10
F E SYS INTEG 11
C PERIPH SPEC 12
DEL MINIMUM PERIPH 13
DEL & INTEG FULL PERIPH 14
C NETWORK SPEC 15
DEBUG NETWORK AVAIL 16
DEL BASIC NETWORK 17
DEL & INTEG FULL NETWORK 18
C GRAPHICS SPEC 19
BASIC GRAPHICS TERM AVAIL 20
DEL FULL GRAPHICS SYS 21
USER LANGUAGE SPEC AVAIL 22
C- PRELIM APP 23
C & CO APP PROG 24
C USER COMPILER SPEC 25
USER COMPILER AVAIL FOR C O 26
C USER COMPILER 27
INTEG COMPILER 28
C MONITOR LANGUAGE SPEC 29
C MONITOR PROG 30
C MONITOR DIAG C O 31
C MONITOR C O 32
C MONITOR COMPILER SPEC 33
PRELIM MONITOR COMPILER AVAIL 34
C FINAL MONITOR COMPILER 35
C MONITOR COMPILER INTEG 36
C DIAG SPEC 37
C DIAG PROG 38
C DIAG PROG C O 39
DIAG SYS INTEG 40
C F E SW SPEC 41
C F E PROG 42
C C O FULL F E SW 43
SYS INTEG 44

COSTS

Basically, costs associated with a program such as the NASF program, the subject of this study, fall into three general categories: development, acquisition, and operating. Development cost is incurred primarily during the early part of a program for study, research, design, and development. Acquisition cost is that associated with the actual fabrication, and/or procurement, and the installation. Operating and support costs are recurring, or on-going, and are associated with the installed and operational facility.

The latter of these is estimated in section 6 of this report. Subsequent phases of this program will incurr development and acquisition costs, and preliminary study estimates are provided in tables 5-43 and 5-44 for sizing purposes.

TABLE 5-43. ESTIMATED DEVELOPMENT COSTS

Phase	Estimated Cost (in millions)
Phase II	0.3 to 0.5
Phase III	3.3

TABLE 5-44. ESTIMATED ACQUISITION COSTS - PHASE IV

Item	Estimated Cost (in millions)
NSS, including main memory, backing storage, I/O	26.6
Disk storage	4.1
Archival storage	2.3
Front-end CPU, including main memory, I/O	3.3
Card equipment	0.2
Printers	0.5
Magnetic tape	0.2
Graphics communications	0.6
Correspondence communcations	0.2
Software acquisition and development support of total acquisition	2.8
Total Estimated Acquisition Cost	40.8 Million

APPENDIX A
NETWORK MESSAGE PROTOCOL

TRUNK FRAME FORMAT

The trunk frame format is defined independent of the hardware used for transmitting or receiving the frame.

FLAG	TO	FUN	←VLF→	CRC1	CRC2	FLAG

A frame is a set of 8-bit bytes. The various bytes are:

FLAG	Flag bytes delimit the frame. There may be more than one flag between frames.
TO	The destination address.
FUN	The function byte. This byte defines the functional meaning of the frame.
VLF	Variable length field. Any number of bytes, including zero.
CRC1	Cyclic redundancy check code. Upper eight bits of 16-bit CRC.
CRC2	Cyclic redundancy check code. Lower eight bits of 16-bit CRC.

TCU FRAME FORMAT - RECEIVED

This definition applies to all frames (control message frames and response message frames) received by a transmission control unit.

FLAG	TO	FUN	FROM	PAR	ODF	CRC1	CRC2	FLAG

FLAG	Same as above.
TO	Same as above.

FUN	Same as above.
FROM	The address of the originator of the frame.
PAR	A parameter/status byte.
ODF	Optional data field. Any length, including zero.
CRC1	Same as above.
CRC2	Same as above.

A TCU will accept all trunk frames which:

- Have six or more bytes (not including the flags). A frame of less than six bytes will be ignored.
- Are addressed TO this TCU.

The third byte of the frame, the FROM byte, is saved in a TCU register. If the message consists of more than one frame, then this register will contain the FROM byte of the final frame.

TCU FRAME FORMAT - TRANSMITTED

BUS INTERFACE - CONTROL MESSAGE OR RESPONSE MESSAGE FRAME

```
| TO | FUN | V | V | ODF |
```

A minimum of four bytes must be presented to the TCU. The V (variable) bytes are defined by the unit to which this TCU is transmitting. If the frame is being sent to another TCU, then the V bytes represent FROM and PAR (as previously défined).

TRUNK INTERFACE

FLAG	TO	FUN	FROM	PAR	ODF	CRC1	CRC2	FLAG

Control Message Frame

The four bytes presented at the bus interface become the first four bytes of the transmitted frame (TO, FUN, FROM, PAR). There may be additional bytes (ODF).

Response Message Frame

Although the initial four bytes are presented to the TCU at the bus interface, the TO and FROM bytes impressed on the trunk originate within the TCU. The TO byte is derived from the holding register mentioned previously. The FROM byte is derived from the address switches of the TCU.

In addition, the lower four bits of the FUN byte are accepted from the bus while the upper four bits of the FUN byte are generated within the TCU.

TCU RESPONSE CODES

One of these response codes is automatically inserted by the TCU when the initial response message frame is transmitted.

ACK	The control message was received correctly by the TCU and accepted by the TCU interface.
BUSY	The control message was received correctly by the TCU but not accepted by the TCU interface.
NAK	The control message was received correctly by the TCU but the TCU interface did not respond (it neither accepted nor rejected).

TCU FUNCTIONS

The TCU recognizes one function, ROTATE PRIORITY. This control message frame is not passed on to the TCU interface.

TCU FRAME SEQUENCING

The TCU recognizes two fixed orders (but not number) of frame movements. These orders are called the active mode and the passive mode.

ACTIVE MODE

In this mode, the TCU is selected to send a control message. By virtue of having transmitted a control message, the TCU conditions itself to receive a response message. The three states of this mode are·

- Send control message state
- Receive response message state
- Idle state

If a response message is not forthcoming within a fixed-time interval, the TCU presents a NO ANSWER status to the TCU interface.

PASSIVE MODE

The TCU is in the passive mode, waiting to receive a control message, whenever not in the active mode. The three states of this mode are:

- Receive control message state
- Send response message state
- Idle state

Having received a control message, the TCU then conditions itself to send a response message. After the response message has been sent, the TCU returns to the idle state.

Mode changes can only occur while the TCU is in the idle state.

REFERENCES

The following is a list of all references that are called out in the text of this section.

Reference Number	Reference
5-1	Iverson, Kenneth E.: A Programming Language. John Wiley and Sons, Inc., 1962, Chapter 1.
5-2	STAR FORTRAN LANGUAGE. Version 2 Reference Manual. Publication no. 60386200, Control Data Corporation, 1977.

SECTION 6
PRELIMINARY SITE REQUIREMENTS

INTRODUCTION

Consideration of an installation such as the Numerical Aerodynamic Simulation Facility (NASF) has many facets ranging from initial conception and study, through design and development, to subsequent installation, operation, and support. Phase I of this program, covered by this report, has included only the initial study for this facility to determine feasibility of such a computational complex. Detailed design was not an objective of this phase, but was to be carried to a level such that an overall system could be defined, procurement and development costs estimated, and operational and support requirements sized.

The system described below has been configured around the NASF computational engine which has been the principal subject of this study. To the extent possible, existing standard products have been used as typical or representive of those required to obtain the desired system function. It is recognized that these are today's products whereas the proposed facility has an operational target date 5 years in the future. However, this is believed to provide the most accurate sizing of site requirements in this preliminary phase. As the program progresses, these standard products can be replaced with the most current, state-of-the-art devices available for the target date of installation.

The system components unique to, or customized for, this facility have been defined in preceding sections of this report. Estimates of requirements such as space, power, and cooling have been made based on the preliminary, high-level design and are included below in the total site requirements.

SYSTEM DESCRIPTION

The typical basic system required for the Numerical Aerodynamic Simulation Facility would be composed of a central processor, a hierarchy of storage devices, multiple mainframe front-end systems, associated peripherals, motor generators, and a refrigeration system. The basic components (products) are described below and include the associated cabinets in which the products are contained. Product descriptions and cabinet(s) are for single product only and not for quantity listed.

CENTRAL PROCESSOR

Quantity	Product	Cabinet(s)	Description
1	NSS	Housed in three cabinets	A CPU mainframe having 16 I/O data channels, 8 x 10^6 words of 40-ns memory, and 256 x 10^6 words of secondary storage (backing storage)

STORAGE SUBSYSTEMS

Quantity	Product	Cabinet(s)	Description
32	819-X	Stand-alone cabinet	Disk storage unit. Capacity of 4.8 x 10^9 bits of data organized into four-bit parallel bytes; transfer rate of 38.7 million bps.
2	844-21	Stand-alone cabinet	Disk storage unit. Capacity of 869 x 10^6 bits when used in an unsectored format or 712 x 10^6 bits with 24 sectors per track, 404 tracks, 6.8 million bps transfer rate.
1	7154-4	Stand-alone cabinet	Mass storage controller. Permits four-channel interface from one to eight 844-XX disk units.

Quantity	Product	Cabinet(s)	Description
4	7639-2	Stand-alone cabinet	Mass storage controller. Contains two independent controllers in a single cabinet. Each controller allows time-shared access from four channels up to eight 819 disk storage units.
2	38501-1	Stand-alone cabinet	Consists of the mass storage coupler (MSC) and the mass storage adapter (MSA): MSC Interfaces between two CYBER channels and one MSA; MSA Interfaces between the MSC and up to eight devices (where a device is either a CSU or MST)
2	69016	Mounts in the 38501-1	16-device option CSU
8	69006	Mounts in the 38501-1	Dual-path option
8	38510-16	Stand-alone unit with MSTs physically attached	Mass storage file. Consists of the cartridge storage unit (CSU) and two mass storage transports (MST).

FRONT-END SYSTEMS

Quantity	Product	Cabinet(s)	Description
2	CYBER 173-8	Housed in a single bay	Central computer with central processor, memory control, 131 kwords of central memory, 10 peripheral processors, two data channel converters, display console/controller, power supplies, and one 3-ton condensing unit (mounted internally in the mainframe bay).

Quantity	Product	Cabinet(s)	Description
2	10318-1	Mounts in CDC CYBER 173 bay	ECS coupler. Allows CDC CYBER 173 to interface to the ECS subsystem.
1	7030-104	Single stand-alone cabinet	Extended core storage (ECS). Provides 524 kwords of magnetic core storage.
2	405	Stand-alone cabinet	Card reader. Reads 1200 cpm (80 column cards); 1600 cpm (51 column cards), 4000-card hopper, 4000-card stacker, 240-card reject stacker.
2	3447	Mounts in the 405 card reader	Card reader controller. Permits direct connection between 405 card reader and one standard channel; full card buffer; BCD conversion, data checking.
1	415-30	Stand-alone cabinet	Card punch with controller. Punches 250 cpm (80-column cards) programmable offset stacking, 1200-card hopper, 1500-card stacker, read check after punch, 80 12-bit word buffer memory.
4	580-20	Stand-alone cabinet	Line printer. Prints 2000 lpm with 48-character train, 135 columns with 6- or 8-lpi spacing includes power stacker, controller with one-line buffer, train image storage and error checking.
2	667-4	Stand-alone cabinet	Magnetic tape transports. 7-track; 556- and 800-cpi NRZI recording; 111.2K and 160K 6-bit characters per second transfer rates; 200-ips tape speed; forward and reverse read; 45-second maximum rewind time. Read-only capability at 200 cpi.

Quantity	Product	Cabinet(s)	Description
2	669-4	Stand-alone cabinet	Magnetic tape transport. 9-track; 800-cpi NRZI recording and 1600-cpi phase-encoded recording, 160K and 320K 8-bit characters per second transfer rates; 200-ips tape speed, forward and reverse read; 45-second maximum rewind time.
1	7021-22	Stand-alone cabinet	Magnetic tape controller. Provides single channel connection to two controls; permits simultaneous read/write on any two of eight 66X intermixed tape units.
1	777-2	Display console cabinet, card reader/controller cabinet, mainframe, and register display cabinet	Interactive graphics terminal. A job entry terminal with display console, controller, 300-cpm card reader/controller and synchronous communications interface.
1	751-10	Desk top mounted stand-alone unit	Interactive correspondence terminal. 24 lines, 80 characters per line display, 128-ASCII character set, TTY compatible, display data editing, and highlight features.
2	2550-2	Stand-alone dual-bay cabinet	Host communications processor (network processing unit). Includes communications processor, 32K 16-bit words of MOS memory, multiplier loop interface adapter communications coupler, loop multiplier maintenance panel, tape cassette, cyclic encoder, and power supplies.
2	2550-100	Mounts in the 2550-2 cabinet	Emulation module. Provides hardware and controlware necessary to emulate one to four 6671/6676 data set controllers. Hardware/controlware together accommodate data interchange between one CDC CYBER 170 data channel and 128 modems.

Quantity	Product	Cabinet(s)	Description
2	2558-1	Mounts in the 2550-2 cabinet	Communications coupler. Provides a bidirectional interface between the 2550-2 HCP and the CDC CYBER 170 computer.
2	2558-2	Mounts in the 2550-2 cabinet	Emulation coupler. Provides the second 667X emulation coupler for use with 2550-100 in dual-coupler mode.
	2560-X	Cables connecting between the 2550-2 and modems	Communications line adapter. Provides software selection and control of half- or full-duplex 5-, 6-, 7-, or 8-bit code lengths, frame synchronization, and loop-back capabilities.

MOTOR-GENERATORS

Quantity	Product	Cabinet(s)	Description
4		Stand-alone unit	125-kVA motor generators. Operates on 460-volt, ac, 3-phase, 60-Hz input. Provides 208-volt, ac, 3-phase, 400-Hz output.
4		Stand-alone cabinet	125-kVA motor generator control cabinet. Provides input/output power connections and control for the MG.

REFRIGERATION SYSTEM

Quantity	Product	Cabinet(s)	Description
4		Stand-alone cabinet	30-ton condensing unit. Provides up to 30-ton of R-12 refrigerant cooling for the NSS computer. Requires closed-loop chilled water system.

CONFIGURATION

Physical specifications (dimensions, power, heat dissipation, environmental, and floor space requirements) are provided in appendixes A, B, and C at the end of this section.

• Machine unit specifications (MUS)	A computer printout, for quick reference, which indicates unit dimensions, weight, heat dissipation, required input power breakers and total floor space occupied by system. Refer to appendix A.
• Equipment data sheets	Individual unit (cabinet) specification sheets which indicate dimensions, weight, heat dissipation, power, power connection type and placement, logic cabling, type of cooling, and environmental requirements for operation and storage. The data sheets also provide photo and equipment layout sketches showing raised floor cabling access cutouts and door-swing clearances. Refer to appendix B.
• Floor plan layout	A typical system layout (within the logic/refrigeration cabling/hose restrictions). This layout is to a scale of 1/4 inch equals 1 foot. Refer to appendix C.

ACCOMMODATION REQUIREMENTS

Due to the complexity of the computer systems, a specially constructed environmentally controlled computer room is required.

Construction of the computer room and installation of its facilities and utilities should comply with applicable building and service codes. Additionally, the site must meet the requirements of the National Fire Protective Association Standard 75 (Protection of Computer/Electronic Data Processing Installation).

Because of the investment a computer system represents, the customer may desire the use of professional consulting services for various areas of site design. These services are extremely important in helping provide a smooth running computer installation.

Control Data provides qualified services in site design, administration of contracts, and supervision of the computer site construction (room or entire building). This work is done on a contractual basis by the Control Data Computer Facility Planning and Construction Division.

FLOORS

In order to accommodate the computer condensing unit's chilled water piping, refrigeration manifolds, piping and hoses, system power, and logic cabling, the system requires the installation of a raised floor. Several types of raised floors will work; however, a metallic pedestal and stringer type has the advantage of serving as an electro-magnetic compatibility (EMC) grid-ground reference plane (which is also a requirement for the computer room).

Floor Clearance

Due to the refrigeration manifold and hose requirements, the raised floor must have a minimum clearance of 0.3 m (12 inches) from the bottom of the support stringer to the surface of the building floor.

If the installation is made in an existing building, one or more ramps should be built between the building floor and the surface of the raised floor. Position the ramps at the room entrances to permit the ease of movement of equipment and carts to/from the computer area.

NOTE

If the computer room is to be in a building under construction, install the building computer room floor lower than the normal building floor. The raised floor installation then can be at the same height as the normal building floor, thus eliminating the need for ramps.

Raised Floor Requirements

In order to accommodate the weight and installation movement of the computing equipment, the raised floor must meet the following requirements.

Panel Size	Maximum of 0.61 x 0.61 m (24 x 24 in).
Construction	Metal or metal-clad wood panels.
Surface covering	The surface of the raised floor panels may be either tiled or carpeted (each has advantages).
Support capability	The raised floor and panels must be capable of supporting 1220 kg/m^2 (250 lb/ft^2) evenly distributed with 454 kg (1000 lb) point concentrations in 5.08 cm (2 in) diameter areas.

CEILING

The recommended ceiling height is 2.74 m (9 ft) (ceiling surface to the top surface of the raised floor). Minimum height is 2.59 m (8.5 ft).

The room may have a natural or false (dropped) ceiling, however, it must meet the ceiling height requirements just mentioned.

Minimum clearance to the ceiling from equipment which exhausts through the top of the cabinet is 0.61 m (2 ft).

EQUIPMENT CLEARANCES

Equipment door swings are shown on the accompanying floor plan and also on the individual equipment data sheets. Refer to appendix B of this section.

Access clearance for maintenance on the mainframes is 1.22 m (4 ft) on all sides.

Worst-case units for crated access clearances and unloading requirements are:

Largest overall unit is the CDC CYBER 173 bay.

Width: 2.44 m (96.2 in)

Depth: 0.89 m (35 in)

Height: 2.03 m (79.8 in)

Add 2.54 cm (1 in) to depth and width for crated clearance.

Add 5.08 cm (2 in) to height to accommodate Rol-a-lift.

Heaviest unit is the 125-kVA motor generator.

Weight: 1848 kg (4075 lb)

Highest unit is the CDC CYBER 173 bay.

Height: 2.03 m (79.8 in)

LIGHTING

The computer room should have a minimum of 583 lux (50 foot candles). Avoid excessive illumination to enable proper visibility of lighted equipment indicators.

Avoid the use of spot lights, flood lights, and/or direct sunlight.

ENVIRONMENTAL

Both the temperature and humidity must be controlled in the computer room.

The NSS, CDC CYBER 173, and ECS bays utilize refrigerant cooling techniques which require strict control of the room dew point in order to avoid condensation which could possibly cause damage to the equipment and/or its associated logic.

Dew Point

Control Data furnishes a dew-point recorder with the system. The dew point must be maintained below 13.3°C (56°F) or the system will automatically shut down.

NOTE

> The temperature monitor and power control (TMPC) panel furnished with the CDC CYBER 173 is designed to provide visual and audible alarms for the dew point. At 12.2°C (54°F) (dew point warning), a visual indication and audible alarm sound; at 13.3°C (56°F) (dew point danger), visual and audible indications are given, and the TMPC initiates a sequential power-down 135 seconds after receipt of the danger signal from the dew point recorder.

Temperature and Humidity

The temperature and humidity ranges for the various equipments are indicated on the respective equipment data sheets.

Design considerations for the computer installation are based upon the following room conditions.

Temperature: 22.2°C (72°F)

Relative Humidity: 50 percent [15.6°C (60.1°F) Wet Bulb]

ADDITIONAL INFORMATION

Additional information on the site requirements such as raised floor leveling, floor loading due to weights of raised floor structure, cables, and so on, and signal and power cable routing can be found in the Section 1, Large and Medium Scale Computer Systems Site Preparation Manual. Refer to this publication in appendix D of this section.

POWER AND COOLING REQUIREMENTS

ELECTRICAL REQUIREMENTS

The system motor-generators (MG) operate on 460 volts, ac, 3-phase, 60-Hz input. (The MGs provide a 208-volt, ac, 3-phase, 400-Hz output to the computers and certain peripherals.)

The 30-ton condensing units require 460-Vac, 3-phase, 60-Hz input.

Other equipment requires either 120-Vac, 1-phase, 60-Hz input or 208-Vac, 3-phase, 60-Hz input. (Refer to the individual equipment data sheets for specific details.)

Main Supply Variations

To the System MG (460-vac, 3-phase, 60-Hz)

Frequency. ±5 percent of nominal

Undervoltage (instantaneous/transients)·

Up to 200 ms (maximum) down to zero volts

Overvoltage (instantaneous/transients):

- Up to 150 percent of nominal for two to three cycles
- Up to 120 percent of nominal for 3 to 5 minutes
- Up to 110 percent of nominal continuous

Long-Term Deviation:

- ±10 percent continuous.
- 80 percent of rated nominal for up to 15 minutes at full load
- Refer to the following chart for deviations at less than full load. (At 130 percent current, do not exceed 15 minutes).

MG Input Current at Reduced Voltage

Percent Rated Volts	Percent Rated Current at Various Loads			
	4/4 Load	3/4 Load	1/2 Load	1/4 Load
100	100	77	57	39
90	111	85	62	40
80	132	96	68	41
70		115	78	45
60		160	95	51
50				63
40				98

To Other System Components

Frequency deviation. 59 Hz to 60.6 Hz (maximum range)

Voltage deviation:

- Long term +5 percent, -10 percent continuous
- Instantaneous/transient; +5 percent, -20 percent for up to 30 ms.

Power Factors

The power factor for 400-Hz equipment is 1.0. Sixty-Hz units utilize blowers, drive motors, and so on, and have power factors in the range of 0.7 to 0.8.

NOTE

The heat dissipation figures on the equipment data sheets have the power factors calculated into kcal/hr and Btu/hr.

Grounding Requirements

Other than the normal safety (green wire) ground, the system also requires an EMC grid ground. The details of the required EMC grid ground network are set forth in part 7 of the Section 1, Large and Medium Scale Computer Systems Site Preparation Manual. (Refer to this publication in appendix D of this section).

In addition to the power and grounding requirements set forth in the site preparation manual, the following grounding and shielding is required for the CDC CYBER 170 installation.

All power and control cable runs must be shielded, and the shielding must be grounded at both ends of the run. The following types of shielding are acceptable:

- Totally enclosed conductive busways
- Steel thin-walled metallic conduit
- Flexible metallic conduit
- Braided metallic-shielded jacketing
- Zip-on flexible metallic-shielded jacketing (for example, Zippertubing)

NOTE

The shielding on all power and control cable runs must be grounded at each end of the run.

For stationary cabinets (mainframe and peripheral units not on casters), the shielding should be run up into the cabinet. If the shielding used is rigid, a short length of flexible shielding (if local codes permit) should be used to connect from the rigid shielding at a point below the surface of the raised floor, up into the cabinet. Where flexible shielding is used, it should be run directly up into the cabinet.

For movable peripherals (those on casters) having a drop cable, the shielding should terminate at a junction box located below the surface of the raised floor at a point immediately below the peripheral equipment floor cable cutout, or as close as possible. Local codes determine the exact placement of the junction box.

NOTE

> All signal cables attached to the mainframe must be shielded and grounded to the frame. If either logic cables or the movable type peripheral equipment drop cables are a source of EMI, they will be shielded by Control Data (on an as-needed basis) with a zip-on type shielding.

In order to keep the EMI at a minimum, signal cables, alternating current cables, and direct current cables must be routed in separate cable troughs.

Additionally, the following power shielding is required for the NSS. Power is supplied to the CPU at several locations. All power cables (50/60-Hz and 400-Hz) must be shielded. Shielding is necessary to protect the advanced hardware technology used in the NSS from electromagnetic and radio interferences possibly transmitted via power leads. Acceptable shielding is obtained by use of:

• Screened power cables	A minimum 90 percent of the outside running perimeter of the cable must be screened. The mesh screening must be conductive; copper is preferred, and aluminum or steel is acceptable.

• Rigid conduits	Normally used between circuit breaker panels and a junction box located under the elevated floor in the immediate vicinity of a cabinet.
• Flexible conduits	Normally used between a junction box under the elevated floor and the CDC-provided cabinet power termination.
• Enclosed Metallic Raceways	Normally used between the main power service stations, frequency converters, and related circuit breaker panels.

Power Distribution

All power and warning system wiring involved with energizing the computer system and associated peripheral equipment must conform to local codes and should be installed by the customer prior to delivery of the computer system.

Power distribution is designed for ease of installation and convenience of use. All main cabinets have terminal strips and/or junction boxes for all power connections.

It is the customer's responsibility for the provision and installation of all items related to each computer system power distribution as defined in the site preparation manual. The customer must supply and install all primary power source circuit breakers, panels, disconnects, ductwork, magnetic contactors, and all power cabling, including cabling required for signal cable terminator power distribution.

NOTE

CDC CYBER 170 systems utilize hot-gas bypass solenoid valves in the cooling apparatus. The solenoids require a 50/60-Hz power cable between TB-3, pins 1 and 2 of the power cable between TB-3, pins 1 and 2 of the power disconnect panel and TB-1, pins 5 and 6 of the compressor for each bay in the system.

Power cables must extend 0.61 m (24 in)† above the surface of the raised floor. Insulation should be stripped back 15.2 cm (6 in) on all cables and 1.27 cm (0.5 in) on individual wires.

Within each computer (controller and/or peripherals included), individual power supplies are protected by a power supply input circuit breaker (and/or fuse). In certain cases such as mainframes, the input power to the power distribution panel is also fused prior to distribution to individual power supplies.

Additionally, the system's power control monitor panels ensure proper power-down of the system in case of line failures.

These safeguards protect the equipment from any damage due to normal power loss circumstances.

In addition, the temperature monitor panels also monitor system condensing units and MG power status and will cause system shutdown in case of failure to those components.

Control Data's site liability is limited to 1 year after acceptance in those cases where Control Data designs and constructs the site. No liability is accepted for sites not designed and constructed by Control Data.

It is recommended that all mainframes, MGs, and condensing units be kept running at all times. (Powering on/off tends to create thermal-shock to the logic elements and greatly reduces life expectancy and reliability of these elements.)

Most peripherals and their controllers may be powered off as necessary, however, certain rotating storage-devices such as large disk files should be left on to maintain the clean air status of these units.

†1.22 m (48 in) for the NSS.

EQUIPMENT COOLING REQUIREMENTS

The various cabinets in the computer system are cooled by means of refrigerant, air, convection, or combinations of these.

Refrigerant-Cooled Equipment

The refrigerant-cooled equipment (cabinets) utilizes condensing units which circulate R-12 refrigerant through the equipment cabinets and chassis. Approximately 80 percent of the heat dissipated is transferred to the refrigerant; the remainder of the heat is dissipated via convection to the ambient room air.

Components (cabinets) which utilize refrigerant-cooling techniques and their associated condensing units are:

- NSS mainframe and associated memory cabinets - four stand-alone 30-ton condensing units
- CDC CYBER 173-8 computer - one 3-ton condensing unit, housed within the mainframe bay
- ECS (7030-104) - one 3-ton condensing unit, housed within the ECS cabinet.

Condensing Unit's Cooling Water

Each condensing unit requires cooling water at a flow rate and head loss pressure as specified on the applicable data sheet (30-ton data sheet or CDC CYBER 173 and ECS data sheets).

The customer-supplied water supply should be a recirculating, closed-loop, cooled water system. This system may use refrigerant-type chillers or secondary chillers that use other water sources on the primary side of the condensing unit.

Water supply temperatures must fall within the nominal range of 10°C to 24.4°C (50°F to 76°F) for the 30-ton condensing units and 15.6°C to 26.7°C (60°F to 80°F) for the 3-ton condensing units. Fluctuations from the nominal must not exceed 2.2°C (4°F). Recommended inlet temperature (at the condensing unit) is 15.6°C (60°F). Maximum pressure is $6.89 \times 10^5 n/m^2$ (100 psig). Water quality should meet the following.

Hardness	Not to exceed 200 ppm $CaCO_3$
PH level	Maintained between 7.0 and 9.5
Suspended solids	Not to exceed 10 ppm

Water Piping

It is the customer's responsibility for the installation of all pumps, heat exchangers, piping, valves, and unions associated with the cooling water systems.

Each 30-ton condensing unit must be furnished with a supply and return line equipped with a water turn-off valve and terminating in a 3.8 cm (1.5 in) pipe union (female) not more than 1.5 m (5 ft) from the water entry point of the condensing unit.

Each 3-ton condensing unit must be furnished with a supply and return line equipped with a water turn-off valve and terminating in a 15.9 mm (5/8 in) flare fitting (male) not more than 2.1 m (7 ft) from the water entry point of the equipment cabinet.

Refrigerant Distribution Networks

Each refrigerant distribution network is shipped in three basic sections to be assembled on site prior to computer system delivery.

The purpose of a distribution network is to supply refrigerant to each chassis section of the CDC NSS central processor. The distribution networks are located under the central processor in the plenum space provided by the elevated floor. The piping is slightly offset from the computer base perimeter in order to facilitate easy access to the piping under the floor panels.

Each supply and return refrigerant distribution network is supported by adjustable support brackets and weighs approximately 172 kg (380 lb). The longest section of a network is approximately 3.5 m (140 in), thus requiring temporary dismantling of some stringers which are part of the elevated floor assembly.

The connection of each refrigerant distribution network to its respective main supply and return refrigerant lines may be accomplished at any one of several optional locations. This provides flexibility and reduces possible refrigerant piping crossovers or interferences with the power distribution conduits.

CAUTION

Due to the complexity in layout of the refrigeration, power cabling, and logic cabling required by the NSS, CDC CYBER 173, and ECS units and the water piping required for the CDC CYBER and ECS, the mounting of ducts, cables, and piping should be avoided under any area occupied by the following.

NSS cabinets
CDC CYBER 173 cabinets
ECS cabinets

Refrigerant Piping

Because of size, acoustical, and maintenance consideration, the 30-ton condensing units are designed to be located in an equipment room rather than in the computer room. Such an equipment room may be on the same level as or one floor above or below the computer room (central computer).

It is the customer's responsibility to provide and install leak-proof copper supply and return refrigerant lines between each 30-ton condensing unit and associated CDC-provided refrigerant distribution network sections (manifolds) prior to delivery of the Control Data NASF. CDC strongly recommends that a refrigeration specialist either be hired or consulted for the design, layout, and installation of the supply and return refrigerant lines.

The length of the refrigerant lines between the condensing units and refrigerant distribution networks depends on many variables such as size of copper tubing, rotating, horizontal and/or vertical rises, quantities of elbows, and so on. For the preliminary layout design studies, it can be assumed that the equipment and computer room are on the same level, and that the condensing units are located within a 30 m (100 ft) radius of the central processor. For the final layout design and installation of the refrigerant lines, the refrigeration specialist should observe the following criteria related to each 30-ton condensing unit.

Maximum pressure drop in refrigerant lines:

	Vertical	Horizontal	Total
Liquid line pressure drop n/m^2 (psig)	1.1×10^5(16)	0.28×10^5(4)	1.38×10^5(20)
Suction line pressure drop n/m^2 (psig)			0.83×10^5(12)

Air-Cooled Equipment

The remainder of the equipment (cabinets) in the system is air cooled. This equipment contains blowers and/or fans and pulls the conditioned room air into the cabinet near the floor and expels the air back into the computer room from louvers located at or near the top of the cabinets.

Additionally, many of the cabinets are equipped with filters immediately in front of the blowers.

HEAT DISSIPATION

The heat dissipation figures indicated on the equipment data sheets is based on maximum (steady state) power consumption. As an example; the power consumption of the CDC CYBER 173 MOS memories is relatively low when not processing data; therefore, the figures on the data sheets reflect an average amount of memory in processing. Additionally, the figures reflect all possible options installed, the heat dissipation figures for tape drives indicates a running (read/write) state as opposed to a stand-by or idle state, and so on.

Total heat dissipation for the system is approximately.

To water via the condensing units: 274,236 kcal/hr (1,088,254 Btu/hr)

To air via conduction and convection: 218,245 kcal/hr (866,066 Btu/hr)

NOTE

Heat dissipation figures are for the computer room only because the 30-ton condensing units and MGs are located in a separate equipment room.

VENTILATION, COOLING, AND AIR CONDITIONING REQUIREMENTS

Air filtration requirements for the computing room are set forth in part 4 of the site preparation manual in appendix D.

Limits of temperature and humidity ranges and appropriate storage or nonuse temperature ranges are set forth on the accompanying equipment data sheets.

NOTE

In those cases where a humidity range is not indicated (such as for storage), the limiting factor is a no condensation state.

In addition to the data set forth on the data sheets, the customer should refer to the Magnetic Recording Media Storage Requirements in part 5 of the site preparation manual in appendix D.

The quantity of heat dissipated by each piece of equipment is indicated in BTU/hr (kcal/hr) on the data sheets in appendix B.

Safeguards to the equipment in case of failure of the site (or equipment) cooling systems are the individual equipment air flow sensors and thermostats, the system dew-point recorder, and temperature monitor control panels. These units provide a mixture of early-temperature and high-temperature warnings and low-temperature and extreme high-temperature shutdown conditions.

SPECIAL REQUIREMENTS/CONSIDERATIONS

In addition to the special requirements and considerations set forth previously, the following requirements and/or considerations are applicable.

CABLE LAYOUT AND ROUTING

Power wiring from the power distribution panels to the various system equipment must be routed beneath the raised floor in ducts, raceways, or other approved devices.

All power wiring ducting and mounting are supplied and installed by the customer and must conform to local electrical and/or building codes and ordinances.

Signal cables transmit data and control signals and status between the individual equipment in the system. Control Data provides all necessary signal cables according to the final system layout. Cables are supplied in the standard lengths indicated on the individual equipment data sheets. In certain instances, longer-length signal cables (up to the maximum length allowable as indicated on the data sheet) can be supplied at an additional cost to the customer.

The signal cables route beneath the raised floor. The cables lie directly upon the surface of the normal building floor (except where the use of trough or ducting is required by specific ordinance or desired by the customer). If a cable is longer than necessary, the cable must be randomly routed (never coiled) to take up the excess length.

NOTE

> Signal cables and power cables must be separately routed. 400-Hz and 50/60-Hz cables must not be run in the same cable trough.

EMC GRID

The computer system requires the installation of both a safety (green wire) ground system and an electro-magnetic compatibility (EMC) reference plane grid-ground. Details of the construction and installation of the EMC grid can be found in part 7 of the site preparation manual in appendix D.

RECORDING MEDIA STORAGE ROOM

A separate room (or rooms) should be set aside for the storage of various recording media such as tapes, disk packs, and so on.

Because paper products produce lint, only sufficient quantities for each day's use of cards, line printer, and typewriter paper should be kept in the computer room. Consideration should be given to a convenient but separate storage room for additional quantities of these products.

Refer to part 5 of the site preparation manual in appendix D for detailed information on the requirements of these storage media rooms.

EQUIPMENT ROOM

Because of size, acoustical, and maintenance considerations, the 30-ton condensing units and MGs should be located in a separate equipment room. Preferably the room should be on the same level as or one floor below or above the computer room.

REFRIGERANT SAFETY PRECAUTIONS

Because of the large quantity of refrigerant contained in the 30-ton condensing units, the installation of a pressure venting network should be considered. The following sets forth the potential danger of concentrated refrigerant gas and information on such a venting network.

Control Data computer cooling systems utilize refrigerant R-12 (Freon) in the condensing units. Each condensing unit is equipped with a temperature/pressure sensitive fusible plug which acts as a safety relief valve to remove any danger of explosion due to an excessive temperature/pressure buildup. Excessive temperature/pressure buildup could occur due to internal (mechanical) failure of the condensing unit or external causes (such as a fire in the computer room).

Hazards of Refrigerant Gas

Small quantities of refrigerant R-12 taken into the respiratory system are harmless; however, large quantities of concentrated refrigerant gas can cause serious illness. Refrigerant R-12 is an inert gas which is heavier than air and tends to collect in low areas (such as under the false floor). A small amount of gas trapped can cause a high concentration level.

The principal hazard with an inert gas is that if the supply of oxygen is replaced by the inert gas, suffocation may occur. The threshold limit value† for refrigerant R-12 is 1000 parts per million. In addition, concentrations of refrigerant R-12 gas†† in the order of 0.5 percent to 2.5 percent in the presence of any open flame represents a lethal life hazard for continuous duration of exposure of 5 to 15 minutes (no ventilation).

WARNING

Toxic gases are formed if refrigerant R-12 gas comes in contact with any open flame.

Calculation of Lethal Refrigerant Levels

Because an average computer room ranges in size from 1133 m^3 to 1699 m^3 (40,000 to 60,000 ft^3), the system with two CDC CYBER 173's and ECS could possibly reach concentrations of 0.5 percent.†††

†Determined by the American Conference of Governmental Industrial Hygienists. The value is the concentration in air which is believed to represent a safe limit for repeated exposure day-after-day without adverse effects. From the E.I. DuPont de Nemours and Co. (Inc.) Freon Fluorocarbons, Properties and Applications product bulletin E-2 (3-71).

††29.5 kilograms (65 pounds) of liquid refrigerant R-12 expands to approximately 5.9 cubic meters (200 cubic feet) of gas.

†††Each 3-ton condensing unit contains 9.34 kg (20.6 lb) of liquid refrigerant R-12; each 30-ton condensing unit contains 108.4 kg (239 lb) of liquid refrigerant R-12.

The concentration level of refrigerant R-12 gas expressed as a percentage for any given space can be approximated by.

Concentration in percent = $\frac{cQ}{V}$

c	Constant = 19.17 m^3/kg (307 ft^3/lb)
Q	Kilograms (pounds) of liquid refrigerant R-12 being released
V	Volume of space in cubic meters (cubic feet)

For example: 90.7 kg (200 lb) of R-12 in a computer room of 1133 m^3 (40,000 ft^3)

$$\frac{cQ}{V} = \frac{19.17 \times 90.7}{1133} = \frac{1739}{1133} = 1.53 \text{ percent potential concentration}$$

or

$$\frac{cQ}{V} = \frac{307 \times 200}{40,000} = \frac{61,400}{40,000} = 1.53 \text{ percent potential concentration}$$

Pressure Venting Network

Each condensing unit is supplied with a flexible 1.2-m (4-ft) metallic hose connected to the pressure relief safety valve on the condensing unit. The other end of the metallic hose terminates in a 9.53 mm (3/8 in) flare union (female) and hangs down into the area of raised floor space underneath the condensing unit.

The provision and installation of such a venting network is the responsibility of the customer. Although the customer is not obligated to install such a system, Control Data strongly recommends installation of the pressure venting network. Detailed information and instructions for such a network may be obtained from the Control Data Site Engineering personnel.

The final decision on installation of a pressure venting network should be based upon calculation of potential lethal refrigerant levels in consideration with the following.

- Physical location of the condensing units
- State and local building codes and regulations†
- Total amount of liquid refrigerant R-12 on site
- Volume of the computer room (or area where the condensing units are located)
- Total occupancy and approximate evacuation time of the site by personnel in any emergency.

WATER ALARM INDICATOR

The computer room floor (under the false floor) should be equipped with one or more devices to warn personnel of the presence of water due to a broken pipe or hose. A float-operated switch, controlling an audible alarm and a sump pump, is recommended. In addition, one or more devices should be installed in the water return line(s) to warn personnel of the absence of water or an abnormal drop in water pressure.

CAUTION

After installation of all piping, valves, and alarm indicators is completed, locations of all shutoff valves, alarm indicators, and sump pumps should be indicated on a master floor plan diagram for easy access during any emergency.

SITE SECURITY

Refer to part 6 of the site preparation manual in appendix D for information on site fire and security precautions.

†These may require the installation of such a network.

PERMISSIBLE RADAR/RADIO FIELD STRENGTH

The maximum permissible radar and radio field strength in which the equipment will perform reliably is 1.0 volt per meter. (This is based upon worst-case; many equipments will perform at higher levels depending upon frequency.)

EQUIPMENT-TO-GROUND RESISTANCE

The resistance between an equipment's frame and earth ground shall not exceed 10.0 ohms.

- Equipment-to-Equipment Resistance

  The resistance between any two pieces of equipment shall not exceed 0.1 ohm.

- Earth Resistance

  The resistance to earth of a continuous underground water system, used as the earth electrode, shall not exceed 5.0 ohms.

  The resistance to earth of metallic building frames, local metallic underground piping systems, metal well casings, or buried (or driven) electrodes, used as the earth electrode, shall not exceed 5.0 ohms.

POWER SYSTEM (MAINS) NEUTRAL ISOLATION

System grounds shall be isolated from the system neutral at the computer system main power panel (not the service entrance). Visually inspect the main power panel to ensure that the neutral and conduit are isolated in the power panel.

RAISED FLOOR PANEL SURFACE COVERING

If the surface of the raised floor panel is to be tiled, it must be of a high-pressure laminate variety in order to withstand the heavy loads to which computer room floors are exposed. (Do not use asphalt, vinyl, or vinyl-asbestos tiles). A high-pressure laminate surface provides minimum maintenance requirements.

Control Data suggests a carpet tile surface because of its acoustical, aesthetic, and comfort properties. However, if carpet tiles are to be used, they must have an acceptable electrostatic rating (Control Data has a list of tested acceptable carpeting).

OPERATIONS AND SUPPORT REQUIREMENTS

A large, complex facility such as the proposed NASF has not only a relatively large initial acquisition cost but has significant on-going costs as well. The initial cost is primarily for system development as well as for procurement and site preparation. The system costs are estimated and presented in previous sections of this report. Procurement and/or site preparation (real estate and a building) are not included in this study. However, this section of the report has presented site requirements which influence the development of proper housing for such a facility.

Once the system is installed and in operation, it becomes a computer center in every respect. Although the NASF will be a specialized computational facility, it nevertheless requires basically the same types of support as other large computer centers. The costs associated with the operation of the proposed NASF must be identified and estimated for future planning since these are continuing or recurring costs. Table 6-1 lists the estimated expenses for operation and support of the NASF proposed in this report. A brief discussion of each item is included. Some of the expense items appear for identification only and have no cost estimated. These costs are either specifically excluded from the study or beyond the scope of the study.

TABLE 6-1. ESTIMATED ANNUAL OPERATION AND SUPPORT COSTS

Expense Item	Shift (Monday through Friday)†			Shift (Saturday and Sunday)†			Estimated Annual Cost($)††
	1	2	3	1	2	3	
Computer operators	3	2	2	2	1	1	166,850
Customer engineers	5	3	3	4	2	2	382,970
System programmers	2	1	1	1	1	1	222,120
System analysts	2	1	1	1	0	0	134,960
Supervisory personnel	3	0	0	0	0	0	112,460
Supplies		-			-		136,320
Software lease		-			-		217,600
Replacement parts		-			-		115,020
Utilities		-			-		314,560
Total		-			-		1,802,860

†Number of personnel per indicated shift.
††Labor figures are unburdened and include allowance for vacation, holiday, and sick leave.

The NASF computer center operation is assumed to be 24 hours per day, 7 days per week. This is the basis of the estimated operation and support costs. The heaviest traffic is expected to be during the prime shift (Monday through Friday), but support requirements are anticipated on other shifts and weekends as well.

OPERATORS

Computer operators would be required to provide counter service for batch jobs, operate various equipments, load and unload card devices, mount or demount tapes and disk packs, and remove and distribute printer listings.

CUSTOMER ENGINEERS

To provide complete preventive and corrective maintenance coverage for the multiplicity of devices in this type of system, a relatively large staff of on-site customer engineers would be necessary. A sufficient staff would be on duty at all times to provide emergency service, if needed, with others on call if a given specialty or expertise is required. Prime shift would have the best coverage with one of the off-shifts having sufficient coverage to perform preventive maintenance.

SYSTEM PROGRAMMERS

Although the proposed facility is not intended as a general-purpose computer center, it would have a considerable amount of system software. Some of the software, particularly for the front-end systems, would be standard products while other software would be tailored to this special application. The system programmers would provide on-site assistance for system utilization, for debugging aid, and for system software maintenance in general.

SYSTEM ANALYSTS

In addition to the system programmers, a more in-depth problem resolution support would be provided by system analysts. This would be particularly helpful during the early operation of the facility. This coverage could perhaps diminish as time passes and the operation stabilizes. At all times, however, system analysts can provide assistance for improving system utilization, maximizing efficiency, isolating problems, and effecting solutions.

SUPERVISORY PERSONNEL

Efficient operation of this facility and effective management of the staff described above requires adequate supervision. This category is the first-level supervisor who would report to a computer center manager; this manager is not included in this report. The supervisory personnel are intended for only a one-shift coverage, with a senior person serving in a lead capacity for off-shift supervision.

SUPPLIES

The supplies estimated in this report are those used directly by the computer center. This includes items such as printer paper, printer ribbons, card stock, magnetic tape, and disk packs. Maintenance supplies for either building or equipment are not included.

SOFTWARE LEASE

With the unbundling of software, which has become commonplace, it is reasonable to expect that at least the standard product software will have a lease or license charge. Typically, this is a one-time charge plus an on-going monthly charge. In this report, the one-time charge is included with acquisition, and in this section the estimate is for the continuing software lease.

REPLACEMENT PARTS

Equipment configurations in this report require replacement parts for routine maintenance. Mechanical parts wear out and electronic parts fail. Reliability is a significant factor in the design of this system, but replacement expense is necessary.

UTILITIES

Although the building for this facility is not part of this study and report, the estimated utility expense is included. This category covers gas, water, and electrical energy. For a system such as this, the bulk of the utility charges are for the electrical energy to power the system and to cool it, for cooling the building, and for normal building requirements such as lighting. Gas and water constitute a relatively small part of the utility expense.

EXPENSES NOT ESTIMATED

The following anticipated expense items are listed in order to specifically identify them. They are not included in the estimate appearing in table 6-1. These items are either specifically excluded from this study or are beyond its scope.

- Building and real estate amortization
- Equipment amortization
- Building maintenance and supplies
- Building and grounds custodian
- Facility engineers (electrician, plumber, and so on)
- Telephone
- Communication line charges

- Applications programming
- Facility management and administration
- Insurance
- Security

Some of these are GSA functions and are best estimated by that body; others are best estimated by the NASA personnel who are planning and/or would be using the proposed facility.

APPENDIX A

MACHINE UNIT SPECIFICATIONS

419

6-A-1

CONTROL DATA CORPORATION
DOCUMENT NO. 2VA30-2
RUN DATE 08/08/77
CUSTOMER: NASA - AMES GENERAL COMPUTING FACILITY

MACHINE UNIT SPECIFICATION

CABINET NAME MODULE/MODULES	PRODUCT	QTY	PHYSICAL PROPERTIES UNIT WIDTH (IN.)	UNIT DEPTH (IN.)	UNIT AREA (SQ FT)	UNIT HEIGHT (IN.)	UNIT WEIGHT (LBS)	UNIT HEAT DISSIP (BTU/HR)	UNIT KVA 400HZ	UNIT KVA 50/60HZ	TPWR	UNIT CIRCUIT BREAKER REQUIREMENTS (02) 400HZ 208V,3P AMPERES	50/60HZ 208V,3P AMPERES	CONN	50/60HZ 115V,1P AMPERES	CONN
CENTRAL COMPUTER BAY 1																
	173-8B	2	96.20	35.00	46.76	79.80	4500	32440(10)	7.0	4.0	B	30	30(06)	(02)		
CONSOLE																
	173-8B	2	32.50	47.00	21.22	46.50	390	3030	0.8	0.3		15			15(07)	(02)
ECS COUPLER																
	10318-1	2	-	-	-	-	-	-	-	-						
PERIPHERAL CONTROLLER B CAB																
	7030-104C	1	42.00	20.50	5.98	56.90	850	3600	0.9	0.4		15			15	(02)
STORAGE CABINET 1																
	7030-104C	1	70.00	40.80	19.83	72.50	1950	23500(09)	4.0	4.0		30	20	(02)		
DDP CONTROLLER																
PO ECS CONFIG.	7030-104C	1	29.00	25.00	5.03	66.00	500	1400	0.3	0.1		15			15	(02)
MASS STORAGE CONT																
FA202 + FA103	7639-2	4	29.00	25.00	20.14	66.00	400	2840	0.7	0.2		15			15	(02)
DISK STORAGE UNIT																
	819-2	32	27.00	45.00	270.00	45.00	800	10000	-	5.0			20	(03)		
MASS STORAGE CONT																
	7154-4	1	29.00	25.00	5.03	76.00	300	2400	0.6	0.2		15			15(05)	(02)
DISK STORAGE UNIT																
	844-21	2	22.00	44.80	13.69	39.50	700	4500	-	1.4			20(08)	(03)		
MAG TAPE CONTROL																
TRIMLINE	7021-22	1	55.60	25.00	9.65	76.00	850	4370	0.9	0.4		15			15(05)	(02)
MAG TAPE TRANSPORT																
	667-4	2	30.50	29.50	12.50	63.50	900	9900	-	2.9			15(04)	(03)		
MAG TAPE TRANSPORT																
	669-4	2	30.50	29.50	12.50	63.50	900	9900	-	2.9			15(04)	(03)		
MASS STORAGE FILE																
CARTRIDGE STORE UNIT																
	38510-2	8	126.00	21.00	149.33	76.80	3300	10580	-	3.6			15			
CARTRIDGE TRANSPORT																
	38510-2	8	26.80	31.00	46.16	71.00	700	6150	-	2.1			15	(02)		
CARTRIDGE TRANSPORT																
	38510-2	8	26.80	31.00	46.16	71.00	700	6150	-	2.1			15	(02)		
MASS STORE ADAPT																
	65125-1	2	53.80	31.00	23.16	71.00	500	19700	-	7.2	A		30	(03)		
MASS STORAGE COUPLER																
	65125-1	2	29.20	25.00	10.14	66.00	500	1430	0.3	0.1						
CARD READER CONT																
IN 405	3447-2	2	-	-	-	-	-	-	-	-	B		PWR FROM 405 CR			
CARD READER/CONTROLLER																
1200 CPM	405	2	57.00	33.00	26.13	46.00	1200	9000	-	3.4	B		15	(03)		
CARD PUNCH/CONTROL																
	415-30	1	21.50	39.50	5.90	45.30	650	3565	-	1.5	B				15	(02)
LINE PRINTER/CONT																
2000 LPM	580-20	4	62.00	31.50	54.25	51.50	1500	15000	-	5.2	B		30(04)	(02)		

Reproduced from best available copy.

```
                                                                                                     UNIT CIRCUIT BREAKER REQUIREMENTS
                                    --------PHYSICAL PROPERTIES-------  UNIT                       T (02)                 C                C
                                    UNIT   UNIT   UNIT    UNIT   UNIT   HEAT                       P 400HZ   50/60HZ      O     50/60HZ    O
 CABINET NAME                   QTY WIDTH  DFPTH  AREA   HEIGHT WEIGHT DISSIP      ---UNIT KVA---  W 208V,3P 208V,3P      N     115V,1P    N
  MODULE/MODULES     PRODUCT        (IN.)  (IN.) (SQ FT) (IN.)  (LBS) (BTU/HR)    400HZ  50/60HZ  R AMPERES AMPERES      N     AMPFRES    N

 HOST COMM PROCESSOR
                    2550-2       2  48.00  34.00  22.67 75.00   950  13300        -      3.9                30(05)  (03)
 CHANNEL COUPLER
  IN HPC            2558-1       2    -      -      -     -      -      -         -      -
 CHANNEL COUPLER
  IN HPC            2558-2       2    -      -      -     -      -      -         -      -
 EMULATION MODULE
  IN 2550-2         2550-100     2    -      -      -     -      -      -         -      -
 DISPLAY TERMINAL
                    751-10       1  20.50  20.50   2.92 15.80    55    665        -      0.2                                  15(05)  (03)
 GRAPHICS TERM CONSOLE
                    774-1        1  33.20  54.00  12.45 57.80   900   6000        -      2.3                15(08)  (03) :
 FMERGENCY OFF     (WALL MOUNT)
                    53369800     1  21.00   7.90   1.15 12.00    35     -         -      -                  15(06)  (02)
 TEMP MON/PWR CONT(WALL MOUNT)
  PO CY170          53368900     2  25.00   4.80   1.67 16.00    40     -         -      -              PRIMARY PWR GRP OR EM OFF
 SYST PWR CONTROL  (WALL MOUNT)
  PO STAR-100       53376500     1  21.00   7.80   1.14 12.00    40     -         -      -              PWR FROM EMERGENCY-OFF
 TERMINATOR POWER  (WALL MOUNT)
                    18182800     1  24.00   6.80   1.13 12.00    50     -         -      -       15
 PERIPH.PWR.CONTR.(WALL MOUNT)
                    53370400     1   9.50   6.80   0.45 12.00    30     -         -      -              POWER FROM EMERGENCY OFF
 DEW POINT RECORD  (WALL MOUNT)
                    53370000     2  15.50   7.50   1.61 36.00    80     -         -      -                                    15(07)  (02)
 MOTOR GENERATOR
   125 KVA       (01)MG125K      4  70.50  42.50  83.23 34.50  4075  79000        -      -               **** CONN TO MG CONT
 NON-NUMERIC DATA IN DATA BASE
                    MG125K       4  56.00  20.00  31.11   -    1100     -         -      -              SIZE DISC FOR 200HP MOTOR

                                                                                TOTAL AREA(EQUIP ONLY) =      961.9 SQ FT
                                                                                TOTAL WEIGHT =   119,030 LBS
                                                                                TOTAL HEAT DISSIPATION = 1,142,320 BTU/HR
                                                                                TOTAL KVA(400HZ) =    25.7
                                                                                TOTAL KVA(50/60HZ) =   305.3

                                                                                NOTE - ABOVE TOTALS EXCLUDE ANY ADDITIONAL DATA
                                                                                THAT MAY ACCOMPANY THIS SPECIFICATION (SEE BELOW)
```

DOCUMENT NO. 2VA30-2

NOTES:
- ALL SPECIFICATIONS ARE ON A PER UNIT BASIS
- WITH EXCEPTION OF UNIT AREA(REFLECTS TOTAL UNIT AREA)

(A) INTERNAL TERMINATOR POWER SUPPLY
(B) EXTERNAL TERMINATOR POWER SUPPLY
(01) MAXIMUM CAPACITY SHOWN: ACTUAL LOAD MAY BE LESS.
(02) TERMINAL STRIP.
(03) LOCKING CONNECTOR.
(04) 380V-3PH. FOR 50HZ VERSION.
(05) 220V - 1PH. FOR 50 HZ VERSION.
(06) 380/415V - 3 PHASE + N FOR 50HZ OPERATION
(07) 220/240V - 1 PHASE + N FOR 50HZ OPERATION
(08) TWO PHASE CIRCUIT BRKR FOR 60 HZ OPERATION
FOR 50 HZ OPERATION USE SINGLE PHASE 220V CIRCUIT BRKR.

(09) WATER COOLED. 90 PERCENT OF HEAT REJECTED TO WATER.
10 PERCENT OF HEAT REJECTED TO ROOM.

WATER REQUIREMENTS PER BAY

INLET TEMP DEG F	INLET TEMP DEG C	FLOW RATE GPM	FLOW RATE L/MIN	HEAD LOSS PSI	HEAD LOSS KG/CM2
80	26.7	2.1	8.0	6.6	0.48
70	21.1	1.5	5.7	3.5	0.27
60	15.6	1.2	4.5	2.3	0.17

(10) WATER COOLED. 80 PERCENT OF HEAT REJECTED TO WATER.
20 PERCENT OF HEAT REJECTED TO ROOM.

WATER REQUIREMENTS PER BAY

INLET TEMP DEG F	INLET TEMP DEG C	FLOW RATE GPM	FLOW RATE L/MIN	HEAD LOSS PSI	HEAD LOSS KG/CM2
80	26.7	5.5	20.8	11.8	0.8
70	21.1	3.7	14.0	6.9	0.5
60	15.6	2.8	10.6	4.1	0.3

(11) WATER COOLED. 300000 BTU/HR (75000 KCAL/HR) REJECTED TO
WATER, 12000 BTU/HR (3025 KCAL/HR)
REJECTED TO ROOM.

30 TON CONDENSING UNIT WATER REQUIREMENTS PER UNIT

INLET TEMP DEG F	INLET TEMP DEG C	FLOW RATE GPM	FLOW RATE L/MIN	HEAD LOSS PSI	HEAD LOSS KG/CM2
80	26.7	19	72	2.8	0.20
70	21.1	15	57	2.3	0.17
60	15.6	12	46	1.8	0.13
50	10.0	10	38	1.2	0.09

ADDITIONAL DATA-

CABINET NAME MODULE/MODULES	PRODUCT	QTY	PHYSICAL PROPERTIES UNIT WIDTH (IN.)	UNIT DEPTH (IN.)	UNIT AREA (SQ FT)	UNIT HEIGHT (IN.)	UNIT WEIGHT (LBS)	UNIT HEAT DISSIP (BTU/HR)	UNIT KVA 400HZ	UNIT KVA 50/60HZ	TPWR (02)	UNIT CIRCUIT BREAKER REQUIREMENTS 400HZ 208V,3P AMPERES	50/60HZ 208V,3P AMPERES	CONN	50/60HZ 115V,1P AMPERES	CONN
NAVIER-STOKES SOLVER	NSS	1	271.00	208.30	392.0	75.00		988000(09)	289.5							
SECONDARY STORAGE	NSS	1	96.00	28.00	18.6	75.00		140000(09)	40.0							
CONDENSING UNIT	NSS	4	90.00	34.00	21.3	48.00	2125	312000(11)		40.0			70	(02)		

HEAT SHOWN AS DISSIPATED BY THE NAVIER-STOKES SOLVER AND SECONDARY STORAGE ARE ALSO REFLECTED IN THE HEAT DISSIPATION OF THE 30 TON CONDENSING UNITS AND SHOULD NOT BE COMBINED. TEN PERCENT OF THE TOTAL CENTRAL COMPUTER AND SECONDARY STORAGE HEAT SHOWN SHOULD BE USED TO CALCULATE COMPUTER ROOM HEAT LOADS.

APPENDIX B

EQUIPMENT DATA SHEETS

APPENDIX B

EQUIPMENT DATA SHEETS

The equipment data sheets are in the following order.

2550 Host Communications Processor

2550 Host Communications Processor Specifications

7021 Magnetic Tape Controller

7021 Magnetic Tape Controller Specifications

7154 Mass Storage Controller

7154 Mass Storage Controller Specifications

7639 Controller

7639 Controller Specifications

125 KVA Motor Generator

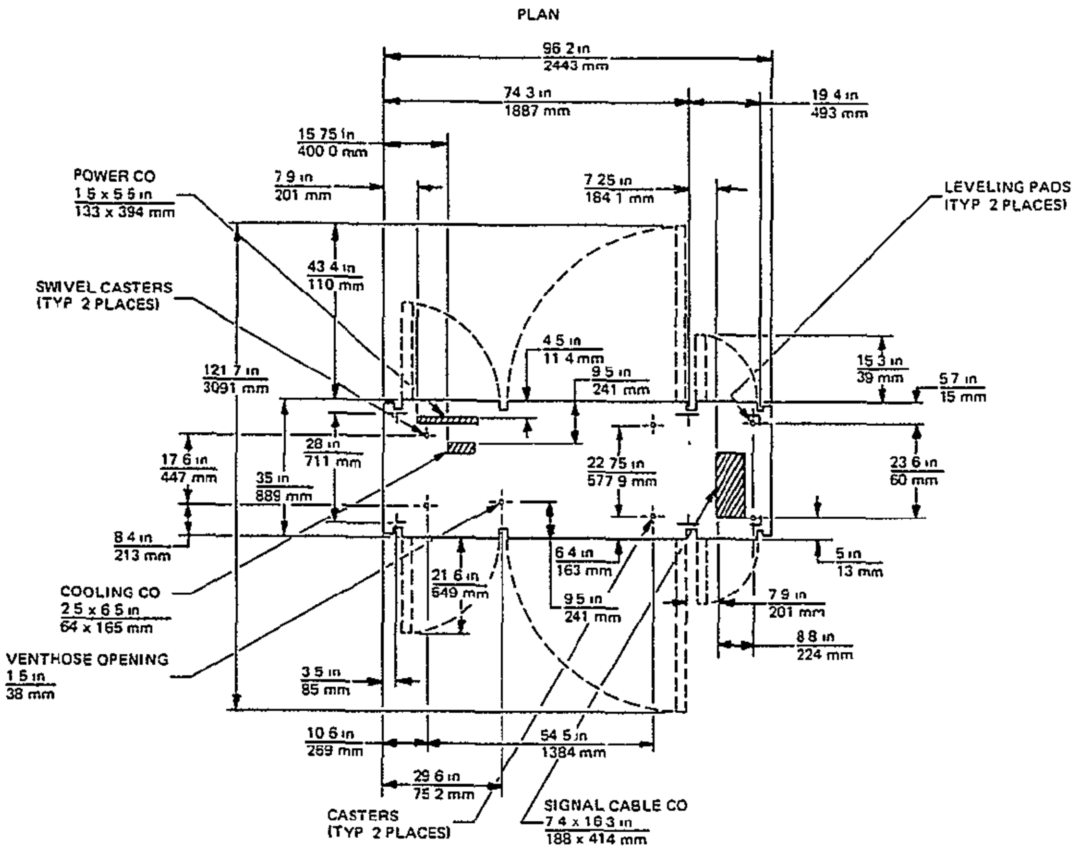
125 KVA Motor Generator Specifications

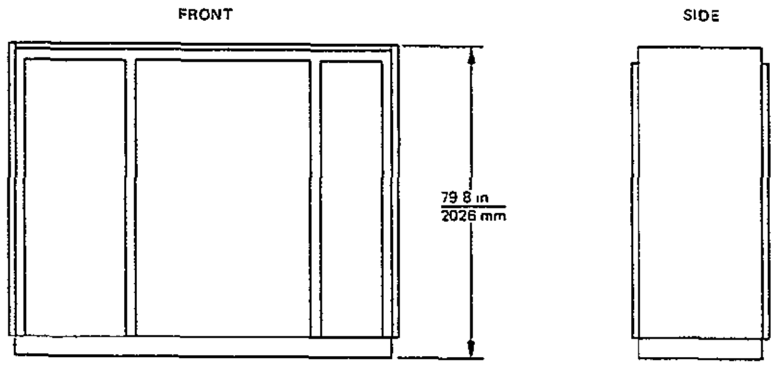
125 KVA Motor Generator Cabinet Specifications

30-Ton Condensing Unit

30-Ton Condensing Unit Specifications

CENTRAL COMPUTER, MODEL 172B/173B
(One-Bay Configuration, S/N 201-600 and 701-XXX)





6AA13A

CENTRAL COMPUTER, MODEL 172B/173B
(One-Bay Configuration, S/N 201-600 and 701-XXX)

Width	96.2 in. (2443 mm)	Weight	4500 lb (2045 kg)
Depth	35 in. (889 mm)	Supported by 4 casters	
Height	79.8 in. (2026 mm)		

For maximum width/depth, all doors extended, refer to floor plan layout

Power consumption, steady state, maximum

400 Hz, 208v, 3 phase, 7.0 kVA

and { 60 Hz, 208v, 3 phase, 4.0 kVA or 50 Hz, 208v, 3 phase, 4.0 kVA or
50 Hz, 380v, 3 phase, 4.0 kVA (estimated) or 50 Hz, 415v, 3 phase, 4.3 kVA (estimated) }

Circuit breakers				Power connector locations above cabinet base	
	400 Hz,	30 amp,	3 phase	Terminal strip	18 in. (457 mm)
and	60 Hz,	30 amp,	3 phase	Terminal strip	18 in. (457 mm)
	or				
	50 Hz,	30 amp,	3 phase	Terminal strip	

Control Data signal cables	Quantity:	Standard:	Maximum length
17X to console	2	65 ft (19.8m)	65 ft (19.8m)
17X to ECS controller	10	65 ft (19.8m)	65 ft (19.8m)
17X to ECS/DDP	2	65 ft (19.8m)	65 ft (19.8m)
17X to any 6000 peripheral controller	2	65 ft (19.8m)	65 ft (19.8m)

Distance from internal signal cable connectors to floor 53 in. (1346 mm)

External terminator power connection required no x yes (40 vdc)

Environmental considerations

Type of cooling, R-12 refrigerant to water

Source of cooling, internal 3-ton condensing unit

Air required at inlet if plenum-cooled, N/A cfm (N/A m^3/hr)

Heat rejection rate, maximum { to air, 6489 Btu/hr (1622 kcal/hr)
to water, 25,954 Btu/hr (6489 kcal/hr) }

	Maximum	Recommended	Minimum	Dew-Point Limitation
Operating temperature	74°F (23.3°C)	72°F (22.2°C)	62°F (16.7°C)	56°F (13°C)
Storage temperature	120°F (49°C)		40°F (4.4°C)	

	Inlet Temperature		Flow Rate		Head Loss	
	°F	°C	US gpm	Liter/Min	psi	kg/cm^2
Water flow and temperature requirements	80	26.7	5.7	20.8	11.8	0.8
	70	21.1	3.7	14.0	6.9	0.5
	60	15.6	2.8	10.6	4.1	0.3

EXTENDED CORE STORAGE 2 VERSION 2 (524K BAY)

Width	70.0 in. (178 cm)	Weight	1950 lb (887 kg)
Depth	40.8 in. (104 cm)	Supported by	Frame
Height	72.5 in. (182 cm)	Use Template - T 273	

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum

400 Hz, 208 v, 3 phase, 4.0 kva

and { 60 Hz, 208 v, 3 phase, 4.0 kva or 50 Hz, 208 v, 3 phase, 4.0 kva } or { 50Hz, 380V, 3 Phase, 4.0KVA or 50Hz, 415V, 3 Phase, 4.2KVA }

Circuit breakers		Power connector locations above cabinet base	
400 Hz, 20 amp,	phase	Terminal Strip	20 in. (51 cm)
and { 60 Hz, 20 amp, or 50 Hz, 20 amp,	phase phase }	Terminal Strip	20 in. (51 cm)

Control Data signal cables:	Quantity·	Standard.	Maximum length.	P/N
ECS bay to contr. (6640)	10	40 ft (12.2 m)	40 ft (12.2m)	91903200
NOTE: Total cable length between contr. & ECS end panel is 45ft (13m) 40ft + 5ft drop cable (6640)		ft (m)	ft (m)	
		ft (m)	ft (m)	

Distance from internal signal cable connectors to floor. 54 in. (137 cm)

External terminator power connection required no ———

Environmental considerations

Type of cooling, R-12 Refrigerant to Water

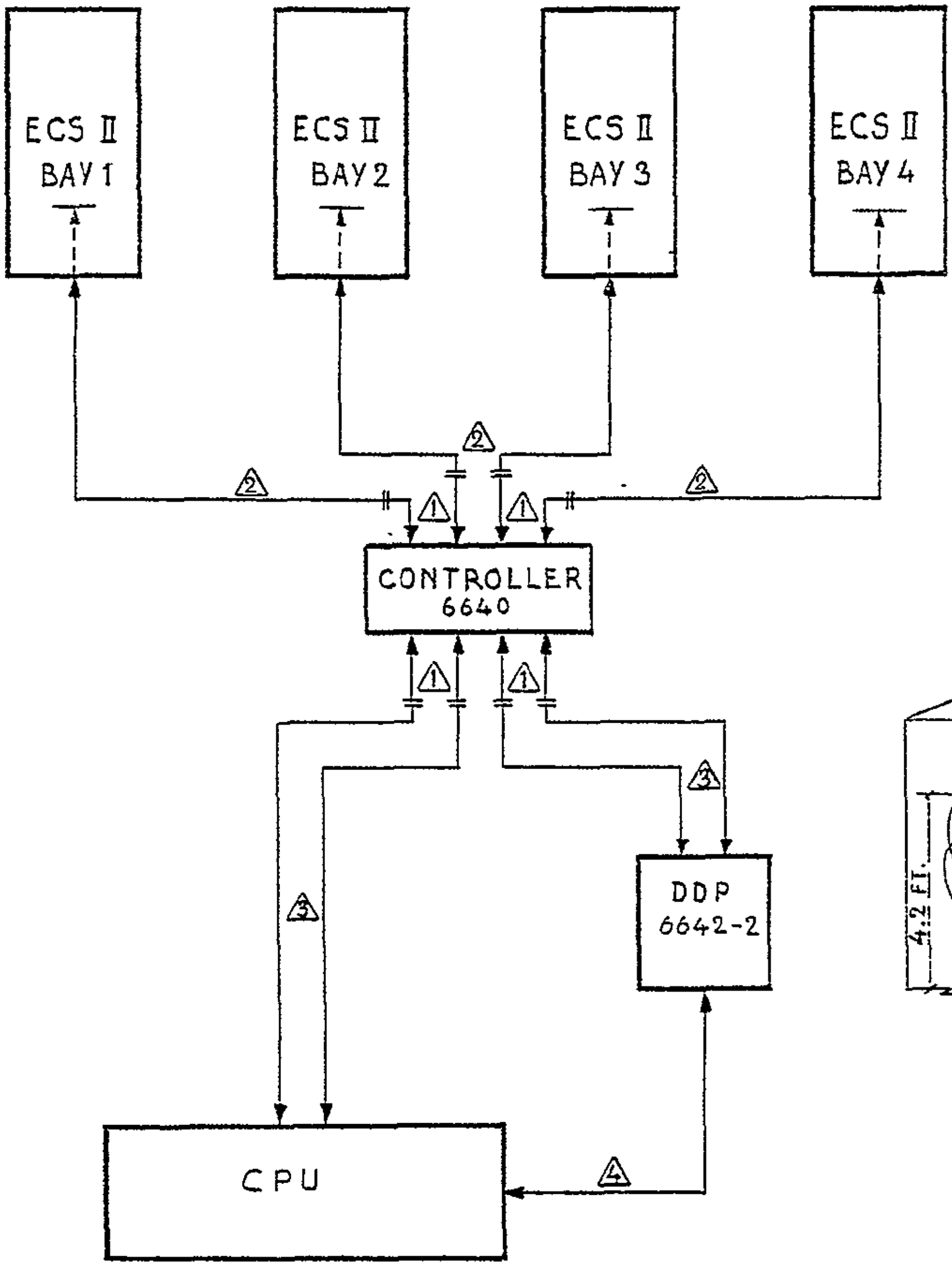
Source of cooling, Internal 2-Ton Condensing Unit and Blowers

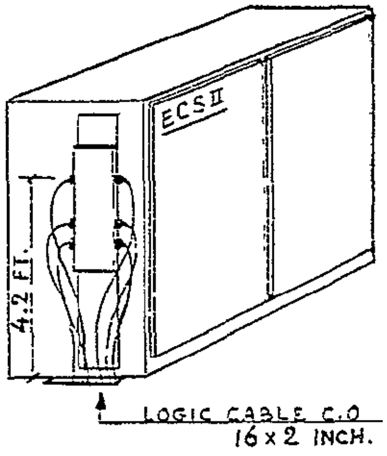
Air required at inlet if plenum-cooled, N/A cfm (m^3/hr)

Heat rejection rate, maximum { to air, 2350 Btu/hr (588 kcal/hr) ; to water, 21150 Btu/hr (5288 kcal/hr) }

	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	74 °F (23.3 °C)	72 °F (22.2 °C)	62 °F (16.7 °C)	56 °F (13 °C)
Storage temperature	120 °F (49 °C)		40 °F (4.4 °C)	

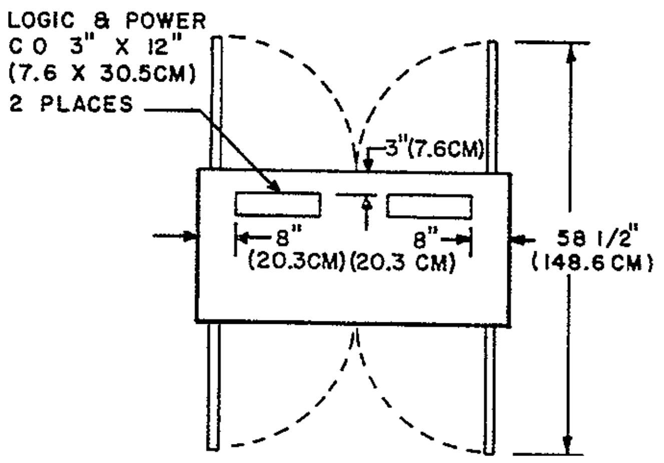
STANDARD LOGIC CABLE CONFIGURATION

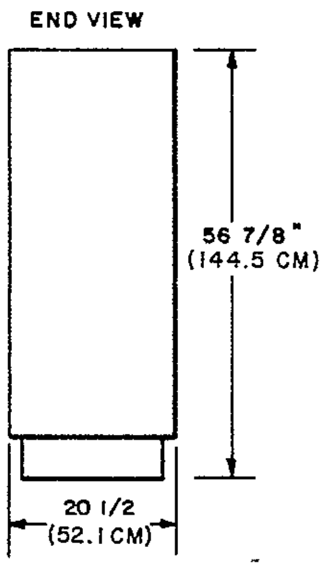


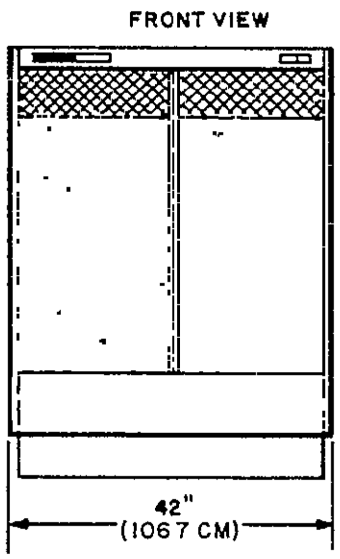


1 5 FT. DROP CABLES, ECS AND CONTROLLER END ARE FACTORY PUNCHED.
2 40 FT. CABLE, 19 PIN, P/N 91903200, 10 CABLES PER ECS BAY.
3 65 FT CABLE, 19 PIN, P/N 19191600, 10 CABLES PER CPU AND 10 PER DDP.
4 65 FT. CABLE, 19 PIN, P/N 19191600, 2 CABLES.

ECS CONTROLLER







ECS CONTROLLER

Width	42 in.	(107 cm)
Depth	20.5 in.	(52 cm)
Height	56.87 in	(144 cm)

For maximum width/depth, all doors extended, see floor plan layout.

Weight	850 lb	(386 kg)
	10 in^2 each	(65 cm^2)

Power consumption, steady state, maximum:

	400 Hz,	230 v,	3-phase, 0.9 kva
and	60 Hz, or 50 Hz,	120 v,	1-phase, 0.4 kva
		120 v,	1-phase, 0.4 kva

Circuit breakers.		Power connections	Location	
	400 Hz, 15 amp,	3-phase Terminal strip	12 in. (30 cm)	above base of cabinet
and	60 Hz, 15 amp, or	1-phase Terminal strip	12 in. (30 cm)	
	50 Hz, 15 amp,	1-phase Terminal strip		

Control Data signal cables (19-pin coaxial):	Quantity	Maximum length
Computer† to controller	10	65 ft (19.8 m)
Controller to ECS	11	50 ft (15.2 m)

Distance from internal signal cable connectors to floor	50in. (127 cm)
External terminator power connection required:	x no

Environmental considerations.

Type of cooling: Forced air (internal blowers)

Source of cooling Ambient room air

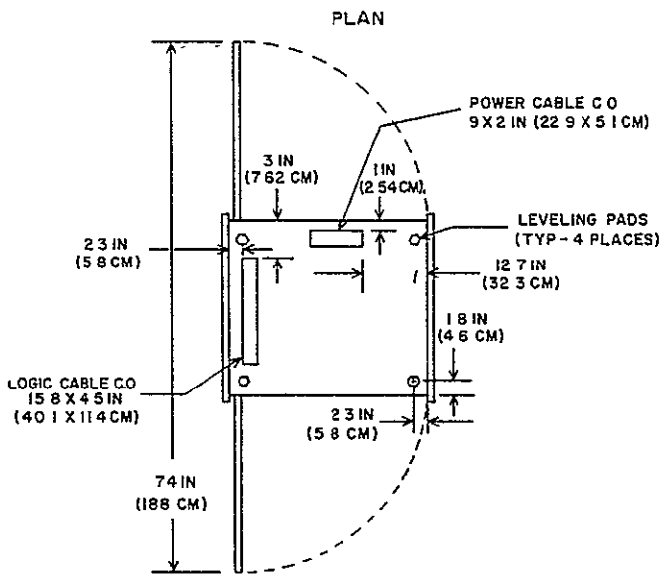
Air required at inlet if plenum-cooled:		N/A cfm	(- m^3/hr)
Heat rejection rate, maximum	to air	3600 Btu/hr	(900 kcal/hr)
	to water	N/A Btu/hr	(- kcal/hr)
Permissible range of room relative humidity		35 % to 60 %	

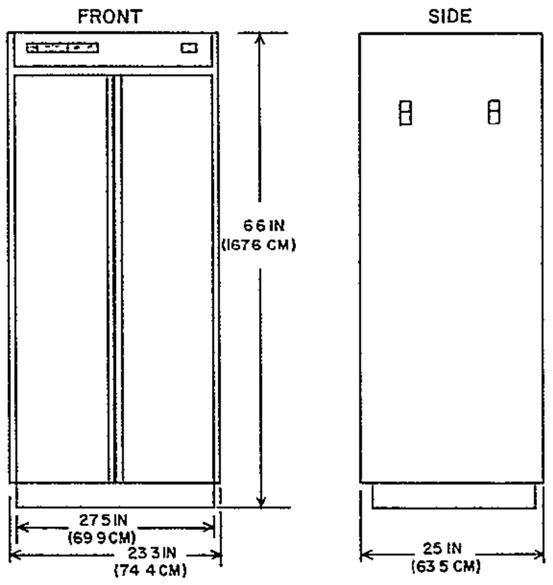
	Maximum	Recommended	Minimum	Dew point limitation
Operating temperature (ambient or plenum inlet temperature)	78 °F (25 5 °C)	72 °F (22.2 °C)	62 °F (16.7 °C)	N/A
Storage temperature	125 °F (51 °C)		-20 °F (-29 °C)	

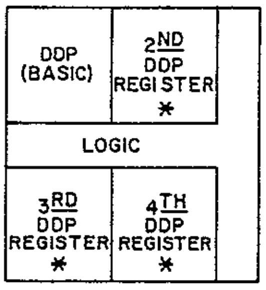
Water flow and temperature requirements N/A

† May be computer or DDP.

DISTRIBUTIVE DATA PATH (DDP) CONTROLLER







OPTIMUM CONFIGURATION
* SEE 10266-X OPTIONS

DISTRIBUTIVE DATA PATH (DDP) CONTROLLER

Width	29.3 in.	(74.4 cm)
Depth	25 in.	(63.5 cm)
Height	68 in.	(167.6 cm)

For maximum width/depth, all doors extended, see floor plan layout.

Weight	500 lb	(226 8 kg)

Power consumption, steady state, maximum·

	400 Hz,	208 v,	3-phase,	0.3 kva
and	60 Hz, or 50 Hz,	120 v, / 220 v,	1-phase, / 1-phase,	0.1 kva / 0.1 kva

Circuit breakers		Power connections	Location·
	400 Hz, 15 amp,	3-phase Terminal strip	12 in. (30.5 cm) above base of cabinet
and	60 Hz, 15 amp, or 50 Hz, 15 amp,	1-phase Terminal strip / 1-phase Terminal strip	12 in. (30 5 cm) above base of cabinet

Control Data signal cables (19-pin coaxial)	Quantity	Maximum length
DDP to ECS controller	10	65 ft (19 7 m)
DDP to computer I/O channel	2 per channel	65 ft (19 7 m)

Distance from internal signal cable connectors to floor	55 in. (140 cm)
External terminator power connection required.	x no

Environmental considerations

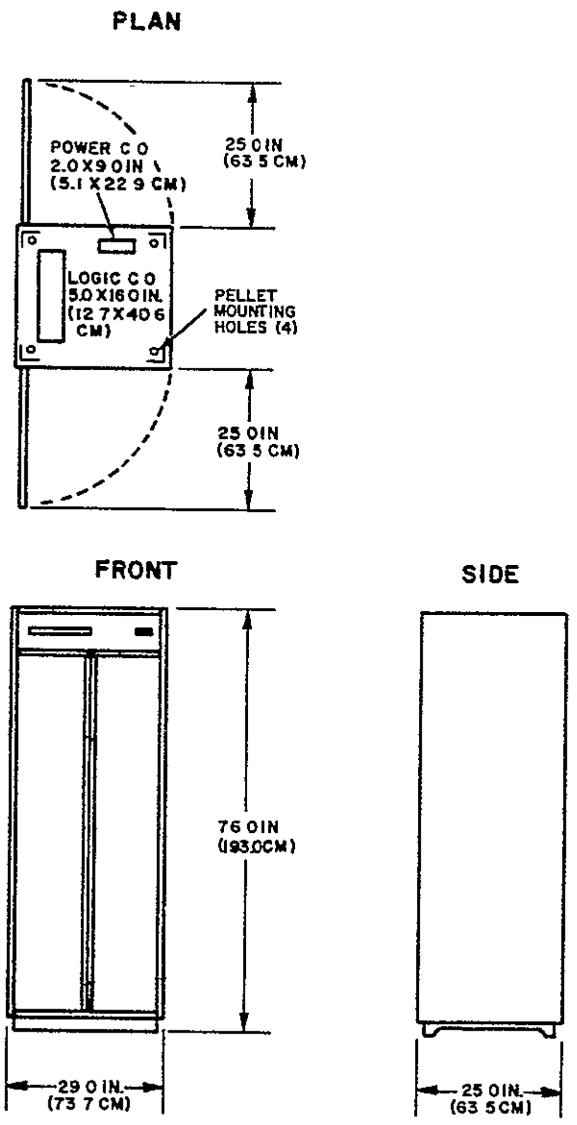
Type of cooling. Internal blowers

Source of cooling Ambient room air

Air required at inlet if plenum-cooled		N/A cfm (- m^3/hr)
Heat rejection rate, maximum	to air·	1400 Btu/hr (354 kcal/hr)
	to water	N/A Btu/hr (- kcal/hr)
Permissible range of room relative humidity		35% to 60%

	Maximum	Recommended	Minimum	Dew point limitation
Operating temperature (ambient or plenum inlet temperature)	90 °F (32.2 °C)	72°F (22 2°C)	59 °F (15 °C)	N/A
Storage temperature	158 °F (70 °C)		-40 °F (-40 °C)	

MASS STORAGE COUPLER



MASS STORAGE COUPLER (MSC)

Width	29.3 in. (74.5cm)	Weight (EST) 500 lb (227 kg)
Depth	25 in. (63.5cm)	Supported by four leveling pads, each
Height	66 in. (167.5cm)	5 in (12.7 cm) in diameter.

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum·

400 Hz, 208 v, 3 phase, 0.3 kva

and { 60 Hz, 120 v, 1 phase, 0.15 kva or 50 Hz, 220 v, 1 phase, 0.15 kva } NOTE: 50Hz input adjustable via taps for 120, 220, 230, 240 and 250 V.

Circuit breakers:	Power connector locations above cabinet base	
400 Hz, 15 amp, 3 phase	Terminal Strip	12in. (30.5 cm)
and { 60 Hz, 15 amp, 1 phase	Terminal Strip	} 12in. (30.5cm)
or		
50 Hz, 15 amp, 1 phase	Terminal Strip	

Control Data signal cables	Quantity·	Standard	Maximum length:
MSC to I/O Channel	2*	70 ft (21.3 m)	70ft (21.3m)
MSC to MSA	2	100 ft (30.5 m)	100ft (30.5m)

* Per Channel

Distance from internal signal cable connectors to floor 60in. (152 cm)

External terminator power connection required X no ___ yes (vdc)

Environmental considerations.

Type of cooling, forced air (internal fans)

Source of cooling, ambient room air

Heat rejection rate, maximum { to air, 1435 Btu/hr (362 kcal/hr) ; to water, N/A Btu/hr (- kcal/hr) }

Permissible range of Relative Humidity: 35 to 60 %

Maximum operating altitude: 8000 ft (2438 m)

	Maximum	Recommended	Minimum
Operating temperature	90 °F (32.2 °C)	72 °F (22.2 °C)	59 °F (15 °C)
Storage temperature	158 °F (70 °C)		-40 °F (-40 °C)

MASS STORAGE ADAPTER (MSA)

Width	53.6 in. (136.2 cm)	Weight	500lb (227 kg)
Depth	31 in. (78.7 cm)		
Height	71 in. (180.4 cm)		

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum:

60 Hz, 208 v,3 phase, 7.2kva
or
50 Hz, 380 v,3 phase, 7.2kva

NOTE: 60Hz is 4-wire Delta
50Hz is 5-wire Wye

Circuit breakers:

60 Hz, 30 amp, 3 phase
or
50 Hz, 30 amp, 3 phase

Power Connector:

Located on 60 in (152 cm) drop cord with Hubble L21-30 connector (CDC P/N 15005700) Attached, mating receptacle supplied is CDC P/N 15005900.

Control Data signal cables.	Quantity.	Standard	Maximum length
MSA to CSU	1	100 ft (30.5 m)	200 ft (61 m)
MSA to MST	1	100ft (30.5 m)	200ft (61 m)

Distance from internal signal cable connectors to floor: - 12 in (30.5 cm)
External terminator power connection required X no yes (vdc)
Environmental considerations

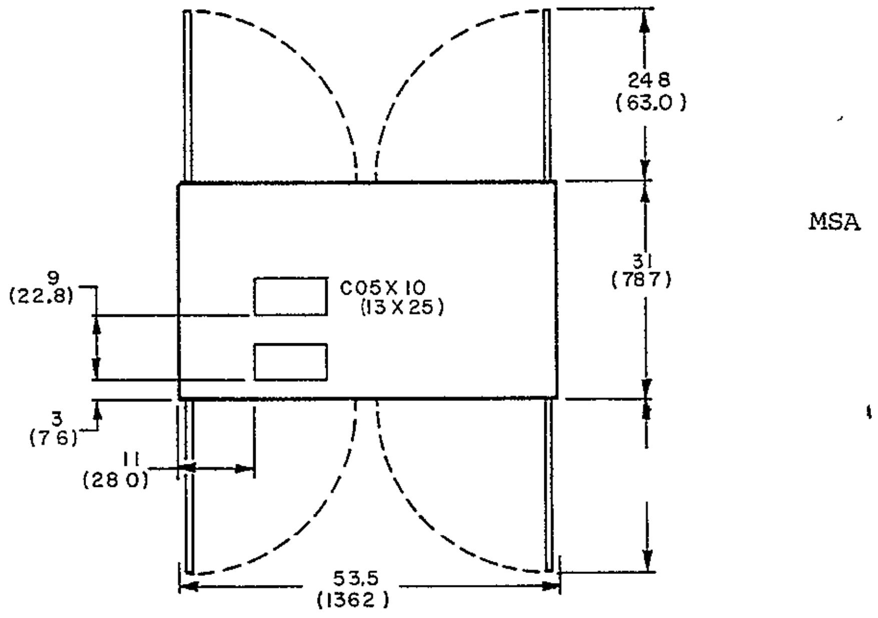
Type of cooling, forced air (internal fans)
Source of cooling, ambient room air
Maximum operating altitude:

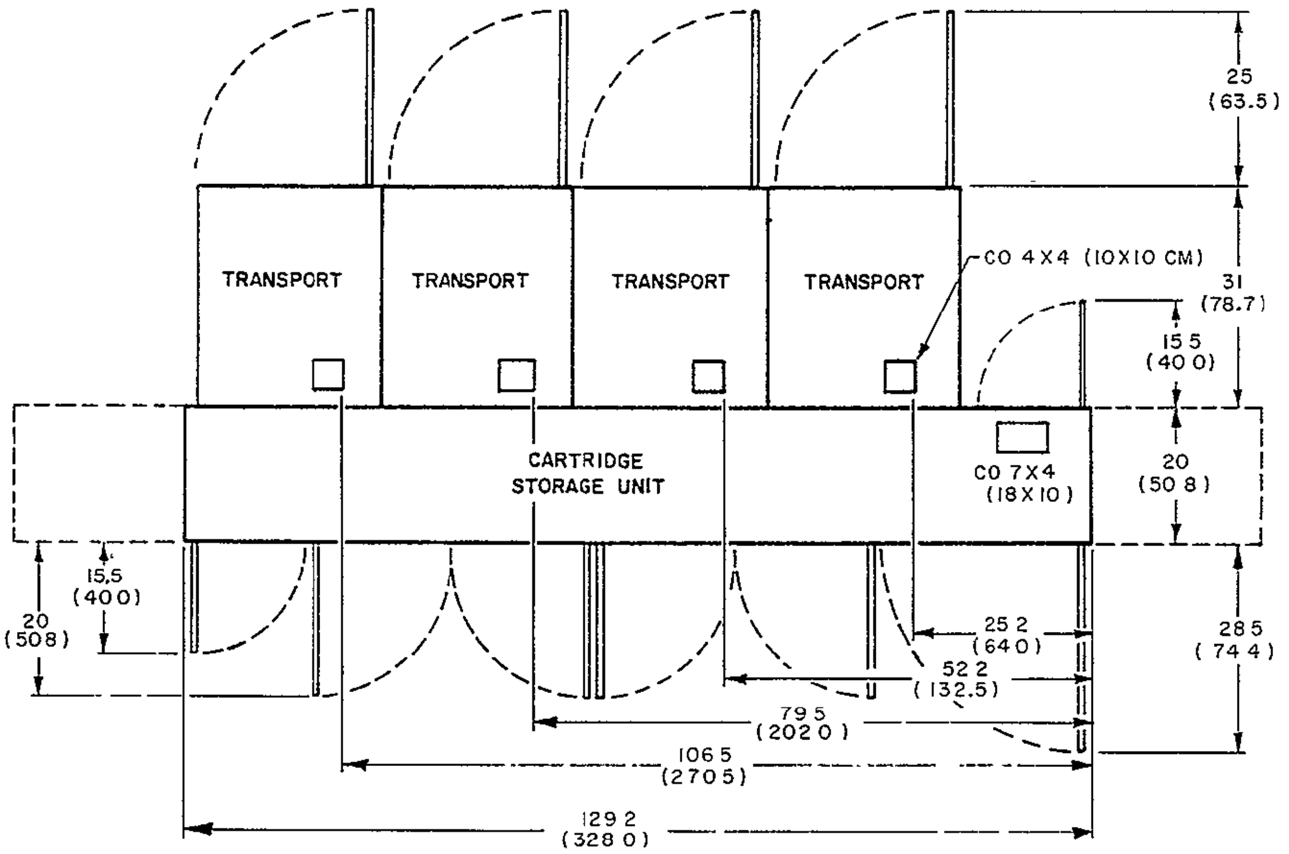
Heat rejection rate, maximum
- to air, 19,700 Btu/hr (4985 kcal/hr)
- to water, N/A Btu/hr (- kcal/hr)

Permissible range of Relative Humidity: 10 to 80%

	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	90 °F (32.2 °C)	72 °F (22.2 °C)	60 °F (15.2°C)	N/A °F (- °C)
Storage temperature	150 °F (°C)		-30 °F (°C)	

FLOOR PLAN CUTOUTS
(MSA, CSU, and Maximum MST Configuration)





MASS STORAGE TRANSPORT (MST)

Width	26.6 in. (78.7 cm)	Weight	700lb (318 kg)
Depth	31 in. (67.6 cm)	Supported by the frame	
Height	71 in. (180.3 cm)		

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum:

{60 Hz, 208 v, 3 phase, 2.1 kva
or
50 Hz, 380 v, 3 phase, 2.1 kva} NOTE: 60Hz is 4-wire Delta; 50Hz is 5-wire Wye

Circuit breakers

{60 Hz, 15 amp, 3 phase
or
50 Hz, 15 amp, 3 phase} Located on 60 in (152 cm) drop cord with Hubble L21-30 connector (CDC P/N 15005700) Attached, mating receptacle supplied is CDC P/N 15005900.

Control Data signal cables

See MSA Data Sheet

Distance from internal signal cable connectors to floor: 48 in (172 cm)

External terminator power connection required X no yes (vdc)

Environmental considerations

Type of cooling, forced air (internal fans)

Source of cooling, ambient room air

Heat rejection rate, maximum {to air, 6150 Btu/hr (1555 kcal/hr); to water, N/A Btu/hr (– kcal/hr)}

Permissible range of Relative Humidity: 30 to 80%

	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	90 °F (32 °C)	72 °F (22.2 °C)	60 °F (15 °C)	N/A °F (– °C)
Storage temperature	122 °F (50 °C)		14 °F (-10 °C)	

CARTRIDGE STORAGE UNIT (CSU)

Width	129.25 in. (328 cm)	Weight	3300 lb (1500 kg)
Depth	20 in. (50.8 cm)	Supported by the frame	
Height	76.25 in. (179 cm)		

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum:

{60 Hz, 208 v, 3 phase, 3.6 kva
or
50 Hz, 380 v, 3 phase, 3.6 kva}

NOTE: 60Hz is 4-wire Delta; 50Hz is 5-wire Wye

Circuit breakers·

{60 Hz, 15 amp, 3 phase
or
50 Hz, 15 amp, 3 phase}

Located on 60 in (152 cm) drop cord with Hubble L21-30 connector (CDC P/N 15005700) Attached, mating receptacle supplied is CDC P/N 15005900.

Control Data signal cables.

See MSA Data Sheet

Distance from internal signal cable connectors to floor: 48 in (172 cm)
External terminator power connection required´ X no yes (vdc)
Environmental considerations.

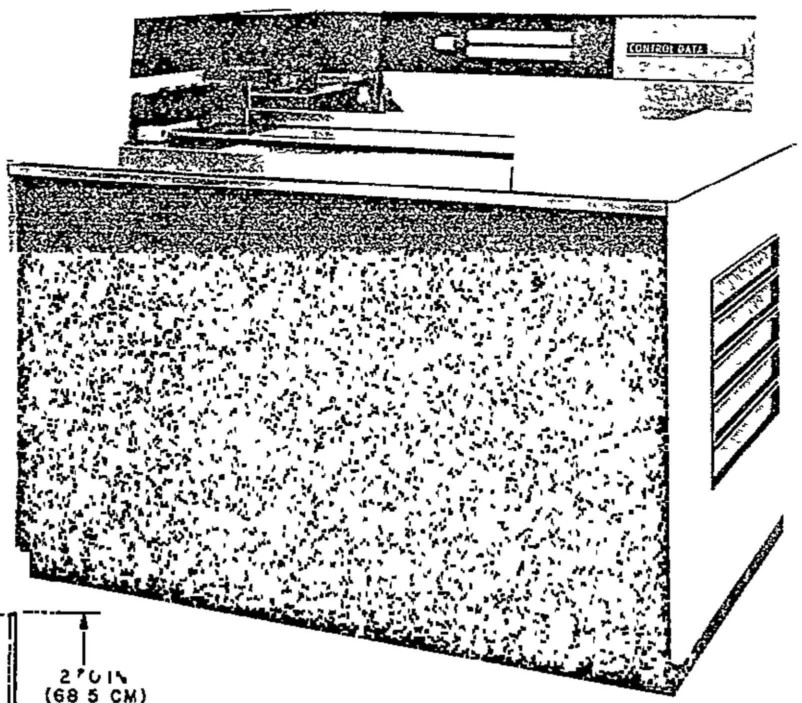
Type of cooling, forced air (internal fans)
Source of cooling, ambient room air

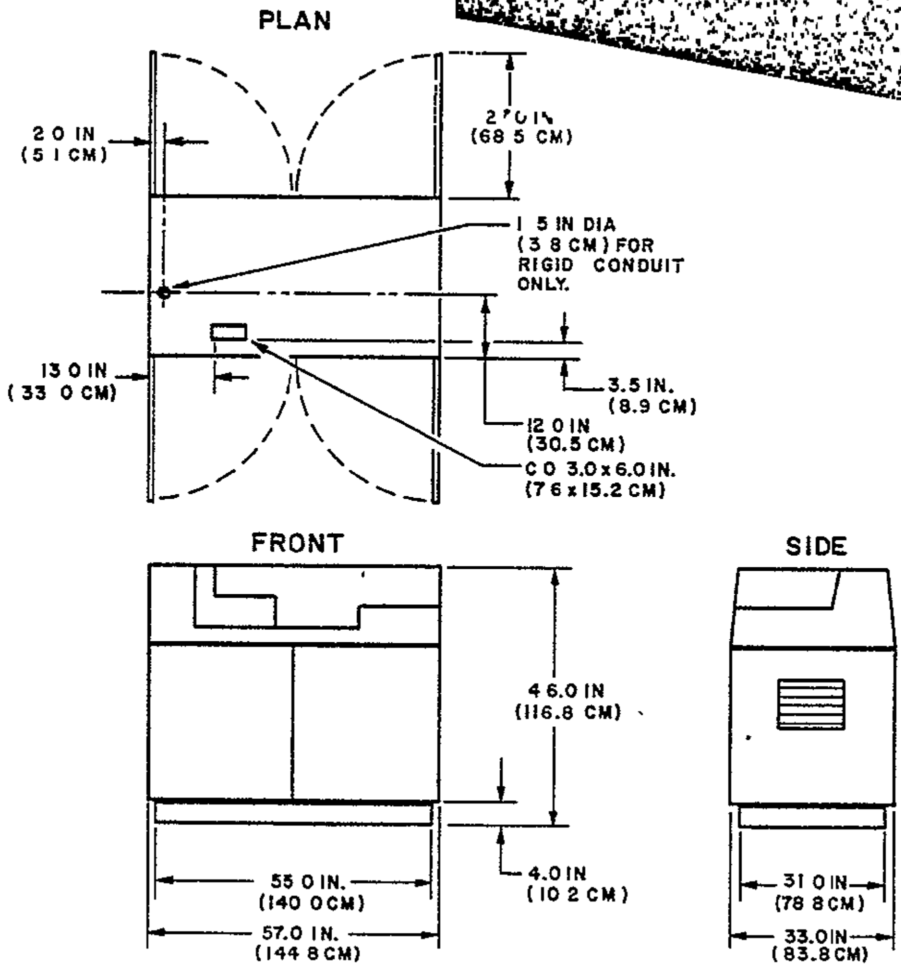
Heat rejection rate, maximum {to air, 10,585 Btu/hr (2680 kcal/hr); to water, N/A Btu/hr (- kcal/hr)}

Permissible range of Relative Humidity: 30 to 80%

	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	90 °F (32 °C)	72 °F (22.2 °C)	60 °F (15 °C)	N/A °F (- °C)
Storage temperature	122 °F (50 °C)		14 °F (-10 °C)	

405 CARD READER





405 CARD READER

Width	57.0	in. (144.8 cm)	Weight	1020 lb (463 kg)	
Depth	33.0	in. (83.8 cm)	Supported by 4 casters		
Height	46.0	in. (116.8 cm)			

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum

60 Hz, 208 v, 3 phase, 3.4 kva
or
50 Hz, 208 v, 3 phase, 3.4 kva

Circuit breakers — Power connector locations above cabinet base

60 Hz, 15 amp, 3 phase	Locking connector	60 in. (152 cm)
or		
50 Hz, 15 amp, 3 phase	Locking connector	

Control Data signal cables· Quantity: Standard length: Maximum length

N/A

Distance from internal signal cable connectors to floor 10 in. (25 cm)
External terminator power connection required no X yes (40 vdc)*
Environmental considerations

Type of cooling, internal fan
Source of cooling, room air

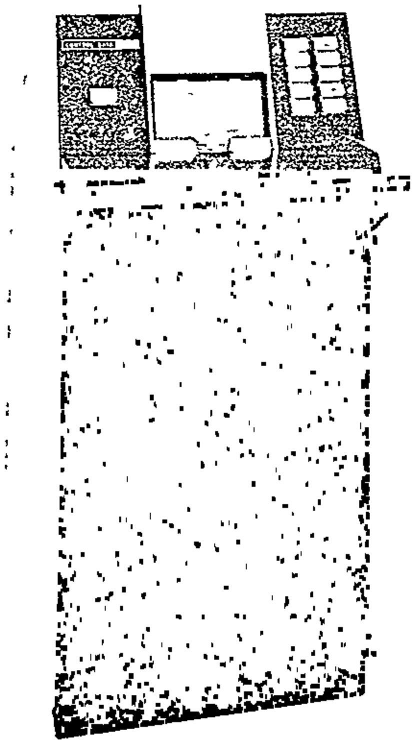
Heat rejection rate, maximum { to air, 9000 Btu/hr (2250 kcal/hr) ; to water, N/A }

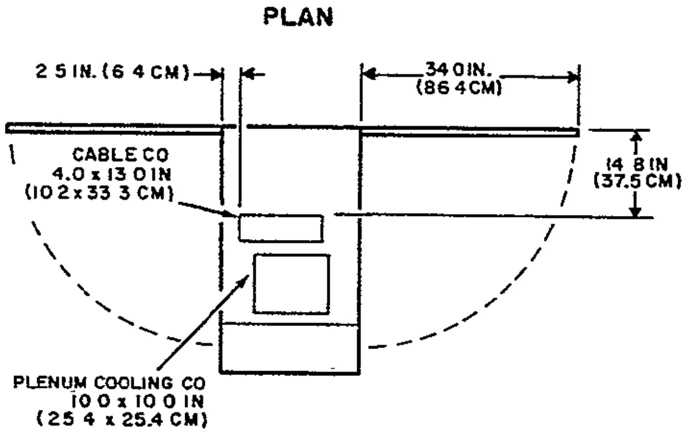
Permissible range of room relative humidity, 35 % to 60 %

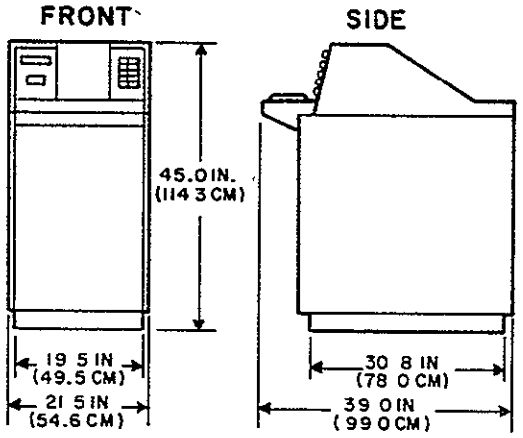
	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	74 °F (23.3 °C)	72 °F (22.2 °C)	62 °F (16.7 °C)	N/A °F (°C)
Storage temperature	120 °F (48.9 °C)		-30 °F (-34.4 °C)	

*For controller only.

415-30 CARD PUNCH CONTROLLER




PLAN
2 5 IN. (6 4 CM)
34 0 IN. (86 4 CM)
14 8 IN (37.5 CM)
CABLE CO 4.0 x 13 0 IN (10 2 x 33 3 CM)
PLENUM COOLING CO 10 0 x 10 0 IN (25 4 x 25.4 CM)



415-30 CARD PUNCH CONTROLLER

Width	21.5	in. (54.6 cm)	Weight	570	lb (258.5kg)
Depth	39.0	in. (99.0 cm)	Supported by 4 casters		
Height	45.0	in. (114.3 cm)			

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum.

60 Hz, 120 v, 1 phase, 1.5 kva
or
50 Hz, 120 v, 1 phase, 1.5 kva

Circuit breakers.	Power connector locations above cabinet base	
60 Hz, 15 amp, 1 phase	Terminal strip	10 in. (25 cm)
or		
50 Hz, 15 amp, 1 phase	Terminal strip	

Control Data signal cables	Quantity:	Standard length:	Maximum length.
415-30 to 3000 Data Channel	2	40 ft (12.2 m)	200 ft (60.9 m)

Distance from internal signal cable connectors to floor 24 in. (60.9 cm)
External terminator power connection required: no X yes (40 vdc)
Environmental considerations:

Type of cooling, internal fan
Source of cooling, room air

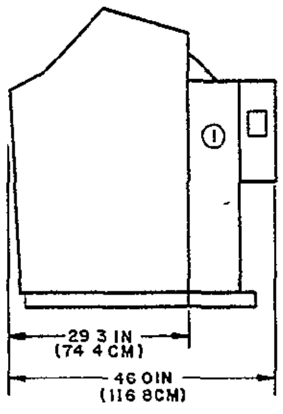
Heat rejection rate, maximum { to air, 3585 Btu/hr (896 kcal/hr) ; to water, N/A }

Permissible range of room relative humidity, 35 % to 60 %

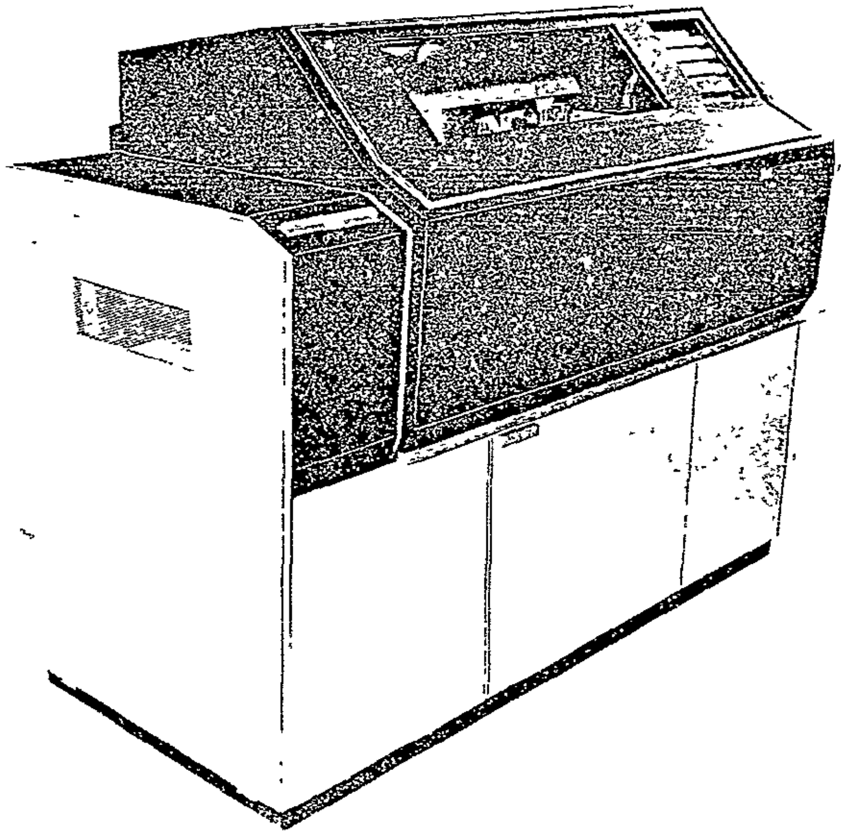
	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	90 °F (32 °C)	72 °F (22 °C)	59 °F (15 °C)	N/A °F (°C)
Storage temperature	122 °F (-50 °C)		14 °F (-10 °C)	

580-20 TRAIN PRINTER SUBSYSTEM

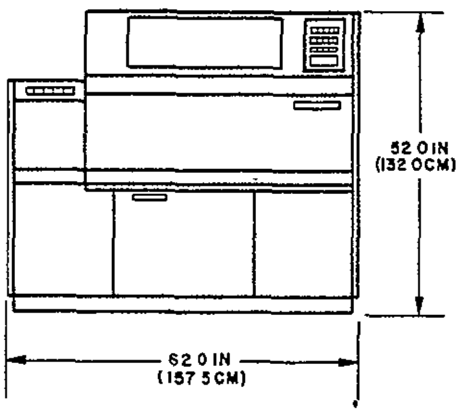
SIDE



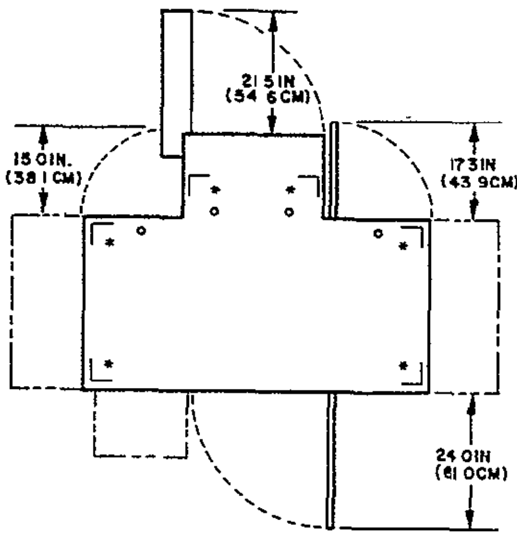
NOTE:
1 STACKER REMOVABLE FOR MOVING PRINTER.



FRONT



PLAN



580-20 TRAIN PRINTER SUBSYSTEM

Width	62.0	in. (157.5cm)	Weight	1500 lb (682 kg)
Depth	46.0	in. (116.8cm)	Supported by frame, 4 casters	
Height	52.0	in. (132 cm)		

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum.

60 Hz, 208 v, 3 phase, 5.2 kva
or
50 Hz, 380 v, 3 phase, 5.2 kva

Circuit breakers. Power connector locations above cabinet base

and	60 Hz, 30 amp, 3 phase	Terminal strip	60 in. (152 cm)*
	or		
	50 Hz, 30 amp, 3 phase	Terminal strip	

Control Data signal cables:	Quantity:	Standard length:	Maximum length·
Logic	2	40 ft (12.19 m)	200 ft (60.96 m)

Distance from internal signal cable connectors to floor 30 in. (80 cm)*
External terminator power connection required· no X yes (40 vdc)
Environmental considerations.

Type of cooling, internal fan
Source of cooling, room air

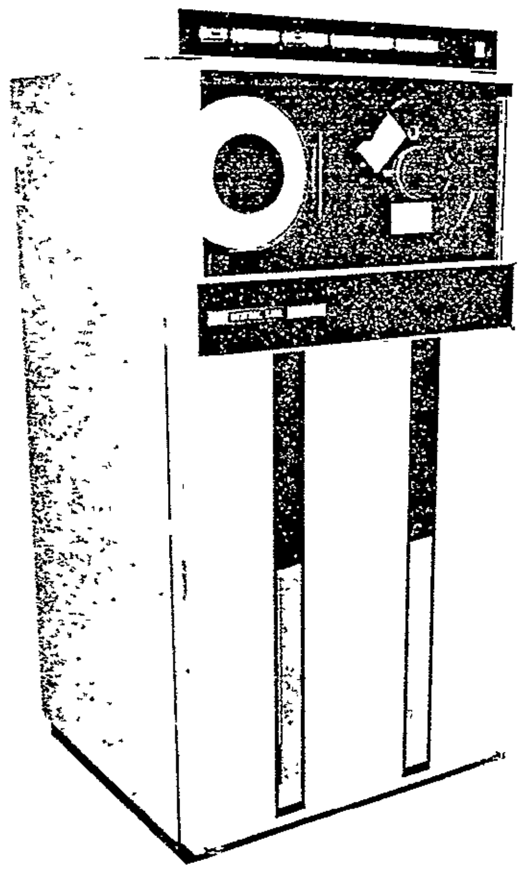
Heat rejection rate, maximum
- to air, 15,000 Btu/hr (3750 kcal/hr)
- to water, N/A

Permissible range of room relative humidity, 30 % to 80 %

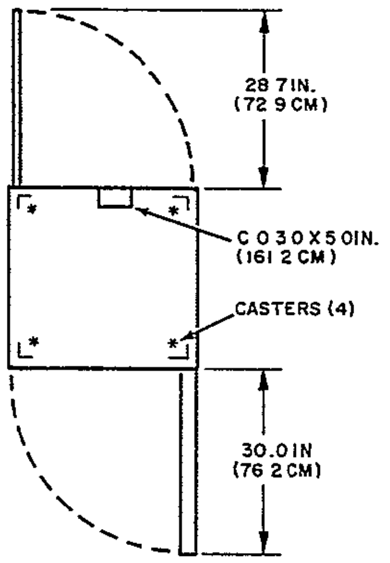
	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	90 °F (32.2 °C)	72 °F (22 °C)	60°F (15.6°C)	N/A°F (- °C)
Storage temperature	150 °F (65.6 °C)		-30°F (-34°C)	

* Estimated

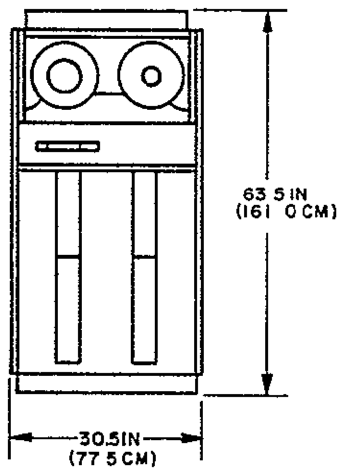
667-2/3/4 AND 669-2/3/4 MAGNETIC TAPE TRANSPORT



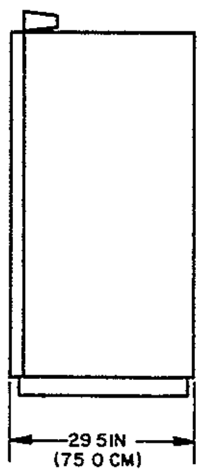
PLAN



FRONT



SIDE



667-2/3/4 AND 669-2/3/4 MAGNETIC TAPE TRANSPORT

Width	30.5	in. (77.5 cm)	Weight	900	lb (410 kg)
Depth	29.5	in. (75.0 cm)	Supported by 4 casters		
Height	63.5	in. (161.0 cm)			

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum

60 Hz, 208 v, 3 phase, 3.0 kva
or
50 Hz, 380 v, 3 phase, 3.0 kva

Circuit breakers

60 Hz, 15 amp, 3 phase
or
50 Hz, 15 amp, 3 phase

Power connector locations above cabinet base·

72-inch line cord from lower rear of cabinet and locking connector N/A in. (cm)

Control Data signal cables	Quantity:	Standard length:	Maximum length.
Logic cable	1	50 ft (15.24 m)	125 ft (38.1 m)

Distance from internal signal cable connectors to floor· 10 in. (25 cm)

External terminator power connection required X no yes (vdc)

Environmental considerations·

Type of cooling, internal fan

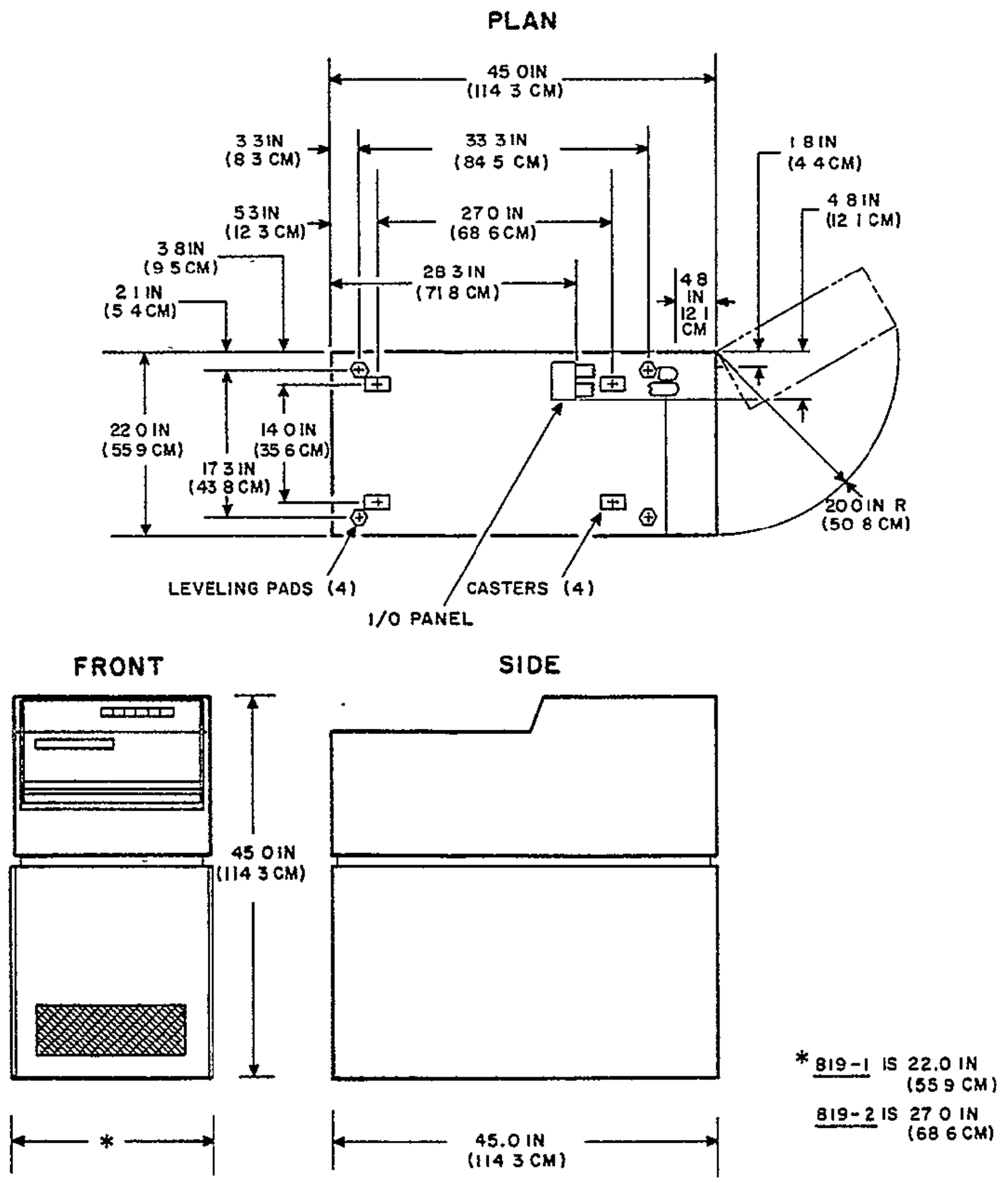
Source of cooling, room air

Heat rejection rate, maximum { to air, 9900 Btu/hr (2490 kcal/hr) ; to water, N/A }

Permissible range of room relative humidity, 30 % to 80 %

	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	90 °F (32.2 °C)	72 °F (22 °C)	60 °F (15.6 °C)	N/A °F (°C)
Storage temperature	150 °F (65.6 °C)		-30 °F (-34 °C)	

819-1/2 DISK STORAGE UNIT



* 819-1 IS 22.0 IN (55 9 CM)
819-2 IS 27 0 IN (68 6 CM)

819-1/2 DISK STORAGE UNIT

Width 22.0 in. (55.9 cm) 819-1			
Width 27.0 in. (68.6 cm) 819-2	Weight	800 lb (363 kg)	
Depth 45.0 in. (114.3 cm)	Supported by 4 leveling pads		
Height 45.0 in. (114.3 cm)			

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum:

60 Hz, 208 v, 3 phase, 5.0 kva
or
50 Hz, 208 v, 3 phase, 5.0 kva

Circuit breakers. Power connector locations above cabinet base.

60 Hz, 20 amp, 3 phase Locking Connector
or
50 Hz, 20 amp, 3 phase Locking Connector } 4 in. (10.2cm)

Control Data signal cables:	Quantity:	Standard length:	Maximum length.
Controller/DSU	2	50 ft (15.2 m)	60 ft (18.3 m)

Distance from internal signal cable connectors to floor· 4 in. (10.2cm)
External terminator power connection required X no yes (vdc)
Environmental considerations·

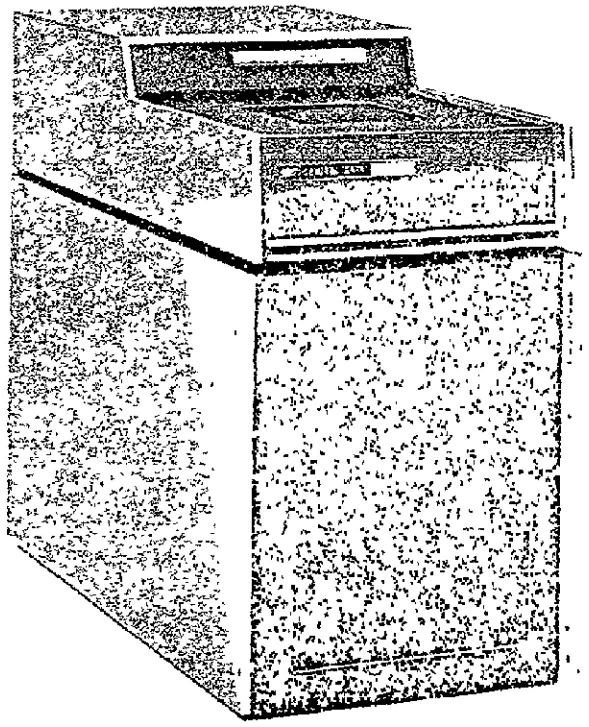
Type of cooling, internal fan
Source of cooling, room air

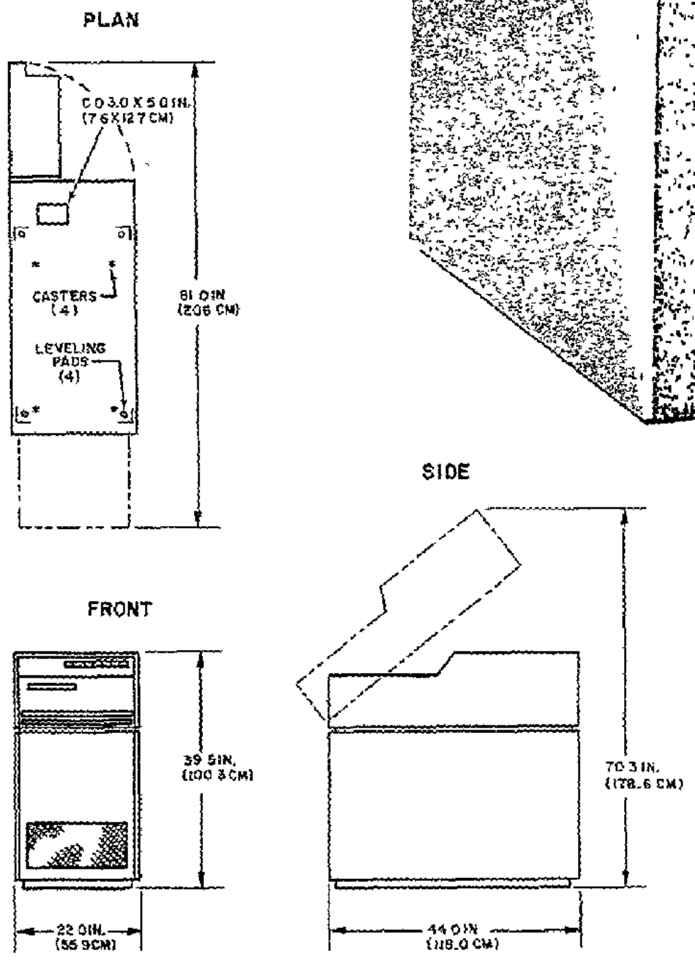
Heat rejection rate, maximum { to air, 10,000 Btu/hr (2500 kcal/hr) ; to water, N/A

Permissible range of room relative humidity, 30 % to 80 %

	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	90 °F (31.7 °C)	75 °F (23.9 °C)	60 °F (15.5 °C)	N/A °F (°C)
Storage temperature	150 °F (65.5 °C)		-30 °F (-34 °C)	

844-21 DISK STORAGE UNIT





844-21 DISK STORAGE UNIT

Width	22.0 in. (55.9 cm)	Weight	700 lb (315 kg)
Depth	44.0 in. (111.8 cm)	Supported by	4 casters, 4 leveling pads
Height	39.5 in. (100.3 cm)		

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum.

and { 60 Hz, 208 v, 1 phase, 1.35 kva or 50 Hz, 220 v, 1 phase, 1.65 kva }

Circuit breakers·

Power connector locations above cabinet base.

and { 60 Hz, 20 amp, 1 phase — Locking connector or 50 Hz, 20 amp, 1 phase — Locking connector } 0 in. (0 cm)

Control Data signal cables:	Quantity:	Standard length:	Maximum length:
Controller to disk drive unit	2	50 ft (15.2 m)	75 ft (23 m)

Distance from internal signal cable connectors to floor 10 in. (254 cm)

External terminator power connection required: X no yes (vdc)

Environmental considerations:

Type of cooling, internal fans

Source of cooling, room air

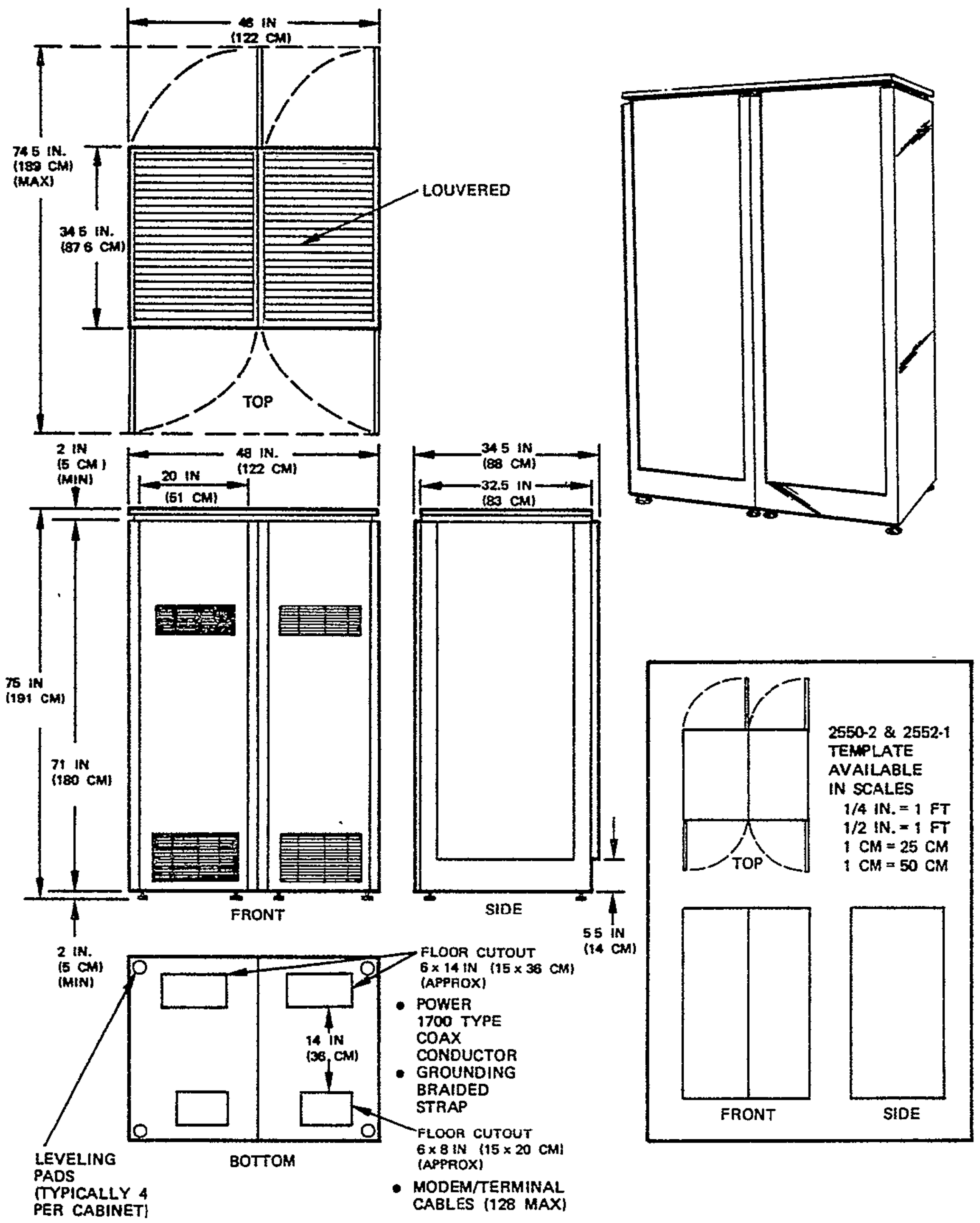
Heat rejection rate, maximum { to air, 4500 Btu/hr (1125 kcal/hr) to water, N/A }

Permissible range of room relative humidity, 20 % to 80 % 844-21

	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	90 °F (32 °C)	75 °F (24 °C)	60 °F (16 °C)	N/A °F (°C)
Storage temperature	150 °F (66 °C)		-30 °F (-34 °C)	

Maximum operating altitude 10,000 feet above sea level.

2550-2/2552-1 HOST COMMUNICATIONS PROCESSOR



2550-2/2552-1 HOST COMMUNICATIONS PROCESSOR

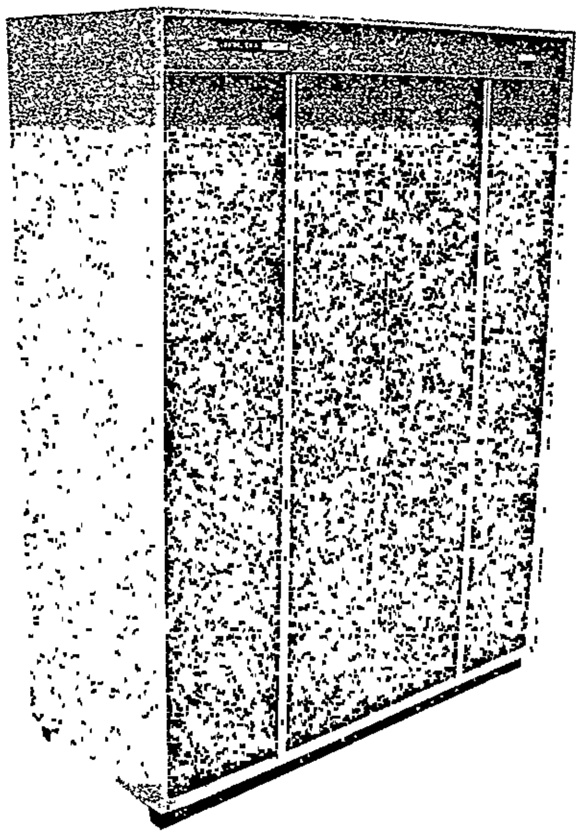
Physical and Functional Characteristics	Model 2550-2	Model 2552-1
Width†	48 in. (1.22 m)	48 in. (1.22 m)
Depth†	34 in. (0.86 m)	34 in. (0.86 m)
Height	75 in. (1.91 m)	75 in. (1.91 m)
Weight (est. maximum - all options present; supported by leveling pads)	950 lb (431 kg)	1050 lb (476 kg)
Heat Dissipation (est.)	13,300 BTU/hr (3350 kg - cal/hr)	15,400 BTU/hr (3870 kg - cal/hr)
Power (maximum)	3.9 KVA	4.5 KVA
Consumption (est.)	Varies with exact configuration	Varies with exact configuration
Power Source††		
60 Hz†††	208Y/120v, 3 phase, 20 amp	208Y/120v, 3 phase, 20 amp
50 Hz	120v+N, single phase, 40 amp	120v+N, single phase, 40 amp
Cabinet Circuit Breaker (60 Hz)	20 amp, 3 phase, wye, 5-wire	20 amp, 3 phase, wye, 5-wire
CLA Signal Cables (50 feet max.)	1 thru 128 lines	1 thru 128 lines
Cooling-Fans (Internal) Room Ambient	1200 CFM	1200 CFM
Temperature Range	From +50°F (+10°C) to +95°F (+35°C)	From +50°F (+10°C) to +95°F (+35°C)
Recommended Temperature	+72°F (+22°C)	+72°F (+22°C)
Humidity	20% to 80% (no condensation)	20% to 80% (no condensation)
Caustic chemical Environment	Not allowed	Not allowed

†For maximum width and depth, all doors extended, see figure.

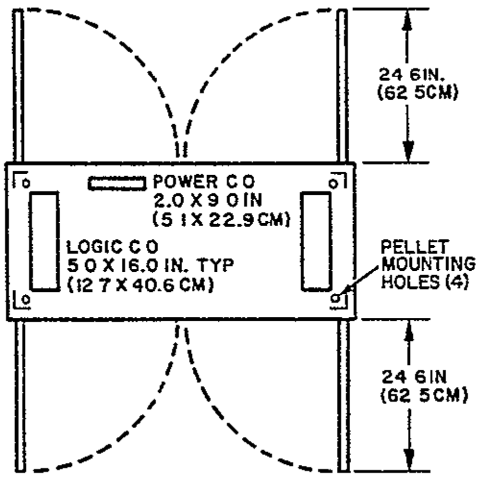
††See figures 3-2 and 3-3 for optional power sources.

†††Can also be wired for 120/240, 2 phase, 30 amp service.

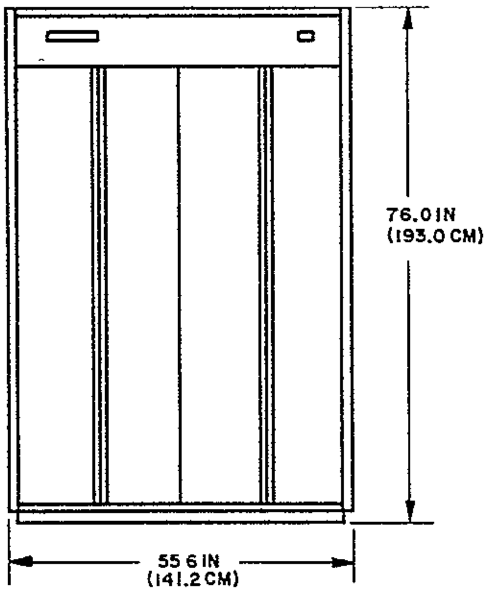
7021-22 MAGNETIC TAPE CONTROLLER



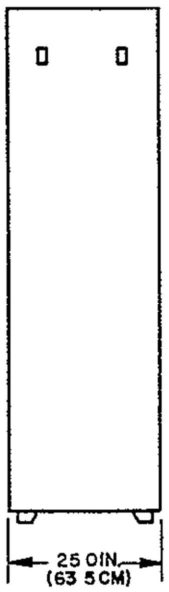
PLAN



FRONT



SIDE



7021-22 MAGNETIC TAPE CONTROLLER

Width	55.6 in. (141.2 cm)	Weight	850 lb (386 kg)
Depth	25.0 in. (63.5 cm)	Supported by	frame
Height	76.0 in. (193.0 cm)		

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum

400 Hz, 120/208 v, 3 phase, 1.0 kva

and { 60 Hz, 120 v, 1 phase, 0.4 kva
or
50 Hz, 220/250 v, 1 phase, 0.4 kva }

Circuit breakers		Power connector locations above cabinet base	
	400 Hz, 15 amp, 3 phase	Terminal strip	4 in. (10 cm)
and {	60 Hz, 15 amp, 1 phase	Terminal strip	} 4 in. (10 cm)
	or		
	50 Hz, 15 amp, 1 phase	Terminal strip	

Control Data signal cables:	Quantity:	Standard length:	Maximum length:
7021-22 to central computer	1	65 ft (19.8 m)	65 ft (19.8 m)
7021-22 to tape unit	1 per unit	50 ft (15.2 m)	125 ft (38.1 m)

Distance from internal signal cable connectors to floor: 72 in. (183 cm)

External terminator power connection required: X no yes (vdc)

Environmental considerations

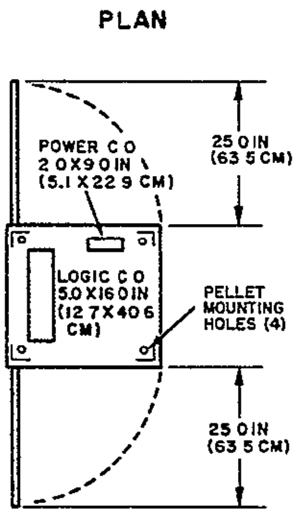
Type of cooling, internal fan

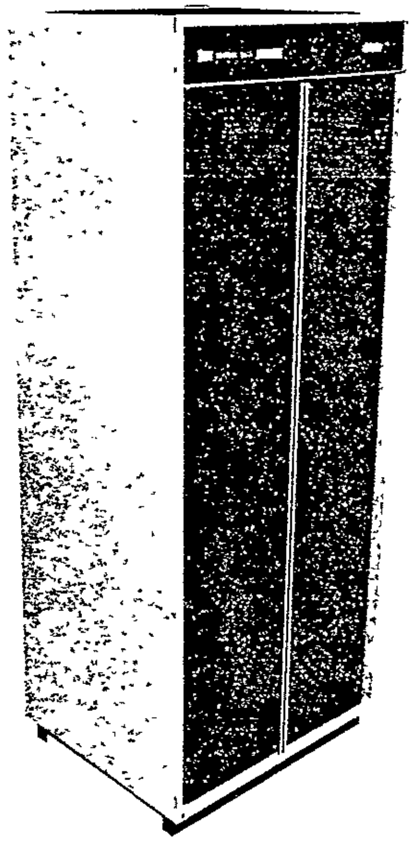
Source of cooling, room air

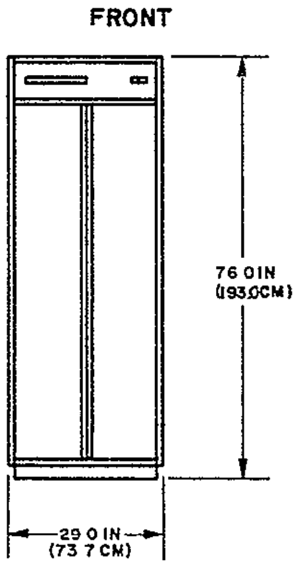
Heat rejection rate, maximum { to air, 4370 Btu/hr (1090 kcal/hr) ; to water, N/A }

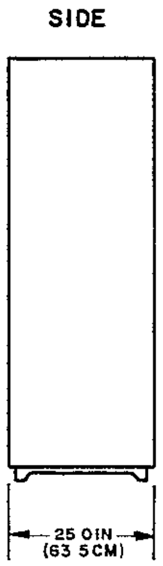
Permissible range of room relative humidity, 30 % to 80 %

	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	90 °F (32.2 °C)	72 °F (22.2 °C)	60 °F (15.6 °C)	N/A °F (°C)
Storage temperature	150 °F (65.6 °C)		-30 °F (-34.4 °C)	


PLAN
POWER C O
2 0X9 0 IN
(5.1 X 22 9 CM)
25 0 IN
(63 5 CM)
LOGIC C O
5.0 X16 0 IN
(12 7X40 6 CM)
PELLET MOUNTING HOLES (4)
25 0 IN
(63 5 CM)




FRONT
76 0 IN
(193.0CM)
29 0 IN
(73 7 CM)


SIDE
25 0 IN
(63 5CM)

7154-1/2/3/4 MASS STORAGE CONTROLLER

Width	29	in. (	74	cm)	Weight	300	lb (	136 kg)
Depth	25	in. (	64	cm)	Supported by	frame		
Height	76	in. (	193	cm)				

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum:

	400 Hz,	120/208 V,	3 phase,	0.6 kVA		
and	60 Hz,	120 V,	1 phase,	0.2 kVA		
	or					
	50 Hz,	220 V,	1 phase,	0.2 kVA		

Circuit breakers: Power connector locations above cabinet base:

	400 Hz,	15 amp,	3 phase	TB5-1, 2, 3, 4	12 in. (30.5 cm)
and	60 Hz,	15 amp,	1 phase	TB4-1, 2	12 in. (30.5 cm)
	or				
	50 Hz,	15 amp,	1 phase	TB4-1, 2	

Control Data signal cables:		Quantity:	Standard length:	Maximum length:
7154	DSU	1 50p	70ft (21.3 m)	70ft (21.3 m)
7154	DSU	1 75p	70ft (21.3 m)	70ft (21.3 m)
7154	I/O Channel	2 19p	70ft (21.3 m)	70ft (21.3 m)

Distance from internal signal cable connectors to floor: 60 in. (152.4 cm)

External terminator power connection required: X no

Environmental considerations:

Type of cooling, internal fans

Source of cooling, ambient room air

Total fan capacity 350 cfm (595 m^3/hr)

Heat rejection rate, maximum { to air, 2400 Btu/hr (600 kcal/hr) ; to water, N/A Btu/hr (kcal/hr) }

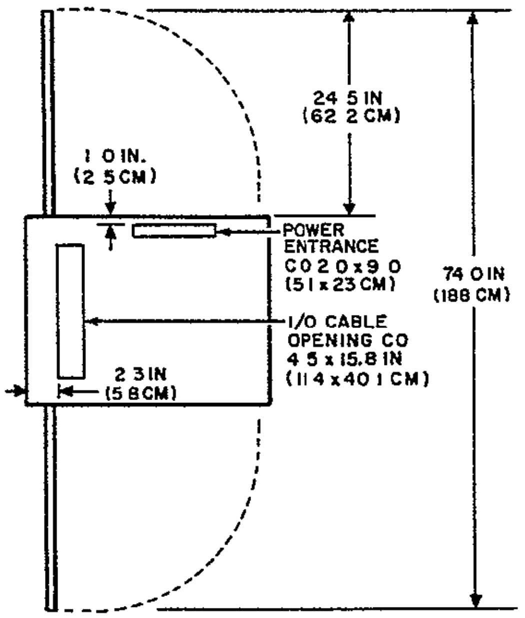
Permissible range of room relative humidity 35% to 60%

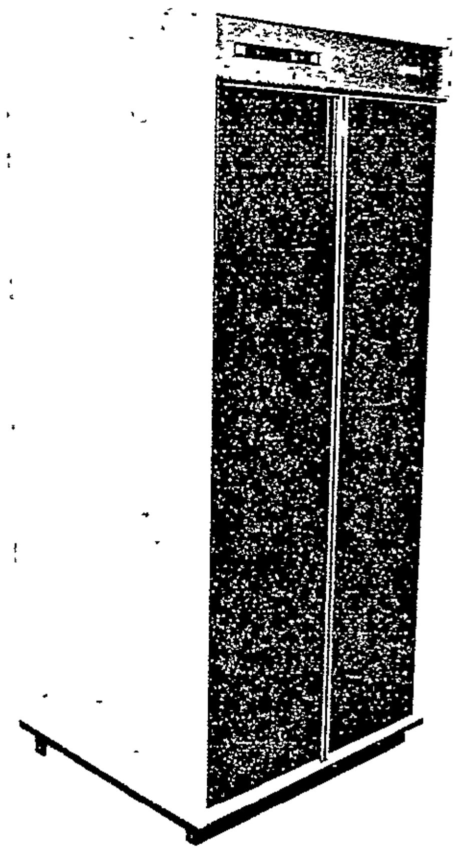
	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	90°F (32.2°C)	72°F (22 2°C)	59°F (15.0°C)	N/A°F (°C)
Storage temperature	150°F (65.6°C)		-40°F (-40°C)	

Maximum operating altitude 9,000 feet (2743M) above sea level.

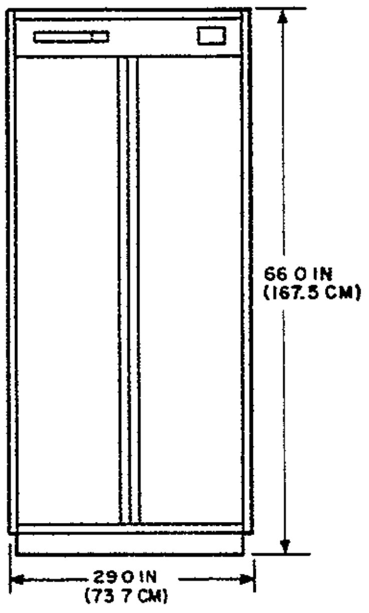
7639-1/2 CONTROLLER

PLAN

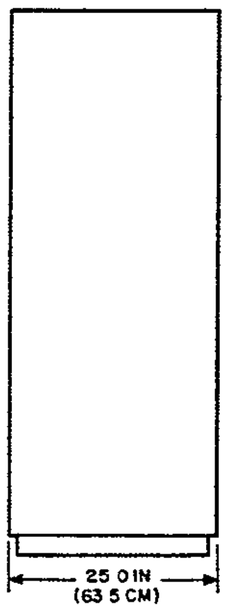




FRONT



SIDE



7639-1/2 CONTROLLER

Width	29 in. (73.7 cm)	Weight	375 lb (170.1 kg) (450 lb., 204 kg*)
Depth	25 in. (63.5 cm)	Supported by	N/A
Height	66 in. (167.5 cm)		

For maximum width/depth, all doors extended, refer to floor plan layout.

Power consumption, steady state, maximum:

400 Hz, 208 v, 3 phase, 0.5 kva (0.7 kva*)

and { 60 Hz, 120 v, 1 phase, 0.2 kva
or
50 Hz, 220 v, 1 phase, 0.2 kva }

Circuit breakers:	Power connector locations above cabinet base:
400 Hz, 15 amp, 3 phase	12 in. (30.5 cm)
50/60 Hz, 15 amp, 1 phase	12 in. (30 5 cm)

Control Data signal cables.	Quantity:	Standard length:	Maximum length:
Controller/PPU	16 max (32*)		60 ft (18.3 m)
Controller/DSU	8 max (16*)		70 ft (21.3 m)

Distance from internal signal cable connectors to floor. 48 in. (123 cm)

External terminator power connection required· X no yes (vdc)

Environmental considerations:

Type of cooling, internal fan

Source of cooling, room air

Heat rejection rate, maximum { to air, 2250 (2840*) Btu/hr (542 kcal/hr*) (715 kcal/hr)
to water, N/A }

Permissible range of room relative humidity, 10 % to 90 %

	Maximum	Recommended	Minimum	Dew Point Limitation
Operating temperature	90 °F (31.7 °C)	75 °F (23.9 °C)	60 °F (15.5 °C)	N/A °F (°C)
Storage temperature	150 °F (65.5 °C)		-30 °F (-34.5 °C)	

* Data differs between the 7639-1 and -2. Figures marked with an asterisk indicate the 7639-2.

125-KVA, 60-HZ KATO M-G SET (SPEC. NO. 52316700)

MOTOR GENERATOR

Width	70.5 in. (180 cm)	Weight	4075 lb (1900 kg)
Depth	42.5 in. (108 cm)	Supported by 4 pads	
Height	37.5 in. (96 cm) includes lifting eye	with areas 30 in.^{2} each (193 cm^2)	

Power Specifications:

	Motor	Generator
Rating	start motor 40 hp, run motor 200 hp	125 kVa
Frequency	60 (±3) Hz	400 Hz
Voltage, nominal	460 (±10%)V	120 and 208V
Full-load current per phase	218 amp	347 amp
Starting current, locked rotor	280 amp	
First-cycle peak of 3-cycle asymmetrical surge current	4360 amp	
Power factor at		
0.5 Generator load	82 %	
0.7 Generator load	87 %	
1.0 Generator load	92 %	
Distance from internal power cable connections to floor	motor 18.1 and 30.1 in. (46 and 76 cm) generator 19.1 and 30.1 in. (49 and 76 cm)	

Environmental considerations:

Type of cooling, turbo centrifugal fan

Source of cooling air, room

Air displacement at 60 Hz, 3450 cfm (5850 m^3/hr)

Heat rejection rate 79,000 Btu/hr (19,900 kcal/hr)

Permissible range of room relative humidity 10% to 90%
(Providing no condensation occurs)

	Maximum	Recommended	Minimum
Operating temperature	+104°F (+40°C)	+77°F (+25°C)	-65°F (+18°C)
Storage temperature	+125°F (+52°C)		-65°F (-54°C)

125-KVA, 60-HZ KATO M-G SET (SPEC. NO. 52316700)

MOTOR-GENERATOR CONTROL CABINET

Width	56 in. (143 cm)	Weight	1050 lb (475 kg)
Depth	20 in. (51 cm)	Supported by	cabinet frame
Height	81 in. (207 cm) includes lifting eye		

Power specifications

Main power, incoming	60 Hz,	460V,	3 phase,	3 wires
Remote on/off control, incoming	60 Hz,	120V,	1 phase,	2 wires

Line protection — Three-phase gang circuit breaker, manually-operated; size determined by MG and incoming line voltage. Fusible switch disconnects are not preferred.

Line restrictions — Line sizes, lengths, and quality for each phase shall not cause voltage losses between generator and loads to exceed 2 percent of the generator output voltage.

Power connections

		Wires per Run	Recommended Wire Size	Distance from Connector to Cabinet Base
Main power	460V	3	300 MCM (155 mm^2) based on max of 272 amp	45 in. (114 cm)
Start-motor power ③	460V	3	4 AWG (21.2 mm^2)	10 in. (26 cm)
Run-motor power ③	460V	3	300 MCM (155 mm^2)	8 in. (21 cm)
Exciter-field power		2	14 AWG (2.08 mm^2)	60 in. (153 cm)
Generator-output power ④		8	two 0000 AWG (107 mm^2 ea)	50 in. (127 cm)
Power to load ⑤ ⑥ ⑦		16	four 0 AWG (53.3 mm^2 ea)	22 in. (56 cm)
Remote on/off control		2	14 AWG (2.08 mm^2)	56 in. (142 cm)
Remote voltage adjust		3	14 AWG (2.08 mm^2)	56 in. (142 cm)
Standby-to-normal 400-Hz power ⑧		8	two 0000 AWG (107 mm^2 ea)	28 in. (71 cm)
Standby-to-normal interlocutor ⑧ ⑨		2	14 AWG (2.08 mm^2)	66 in. (168 cm)
Remote sensing		N/A		
Standby-to-normal voltage adjust		2	14 AWG (2.08 mm^2)	56 in. (142 cm)

Cooling: natural convection, ambient air. Temperature and humidity refer to M-G data sheet.

Notes:

1. Recommended wire size is based on insulated copper wire type THW 75°C.
2. Wiring from the standby control cabinet to the MG, main power, and remote sensing device (if used) has the same wiring data as the normal control cabinet.

③ This wiring is in same conduit.

④ Two raceways with the same physical characteristics are required. Each raceway must contain only one wire per phase and one wire for neutral, if used. The wires of each raceway must be the same length and material and have the same type insulator.

⑤ Based on 100-ft conductor lengths between M-G control cabinet and load circuit breaker panel.

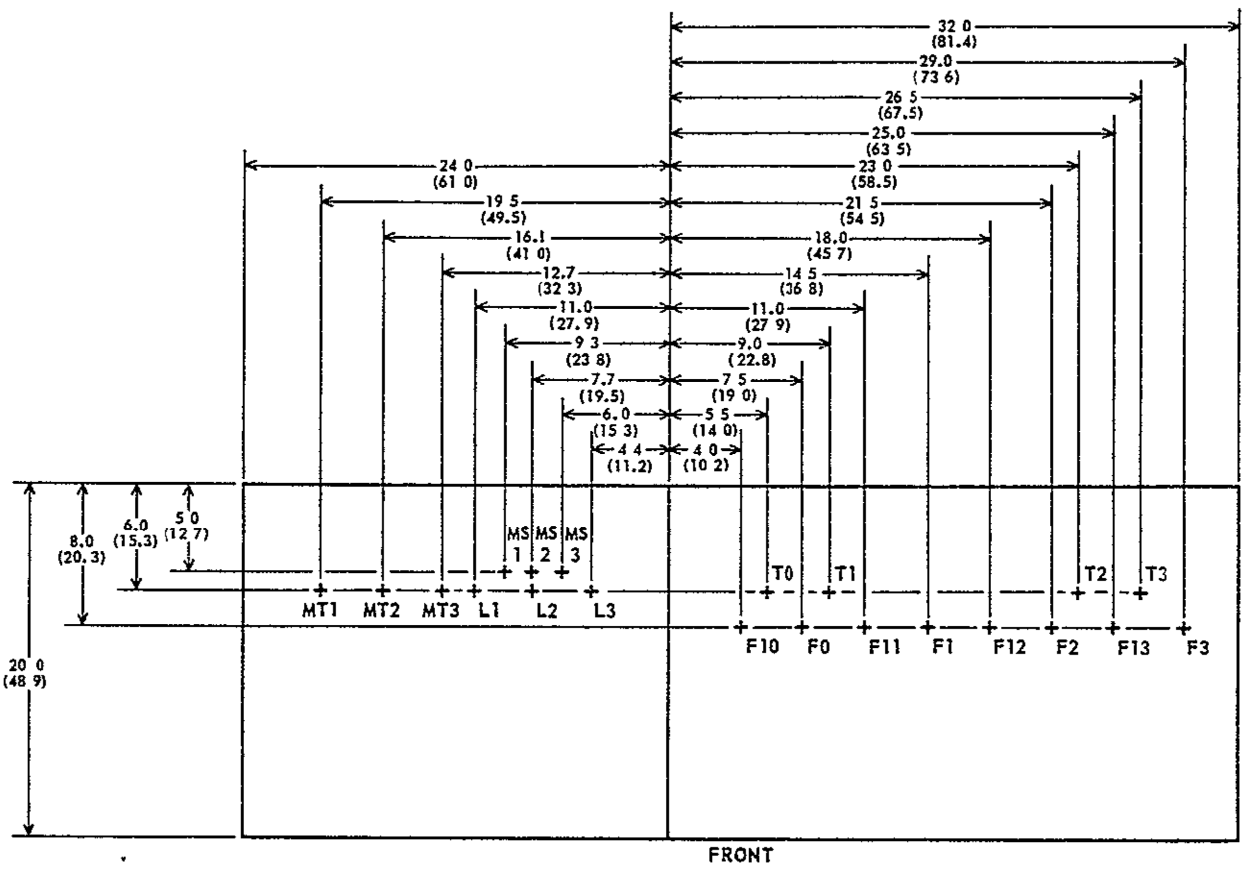
⑥ The neutral wires need only be four 4 AWG (21.2 mm^2) conductors.

⑦ Four aluminum conduits are required with four wires.

⑧ If the standby control cabinet provides 400-Hz power to a second normal control cabinet, 400-Hz and interlocutor wiring must be duplicated to the second normal control cabinet.

⑨ This wiring requires separate wiring runs.

MOTOR-GENERATOR CONTROL CABINET TERMINAL LOCATIONS

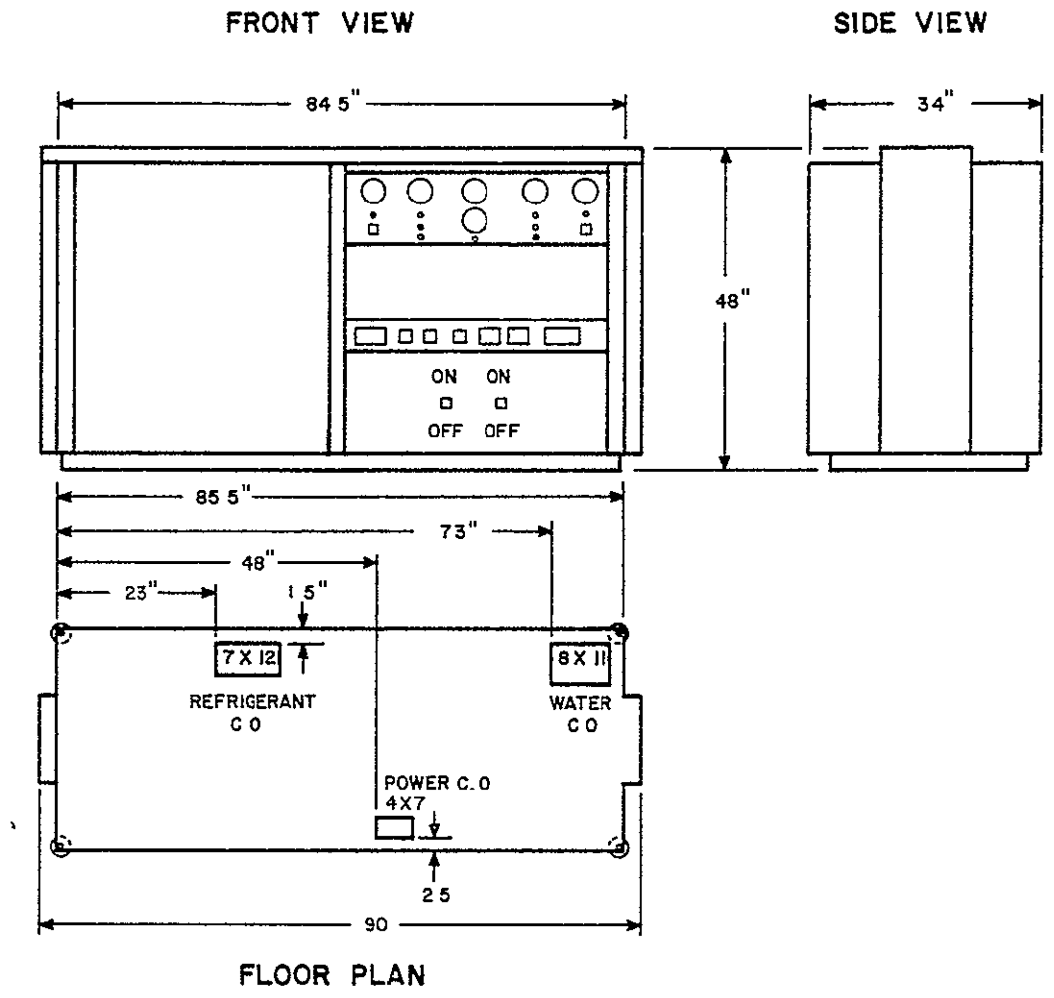


CABINET FLOOR PLAN

NOTES

1. DISTANCES ARE SHOWN IN INCHES AND CENTIMETERS. CENTIMETERS ARE IN PARENTHESES
2 WIRE ENTRY LOCATIONS (+) HAVE DESIGNATORS THAT CORRESPOND TO CABINET WIRE TERMINALS
3 APPROXIMATE DISTANCES FROM CABINET TERMINALS TO FLOOR ARE.
   MS1, MS2, MS3, MT1, MT2, AND MT3 = 8.0 IN. (20.3 CM)
   L1, L2, AND L3 = 51.0 IN (130.0 CM)
   T0, T1, T2, AND T3 = 56.0 IN (145 0 CM)
   F0, F1, F2, AND F3 = 21.0 IN. (53.4 CM)
   F10, F11, F12, AND F13 = 25 5 IN. (64.8 CM)

30-TON CONDENSING UNIT



Layout restrictions· The following minimum access clearances must be observed to adequately service the unit and permit the possible removal of a compressor.

Front and rear:	24 inches
Both sides·	36 inches

30-TON CONDENSING UNIT

Width	90 inches
Depth	34 inches
Height	48 inches

For maximum width/depth, all doors extended, see floor plan layout.

Weight	2125 lb
Supported by support beams	132 in^2 each
Number of leveling pads	4
Area of each pad	5 in^2 each

Power consumption, steady state:

60 Hz 460 V, 3 phase 40.0 KVA

Circuit breakers:	Power connections	Locations
60 Hz, 70 A, 3 phase	Field Terminals	6 inches above base of cabinet

External Terminator power connection required· none

Environmental considerations:

Type of cooling	Forced convection and water cooled heat exchanger
Source of cooling·	Ambient room air and building water supply
Heat rejection rate maximum	12,000 BTU/hr (to air)
	300,000 BTU/hr (to water)

Permissible range of room relative humidity· 10 to 85 percent

	Maximum	Recommended	Minimum	Dew point limitation
Operating temperature (ambient or plenum inlet temperature):	100 °F (37.8 °C)	75 °F (24 °C)	55 °F (13 °C)	Refer to the following note
Storage temperature	120 °F (°C)		40 °F (°C)	

NOTE

The selection and combination of operating conditions for ambient room air and water inlet temperatures may cause condensation. It is recommended the customer provide a condensation proof insulation barrier on all water supply lines.

Design water flow and temperature requirements (per unit)·

Inlet temperature of	Flow rate US GPM	Pressure Drop Through Open Condenser PSIG	Minimum Operating Pressure Differential PSIG
50	10.0	1.2	10
60	12.0	1.8	10
70	15.0	2.3	10
80	19.0	2.8	10

NOTE

The inlet water pressure shall not exceed 100 psig.

APPENDIX C

FLOOR PLAN LAYOUT

APPENDIX D

CONTROL DATA SITE PREPARATION MANUAL

60275100

CONTROL DATA
CORPORATION

CONTROL DATA® LARGE AND MEDIUM SCALE COMPUTER SYSTEMS

SECTION 1 GENERAL INFORMATION

SITE PREPARATION MANUAL

REVISION RECORD

REVISION	DESCRIPTION
A	Manual released.
B	Manual revised; Engineering Change Order 25957. Page 8-9 is deleted. Pages iii, iv, v, vi, 1-1,
	1-2, 1-3, 2-2, 4-1, 5-1, 5-3, 5-5, 5-11, 5-18, 5-20, 8-1, 8-5, 8-7, and 8-8 are revised.
C	Manual revised; Engineering Change Order 34459, publications change only. Page 8-6 is revised.
(11-29-73)	
D	Manual revised; Engineering Change Order 36986, publications change only. This edition obsoletes
(2-13-76)	all previous editions.
Publication No 60275100	

REVISION LETTERS I, O, Q AND X ARE NOT USED

Address comments concerning this manual to

Control Data Corporation
Publications and Graphics Division
4201 North Lexington Avenue
St. Paul, Minnesota 55112

or use Comment Sheet in the back of this manual

LIST OF EFFECTIVE PAGES

New features, as well as changes, deletions, and additions to information in this manual are indicated by bars in the margins or by a dot near the page number if the entire page is affected. A bar by the page number indicates pagination rather than content has changed.

PAGE	SFC †	REV
Cover		-
Title Page		-
ii		D
iii		D
v		D
vi		D
vii		D
viii		D
1-1		D
1-2		D
2-1		D
3-1		D
4-1		D
4-2		D
4-3		D
4-4		D
4-5		D
4-6		D
4-7		D
4-8		D
4-9		D
4-10		D
5-1		D
5-2		D
5-3		D
5-4		D
6-1		D
6-2		D
7-1		D
7-2		D
7-3		D
7-4		D
7-5		D
8-1		D
8-2		D
A-1		D
A-2		D
A-3		D
B-1		D
B-2		D
B-3		D
B-4		D
Comment		
Sheet		D
Return		
Envelope		-
Back Cover		-

†SFC Software Feature Change

A~ 5935

PREFACE

Control Data Corporation maintains an Engineering and Architectural Services Division to assist the customer in preparing his site for a computer system installation. To aid in preparing the site for the installation, Control Data publishes this and other site preparation manuals. The manuals provide system and individual equipment references for use before and after the installation.

The manuals are divided into two groups (refer to Site Preparation Publication Index on following page), Computer Systems and Mini Computer Systems. In each group, sections 1 through 3 are required to document the site preparation information for a specific system. An additional manual, useful after completion of the site preparation phase, is the Site Environmental Maintenance Handbook. A more detailed description of the sections and handbook follows.

Control Data urges the customer to consult local authorities if the requirements in the manuals conflict with local building, electrical, or fire codes or ordinances. Any deviations from the manual procedures in sections 1 througn 3, except to comply with local codes or ordinances, must be approved by the Engineering and Architectural Services Division.

SECTION 1. GENERAL INFORMATION

Section 1 details computer installation information that is common to more than one computer system. There are three separate section 1 manuals; one applies to mini † computer systems, one applies to small scale †† computer systems, and one applies to large and medium scale ††† computer systems. A section 1 manual contains information on.

- Site planning procedures
- Equipment layout
- Signal cabling layout and location
- Maintenance personnel area
- Building and environmental requirements
- Paper tape punch card and magnetic recording media storage
- Fire and security precautions
- Grounding systems
- Switchgear, converters, and system power requirements

SECTION 2. SYSTEM DATA

Section 2 contains information related to the physical and environmental specifications of a specific computer system. A section 2 manual contains·

- Specifications for the computer system including the number, function, and placement of system cabinets and/or units
- Equipment data sheets that detail system power and environmental requirements, individual equipment dimensions, and physical configurations
- Electrical schematics that document external power connections among the system equipment and between the system and electrical switchgear

SECTION 3. PERIPHERAL EQUIPMENT DATA

Section 3 documents only peripheral equipment which connects to more than one type, series, or family of computers. There are two section 3 manuals; one applies to mini computer systems, and one applies to all other peripherals. A section 3 manual contains:

- Table of peripneral controllers and equipment that gives a brief description of each peripheral unit and indicates its cabinet type
- Equipment data sheets that include power and environmental requirements and the physical configuration and dimensions of each piece of equipment
- Motor-generator data sheets that include physical, power, and environmental specifications and wiring information for the motor generators and motor-generator control cabinets

† CONTROL DATA® MP-17, System 17, 1500 equipment, and similar systems.
†† CONTROL DATA® SC1700, 1700, 8090, 160-A and similar systems.
††† CONTROL DATA® 3000, 6000, 7000, CYBER, STAR, and similar systems.

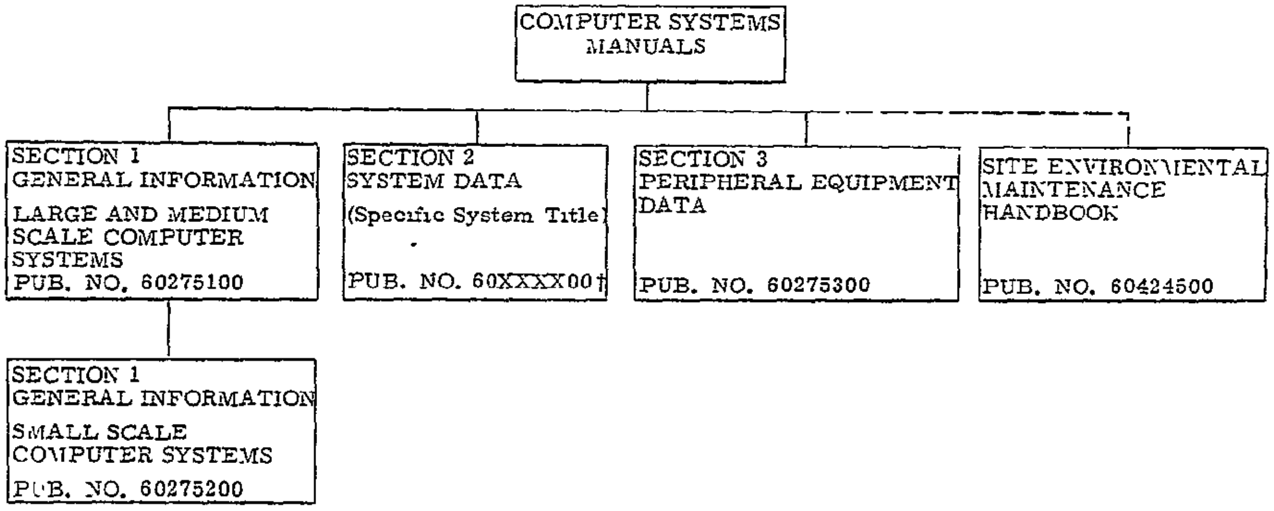
471

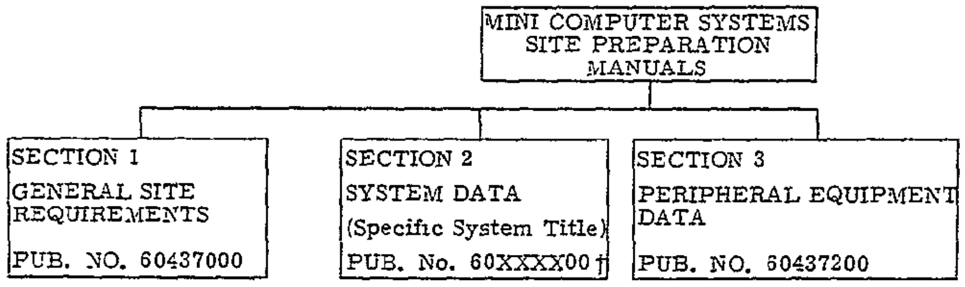
SITE ENVIRONMENTAL MAINTENANCE HANDBOOK

This manual contains information applicable to the computer system site. It describes site maintenance recommendations and procedures necessary for reliable computer operations and minimum suggested maintenance schedules and guidelines.

The Site Environmental Maintenance Handbook applies mainly to medium and large scale computer sites; however, parts of the handbook are applicable to small scale and mini computer systems. The handbook contains:

- Guidelines and recommendations for the proper implementation of general site maintenance such as cleanliness, floor coverings, and safety
- Guidelines and recommendations for the proper implementation of technical site maintenance such as environmental control, water treatment and handling, cabling, lighting, and power


COMPUTER SYSTEMS MANUALS
SECTION 1 GENERAL INFORMATION LARGE AND MEDIUM SCALE COMPUTER SYSTEMS PUB. NO. 60275100
SECTION 2 SYSTEM DATA (Specific System Title) PUB. NO. 60XXXX00†
SECTION 3 PERIPHERAL EQUIPMENT DATA PUB. NO. 60275300
SITE ENVIRONMENTAL MAINTENANCE HANDBOOK PUB. NO. 60424500
SECTION 1 GENERAL INFORMATION SMALL SCALE COMPUTER SYSTEMS PUB. NO. 60275200


MINI COMPUTER SYSTEMS SITE PREPARATION MANUALS
SECTION 1 GENERAL SITE REQUIREMENTS PUB. NO. 60437000
SECTION 2 SYSTEM DATA (Specific System Title) PUB. No. 60XXXX00†
SECTION 3 PERIPHERAL EQUIPMENT DATA PUB. NO. 60437200

SITE PREPARATION PUBLICATION INDEX

†Refer to the Literature Distribution Services (LDS) catalog for appropriate system publication number.

472

CONTENTS

APPENDIXES

FIGURES

SITE PLANNING PROCEDURES

INTRODUCTION

Detailed planning and coordination are essential for the successful installation of a Control Data computer system. Because the completion of each step in the site preparation phase is presumed by the next step, it is essential that the customer formulate and adhere to a planned and carefully sequenced schedule. Prior to establishing a schedule, the customer should become thoroughly familiar with the sections of the site preparation manual pertaining to his system.

The customer should provide a liaison engineer for the purposes of coordinating events and communicating information with the Control Data site planning representative. The key to successful completion of the site preparation program is an effective liaison engineer.

The completed installation site must provide the environmental and operational requirements established by Control Data and described in this manual, but factors such as construction methods are left to the discretion of the user. The Control Data site planning representative is regularly available for advice and assistance.

Because of the investment a computer system represents, the customer may desire the use of professional consulting services for the various areas of site design. These services are extremely important in helping to provide a smooth running computer installation.

Control Data provides qualified services in the design, administration of contracts, and supervision of the computer site construction. This work is done on a contract basis by the Control Data Engineering and Architectural Service division. Additional site service information is available from the Control Data sales representative.

SITE PLANNING SUPPORT

Control Data maintains a site planning department in the Engineering and Architectural Services Division to assist the customer in making a smooth and rapid computer site installation. The department specializes in site preparation and provides to the customer, as part of its services, a set of site planning drawings specifically tailored to the customer's facility. The drawings serve as a guide to the site design engineer and an aid to the preparation of his working drawings. The site planning drawings include the following.

1. A dimensional floor plan indicating equipment location and the appropriate floor cutouts.
2. A power distribution diagram depicting all intercabinet power and control wiring (but not signal cabling) and switch gear connections
3. A machine unit schedule that outlines the physical properties of each cabinet in the system

In addition, the site planning department provides recommendations on air conditioning design, system grounding, security, noise, and general power requirements.

SITE PLANNING CONFERENCES

Due to the amount of technical information exchanged between Control Data and the customer, a Control Data site planning representative engineer normally conducts three site planning conferences.

475

CONFERENCE 1 - ORIENTATION

The primary purpose of this conference is to develop a final floor plan which best suits the customer's needs and system requirements. In addition to the floor plan requirements, the site planning representative

1. Explains the floor layouts, power wiring, and equipment specifications (prepared by Control Data for the customer)
2. Establishes milestone dates and charts for the site preparation process
3. Establishes customer and Control Data responsibilities pertinent to the preparation of the site

CONFERENCE 2 - SCHEDULING

This conference is conducted with the personnel responsible for designing and installing the air conditioning, power distribution, and raised floor. The site planning representative·

1. Explains in detail Control Data drawings and specifications
2. Offers recommendations
3. Answers questions pertaining to the computer site environment

CONFERENCE 3 - SITE INSPECTION

The site planning representative holds this conference with the customer to inspect the completed site, ensuring that it meets the previously discussed drawings and specifications.

NOTE

Delivery of the computer system is dependent on satisfactory completion of site construction. The customer must therefore agree to have the site ready, prior to scheduled equipment delivery date.

ADDITIONAL CONFERENCES

Any additional conferences will be scheduled as required.

476

EQUIPMENT LAYOUT 2

INTRODUCTION

The customer and the Control Data site planning engineer share the responsibility for the development of an acceptable system layout. Determining factors in laying out the system include the numbers and types of system components, operational considerations, computer room dimensions, area column and door locations, signal cable lengths, and system aesthetics. Sections 2 and 3 of the site preparation manual provide physical dimensions and other pertinent information on each piece of system equipment. Within these physical constraints, careful consideration should be given to personnel traffic flow, material handling, equipment visability, operating characteristics, and acoustical considerations.

TRAFFIC FLOW AND MATERIAL HANDLING

As an aid to efficient site operation, operating and maintenance personnel must have easy access to paper and card handling equipment such as printers, punches, and readers and to magnetic peripherals such as tape handlers and disk storage units.

Equipment layout should permit sufficient space for the passage of carts, oscilloscopes, and personnel and for the placement and removal of waste or take-up containers. Since some computer operations consume large amounts of paper materials in short periods of time, Control Data recommends that a separate storage area for paper products be located adjacent to the computer area (refer to Paper Tape, Punch Card, and Magnetic Recording Media Storage, section 5).

Portable equipment and some movable equipment are often serviced in the customer engineering area. Access to this area should be convenient and should be on the same floor level as the computer area. For safety purposes, do not move equipment such as magnetic tape transports on ramps from one level to another.

VISIBILITY AND OPERATING CHARACTERISTICS

A most efficient operation results when the operator can visually monitor the physical activity of the computing system. Many devices have no moving parts and do not require operator intervention; however, some units utilize lights to indicate various operating conditions. Arrange all such units so that the indicators are visible from the console.

The operating characteristics of certain equipments such as printers, card punches, and similar devices produce appreciable amounts of waste material. This material may find its way into dust-sensitive mechanisms such as card readers, magnetic tape transports, and disk file equipment. Waste-producing devices should, therefore, be placed as far as possible from dust-sensitive mechanisms and as close as possible to the room air exhaust grills.

Since the printers and punches are comparatively noisy, place them in an area where a moderately high noise level is not objectionable.

SIGNAL CABLES

Signal cables transmit data and equipment signals and status conditions between individual pieces of equipment in the computer system. Control Data provides all necessary signal cables according to the final system layout. Cables are normally supplied in the standard lengths† specified on the data sheets in the sections 2 and 3 site preparation manuals. In certain instances, longer-length signal cables are required. The longer than standard length cables, up to the maximum lengths indicated in the sections 2 and 3 site preparation manuals, can be supplied at an additional cost to the customer.

The signal cables route beneath the raised floor. The cables lie directly on the normal building floor except when the use of troughs, raceways, or other routing devices are required by specific ordances or preferred by the customer. If a signal cable is longer than necessary, the cable must be randomly routed (never coiled) to take up the excess length.

NOTE

Signal cables, ac power cables, and dc terminator power cables must each be routed separately. 400-Hz and 50/60-Hz cables must not be run in the same cable trough.

TERMINATOR POWER CABLES

The signal transmission technique employed in some computer equipment requires terminator power cables which connect between signal terminators and an external ± 20 vdc terminator power supply. Control Data furnishes the terminator power supply with the computer system; however, it is the customer's responsibility to furnish and install the terminator power cables.

Equipments requiring terminator power are so specified on their associated equipment data sheets in the sections 2 and 3 site preparation manuals.

POWER CABLES

Power wiring from the power distribution panels to the various system equipments must be routed beneath the raised floor in ducts, raceways, or other approved devices.

All power wiring, ducting, and mounting are supplied by the customer and must conform to local electrical and/or building ordinances and codes. (Refer to section 8 for additional power requirements information.)

† Many signals are time-critical and have a direct relation upon the determination of signal cable lengths.

INTRODUCTION

The area selected to house the computer system should meet the space and convenience requirements of the equipment layout and should provide for the possibility of future expansion. If the area will be in an existing building, examine it carefully to ensure that it can be made to comply with all of the system installation and operating requirements. If a new building will be constructed, inform the architect of the equipment layout and system requirements.

MAINTENANCE PERSONNEL AREA

An area must be provided for customer engineering personnel. The area should be on the same level as the computer area and preferably, adjacent to it.

The size of the maintenance area should be reasonably close to the following specification.

$$\text{Square Feet} = 100 + \frac{(\text{Basic Monthly Maintenance})}{80}$$

Example A certain configuration has a basic monthly maintenance income of \$4000. The size of the maintenance area would be $100 - \frac{4000}{80} = 150$ square feet.

Include adequate heat, light, ventilation, and electrical outlets for the use of Control Data's maintenance personnel.

The area must have an acceptable level of acoustical noise and the following power available.

- 50/60 Hz, 3-phase, 208 volt, 20 amp, to accommodate Hubbel Twistlok connector
- 50/60 Hz, single-phase, 120 volt, 30 amp, to accommodate single-phase locking connector

The customer provides the following furnishings.

- Ready access to a telephone with an outside line
- Desks and chairs based on the following formula

$$\text{Number of Desks} = 1 - \frac{(\text{Basic Monthly Maintenance} - \$3000)}{(\$3000)}$$

- Storage cabinets based on the following formula

$$\text{Number of Cabinets} = 1 - \frac{(\text{Basic Monthly Maintenance})}{(2000)}$$

Typical cabinet dimensions are 36 inches wide by 72 inches high by 18 inches deep.

- A workbench allowing for off-line repair of small equipment and assemblies
- Suitable bookshelves for storage of maintenance manuals

COMPUTER ROOM CONSTRUCTION

Construction of the computer room and installation of its facilities and utilities should comply with applicable building and service codes. Additionally, the site must meet the requirements of the United States National Fire Protective Association Standard Number 75. (In other countries, equivalent codes and fire standards shall apply.)

FLOORS

WARNING

Most raised floors are not watertight and if an open or nonwaterproof raceway is used beneath them, wet mopping of the floor may result in a serious shock hazard to cleaning personnel. It is therefore recommended that suitable precautions be taken to protect all underfloor power wiring.

A raised floor permits all power and signal cables to enter the system equipment cabinets through the cabinet bases. The raised floor permits unrestricted mounting of cables and allows easy remounting in the event of a system layout change. Additionally, the raised floor permits the installation of plumbing and venting networks for systems with condensing units.

Several types of raised floors will work, however, the pedestal and stringer type (figure 4-1) has the advantage of serving as the grid ground reference plane, refer to section 7.

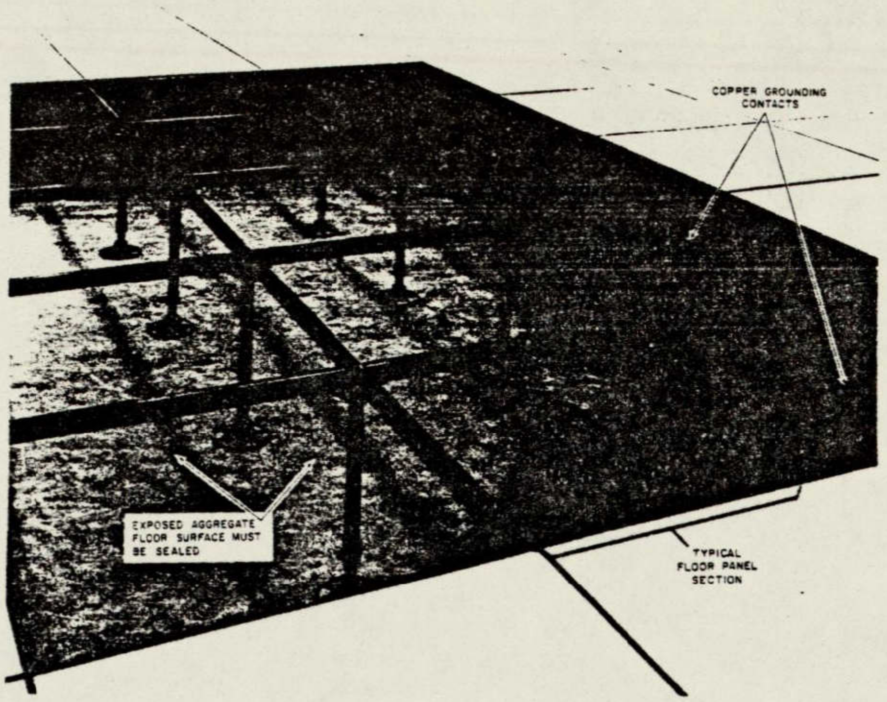

COPPER GROUNDING CONTACTS
EXPOSED AGGREGATE FLOOR SURFACE MUST BE SEALED
TYPICAL FLOOR PANEL SECTION

Figure 4-1. Pedestal- and Stringer-Type Raised Floor

Raised floors require an 8- to 12-inch (15- to 30-centimeters) clearance from the bottom of the support stringers to the surface of the original floor. If the installation is made in an existing building, build one or more ramps between the building floor level and the surface of the raised floor. Position these ramps at the room entrances to permit the ease of movement of equipment and carts to and from the computing room.

If a raised floor will be used in a building under construction, install the building computer room floor lower than the normal building floor. The raised floor installation then can be at the same level as the normal building floor, thus eliminating the need for ramps.

In either case, the raised floor should meet the following requirements.

1. Elevation above building floor: 6 to 12 inches (15 to 30 centimeters) are required if the equipment is plenum cooled
2. Floor panels: maximum of 24 by 24 inches (61 by 61 centimeters); panels may be smaller (smaller panels usually withstand greater loads and are easier to handle during layout changes)
3. Construction: metal or metal-clad wood panels
4. Covering: high pressure laminate or carpet titles
5. Support capability: must be capable of supporting 250 pounds/feet^2 (100 kilograms/meters^2) evenly distributed with 1000-pound (500-kilogram) point concentrations in 2-inch (5-centimeter) diameter areas

SURFACE

The surface of the raised floor may be either tiled or carpeted; each has its own advantages. If the surface will be tiled, it must be of a high-pressure laminate variety to withstand the heavy loads to which computer room floors are exposed. Do not use asphalt, vinyl, and vinyl-asbestos tiles. A high pressure laminate surface provides minimum maintenance. (Refer to Floors in the Site Environmental Maintenance Handbook for further details.)

Control Data suggests carpet tile surfaces for the computing area because of their acoustical, aesthic, and comfort properties. If carpet tiles are to be used, they must have an acceptable electrostatic mating. (Refer to Floors and appendix A in the Site Environmental Maintenance Handbook for further details.)

LOADING

Calculate floor loading by using the weights listed in the equipment data sheets in the sections 2 and 3 manuals. Equipments with casters or leveling pads present concentrated loads of up to 300 pounds per square inch over 3 square inches (20 kilograms per square centimeter over 20 square centimeters).

NOTE

Check floor loading for the raised floor and the normal building floor, prior to installation of the raised floor.

Calculations for the normal building floor include those for the raised floor as well as the following.

- Weight of the raised floor structure: approximately 10 pounds/feet^2 (50 kilograms/meters^2)
- Weight of each signal cable: approximately 0.4 pound/foot (0.6 kilogram/meter) of length
- Weights of power cables and raceways

If loading calculations raise any doubts as to the ability of the normal building floor to support the system equipment, the customers should consult a structural engineer.

LEVELING

The raised floor should be level within 1/16 inch per 5 feet (0.17 centimeter per meter) and within 1/8 inch (0.32 centimeter) over any continuous length of cabinet or cabinets joined end-to-end.

PROTECTION

During installation or other movement of heavy equipments, protect the raised floor from concentrated loads to prevent surface damage.

The surface protection should be temporary and should consist of not less than 1/4-inch plywood or 1/4-inch tempered hardboard. Also, use this protection for heavy personnel or equipment traffic over the floor by the contractor (or subcontractor) during any building construction or modification.

MAINTENANCE

Refer to the Site Environmental Maintenance Handbook for scheduled maintenance procedures and frequencies for the floor surfaces.

LIGHTING

The computing room lighting should have a minimum of 50-foot candles (538 lux). Avoid excessive illumination to enable proper visability of lighted equipment indicators. Avoid use of spot lights, flood lights, and /or direct sunlight.

SIGNAL AND POWER CABLE ROUTING

Refer to Signal and Power Cabling Layout and Location, section 3.

CONVENIENCE OUTLETS

The customer provides and installs electrical convenience outlets within 15 feet (4.6 meters) of each system cabinet to enable the use of test equipment at any of the system cabinets. The outlet locations may be in the computer room walls on building pillars or stantions or mounted in the raised floor sections.

All outlets contain single-phase grounded type receptacles connected to 50/60-Hz power.

NOTE

> The 50/60-Hz power to the convenience outlets must connect to the same source as the computer 50/60-Hz power.

The outlets must not be controlled by the computer or peripheral power panels. For 60-Hz installations, the nominal voltage is 120-vac, and for 50-Hz installations, the normal voltage is 220, 230, or 240 vac. Refer to section 7 for grounding requirements.

ROOM MAINTENANCE

Room cleaning and maintenance is important in keeping air contamination within safe and acceptable levels. Give special attention to filter changing, cleaning beneath the raised floor, and preventing smoking, eating, and drinking in the computer room. The Site Environmental Maintenance Handbook details all aspects necessary to proper site maintenance cleaning procedures and schedules.

TEMPERATURE AND HUMIDITY

Control temperature and humidity so that they remain within the limits of the system. The system limits are determined by the most restrictive cabinet (or equipment) within the system. In many instances, the most restrictive limit in a system is determined by the limitations of the associated peripheral equipments. Refer to the relative humidity range and/or dew point limitation listed in the appropriate system sections 2 and 3 manuals.

The installation of a temperature and humidity monitoring recording device in the computer room helps to ensure proper operating conditions. The device may contain automatic recording charts and should provide for visual and/or audible alarms if the temperature or humidity approaches limiting conditions.

DEW POINT

CAUTION

Should condensation occur, serious component damage may result.

Most large scale computers use refrigerant and/or water-cooling techniques. Keep them below the specified dew point limit of the system to prevent moisture condensation upon system components. The data sheets in the appropriate system section 2 manual specify the system's dew point limitation.

On systems having a dew point limit, Control Data supplies a dew point recorder with the system. Refer to Computer System Requirements in the appropriate system section 2 manual for additional information and requirements.

LIGHTNING PROTECTION

Lightning protection is necessary if the computer installation is in a locality where electrical storms are common, particularly if overhead power lines supply the primary power to the facility.

Install lightning protection in the computer power distribution system to protect the electronic components against power surges and also, to serve as an additional fire prevention measure.

ROOM ENVIRONMENT

Control of the room environment is extremely important in minimizing system troubles. Some control is achieved through the proper selection of a room in an existing building, more control is achieved through effective design in a building to be constructed, and maximum control is achieved through proper selection, design, and the use of special equipments such as air filters, air conditioning, and humidifiers.

AIR FILTRATION

Air cleanliness is important in minimizing system troubles particularly in equipment such as magnetic tapes and rotating magnetic storage units. In order to properly maintain air cleanliness, the room requires an efficient air filtering system in conjunction with a proper cleaning and maintenance program. (Refer to the Site Environmental Maintenance Handbook for recommended cleaning precedures and schedules.

The air filtering system removes dust and dust particles from the air and normally consists of mechanical or electrostatic-type filters located in the system equipment cabinets and the computer room air conditioning units.

The equipment specifications for mechanical filters state an efficiency of 80 percent with a particle size of 5 microns, determined by the National Bureau of Standards (NBS) discoloration test using 96 cottrell precipitate and 4 percent cotton linters. The efficiency is 30 percent when determined by the NBS discoloration test using atmospheric dust.

The equipment specifications for electrostatic filters state an efficiency of 90 percent at a velocity of 500 feet per minute (153 meters per minute), determined by the NBS dust-spot test method using atmospheric air without the addition of dust or contamination.

NOTE

Equipment filters are supplied by Control Data and are an integral part of the equipment cabinet assembly.

Air conditioning system filters are customer-furnished and installed.

Filters in the air conditioning system must equal the filtering ability of the filters in the computing system equipment. Air conditioning filters must also have a low pressure loss capability.

The air contamination specifications for Control Data rotating storage devices have more stringent limits than the specifications of most other computing equipment. Therefore, a computer site which meets the requirement of the rotating storage device specification also meets the requirements for other computing equipments.

Filtering specification requirements for rotating storage devices are shown in figure 4-2. The specification is given in acceptable particle sizes (in microns) versus particle concentrations for given volumes of air.

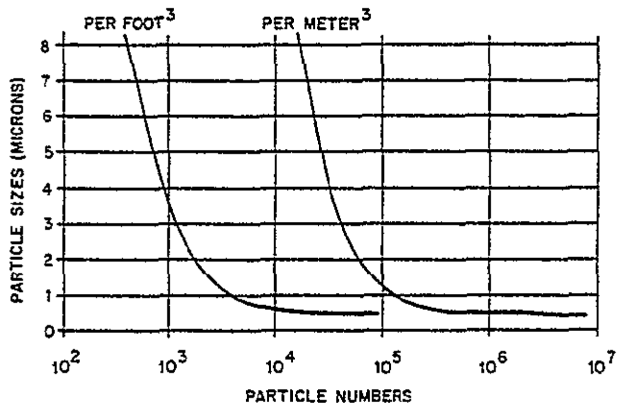


Figure 4-2. Acceptable Particle Size Versus Concentrations for Rotating Storage Device Filters

The following system discussions provide the information needed by an air conditioning engineer to prepare load calculations, select the best type of air-conditioning system, prepare working drawings from which a contractor may accurately prepare cost estimates for, and properly install an air-conditioning system which meets the needs of a specific computer site.

No attempt is made here to delineate the limits of operating conditions for the computer equipment. Refer to the individual equipment data sheets in sections 2 and 3 manuals for allowable ranges of inlet air temperatures for individual units in the system.

The inside design considerations for a computer installation are based on the following room conditions.

1. Temperature - 72°F (22°C)
2. Relative humidity - 50% (60.1°F. WB)

In computer installations having outside walls, maintain the inside wall and glass surface temperature above the maximum dew point of the room air. In geographic areas having outside design temperatures below −10°F (-12°C), provide a controlled heat source near the floor along all outside exposures.

The heat source should have a capacity sufficient to meet 50 percent of the total transmission loss through the wall and glass areas. When outside design temperatures are above −10°F (-12°C), it is not necessary to have a controlled heat source near the floor.

AIR CONDITIONING

Air conditioning provides the means for keeping the ambient air of the computer room at the desired temperature for the air-cooled computing equipments. Other than the large scale computers and a few peripherals which are refrigerant-cooled, most system equipment cabinets are cooled by the ambient room air. The air is forced through the cabinets by self-contained internal blowers† which draw room air into the cabinets through openings near the bottom of the cabinets, exhaust the air into the room from openings near the bottom of the cabinets, and exhaust the air into the room from openings in the top of the cabinets.

The ambient room air must sufficiently cool the heat-dissipating components in the cabinet as the air passes through. The equipment data sheets in the sections 2 and 3 manuals indicate the maximum heat rejection rates of the equipment to aid in determining room air conditioning requirements.

Two types of air conditioning systems are suggested. The first type (room-temperature system) uses only the ambient room air for cooling and the second type (mixed-temperature system) uses the ambient room air to cool some of the equipment and requires air at lower temperatures than the room air to cool other equipment.

ROOM TEMPERATURE SYSTEM

The following paragraphs give a basis for determining the requirements for a computer installation where the equipment is cooled with air at room temperature.

Air Distribution System

The air distribution system is either a ceiling plenum, which uses the suspended ceiling as the supply diffuser or overhead ducts with individual ceiling diffusers (figure 4-3).

The quantity of air required for specific areas within the computer room is dictated by the equipment layout. Air quantities must be supplied in relation to the dissipation of heat which occurs in the computer room. When a ceiling plenum air distribution system is used, it may be necessary to provide a supplementary air distribution device in the suspended ceiling to deliver the increased volume of air required to absorb the dissipated heat.

Certain computer equipment cabinets employ self-contained refrigerant-type cooling devices. Do not deliver supply air within 6 feet in any direction of a cabinet of this type.

The return air inlets should be located either high on the walls or in the ceiling. It is desirable to have the return air inlets near the areas of greatest heat rejection. Dust-producing equipment should be located near the return air inlet with respect to other equipment in the room.

Balance the air conditioning duct system after the computer equipment is installed and operating such that the inlet air to any equipment is within the limits specified in sections 1 and 2 in the site preparation manual.

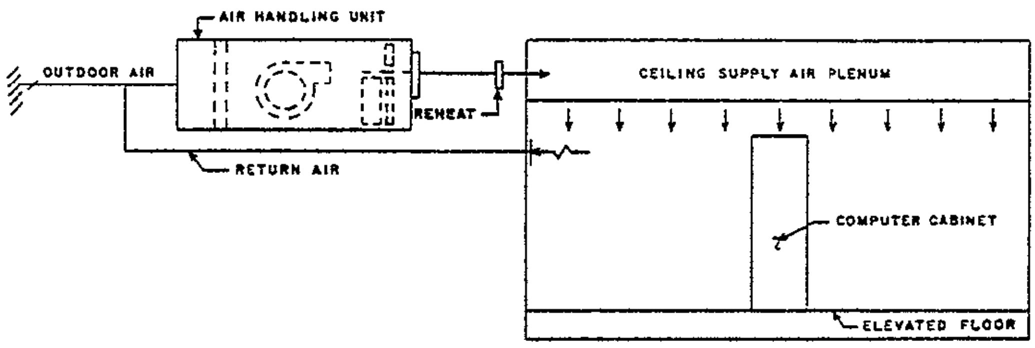
It is not necessary to provide thermal insulation on either the supply or return duct in the ceiling space unless there is abnormal heat gain from the slab above.

Alternate Air Distribution System

This discussion describes a room air distribution system which uses the underfloor space as a supply air plenum. This system is an alternate method of air distribution where the overhead air distribution system is impracticable.

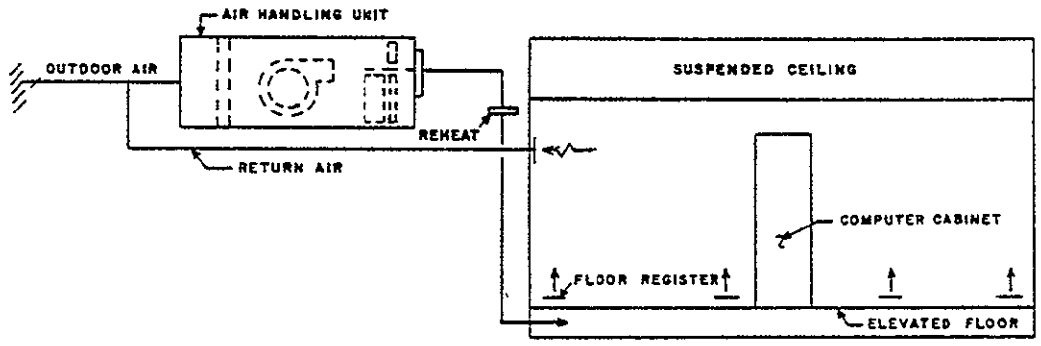
The supply air is ducted directly from the air conditioning unit to the underfloor plenum (figure 4-4) for distribution to the room in lieu of an overhead duct system. The required room air distribution can be accomplished by locating floor registers around the perimeter of the room and near equipment in relation to the heat disipation. Care should be given to the placement of floor registers to provide the necessary cooling while avoiding conditions which may cause operator discomfort.

† A few systems use plenum-type cooling wherein the conditioned air is forced through a plenum located beneath the raised floor. The air enters the cabinets from the plenum through cutouts in the surface of the raised floor which correspond to openings in the cabinet base. The air passes up through the cabinet and is exhausted into the computer room.



ELEVATION

Figure 4-3. Air Conditioning System for Ambient-Cooled Computer Equipment



ELEVATION

Figure 4-4. Alternate Air Conditioning System for Ambient-Cooled Computer Equipment

Because of noise and draft conditions, it is recommended that the floor registers be sized to deliver the quantity of air desired at a velocity of not more than 650 fpm.

Provide supply registers with volume dampers to facilitate balancing air quantities and to provide maximum flexibility for future room changes.

The underfloor static pressure is sufficient to maintain 0.05-inch pressure at the floor register.

Air Side Equipment

Basically, a computer installation is an office area with added internal heat-producing equipment. A typical system of air-side equipment, which would meet the requirements of the space, is a multizone air handling unit with the hot deck coil omitted and zone terminal reheat provided. This type of unit provides a bypass apparatus with the capability of giving zone control by terminal reheat. By using a bypass apparatus, operating efficiencies are improved, because the space sensible heat in the return air can be used as reheat.

The cooling coil must provide the desired leaving air temperature while maintaining the required apparatus dew point. Suggested considerations for the selection of the coil are·

1. Larger-than-usual surface area becuase of the high sensible-heat ratio in the computer center.
2. No more than 10 fins per inch
3. Approximately, a 15°F (8.3°C) temperature rise in the chilled water
4. Maximum face velocity of 600 fpm (183 mpm)

Refrigerant Side Equipment

In order to ensure continuity of operation and minimum maintenance, select chilled water for the cooling medium. The water-chilling equipment snould be the multiple-refrigeration circuit-type so that the loss of one circuit will not result in a loss of capability to carry the design load.

Base the chilled water system on a 15°F (8.3°C) rise in the water at design conditions. Sequence the refrigeration units to maintain leaving water temperatures as indicated by the coil requirements; however, the leaving water temperature need not be lower than 42°F (5.6°C).

It is recommended that the chilled water circulating system utilize a primary-secondary pumping program. The primary pump circulates chilled water around the primary pipe loop and through the water chiller. A secondary pump is used for each apparatus to provide the specific amount of chilled water required. In order to provide continuity of service and allow opportunities for maintenance, provide dual pumps for each purpose.

Condensing Medium

On systems of 100-ton capacity or less, air-cooled condensing is recommended with a condenser provided for each refrigeration compressor on the water chiller.

On systems in excess of 100 tons, cooling towers and water-cooled condensers are recommended; however, a program of continuous water treatment is required to ensure continuous operation of the system.

Design Procedures and Considerations

The design procedure for a computer installation should follow standard engineering practices in calculations of heat gains and losses. However, an understanding of equipment layout within the space is necessary to provide heat gain calculations that are accurate in relation to space use. The following design procedure is recommended.

Internal Equipment Loads

The design engineer places the heat rejection from each system component on the system layout plan. This procedure will identify the areas of high heat dissipation and will assist in determining the air distribution requirements. Some computer equipment may be subject to variation in use; however, the calculations should be based on worst-case conditions with maximum heat rejection from all equipment. The total heat load should include a consideration for future computer system expansion. If specific future expansion plans are not available, a factor of approximately 25 percent of present equipment heat load is added.

Building Loads

When all the system component loads have been indicated on the layout drawing, the design engineer prepares the heat gain and heat loss requirements for the building or area. The procedures for determining these loads are the same as the procedures used in an office building or area.

Zoning

A study of the internal heat gains by areas and a study of the building heat gains or losses allows the designer to establish whether zoning is necessary. In most instances, one zone is required to serve the interior area. If the space has exterior walls, it may be necessary to provide a zone to handle the entire perimeter.

Outside Air

The use of outside air as a cooling medium is not recommended, because outside air is subject to wide variations of temperature and humidity. Since in most instances, a leaving dry bulb temperature of approximately 54°F (12°C) is required for the space, outside air at that temperature may require the addition of large amounts of moisture to prevent the relative humidity from becoming too low.

The fresh air requirements are few and are based upon the number of persons occupying the area for periods of 1 hour or longer. Forty cmf of outside air is recommended for each person; this quantity will not cause undue swings of humidity within the space.

Temperature Control

The space temperature is regulated with minimum control and maximum stability if terminal reheat is employed.

The space thermostat should be capable of controlling the dry bulb temperature within 15 degrees (F) above and below the set point of 72°F (22°C).

The key to good final control within each space lies in obtaining the proper apparatus dew point with the corresponding leaving wet bulb and dry bulb air temperatures. With these conditions properly maintained, it is necessary to have only a single space thermostat in each zone. This space thermostat regulates the zone face and bypass dampers and controls the reheat coil, in that order, adding reheat in direct relation to the reduction of heat from internal equipment and external sources.

The outside air opening is provided with a motorized damper to open and close as the air handling unit is running or shut down. The quantity of fresh air taken into the system is controlled by a manual damper in the outside air opening.

The use of multiple-circuit refrigeration units on a water chiller controlled by a sequencing switch or unloaders makes it possible to accurately maintain the proper temperature of water to the cooling coil. The temperature of water entering the cooling coil is controlled by a three-way valve that is positioned in relation to the leaving air dry bulb temperature from the apparatus. By means of this control program, it is possible to have a constant leaving dry bulb temperature with the corresponding leaving wet bulb temperature.

The placement of the zone thermostats is important, and their location should be made on the same computer equipment layout from which the heat gain calculations were made. The designer can then be certain that the particular zone thermostat will sense conditions that are average for its specific area and will not be subject to the performance of a specific component within the zone.

Humidity Requirements

In areas with outside design conditions lower than - 10°F (-12°C), it is necessary to install a humidifier to add moisture to the air to compensate for the dry air entering the space by infiltration and ventilation. The relative humidity in a computer installation is maintained at 50 percent. The space humidistat is capable of controlling the relative humidity within 5 percent of the set point. In cold climates, minimize the area of outside-exposed glass to prevent condensation on the inside glass surface.

Select a humidifier that is easily maintained and provides moisture without adding sensible heat to the air stream. The atomizing spray-type performs satisfactorily in areas where the mineral content of the makeup water has a hardness of less than 10 grains and where the unit is located upstream from the filtering section. In areas where the mineral content of the makeup water exceeds a hardness of 10 grains, it becomes necessary to treat the makeup water to reduce the mineral content or spray the water against a metallic filter. The mineral deposits will then collect on the filter which can be replaced as necessary. A humidifying section in the air handling unit that sprays water directly on the cooling coil is not recommended, because the mineral buildup that will occur on the coil surfaces will result in loss of efficiency on the air and water sides of the system.

Monitoring Programs

It is essential to provide continuous service for a computer installation; therefore, establish a monitoring program, whereby abnormal conditions are sensed prior to failure of the system. Provide alarm devices to indicate an abnormal condition on the chilled water to the cooling coil, the dry bulb temperature of the air leaving the apparatus, the space conditions for the respective areas, and the pressure drop across the filter bank. These alarm devices must be independent of any alarm devices built into the computer equipment. In addition, the computer system operating personnel should maintain a daily log of the operation of the air conditioning system with respect to chilled water temperature and leaving air dry bulb temperature.

Establish a scheduled maintenance program on the equipment maintaining the space environment. Update a log on each piece of environmental equipment at each scheduled service period, thereby permitting maintenance personnel to actually measure the conditions provided in relation to design conditions established on the log sheet.

MIXED TEMPERATURE SYSTEM

The following paragraphs are a basis for dete the requirements for a computer installation w some of the equipment is cooled with air at ro temperature and other equipment is cooled wit at lower than room temperature.

Air Distribution System

The air distribution system consists of either plenum, which uses the suspended ceiling as t supply diffuser or overhead ducts with individu ceiling diffusers (figure 4-5).

The equipment layout dictates the quantity of a required for specific areas within the compute Air quantities must be supplied in relation to t dissipation of heat occurring in the computer When a ceiling plenum air distribution system it may be necessary to provide a supplementa distribution device in the suspended ceiling to the increased volume of air required to absorb dissipated heat.

Certain computer equipment cabinets employ s contained cooling devices. Do not deliver sup within 6 feet in any direction of a cabinet of th

Locate the return air inlets either high on the in the ceiling. It is desirable to have the retu inlets near the areas of greatest heat rejectio

Balance the air conditioning duct system after computer equipment is installed and operating that the inlet air to any piece of equipment is the limits specified by the site preparation and stallation manuals.

It is not necessary to provide thermal insulati either the supply or return duct in the ceiling unless there is abnormal heat gain from the s above.

Equipment which must be supplied with tempe takes this air in through the bottom of the cab Deliver air to the cabinet intakes from an und distribution system of a plenum created by the floor (refer to figure 4-5) or ducts installed in

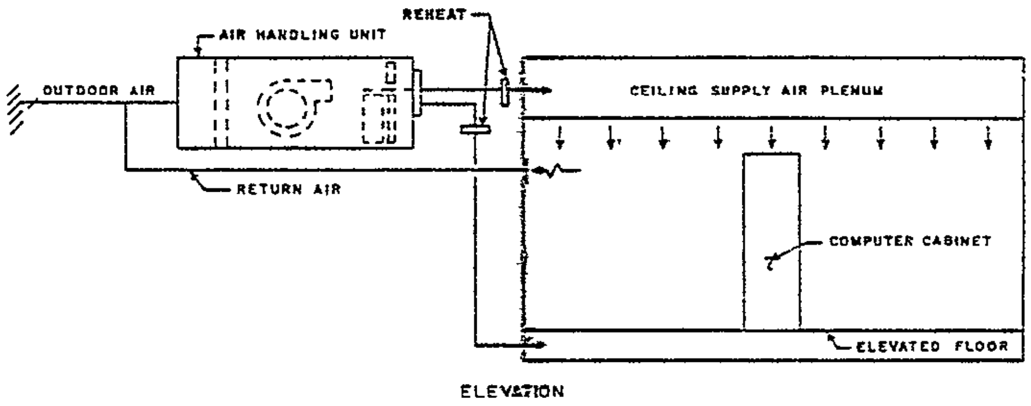


Figure 4-5. Air Conditioning System for Plenum-Cooled Computer Equipment

underfloor space delivering air directly to the cabinet intake locations. Take extreme care when using ducts to ensure a uniform velocity and air distribution to the cabinets. Provide equalizing grids and dampers to accomplish this.

The quantity of air required is equal to the summation of air quantities for the cabinets plus an allowance of 5 percent to 10 percent to assure a positive pressure of 0.02-inch above room pressure at the inlet to the cabinets.

Alternate Air Distribution System

A room air distribution system which uses the underfloor space as a supply air plenum is an alternate method of air distribution where the overhead air distribution system is impracticable.

The supply air is ducted directly from the air conditioning unit to the underfloor plenum for distribution to the room in lieu of an overhead duct system (figure 4-6). Since tempered air must be supplied to certain equipment, duct this air from the air handling unit to equipment directly or to a plenum around this equipment only.

Air Side Equipment

Refer to Room Temperature System.

Refrigerant Side Equipment

Refer to Room Temperature System

Condensing Medium

Refer to Room Temperature System.

Design Procedures and Considerations

Refer to Room Temperature System.

Monitoring Programs

It is essential to provide continuous service for a computer installation; therefore, establish a monitoring program, whereby abnormal conditions are sensed prior to failure of the system. Provide alarm devices to indicate an abnormal condition on the chilled water to the cooling coil, the dry bulb temperature of the air leaving the apparatus, the space conditions for the respective areas, and the pressure drop across the filter bank. These alarm devices must be independent of any alarm devices built into the computer equipment. In addition, the computer system operating personnel should maintain a daily log of the operation of the air conditioning system with respect to chilled water temperature, leaving air dry bulb temperature, and temperature of underfloor air supplied directly to computing equipment.

Establish a scheduled maintenance program on the equipment maintaining the space environment. Update a log on each environmental apparatus at each scheduled service period, thereby permitting maintenance personnel to actually measure the conditions provided in relation to design conditions established on the log sheet.

The required room air distribution is accomplished by locating floor registers around the perimeter of the room and near equipment in relation to the heat release. Pay close attention to the placement of floor registers to provide the necessary cooling while avoiding conditions which may cause operator discomfort.

Because of noise and draft conditions, size the floor registers to deliver the quantity of air desired at a velocity of not more than 650 fpm.

Provide supply registers with volume dampers to facilitate balancing air quantities and to provide maximum flexibility for future room changes.

The underfloor static pressure is sufficient to maintain 0.05-inch pressure at the floor register.

ACOUSTICS

The principal sources of noise in the computing area are the electromechanical devices such as line printers and card punches. Place these devices in an area of the computing room where a relatively high noise level is not objectionable.

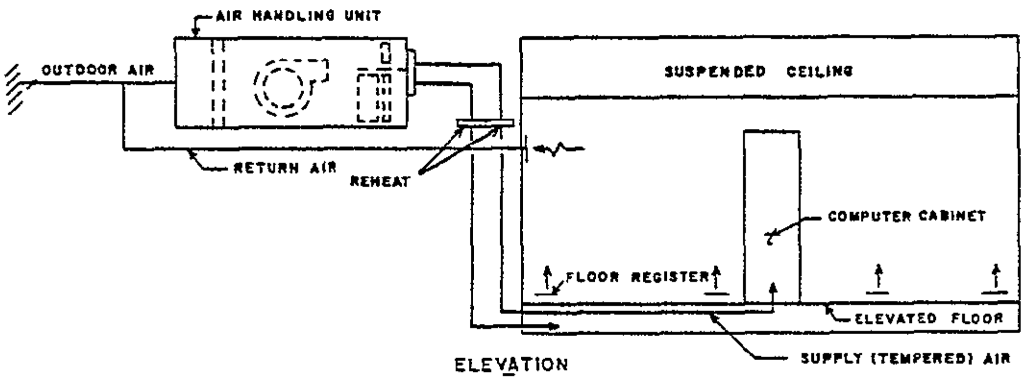


Figure 4-6. Alternate Air Conditioning System for Plenum-Cooled Computer Equipment

487

Acoustical treatment of the computer room is desirable but not essential for system operation. For the best room acoustics, the customer should follow the recommendations of an acoustical consultant. However, if a consultant is not used, Control Data has an Acoustics Handbook which delimiates various tests, considerations, and treatments to acoustical problems.

SHOCK AND VIBRATION

The customer should contact the Control Data site representative if the installation will be in a building or area that is subject to vibrations or shock effects. Computer systems can withstand intermittent low frequency vibrations or minor shocks on the Richter Scale; however, shock and vibration effects are accumulative over long periods of time and may cause unpredictable degradation of performance. Control Data does not guarantee a vibration tolerance, and each installation must be individually inspected to certify operation.

ALTITUDE

Control Data certifies satisfactory operation of all its products at altitudes of up to 6000 feet (1830 meters). The customer should inform Control Data of the need for a special requirement for any installation that will be above 6000 feet.

MOTOR-GENERATOR UNITS

The computers require motor-generator (M-G) units for frequency/power conversion and/or power stabilization. The MGs are designed to run continuously with a minimum of attention; however, some periodic maintenance is required to ensure trouble-free operation. Consider the M-G maintenance during the site selection and design phases.

Because of their noise level and heat generation, the MGs should be in an area other than the computer room, preferably in a room adjacent to or directly below the computer room. When selecting the M-G area, consider the following.

1. Allow access to the MG and control cabinet for cleaning, servicing, and repair. Place the M-G control cabinet so as to minimize the length of lead wires.
2. Allow sufficient space to permit the removal of the exciter and start motor shafts.
3. Allow sufficient space to permit free or ductec air flow to air intakes with discharge to outside of the M-G room.
4. Do not install units in locations where there is any possibility of exposure to steam, corrosive atmosphere, or danger of flooding.
5. If the MGs will be in a restricted area, provide for hoisting bolts above the units for ease of maneuvering the equipment.
6. Post appropriate warning/danger signs in conspicuous locations. Consider locking the M-G room to prevent unauthorized access.

Refer to the manufacturer's literature for specific installation instructions. In all cases, installation must conform to local codes and/or ordinances.

RADIO FREQUENCY AND ELECTROMAGNETIC INTERFERENCE

Do not expose the computer room (and/or site) to or in a direct-line-of-sight path of high intensity radio frequency interference (RFI). Grounding Systems, section 7 and appendix A delimiate the need and requirements for RFI and EMC control of the computer site.

PAPER TAPE, PUNCH CARD, AND MAGNETIC RECORDING MEDIA STORAGE 5

INTRODUCTION

When planning the computer room or site, consider storage areas for recorded media. Because paper and cardboard products produce fibrous lint and dust, keep only sufficient quantities for each day's use of typewriter and line printer paper and unpunched card stocks in the computing area.

Set aside a convenient area adjacent to the computer room for supply storage of paper and cardboard products and magnetic recording media.

NOTE

Dependent upon the nature of the customer's business, loss of recorded information may be catastrophic. Consider special security precautions and/or master file duplication. (Refer to Major Catastrophe under Magnetic Recording Media in this section.)

PAPER TAPE AND PUNCH CARD STORAGE

In order to reduce errors due to misalignment, store paper tape and punch cards in an area that has the same temperature and humidity as the computer room. Avoid extremes in temperature and humidity. The recommended temperature range is 62°F to 78°F (16.7°C to 25.0°C) with a relative humidity between 35 percent and 60 percent.

Condition paper tapes and/or cards which have been subjected to environmental extremes outside the recommended ranges prior to use by exposure to the computer room ambient conditions. Conditioning time varies from 4 to 6 hours, dependent upon the conditions to which they were subjected.

NOTE

Never use direct heat such as lamps and heating coils to warm tapes or cards.

MAGNETIC RECORDING MEDIA

There are two main types of magnetic recording media: fixed (such as disk files, drums, etc.) and portable (such as disk packs, tapes, etc.). The fixed recording media are housed in their own cabinets within the computing area and are subject to the same control and conditions as the central computer; accordingly, only the portable recording media are discussed.

MAGNETIC DISK PACK STORAGE

Store magnetic disk packs in individual self-sealing cases. These cases protect the pack from dust and sudden environmental changes that may occur as the packs are being transported between the storage and the computing areas.

Store the packs horizontally in steel cabinets (figure 5-1). (Control Data does not recommend stacking disk packs on top of one another.) Keep the cabinets within the following environmental limits for disk pack operations.

- Temperature range: 60°F to 120°F (16.6°C to 48.8°C)
- Relative humidity range. 8 percent to 80 percent with a wet bulb reading not to exceed 70°F (25.5°C).

Wider environmental limits are tolerable for extended periods of storage (recorded or unrecorded) for up to 5 years as follows.

- Temperature range: -40°F to 150°F (-40°C to 65.5°C)
- Relative humidity range: 8 percent to 80 percent with a wet bulb reading not to exceed 85°F (29.4°C).

CAUTION

Whether in use, transport, or storage, never expose disk packs to magnetic fields. Exposure to any magnetic field, having an intensity of 50 gauss or greater, may cause loss of data.

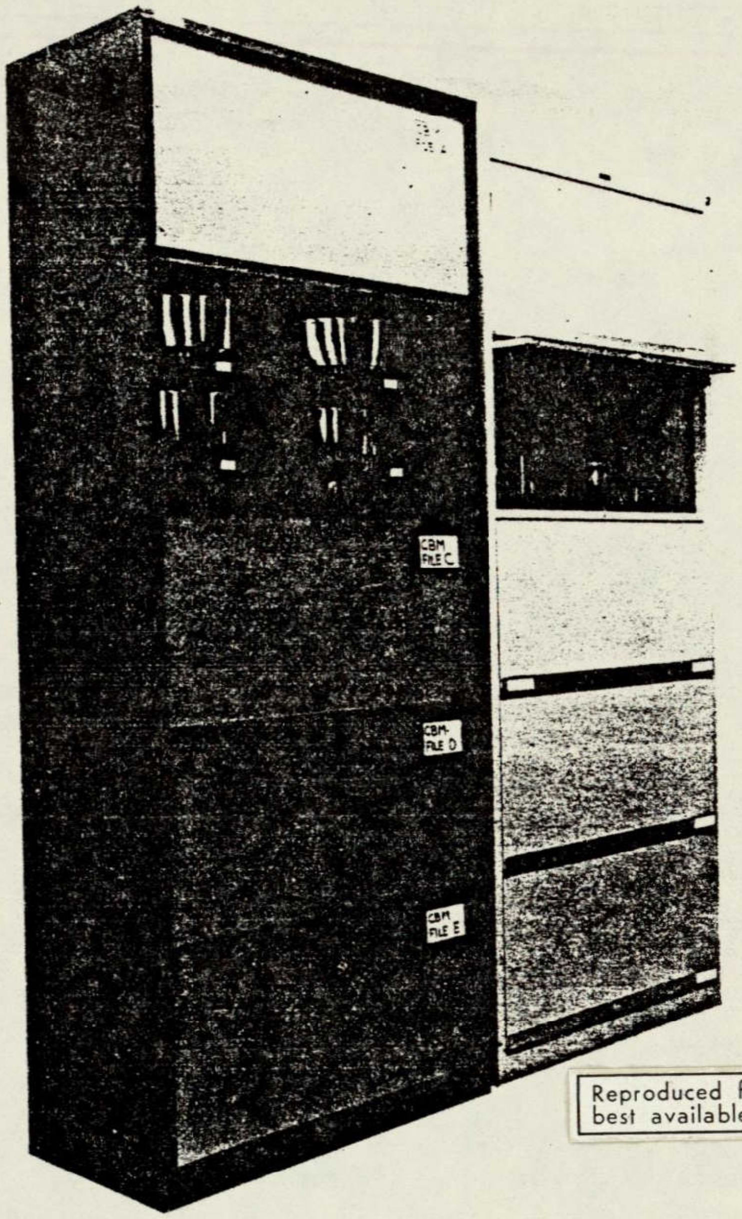
If the disk pack was exposed to environmental extremes or the storage area environment is different than that of the computer room, condition the packs by bringing them into the computer room environment for a minimum of 2 hours prior to use.

MAGNETIC TAPE STORAGE

Store magnetic tapes or reels in individual self-sealing cases (figure 5-2). Store tapes vertically in bins or on racks in an area having the same environmental characteristics as the computer room.

Avoid exposure of magnetic tapes to extreme environmental conditions. Recommended operating and storage limitations are as follows

- Temperature range 62°F to 78°F (16.7°C to 25.5°C).
- Relative humidity range 35 percent to 60 percent.



Reproduced from best available copy.

Figure 5-1. Two Types of Disk Pack Storage Cabinets

CAUTION

Never use heat lamps or coils to warm tapes or speed up conditioning time.

Condition any tapes subjected to environmental extremes by bringing them into the computer room environment prior to use. Dependent upon the conditions and exposure time to which the tapes were subjected, conditioning time will range from 4 to 16 hours.

Whether recorded or not, rewind tapes once or twice yearly to release any stress buildup due to expansion and/or contraction.

CAUTION

Do not expose tapes to any magnetic fields or material. (Do not store reels in cabinets having magnetic latches.) Any magnetic field intensity greater than 70 gauss may cause loss of data.

MAJOR CATASTROPHE

The following paragraphs cover the reaction of and resultant damage to magnetic tape for two forms of major catastrophes, fire and nuclear radiation.

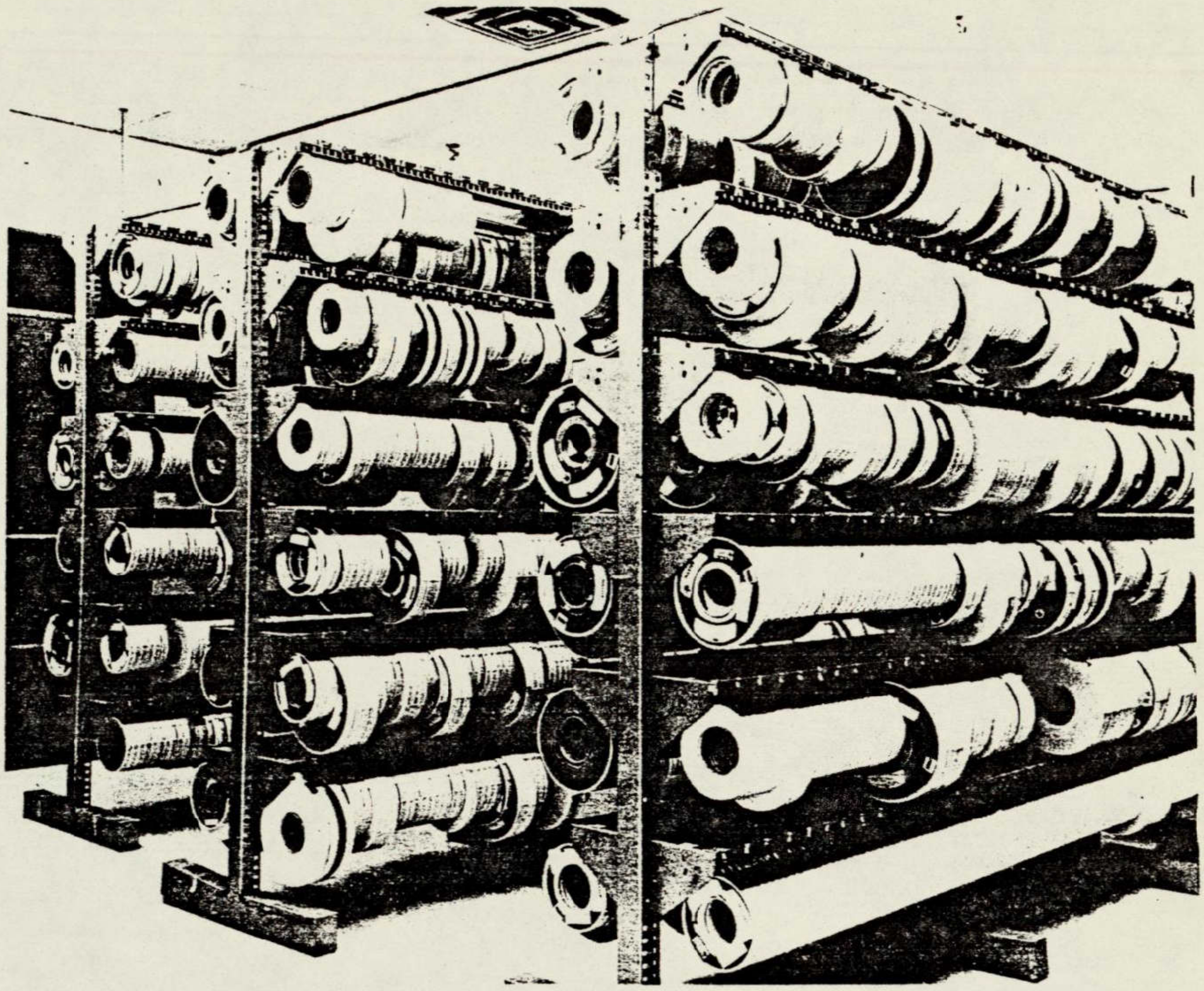


Figure 5-2. Typical Magnetic Tape Storage Racks

Fire

In order for any substance to burn, a breakdown of the organic materials contained in it must exist. The organic contents of magnetic tape are the plastic backing and the binder. To burn, the contents must first vaporize (increasing their exposure to the oxygen in the atmosphere) and then rapidly oxidise (burn) to form light and heat. An ample supply of oxygen is required to sustain burning. Because magnetic tapes do not contain builtin oxidizers, it cannot burn in the absence of air.

While the self-ignition temperature of a polyester-backed tape is in the neighborhood of 1000°F (538°C), damage commences around 250°F (121°C). The following indicates the effects that occur if a magnetic tape is heated to the approximate temperatures indicated and then cooled.

- 250°F (121°C) - backing distortion
- 300°F (149°C) - 1 1/2 percent shrinkage of polyester film
- 320°F (160°C) - softening of both the backing and binder with some adhesion of adjacent layers
- 325°F (163°C) - 25 percent shrinkage of polyester film
- 550°F (288°C) - darkening and embrittlement of the backing and binder
- 1000°F (538°C) - charring of the backing and binder

491

Once charring occurs, the tape cannot be unwound from the reel, because it will flake when touched.

Winding and storing magnetic tape properly will lessen the possibility of damage in the event of fire since tape is a poor conductor of heat. It is sometimes possible to recover information from a tape receiving slight fire damage by carefully rewinding it at minimum tension. Transfer the information it contains immediately to another reel of undamaged tape.

The use of a CO_2-type of fire extinguisher is recommended for combating burning magnetic tape. CO_2 is clean and contains no chemicals that could harm the tape. If water reaches the tape, it probably will not cause complete failure, but there may be some evidence of cupping (transverse curvature). The amount of cupping depends on the quality of the wind and the length of time the roll was exposed. If the wind is loose or uneven, the water can more easily reach the oxide surface, and the cupping would be more pronounced. Remove the tape from the water as soon as possible, and certainly within 24 hours.

After removal, allow the rolls to dry on the outside at normal room temperature and then rewind them a minimum of two times. This aids the drying operation and also helps the rolls to return to equilibrium faster.

If a temperature increase is incurred while the tape is water soaked, steam or high humidity is present. This is more likely to cause damage than water alone. A temperature in excess of 130°F with a relative humidity above 85 percent may cause layer-to-layer adhesion as well as some physical distortion.

Tape stored in canisters, if closed properly, will keep water spray from a sprinkler system from reaching the tape and will tend to protect the tape from the radiant heat of a nearby fire.

To prevent fire involving magnetic tape, store tape in a noncombustible area (for example, a room with metal shelves and sheet-metal walls) and ensure that no combustible materials are stored in the vicinity For maximum fire security, store magnetic tape in a fireproof vault that is capable of maintaining a desirable internal temperature and relative humidity for a reasonable length of time.

Nuclear Radiation

Usually, magnetic tape is unaffected by nuclear radiation until the dosage approaches a level 200,000 times greater than that at which death occurs in 50 percent of the exposed humans. At this level (100 megareps), radiation tends to increase the layer-to-layer signal transfer (print-through) by approximately 4 db. This is not serious enough to prevent information retrieval.

At this level, some physical effects are evident; the backing shows significant embrittlement and its wear life may be reduced by as much as 60 percent.

It is reasoned that whatever electromagnetic field might result from a nuclear detonation it would not be of sufficient intensity to adversely affect the tape, therefore, the threat of signal erasure is virtually nonexistent. The effect of neutron bombardment is limited to activation of the iron-oxide in the coating. This produces a radioactive isotope that might become a source of further radiation, but it is theorized that such activation would not produce a change in the overall magnetic properties of the coating.

Radioactive dust or fallout is not capable of producing the dosage necessary to adversely affect magnetic tape. The earlier recommendations to protect the tape from normal contamination are also applicable.

FIRE AND SECURITY PRECAUTIONS

It is important to provide for loss prevention during the planning stages of the computer site. The two types of loss prevention are fire protection and site security.

The extent of the measures taken to prevent a loss will vary dependent upon the system application. The following are evaluated as they relate to a particular application.

1. Cost of equipment
2. Location of data processing facility
3. Location of the equipment within the processing facility
4. Data processing essentiality
5. Sensitivity of the data
6. Site/system supervision

After the customer evaluates his application, consideration may be given to fire and site security protection. Control Data is experienced in fire and site security protection and has personnel available to advise and provide service in these areas.

FIRE PROTECTION

The National Fire Protection Association Bulletin 75 (NFPA 75), "Standard for the Protection of Electronic Computer Systems" details the minimum requirements for the protection against fire in a computer site. This bulletin considers computer facility construction and location, protection of computer rooms, protection of equipment, protection of records, and emergency procedures.

Control Data recommends that the computer site be located in a building constructed of noncombustible materials and removed from adjacent areas where hazardous processes are taking place. Use noncombustible construction materials in the computer site, consider all materials including raised floor, walls, partitions, ceilings, etc..

Computer facilities, where required by NFPA 75, are protected by any of several different types of fire extinguishing systems available. Sprinkler, fixed carbon dioxide, halogenated gas, and dry and wet chemical systems are more popular. Control Data considers a sprinkler system the safest and most effective means of controlling any fire.

CAUTION

Equipment should be deenergized prior to discharge of the extinguisher or system.

Protect against discharge while the computer is energized through the use of detection devices, alarms, automatic power-down of the equipment, and if necessary, a delayed-action sprinkler system utilizing a valve which is actuated by a product of combustion and/or rate-of-heat-rise sensing devices. Hand-held class A† extinguishers are available for extinguishing paper fires. There are a number of systems available for the extinguishing of fires that may originate within the computer equipment or under a raised floor (in cables or wiring). While this hazard is minimal, some provision should be considered. Control Data recommends the use of portable class C†† Halon 1301 extinguishers or the installation of a fixed Halon 1301 under-the-floor flooding system to serve critical cabinets.

Maintain practical record's storage outside of the computer room. Sprinkle the storage room and have portable extinguishing equipment on hand. Duplicate records vital to a business or operation and store in a separate area.

The most important key to an effective fire protection system is the establishment and regular practice of emergency procedures. Train all personnel in good housekeeping practices, operation of extinguishing devices, proper equipment shutdown methods, site evacuation procedures, and notification of proper authorities.

SITE SECURITY

Protection of the computer site and system against malicious damage, theft of proprietory information, and accidental damage must be considered in the overall loss prevention program. Protection considerations vary with application but include computer facility design, site personnel access, and contingency plans.

Control Data recommends that the computer facility be designed without viewing windows and that it be located within the interior of a building. Use a fireproof separate but contiguous room for the storage of magnetic records. Minimize openings into the computer room, with an anteroom for each opening, to control personnel access. Locate critical power and air conditioning components to minimize the possibility of malicious damage. Some applications may require the installation of standby power generation systems for the computer air conditioning.

The control of personnel into and out of a computer facility is absolutely essential in minimizing risks to equipment or program damage and theft. Control Data, in critical applications, recommends that the anteroom at each entrance be equipped with devices to limit access, equipment, wherever necessary, to detect the presence of magnetic material (including tapes), and a closed circuit TV system monitored by a security guard.

† Class A fires are combustibles such as wood, paper, cloth, etc..
†† Class C fires are electrical.

To minimize business interruption in the event damage does occur to the site or system, establish a contingency plan. Control Data recommends that all vital records be duplicated and stored at a location separate from the computer facility. Also, arrangements should be made for the temporary use of another computer system either from the manufacturer or other user who has duplicate equipment.

INTRODUCTION

A computer system requires the installation of both a protective (safety) grounding system and a reference-plane grid grounding system (figure 7-1). The protective system guards personnel against potential shock hazards and protects the equipment from damage in case of electrical malfunction. The grid grounding system controls the circuit paths of radio frequency interference (RFI) between the pieces of equipment and the surrounding environment to achieve an electromagnetic compatibility (EMC).

NOTE

If any doubt exists as to the intensity levels of a potential or nearby interference source, conduct a site survey. Appendix A contains an explanation of RFI causes, sources, and site survey procedures and equipment.

It is the customer's responsibility for the proper installation of both grounding systems in accordance with local codes and the information and requirements set forth here and in the appropriate system's section 2 site preparation manual.

PROTECTIVE GROUNDING SYSTEM

CAUTION

The neutral (usually white) and ground (usually green) wires of the power cables must not be electrically connected except at the building service entrance.

The protective grounding system connects all of the computer system cabinets, switch boxes, frequency converters, air conditioners, and computer-related equipment to an earth ground. The conductors for the ground connections are either metallic electrical conduit or the ground wires of the equipment power cables.

REFERENCE-PLANE (EMC) GRID GROUNDING SYSTEM

The EMC grid ground system controls the radio frequency interference paths between equipments. The RFI may be from such sources as adjacent logic circuits, power and/or control circuits, or remote radio circuits.

The EMC grid grounding system provides separate ground connections to computer system logic chassis and cabinet frames from an artificial earth ground. Unlike the protective grounding system, the grid grounding system does not connect to switch boxes, frequency converters, air conditioners, or other computer-related equipment. (An additional exception is small table-top equipment such as display terminals and small printers. These equipments are designed to be grounded only through the green ground (safety) wires in the equipment power cords.)

The EMC artificial ground is a conducting grid that is either the metal-supporting structure (figure 7-2) of the computer room's raised floor or a conducting wire-mesh grid (figure 7-3) beneath the raised floor.

NOTE

Because some manufacturer's metal supporting structures do not make satisfactory artificial grounds, the customer must obtain approval from the site planning engineer before using this type of grid.

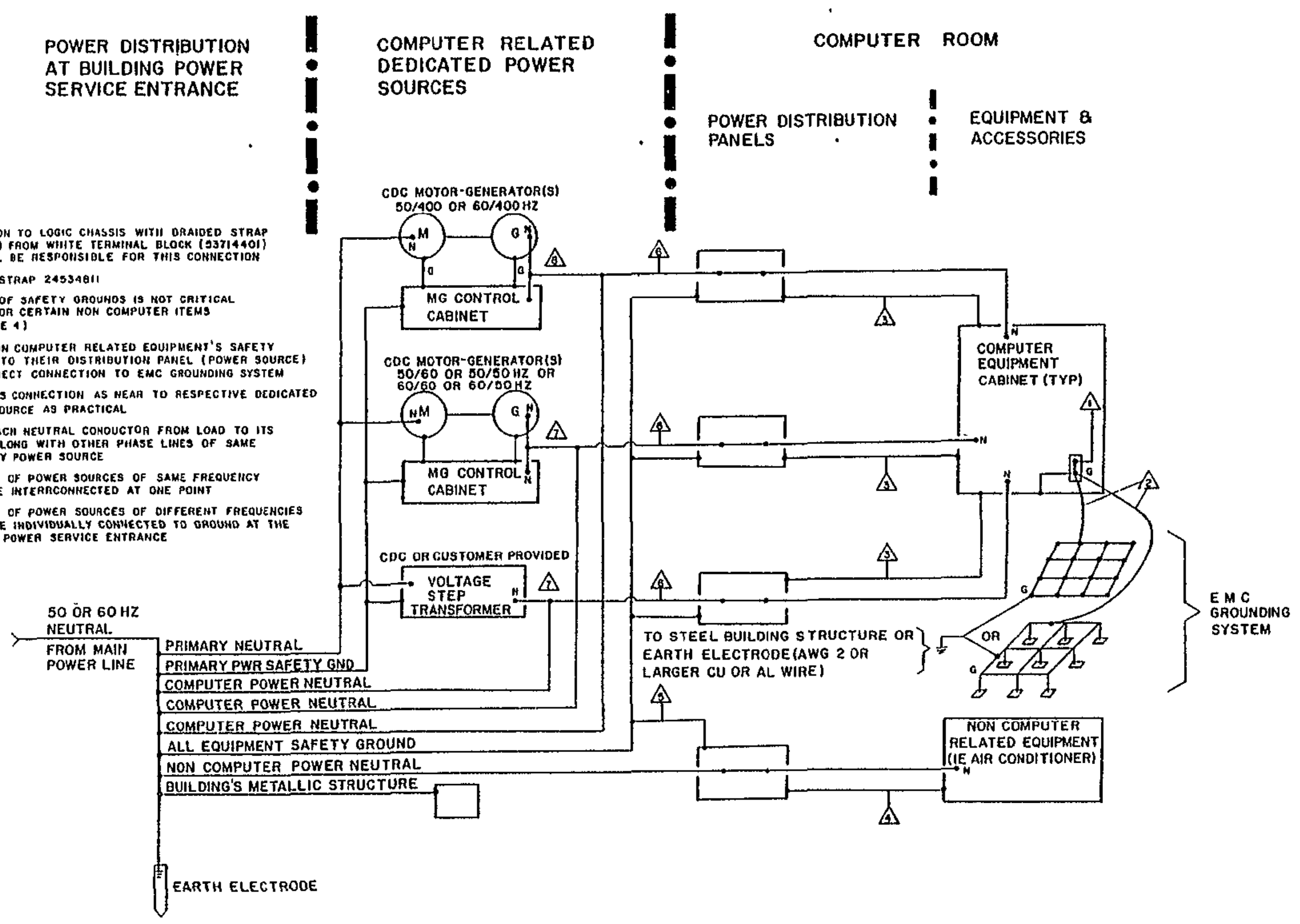
When the raised floor metal supporting structure is not acceptable as a grid or is not used, install a wire mesh grid. This grid is constructed from heavy copper or aluminum wire and lays directly on the building floor, under the raised floor.

Bonding straps provide the means of electrically connecting the raised-floor structure grid or the wire mesh grid to the equipment logic grounds. The bonding straps are short braided conductors which connect individual equipments to the grid as shown in figures 7-2 and 7-3.

The reference grid grounding system provides electromagnetic compatibility for most computer installations. Certain types of installations require additional or different techniques for controlling RFI and must be treated individually. These types of installation are the following.

1. Areas where analog or low signal-level equipment is used
2. Areas where remote display or communications equipment is used
3. Areas where high radio frequency radiation levels exist.

Discuss these or other special computer installations requiring deviations from the following requirements with the site planning representative.


POWER DISTRIBUTION AT BUILDING POWER SERVICE ENTRANCE
COMPUTER RELATED DEDICATED POWER SOURCES
COMPUTER ROOM
POWER DISTRIBUTION PANELS
EQUIPMENT & ACCESSORIES
CDC MOTOR-GENERATOR(S) 50/400 OR 60/400 HZ
M
G
N
MG CONTROL CABINET
CDC MOTOR-GENERATOR(S) 50/60 OR 50/50 HZ OR 60/60 OR 60/50 HZ
MG CONTROL CABINET
CDC OR CUSTOMER PROVIDED
VOLTAGE STEP TRANSFORMER
COMPUTER EQUIPMENT CABINET (TYP)
50 OR 60 HZ NEUTRAL FROM MAIN POWER LINE
PRIMARY NEUTRAL
PRIMARY PWR SAFETY GND
COMPUTER POWER NEUTRAL
COMPUTER POWER NEUTRAL
COMPUTER POWER NEUTRAL
ALL EQUIPMENT SAFETY GROUND
NON COMPUTER POWER NEUTRAL
BUILDING'S METALLIC STRUCTURE
TO STEEL BUILDING STRUCTURE OR EARTH ELECTRODE (AWG 2 OR LARGER CU OR AL WIRE)
OR
EMC GROUNDING SYSTEM
NON COMPUTER RELATED EQUIPMENT (IE AIR CONDITIONER)
EARTH ELECTRODE

NOTES'

1. CONNECTION TO LOGIC CHASSIS WITH BRAIDED STRAP (24534811) FROM WHITE TERMINAL BLOCK (93714401) CDC WILL BE RESPONSIBLE FOR THIS CONNECTION
2. BONDING STRAP 24534811
3. ROUTING OF SAFETY GROUNDS IS NOT CRITICAL EXCEPT FOR CERTAIN NON COMPUTER ITEMS (SEE NOTE 4)
4. ROUTE NON COMPUTER RELATED EQUIPMENT'S SAFETY GROUNDS TO THEIR DISTRIBUTION PANEL (POWER SOURCE) AVOID DIRECT CONNECTION TO EMC GROUNDING SYSTEM
5. MAKE THIS CONNECTION AS NEAR TO RESPECTIVE DEDICATED POWER SOURCE AS PRACTICAL
6. ROUTE EACH NEUTRAL CONDUCTOR FROM LOAD TO ITS SOURCE ALONG WITH OTHER PHASE LINES OF SAME FREQUENCY POWER SOURCE
7. NEUTRALS OF POWER SOURCES OF SAME FREQUENCY ARE TO BE INTERRCONNECTED AT ONE POINT
8. NEUTRALS OF POWER SOURCES OF DIFFERENT FREQUENCIES ARE TO BE INDIVIDUALLY CONNECTED TO GROUND AT THE BUILDING POWER SERVICE ENTRANCE

959

Figure 7-1. Neutral and Grounding Requirements

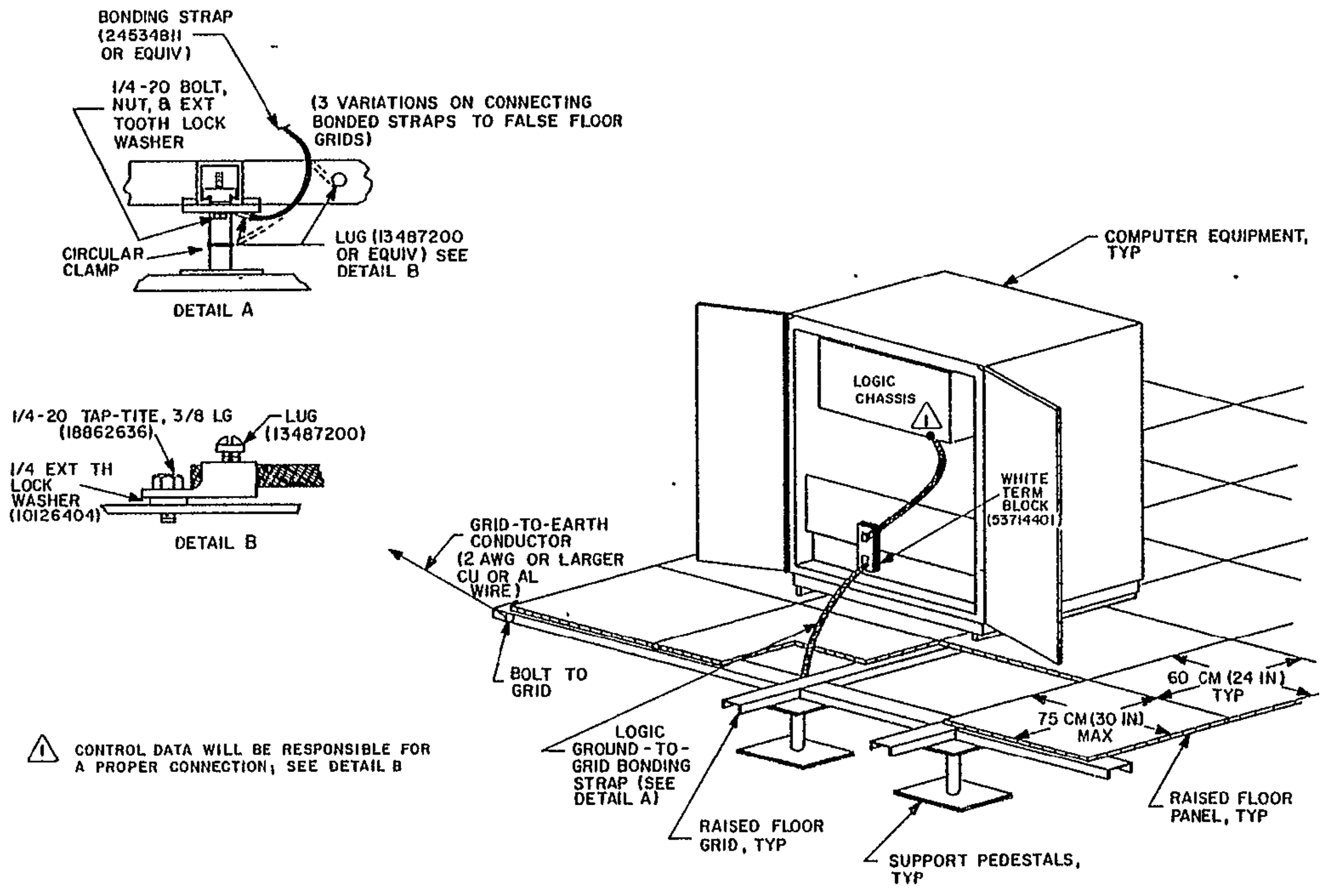
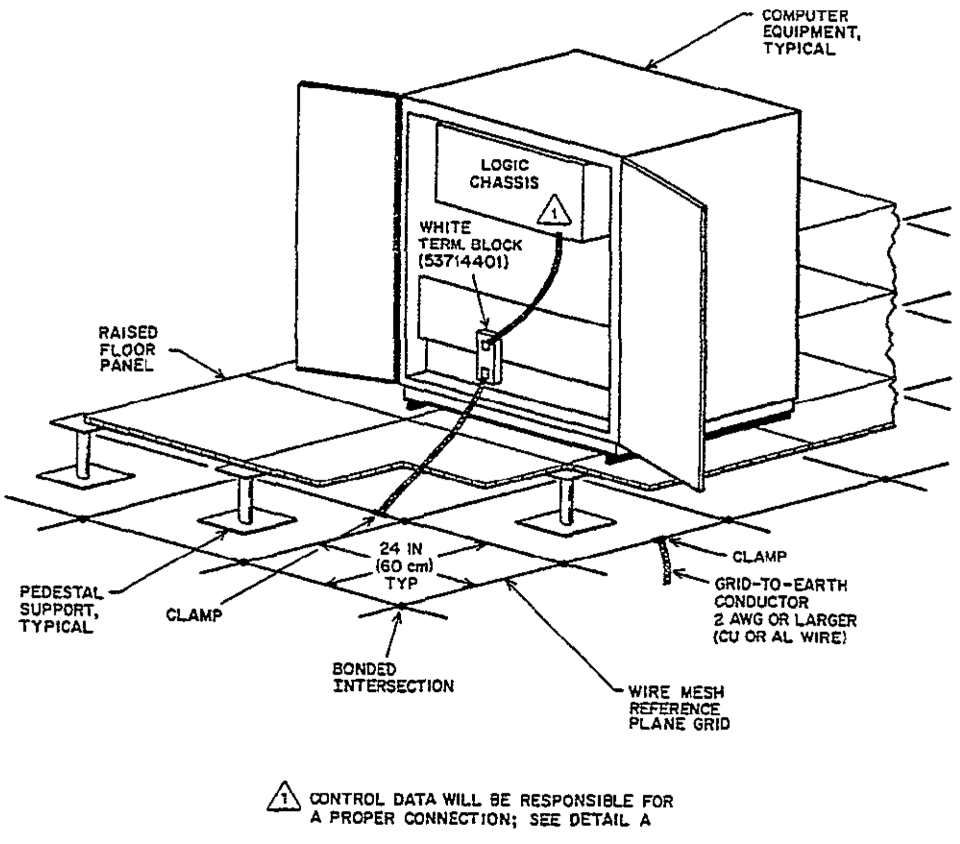

BONDING STRAP (24534811 OR EQUIV)
1/4-20 BOLT, NUT, & EXT TOOTH LOCK WASHER
(3 VARIATIONS ON CONNECTING BONDED STRAPS TO FALSE FLOOR GRIDS)
CIRCULAR CLAMP
LUG (13487200 OR EQUIV) SEE DETAIL B
DETAIL A
1/4-20 TAP-TITE, 3/8 LG (18862636)
LUG (13487200)
1/4 EXT TH LOCK WASHER (10126404)
DETAIL B
COMPUTER EQUIPMENT, TYP
LOGIC CHASSIS
WHITE TERM BLOCK (53714401)
GRID-TO-EARTH CONDUCTOR (2 AWG OR LARGER CU OR AL WIRE)
BOLT TO GRID
LOGIC GROUND-TO-GRID BONDING STRAP (SEE DETAIL A)
RAISED FLOOR GRID, TYP
SUPPORT PEDESTALS, TYP
60 CM (24 IN) TYP
75 CM (30 IN) MAX
RAISED FLOOR PANEL, TYP
CONTROL DATA WILL BE RESPONSIBLE FOR A PROPER CONNECTION; SEE DETAIL B

Figure 7-2. Raised-Floor Gird Grounding

437


COMPUTER EQUIPMENT, TYPICAL
LOGIC CHASSIS
WHITE TERM. BLOCK (53714401)
RAISED FLOOR PANEL
PEDESTAL SUPPORT, TYPICAL
CLAMP
24 IN (60 cm) TYP
BONDED INTERSECTION
CLAMP
GRID-TO-EARTH CONDUCTOR 2 AWG OR LARGER (CU OR AL WIRE)
WIRE MESH REFERENCE PLANE GRID
1 CONTROL DATA WILL BE RESPONSIBLE FOR A PROPER CONNECTION; SEE DETAIL A

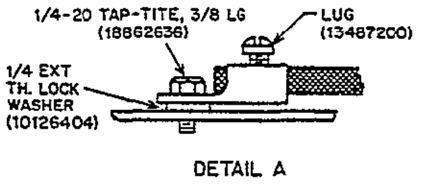

1/4-20 TAP-TITE, 3/8 LG (18862636)
LUG (13487200)
1/4 EXT TH. LOCK WASHER (10126404)
DETAIL A

Figure 7-3. Wire Mesh Grid Grounding

RAISED-FLOOR STRUCTURE GRID REQUIREMENTS

Material

The basic structural members are conductive metal, aluminum is preferred and steel is acceptable.

Demensions

The structural floor members extend a minimum of 3 feet (1 meter) beyond the periphery of the computer system.

The horizontal structural floor supports have a minimum cross-sectional dimension of 6 inches (15 centimeters).

When a single structural floor cannot extend under the entire computer system, separated structural floor supports must be used and electrically interconnected.

Construction

The structural floor supports must be continuous wherever possible. Where continuous construction is not possible, bond individual supports together by welding, bolting, or electrical bonding straps. Thoroughly clean all contact surfaces before bolting or bonding together.

WIRE MESH GRID REQUIREMENTS

Material

The grid is constructed of American wire gage (AWG) number 2 or larger, copper or aluminum, stranded wire.

Dimensions

The wire mesh extends a minimum of 3 feet (1 meter) beyond the periphery of the computer system.

Mesh conductors are formed in a pattern of squares or rectangles with all conductor intersections bonded. The recommended distance between intersections, along any conductor, is 24 inches (60 centimeters); a maximum of 30 inches (75 centimeters) is acceptable.

The horizontal plane of the grid is located within 18 inches (45 centimeters) below the surface of the raised floor.

When a single grid cannot be constructed to extend under the entire computer system, use individual grids and interconnect them by the use of braid or wire jumpers.

Construction

The wire mesh grid conductors are continuous and without splices wherever possible. Where continusous construction is not possible, splice individual conductors by split-bolt clamps, welding, or any other means that provides low-impedance contact with immunity to deterioration. Solder, used alone, is not acceptable.

The grid conductors are uniformly spaced, flat, and free of kinks, loops, and coils. The conductors are insulated, except at bonded intersections where the conductors are brazed or clamped for good electrical connections.

BONDING STRAP REQUIREMENTS

Material

Each bonding strap is a flat, flexible, braided, tinned-copper conductor. The strap is comprised of approximately 850 awg number 36 tinned-copper wires with an approximate area of 21,000 circular mils. Each strap is electrically insulated.

Dimensions

Each bonding strap has a minimum width of 1 inch (2.5 centimeters) and a maximum length of 2 feet (60 centimeters), measured from equipment cabinet opening to the grid connection point.

Routing

Each bonding strap is routed to minimize its length and prevent coiling, kinking, and twisting. Routing considerations should include equipment disassembly requirements, future movement within the room, and logic ground attachment point relative to the grid attachment point.

Attachment

Attach the equipment-end of the bonding strap to the logic ground, usually at a white terminal block and at the grid, using one of the methods shown in figures 7-2 and 7-3. Clamps used in connecting the strap to a grid must be electrochemically compatible with the grid connection point. The grid connection point must be free of all insulating materials prior to attachment of the strap. The application of nonpenetrating protective and corrosion inhibiting finishes is permissible after attachment.

499

INTRODUCTION

This section contains a general discussion of the power requirements for Control Data computer systems. More detailed information about power characteristics and requirements for a particular system is available from the Control Data planning representative and is in the associated system's section 2 site preparation manual. Typical system electrical schematics are also in the section 2 site preparation manual. Individual equipment cabinet electrical schematics are available from Control Data upon request.

GENERAL POWER REQUIREMENTS

Control Data computer systems are available for site installations that have either 50-Hz or 60-Hz power. For certain installations, use a 60-Hz system in place of a 50-Hz system if a 50-Hz system is unavailable. When this situation occurs, Control Data supplies a 50-Hz to 60-Hz frequency converter (consisting of M-G set and associated control cabinet) to provide compatibility between the site power and the system. In all installations, the computer systems use a frequency converter to change the 50-Hz or 60-Hz power to the 400-Hz power which is required by the computer system power supplies.

GENERAL POWER DISTRIBUTION

Design power distribution to the computer system for ease of installation and convenience of use in accordance with applicable electrical codes. Within the computer system equipment cabinets, all terminal strips, locking connectors, grounding, and internal wiring conforms to the National Electrical Code. Refer to section 7 for the system grounding requirements.

It is the customer's responsibility to provide and install items of the power distribution system. When installing ac power runs, the customer must not run 50-Hz, 60-Hz, and terminator power cables through common troughs or conduits. Other items of responsibility are the following.

1. Disconnects for the isolation of the main power line and/or the main subpower feeders
2. Magnetic contactors (activated by 120 vac) for energizing and deenergizing groups of circuit breakers in a predetermined sequence
3. Circuit breakers and associated mounting panels for 50/60-Hz and 400-Hz power distribution and individual equipment line protection.
4. Cables for distribution of 50/60-Hz and 400-Hz power from breaker panels to each piece of system equipment
5. Control wiring for power control panels and magnetic contactors
6. Power wiring for terminators, if required
7. Convenience outlets near each piece of equipment for test equipment
8. Ducting for power cables

POWER CONNECTIONS

Power connections to Control Data equipment are made with locking connectors or terminal strips. Control Data supplies the locking connectors consisting of caps and bodies. The connector caps are installed on the equipment before it leaves the factory. The customer must install the connector bodies at the computer site.

The customer also installs the power wiring to equipment terminal strip connections. These installations are done under the guidance of a Control Data representative. The installations require the power cables to extend 4 feet above the floor cutouts providing for slight movement of castered equipment. Strip armored coverings 6 inches from the wire ends and strip the insulation 1/2 inch from the wire ends. The customer must also provide strain reliefs to the power wires wherever they go through a junction box.

CABLE CUTOUT SEALING AND ACCESS

Air conditioning considerations may require sealing the cable cutouts in a raised floor If this is necessary, the customer must provide and install the seals prior to the computer system installation. The design of the seals must allow the passage of locking-type connectors.

PHASE ROTATION

For proper phase rotation, wire the receptacles used to connect the system M-G set and the various 3-phase peripheral equipment to the 50/60-Hz, 3-phase, 5-wire primary power source.

The color code/phase relation differs between various manufacturers, areas, and countries. Therefore, determine the proper phase relation from the wiring diagrams and existing local codes and practices. In the United States, the following code is standard.

- White is neutral

500

- Green is ground
- Black is phase A (connector pin X)
- Red is phase B (connector pin Z)
- Orange is phase C (connector pin Y)

Do not necessarily connect the wires distributing the 400-Hz, 3-phase, 208-vac power from the M-G set in sequence. The 400-Hz power is used only to energize ac supplies, and phase rotation is not important.

MOTOR GENERATOR

Each M-G set is a brushless frequency converter consisting of a 50-Hz or 60-Hz, 3-phase induction motor that drives an integral 400-Hz, 3-phase generator (0.9 power factor). The motor is a standard National Electrical Manufactures Association, United States (NEMA) design B, squirrel cage induction motor. The generator is a salient pole, rotating field, synchronous machine with a 3-phase ac exciter. A 3-phase rectifying circuit, mounted on the converter shaft, rectifies the output of the ac exciter and supplies dc to the generator field. To ensure proper alignment, the motor and generator stators are mounted in a common frame, and the rotors are mounted on the same shaft. For installation purposes, install the M-G set on any flat surface capable of supporting weight. Do not necessarily bolt the unit to the floor. Size the wires carrying the 400-Hz power from the M-G set to the distribution panel to allow no more than 2 percent voltage drop over the entire length of the run.

The M-G sets convert 50/60-Hz, 3-phase power to 400-Hz, 208-volt, 3-phase power for the computer and other devices. Control Data furnished the required M-G set(s) and associated controller cabinet(s) for the system.

Data sheets for the M-G sets and their control cabinets are in the site preparation manual. The size and number of units furnished depends on the system configuration and is determined by the site planning representative.

To minimize computer room heat gain and noise levels, locate the M-G sets in a power room separate from the computer center. The M-G sets are not designed for outdoor operation; install them in an area where temperature and humidity is controlled within the limits listed on the appropriate data sheets.

Connection diagrams, maintenance, and part manuals for the M-G sets and control cabinets are furnished with each system.

ELECTROMAGNETIC INTERFERENCE AND COMPATIBILITY TESTING A

INTRODUCTION

The need for electromagnetic compatibility (EMC) of computer equipment is a result of the proliferation of electronic devices throughout the world. Various pieces of computer equipment must be compatible, as well as the computer equipment itself must be protected from external sources of electromagnetic interference (EMI).

Most industrialized countries of the world have set (and enforced) fairly uniform standards constituting maximum acceptable levels of emitted or generated EMI from the computer equipments. It is the policy of Control Data to accommodate such standards where required.

Unfortunately, different regulations exist for different types of electronic devices. Accordingly, such devices as radar, radio transmitters, television, welding, diatherny, and other devices are designed with EMC standards and/or considerations in mind, however, they still produce high levels of electromagnetic energy which may cause interference to other electronic devices not designed to operate in their proximity.

Most manufacturers of computing equipment adhere to some standards pertaining to their individual susceptibility to EMI. Unfortunately, the higher the desired tolerance to EMI, the higher the price is. A typical tolerance to EMI of CDC's standard hardware products was established in early 1967, and since then, all its standard products meet or exceed this tolerance. A detailed description of these tolerances and reference test-methods is documented in CDC's Engineering Standard 1.30.22. †

The following guidelines aid in determining and providing measures to reduce or eliminate the undesired intensity, frequency, mode of propagation, direction of arrival, and rate of occurrence of high intensity, externally-generated radio frequency (RF) emanations

Such emanations, peculiar to a computer site, may constitute potential interference to existing or planned computer installations. Because of their radiation of high radio frequency energy levels; devices (for example, long-range search and tracking radars) may disturb the proper functioning of a wide variety of well-designed electronic equipment located nearby. Since most digital computer circuitries must distinguish between only two voltage levels (where difference is comparatively large), standard computer products (relatively unprotected) are tolerant to RF environments, although not immune to high intensity disturbances.

Protection of standard computer systems from both extreme (low and high frequency) radiation levels and excessive power line-transmitted transients may be provided (or become necessary) externally to the system. This cursory site survey is normally sufficient to assess the interference potential and provide ways and means to achieve compatible operation of the computer system in a threatening RF environment.

SURVEY

Although the possibility of simultaneous radiations of extreme intensity from more than one source exists, generally, direct attention to a single interference generator peculiar to the considered site. Visually spotting a large antenna structure, as employed for radio broadcasting, television, ground-based high-power communications, and radar installations is evidence of the increased probability or likelihood of excessive radiation level.

When antennas for high-powered transmissions are spotted within approximately 1/2 mile (800 meters) of a proposed or existing computer site, field intensity measurements are recommended. Similarly, extend primary concern to an approximate 3-mile radius (5 kilometers) in instances where high power radar beams in the 1- to 10-GHz frequency range are directed over clear line-of-sight at the computer installation.

Evidence of external RFI to existing, properly installed computer systems is usually obtained by conventional troubleshooting techniques employing oscilloscopes first. The modulation or pulse characteristics of the interfering RF signal is helpful in identifying both the interference source and the general areas of the computer systems specific exposure.

Without rather conclusive symptoms of computer malfunctions or other evidence of nearby high power transmitter installations, a computer site survey for thorough analysis of EMI is seldom justified economically and is usually unnecessary. The search or scanning of a wide frequency spectrum involves much specialized test equipment and operator time and is therefore, expensive.

SURVEY EQUIPMENT AND PROCEDURE

This section outlines the minimum apparatus and procedure required for obtaining the desired information concerning radiated and conducted EMI

Radiation

Employ tunable, narrow band test receivers, together with precalibrated antennas specifically designed to measure the field intensity, frequency, modulation characteristics, and polarization of the

†Obtain a copy from your local CDC site planning design engineer.

electric field vector of a radiation. The Singer Model NP-105 series receivers and VR-105, LG-105, LP-105, VA-105, and DM-105 series antennas are in wide use and fully meet the site survey requirements for radiation scanning in the 14-kHz to 1.0-GHz frequency range. The frequency range of 1.0-GHz to 10.0-GHz are covered with the Singer Model NF-112 series receiver and model AT-112 antenna. Test equipment of equivalent function and accuracy other than Singer is used; and Hewlett-Packard spectrum analyzers are also recommended.

Several restrictions apply to the performance of radiation tests pertaining to the statistical nature of field samplings and associated problems regarding multiple reflections which significantly affect measurement accuracy. Sample radiation field at several locations throughout the proposed or existing computer site. Measure the maximum field intensity by appropriate positioning and proper polarization of receiving antennas. Restrict measurements of the peak intensity of the electric field radiation vector to the 14-kHz to 10-GHz frequency range. This range covers nearly all sources of extreme level radiation to which computers are likely to be exposed to. Should an exception be anticipated or become evident, coordinate the site survey with Engineering and Architectural Services Division and EMC Engineering Unit representatives of CDC.

In general and normal cases, measurements of vertical polarization are most practical for the 14-kHz to 25-MHz frequency range, whereas horizontal polarization is most commonly used at higher frequency ranges. Nonetheless, always attempt determination of the polarization of incident radiation for the entire frequency range above 88-MHz.

Always scan the asimuthal direction of incidence of radiation and ascertain the higher frequency ranges receiving antennas. Determination of radiation polarization at frequencies above 1.0-GHz is important, because subsequent devised shielding intended to reflect radiations may be effective for one polarization and may prove useless for another.

When at all possible, make field intensity measurements coincidentally with observed disrupting effects of radiations. In the case of scanning variable output radar radiations, coordination with cognizant personnel at the interference source will determine the time frame and frequency range to survey. Such relationship is shown by correlating modulation characteristics (such as pulse repetition rate, pulse width, scan period of source, aural or visual recognition of voice/data being exercised during disturbing effects and their incidences). Amplitude accuracy better than +.2 decibels is not necessary, because it will not significantly contribute to the survey's evaluation. Similarly, precision in assessing the disturbance frequency is valuable only because it aids in revealing the narrow band interference source.

Power Line Transients

Measure transient voltage variations occurring on primary power lines and record them continuously by means of a memory voltmeter and strip-chart recorder combination. This equipment must be capable of simultaneously recording the time of occurrence and the absolute amplitude of either positive or negative transient values, whichever is greater at any particular time or event. Resolution of peak voltage excursions as small as 10 percent of the nominal power line voltage must be possible and provision for measuring and recording the peak amplitude of single, isolated superimposed pulses of 100-nanosecond pulse width or more must also be made. Sensitivity range must be selectable for compatibility with the nominal power line voltage and peak transient levels. Since monitoring must cover periods of 24- to 170-hours or more, unattended operation of such tests is desirable; the Micro Instrument Model 5201 BR is ideally suited for such usage.

Measure short-term power line disturbances by direct electrical connection of test equipment to the power line which is or will be supplying primary power to the computer system (usually 60-Hz or 50-Hz). Correlation of line-to-ground readings to time of day, computer, or other related equipment malfunctions will provide valuable information leading to potential site solutions.

Survey Report

To facilitate the interpretation of test results, the generation of a brief report is recommended. Generally, only the reporting of the following radiation levels above the threshold of susceptibility values are required.

- Radiated levels should not exceed

  2.0 v/m peak between 150 kHz and 54 MHz

  3.2 v/m peak between 54 MHz and 470 MHz

  5.0 v/m peak between 470 MHz and 1 GHz

  8.0 v/m peak between 1 GHz and 10.9 GHz

  10.0 v/m peak at 13.56 MHz, 27.12 MHz, 40.68 MHz, 433.92 MHz, 915 MHz, 2.4 GHz, and 5.80 GHz

- Conducted levels shall not exceed 10 volts rms on the power lines over 500 kHz to 220 MHz.

DIAGNOSES AND IMPLEMENTATIONS

Theoretically, there are three distinct solutions to EMI control, any one of which can eliminate a given interference.

1. Suppression, preventing the generation of EMI at the source
2. Shielding, isolating the emitter so that no interference is transmitted
3. Desensitization, rendering the recipient device nonsusceptible to such interference

Practically, not one of these solutions can singly be achieved. Therefore, the only feasible and economically sound approach to the control of EMI is the study of all three solutions to determine a sensible compromise permitting acceptable trouble-free computer operation by the customer at the most economical price.

Because of the experience and comparative data-base developed through the years, CDC's Engineering and Architectural Services Division is highly qualified to conduct site surveys and recommend the most economical solution.

The following manual, the Control Data Large and Medium Scale Computer Systems Site Preparation Manual, publication number 60275100, revision D, is made available to NASA/Ames as part of the Final Report for contract number NAS2-9457. This manual may be reproduced by NASA/Ames as is reasonable and necessary, in conjunction with this contract.

[illegible]

D. B. Bonstrom
Vice President
Research and Advanced Design Laboratory