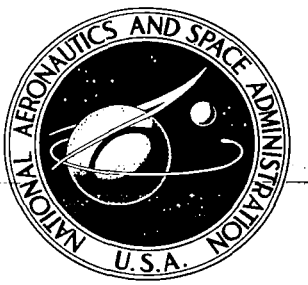
NASA CONTRACTOR REPORT



NASA CR-1820

SYSTEM CONFIGURATION AND EXECUTIVE REQUIREMENTS SPECIFICATIONS FOR REUSABLE SHUTTLE AND SPACE STATION/BASE

by J. R. Kennedy, R. T. Curran, W. S. Fitzpatrick, and J. M. Johnson

Prepared by
COMPUTER SCIENCES CORPORATION
Huntsville, Ala. 35802
for George C. Marshall Space Flight Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • MAY 1971

1. REPORT NO.	2. GOVERNMENT ACCESSION NO.	3. RECIPIENT'S CATALOG NO.
NASA CR-1820		

4. TITLE AND SUBTITLE	5. REPORT DATE
System Configuration and Executive Requirements Specifications for Reusable Shuttle and Space Station/Base	May 1971
	6. PERFORMING ORGANIZATION CODE
7. AUTHOR(S) J. R. Kennedy, R. T. Curran, W. S. Fitzpatrick, and J. M. Johnson	8. PERFORMING ORGANIZATION REPORT # M534
9. PERFORMING ORGANIZATION NAME AND ADDRESS Computer Sciences Corporation Field Services Division, Aerospace Systems Center 8300 South Whitesburg Drive Huntsville, Alabama 35802	10. WORK UNIT NO. 11. CONTRACT OR GRANT NO. NAS8-24930
12. SPONSORING AGENCY NAME AND ADDRESS National Aeronautics and Space Administration Washington, D. C. 20546	13. TYPE OF REPORT & PERIOD COVERED Contractor Report 14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

Work performed for George C. Marshall Space Flight Center
Computation Laboratory

16. ABSTRACT

Presented in this report are computer executive system requirements for the Reusable Shuttle and Space Station/Base. Conceptual hardware configuration aspects that directly influence the respective executive systems are delineated. Performance requirements are defined which constrain or impel system software to a specific configuration. Functional requirements are derived for each respective vehicle's executive system. Executive design requirements comprising preliminary guidelines, objectives, and standards for system software functional design are established. This report defines the software modules necessary for effective operation of intricate Space Shuttle and Station/Base computational complexes.

17. KEY WORDS		18. DISTRIBUTION STATEMENT
Space Station	Data Management Systems	Unclassified - Unlimited
Executive Routine	Information Management Systems	
Reusable Shuttle		
Avionics		
Real Time Monitor		
Onboard Computer		

19. SECURITY CLASSIF. (of this report)	20. SECURITY CLASSIF. (of this page)	21. NO. OF PAGES	22. PRICE*
Unclassified	Unclassified	175	3.00

*For sale by the National Technical Information Service, Springfield, Virginia 22151

FOREWORD

The work reported herein was administered in the Systems Research Branch, Computer Systems Division, Computation Laboratory, MSFC, with Bobby C. Hodges assigned as Contracting Officer's Representative. In addition to his routine duties as Technical Monitor, Mr. Hodges has added significantly to our insight into and understanding of related NASA programs through careful planning, coordination with in-house effort, and encouragement.

SYSTEM CONFIGURATION
AND
EXECUTIVE REQUIREMENTS SPECIFICATIONS
FOR
REUSABLE SHUTTLE
AND
SPACE STATION/BASE

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

DEFINITION OF SYMBOLS

Symbol	Definition
ACK	Acknowledge
AIOP	Analog Input/Output Package
ALU	Arithmetic Logic Unit
AM	Amplitude Modulation
A/D	Analog to Digital
ASM	Assembly
BIT	Built In Test
CC	Central Computer
CCC	Central Computer Complex
CDD	Central Data Display
COSY	Checkout Operating System
CPU	Central Processing Unit (including ALU)
CPSU	Central Processor Subunit
CRT	Cathode Ray Tube
DAU	Data Acquisition Unit
DBI & CU	Data Bus Interface and Control Unit
DCC	Crew Display and Control Computer
DIOP	Digital Input/Output Package
DIU	Data Interface Unit
DMS	Data Management System
DRSS	Data Relay Satellite System

Symbol	Definition
ECLS	Environmental Control and Life Support
ECR	Executive Control Routines
ELD	Error Logging Device
EOM	End of Message
ERD	Error Recording Device
EVA	Extra Vehicular Activity
FCS	Flight Control Subsystem
FDM	Frequency Division Multiplexing
FF	Free Flight
FO/FO/FS	Fail Operational, Fail Operational, Fail Safe
FWA	First Word Address
G&N	Guidance and Navigation
GNC	Guidance, Navigation, and Control
HE	Hardware Executive
IAS	Integrated Avionics System
IF	Intermediate Frequency
IMU	Inertial Measurement Unit
IOCU	Input/Output Control Unit
IOP	Input/Output Package
I/O	Input/Output
JEP	Jet Engine Processor
K	One Thousand (or 1,024)
KBS	Kilo Bits/Second
KB	Keyboard

Symbol	Definition
KB/sec.	Kilo Bits/second
LRU	Line Replaceable Unit
LU	Logical Unit
LSI	Large Scale Integration
MBPS	Million Bits/Second
MDC	McDonnell Douglas Corporation
MODEM	Modulator/Demodulator
MSA	Main Store Allocator
MUX	Multiplexer
NSC	Navigation and Sensor Computer
OCS	Onboard Checkout System
OS	Operating System
PFCD	Primary Flight Control Display
POL	Problem Oriented Language
RDAU	Remote Data Acquisition Unit
REP	Rocket Engine Processor
ROM	Read Only Memory
R/W	Read/Write
SDI	Standard Data Interface
SHF	Shared High Frequency
SMD	Systems Monitor Display
SOM	Start of Message

Symbol	Definition
TBD	To Be Defined
TDM	Time Division Multiplexing
TLC	Time Line Controller
TTY	Teletype
UHF	Ultra High Frequency
XMIT	Transmitter

SYSTEM CONFIGURATION AND EXECUTIVE REQUIREMENTS SPECIFICATIONS FOR REUSABLE SHUTTLE AND SPACE STATION/BASE

SUMMARY

The two major concerns of this document have been descriptions of the computational systems hardware and the requirements analysis of their impact on executive software development. The report outlines what are believed to be the current conceptual trends for both reusable shuttle and space station computational hardware. These discussions are in light of their effect on software executive requirements, of course. Specification of performance and functional executive software requirements for computer complexes on the reusable shuttle and on the space station are derived separately. A single set of design requirements is developed to apply to both systems. Design procedures and evaluation criteria have equal applicability to both systems.

The reusable shuttle executive software system operates in the environment of a distributed multi-computer system with dedicated computer subsystems. Logical procedure modules are derived to satisfy the requirements arising from interaction of mission requirements, avionics hardware and applications.

The space station executive software operates in an environment with many functional similarities and some important differences in hardware and applications. Hardware differences lie in the fact that the space station computer system does not exhibit dedicated characteristics to the degree that these characteristics are exhibited in the reusable shuttle. The applications or environment difference is caused by the existence of requirements to support experiments with high data rates, archival of results, and other more complex data management requirements.

Differences in mission, hardware, and level of available information cause separate treatments of the two executive software systems. Experience in computer applications with similar attributes has been drawn on in the definition of the executive software functional requirements for each.

SECTION I. INTRODUCTION

A. General

The development of design specifications for flight operational executive systems to support major NASA spaceborne electronics systems in the coming decades is an activity currently receiving growing attention. A primary objective of the Marshall Space Flight Center Computation Laboratory's Systems Research Branch has been a continuing effort to broaden the scope of awareness in the areas of future on-board executive system concept formulation and design philosophy. The purposes of this objective are manifold, in that many related benefits accrue from a coherent and persistent effort to refine the overall understanding of spaceborne executive systems. Some obvious benefits are:

- NASA can guide efforts in this area with clearer goals in mind,
- Proposed concepts can be more intelligently evaluated,
- The general state-of-the-art of the computing sciences may be enhanced, and
- Greater cost effectiveness can result.

to name but a few. This report, therefore, supports major NASA objectives through delineation and outline of spaceborne computer systems hardware features and executive system requirements for both the reusable shuttle and earth orbiting space station.

B. Scope

The development effort required for each of the vehicle software systems consists of Flight System Analysis and Software Development.

1. Flight System Analysis. This activity includes:
   - Functional Specification
   - Operational Specification
   - Performance Specification

- Concept Formulation
- Algorithm Identification
- Simulation
- Equation Formulation
- Algorithm Evaluation
- System Functional Alternatives
- Trade Analyses

2. Software Development. This activity includes:

- Requirements Analysis
- System Design
- Program Functional Design
- Program Component Detail Design
- Program Component Coding
- Program Component Checkout
- Software Integration and Test
- Flight Readiness Verification

Flight System Analysis consists of mission problem analysis oriented toward determining and defining the solutions to mission problems such as vehicle guidance and control, experiments control, etc. Software Development, which can, to a limited degree, run concurrent with Flight Systems Analysis but generally follows it, consists of the total process of defining, designing, fabricating, and testing of the required software/hardware complexes.

The scope of this report is not broad enough to address any aspects of Flight System Analysis. This activity is currently being pursued under other NASA contracts, and preliminary reports are only now becoming available.

The two major concerns of this report have to do with descriptions of the computer systems hardware and the requirements analysis aspect of Software Development as it applies to executive systems for control of the space system computers.

C. Vehicle Electronics System Descriptions

Subheadings in Sections II and III of the report outline what are believed to be the most recent trends in concepts for flight hardware for both the shuttle and station. Every effort was made to critically survey all formal documentation describing requirements and trade studies leading to the designs depicted in this report; in many cases, preliminary and informal documentation was correlated to achieve a better perspective. However, much of the on-going design process is being executed in the fashion of a painter capturing a changing scene on canvas. For this reason, and also because we have attempted to observe the results of a process while it is evolving, we have managed to extract and describe only a "broad-brush" version of the system hardware; very little architectural detail has been defined and is therefore not described in the report. Since the primary concern here is with the computers and their gross relation to other subsystems, detail concerning other subsystems was, for the most part, intentionally supressed.

The report then concentrates on the computer complexes, mass and/or auxiliary storage, and the data bus subsystems. This is felt to be sufficient for support of a future executive functional design effort.

D. Executive Requirements

Requirements are considered to be grouped into classes depending on their nature as follows:

- Design Requirements
- Performance Requirements
- Operational Requirements
- Functional Requirements

Design requirements have to do with the specification of how a design is to be performed. Documentation and implementation standards, level of design detail, design flexibility, degree of modularity, and special features to allow for system testing and performance measurement are but a few typical design requirements.

Performance requirements specify how well a system is to perform. This specification is normally accomplished by the definition of a set of parameters that can be measured or evaluated through system operation. The specification outlines acceptable values for these parameters to aid in design and system performance evaluation. Timing characteristics, accuracy, reliability and capacity are examples of performance requirements.

Operational requirements specify aspects of the system peculiar to its operation under the various constraints and conditions imposed upon it. For example, environmental conditions such as temperature, humidity, pressure and foreign particle population counts constitute operational constraints that influence design. Another important class of operational requirements has to do with man-machine interactions and associated responses.

Finally, functional requirements specify operations to be performed, the inputs and resources to be used, and the associated outputs. Functional requirements are often extracted from not only the nature of the process involved but also from the previously mentioned requirements as influencing factors. Functional requirements are determined not only by the hardware architecture, but also explicitly and implicitly by applications programs. For example, the presence of a CRT display in the system indicates that the executive system must provide program modules to support these devices; complex display application programs may require a display file management program module in the executive.

A particular functional requirement may be satisfied in a design by any one of several logically different design procedures. These procedures may vary in execution time, core requirements, reentrancy requirements, or other attributes. Functional requirements and performance requirements are related by the fact that performance requirements may force selection of a particular technique from among the several possible techniques available. For example, in the calculation of spacecraft attitude, accuracy requirements may dictate that sampling and calculation routines must be entered every 100 ms rather than every 500 ms. If a particular task scheduling or dispatching technique does not permit execution of a program module at intervals of 100 ms or less, then it is eliminated from consideration. Failure mode requirements (FO/FO/FS) may require that multilevel interrupt techniques be employed rather than a single level priority interrupt scheme.

The report deals with performance and functional requirements for executive systems for both the shuttle and station separately. Because of the asynchronous phasing of these two NASA programs, the level of detail is not comparable for the two vehicles. A single set of design requirements was developed in a separate section to apply to both systems because of the dual applicability.

The development of meaningful performance and operational requirements specifications necessitates an in-depth analysis of all flight system application program and hardware requirements. Although this is clearly beyond the scope of the report, the major reason for lack of description in this area is the general lack of availability of documented results of flight systems analysis.

The major area of interest in the report is the specification of functional requirements. Every effort has been made to outline these requirements with clarity and comprehensiveness. In certain discussions, a tutorial approach was taken because the subject warranted it, while in other discussions, conciseness was the guideline because the concepts were assumed to be well understood.

E. Conventions

While every reasonable effort has been made to adhere to a common set of definitions and terminology, time has had a mitigating impact. For this reason, some of the areas of possible difficulty are clarified here in the interest of promoting understanding.

- CPU, ALU, computer and processor are frequently used interchangeably. In most cases, the arithmetic logic and control unit is the hardware device being discussed, but frequently, the combined (synergistic) arithmetic logic and control unit and main (macroprogram and data) memory complex is the functional entity of interest. In all cases, context should clarify the actual meaning.

- Bulk store, mass storage, mass memory, and auxiliary storage are interchanged promiscuously. Again, context should clarify the actual storage of interest.

SECTION II. REUSABLE SHUTTLE

A. General

This section of the report discusses the reusable shuttle by considering a hypothetical vehicle consisting of a composite of the booster and orbiter. The discussion is divided into three parts as follows:

- Integrated Avionics System Description,
- Executive System Functional Requirements, and
- Executive System Performance Requirements.

The first of these parts represents a comprehensive organizational delineation of the major hardware elements related to computing within the avionics system complex. Particular attention is given to reviewing the major avionics functions and discussing hardware specifics insofar as they are defined at this evolutionary stage. Major emphasis is placed on those elements relating to the computer complexes and their associated closely-related devices and peripherals. Particular attention is also directed to the bus structure and its related control and data transfer concepts. Due to the scope of the study, no details regarding non-computing subsystems are given.

The second part, based heavily on the hardware, concerns itself with the determination and description of functional requirements for a set of executive routines, or real-time monitors, for the various computer complexes within the shuttle avionics system.

The final part of the section deals with performance requirements for the executives. Ideally, the following methodology would be invoked to determine performance requirements. First, all application (problem- or mission-oriented) tasks would be modeled by analysis and simulator design to determine cumulative distribution functions for each of the following task attributes:

- Time interval allowed for task execution,
- Memory space,
- Number of normalized operations per execution,

- Number of input and output requests per execution, and
- Buffer size for input and output requests; next,

a typical mission profile would be developed to show when, on a relative time scale, each task would be activated for execution; this profile would then serve as the basis for a system load model that is used to envelope the following system load attributes:

- Memory space as a function of time,
- Number of operations per unit of time,
- Number of input and output requests per unit of time, and
- Data transfer volume per unit of time.

Certainly other task and system load attributes could be the object of determination, but these are sufficient to show the method.

The required number of operations per unit time is a function of the timeline task execution frequency, the time intervals allowed for task execution, and the number of operations per task execution. Task execution rate is the number of required operations divided by the allowed time interval. This rate must be maintained throughout the allowed time interval. Given the timeline scale showing when each task is activated, we can superimpose overlapping intervals additively to obtain required execution rate as a function of time. If we assume that this rate is absorbed mostly by accessing main memory (the validity of this assumption depends on how task operations are "normalized"), then we have developed a measure of required main memory access rate as a function of time. The volume of input/output distributed over the allowed execution time gives the other component of main memory access rate which, when combined with operation rate, gives a composite measure of required main memory speed.

This speed represents an "effective" speed which can be obtained in many ways such as by using one very large and fast processor/memory system, or by using several slower devices in parallel. Associated with each choice, of course, is a set of required executive routines for support. The cost and complexity of these routines must be merged into the overall trade analysis to determine an optimal configuration satisfying the mission timeline task requirements.

The trade analysis will result in specification of performance requirements for executive routines. These requirements might include:

- Maximum time to activate a task,
- Maximum time to initiate an I/O request,
- Maximum main memory used by the executive, etc.

The point of this discussion is now clear. A significant deficiency exists with regard to performance requirements for the shuttle executives. The reason for this is that comprehensive performance requirements are derived, as shown above, almost exclusively from application program performance requirement specifications, and these are not specified in sufficient detail at this time. Functional requirements, on the other hand, are based primarily on the hardware architecture and application program functions. Although considerably more definitive detail is required in these areas, a priori knowledge was applicable due to the nature of the mission functions and the applicability of past experience with related and/or similar systems.

B. Integrated Avionics System Functions

1. General. This section outlines the Integrated Avionics System for the reusable space shuttle. Specific mission oriented functions required to be performed by the complete system are described. Characteristics of the avionics hardware conceptual configuration described in various documents and presentation notes that directly impact upon the executive system requirements are delineated. Major emphasis is directed to the Arithmetic Logic Units (ALU's), storage, and data bus subsystems since they have the greatest influence on the executive design. Other aspects of the system are not considered in detail for this report.

2. Avionic System Functions. The space shuttle will utilize an on-board computerized integrated avionics system to provide information processing and system control required for autonomous vehicle operation. It will relieve the crew of excessive workload by automatically performing time critical functions and providing priority sorting and data compression of that information needed by the crew.

The general avionic functions to be performed are:

a. Engine control - jet and rocket.

b. Communication to ground and space station.

c. Attitude reference.

d. Flight control - aerodynamic and reaction jets - manual and automatic.

e. Navigation in orbit and atmosphere.

f. Guidance during boost, orbital burns, entry and landing.

g. Displays for pilot and copilot.

h. Power control and conditioning.

i. Control, checkout and status monitoring of vehicle subsystems.

- Environmental Control and Life Support (ECLS)
- Hydraulics

- Landing Gear
- Lights
- Hatches and Doors
- Electrical Power
- Propellant
- Separation
- Payload

j. Flight recording: maintenance and emergency.

k. Information management.

l. Software for on-board checkout, ground checkout, mission planning.

m. TV cameras, alignment, closed circuit.

3. Avionics System Hardware Description. The space shuttle Integrated Avionics System baseline is shown in Figure II-B-1(1). References, shown in parentheses, are given at the end of each section. The Integrated Avionics System (IAS) is comprised of a central computer for centralized data management and several dedicated processors for special hi-iteration calculations. All other subsystems requiring IAS attention in the form of data communication, or command and control are interfaced through a network of multiplexed data buses. Data bus-to-device interface is accomplished through a standardized digital interface unit (DIU). This unit is to be designed with general purpose flexibility sufficient to allow direct computer communication with any external subsystem while requiring only minimal special interface circuitry design.

The major functional elements of the complete Integrated Avionics System and corresponding subsystems are shown in detail in Figure II-B-2(2). Quantity numbers are assigned to processing units and digital interface units (DIU's) reflecting the redundancy required for meeting failure mode criteria (i.e., fail operational, fail operational, fail safe).

a. Computers. The IAS system computers provide computational capability required by all other subsystems during all phases of the space shuttle mission. The majority of these computations are presently planned to

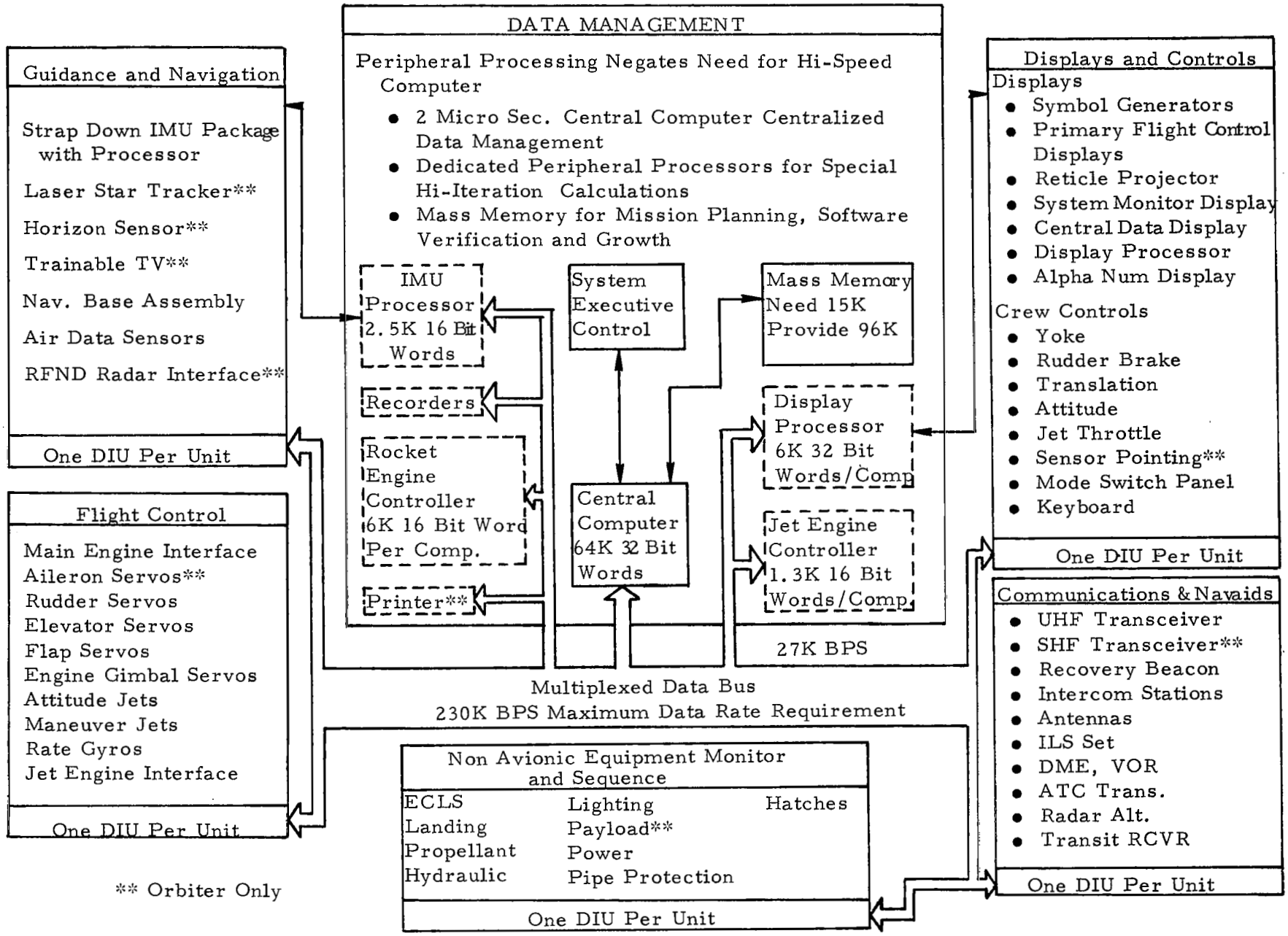


FIGURE II-B-1
INTEGRATED AVIONICS SYSTEM BASELINE

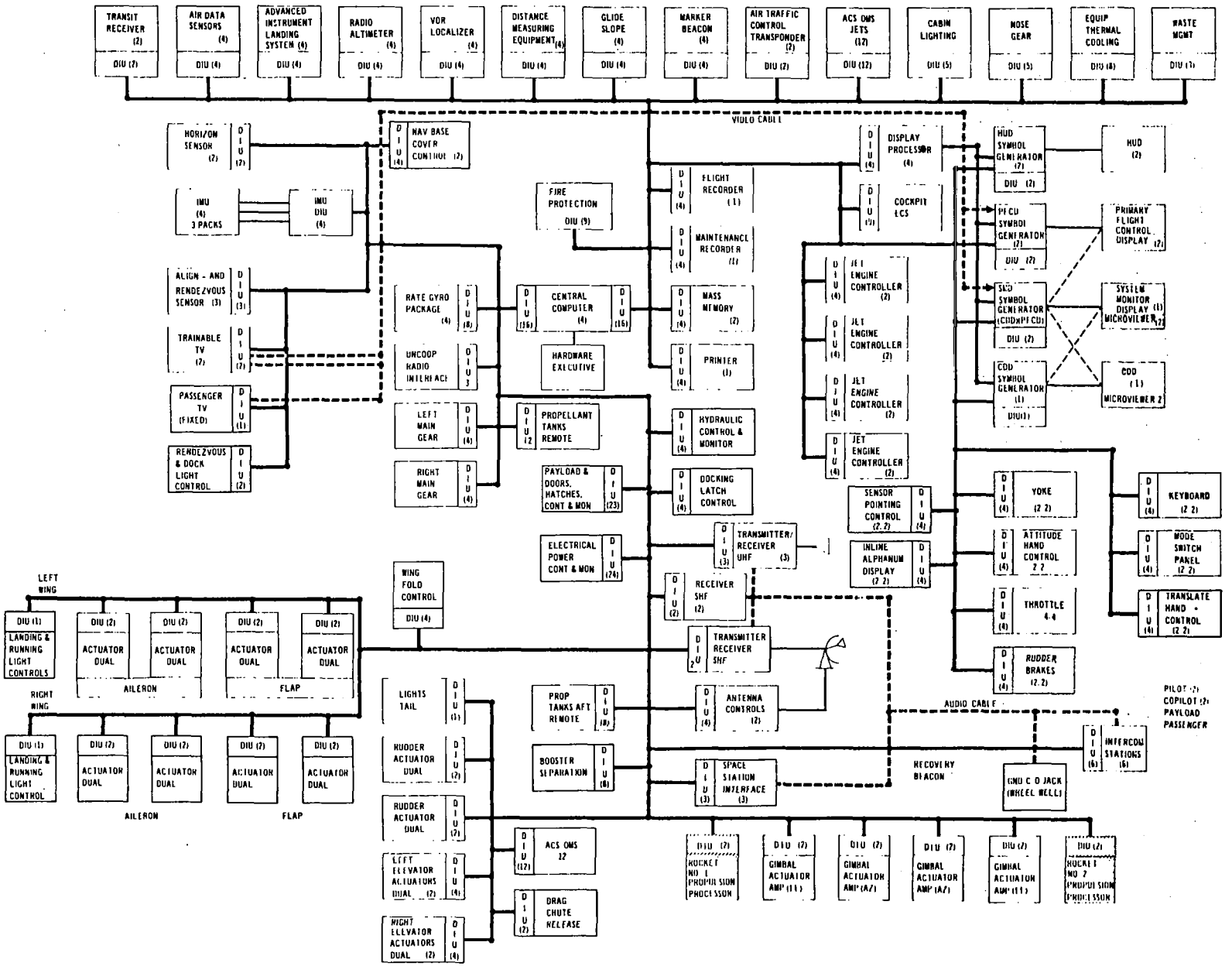

TRANSIT RECEIVER (2)
AIR DATA SENSORS (6)
ADVANCED INSTRUMENT LANDING SYSTEM (6)
RADIO ALTIMETER (6)
VOR LOCALIZER (4)
DISTANCE MEASURING EQUIPMENT (4)
GLIDE SLOPE (4)
MARKER BEACON (4)
AIR TRAFFIC CONTROL TRANSPONDER (2)
ACS OMS JETS (12)
CABIN LIGHTING
NOSE GEAR
EQUIP THERMAL COOLING
WASTE MGMT
VIDEO CABLE
HORIZON SENSOR (2)
IMU (4) 3 PACKS
IMU DIU (4)
NAV BASE COVER CONTROL (2)
FIRE PROTECTION DIU (9)
FLIGHT RECORDER (1)
MAINTENANCE RECORDER (1)
DISPLAY PROCESSOR (4)
COCKPIT ECS
HUD SYMBOL GENERATOR (2)
HUD (2)
PFCU SYMBOL GENERATOR (2)
PRIMARY FLIGHT CONTROL DISPLAY (2)
SKD SYMBOL GENERATOR (CRDxPFCD)
SYSTEM MONITOR DISPLAY (1) MICROVIEWER 1
COD SYMBOL GENERATOR (1)
COD (1) MICROVIEWER 2
ALIGN - AND RENDEZVOUS SENSOR (3)
TRAINABLE TV (2)
PASSENGER TV (FIXED)
RENDEZVOUS & DOCK LIGHT CONTROL
RATE GYRO PACKAGE (4)
UNCOOP RADIO INTERFACE
CENTRAL COMPUTER (4)
HARDWARE EXECUTIVE
MASS MEMORY (2)
PRINTER (1)
JET ENGINE CONTROLLER (2)
JLT ENGINE CONTROLLER (2)
JLT ENGINE CONTROLLER (2)
JET ENGINE CONTROLLER (2)
LEFT MAIN GEAR
RIGHT MAIN GEAR
PROPELLANT TANKS REMOTE
PAYLOAD & DOORS, HATCHES, CONT & MON (23)
ELECTRICAL POWER CONT & MON (24)
HYDRAULIC CONTROL & MONITOR
DOCKING LATCH CONTROL
TRANSMITTER/ RECEIVER UHF (3)
RECEIVER SHF (2)
TRANSMITTER RECEIVER SHF
ANTENNA CONTROLS (2)
SENSOR POINTING CONTROL (2,2)
INLINE ALPHANUM DISPLAY (2,2)
YOKE (2,2)
ATTITUDE HAND CONTROL 2,2
THROTTLE 4-4
RUDDER BRAKES (2,2)
KEYBOARD (2,2)
MODE SWITCH PANEL (2,2)
TRANSLATE HAND + CONTROL (2,2)
LEFT WING
WING FOLD CONTROL
LANDING & RUNNING LIGHT CONTROLS
ACTUATOR DUAL
AILERON
FLAP
RIGHT WING
LIGHTS TAIL
RUDDER ACTUATOR DUAL
LEFT ELEVATOR ACTUATORS DUAL (2)
RIGHT ELEVATOR ACTUATORS DUAL (2)
PROP TANKS AFT REMOTE
BOOSTER SEPARATION
SPACE STATION INTERFACE (3)
ACS OMS 12
DRAG CHUTE RELEASE
ROCKET NO 1 PROPULSION PROCESSOR
GIMBAL ACTUATOR AMP (EL)
GIMBAL ACTUATOR AMP (AZ)
ROCKET NO 2 PROPULSION PROCESSOR
AUDIO CABLE
INTERCOM STATIONS (6)
RECOVERY BEACON
GND C O JACK (WHEEL WELL)
PILOT (2)
COPILOT (2)
PAYLOAD
PASSENGER

FIGURE II-B-2

be performed in the central computer complex. However, dedicated special purpose computational devices are utilized to satisfy unique computational requirements. Specific characteristics of each of these computer complexes are as follows:

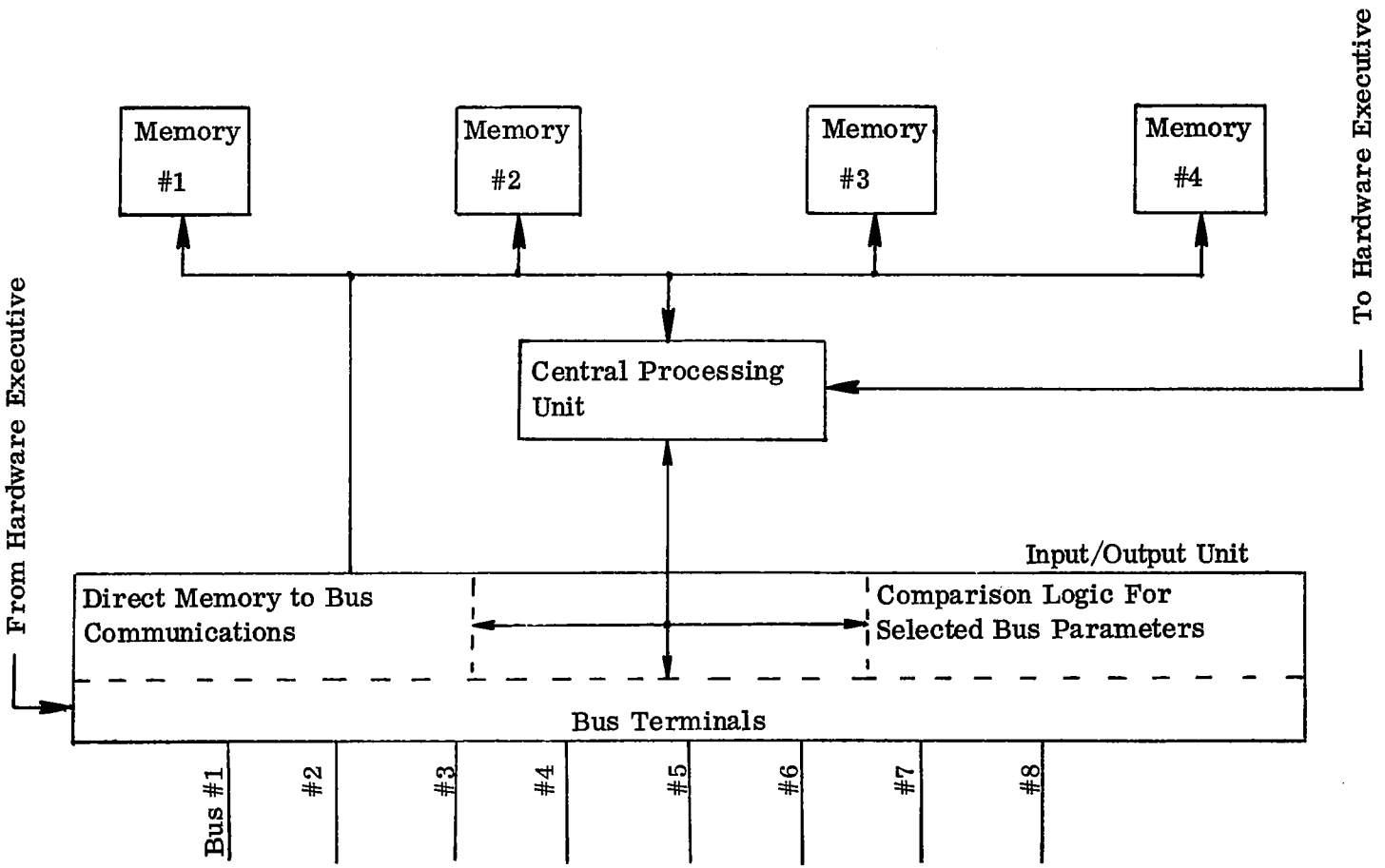
(1) Central computer (CC). A functional diagram of the Central Computer is shown in Figure II-B-3(1). The unit is comprised of an Arithmetic Logic Unit (ALU), 4-16K memory modules and an input/output (I/O) control unit. Data paths exist between each of the memory modules, the ALU and the I/O control unit. Eight buses are connected to the I/O control unit. Logic capability is available for direct memory to bus transfers or cycle stealing memory transfers.

The following Central Computer characteristics are defined:

- Number of computers (max.) 4
- Memory Size (max. words) 256K
- Memory Modules (max.) 16
- Word Length (bits) 32
- Half Word Length (bits) 16
- Add Time (microseconds) 2
- Multiply Time (microseconds) 10

The central processor is projected to have an instruction set consisting of 100 instructions with eight general purpose registers. Direct addressing will be available to the 64K words of main memory. Each memory module will be powered and switched as 16K x 34 bit modules. Two bits will be utilized for parity.

(2) Digital interface unit processor. Each subsystem will be interfaced with the I/O bus through one of 16 slightly different variations of digital interface units. In most cases these units will consist of a line terminal only. However, in a few specific cases a more sophisticated approach is required. A 16-bit parallel processor with approximately 2.5K of memory is envisioned in addition to the line terminal. Basic building blocks for storage are 512 word blocks of MOS R/W and MOS R/O memory. Other characteristics of the Processor include add times in the 2 microsecond range; local memory is partitioned with approximately 25% having read/write capability and 75% with



BASELINE COMPUTER FUNCTIONAL DIAGRAM

FIGURE II-B-3

read only. Twelve to sixteen units of this type are envisioned.

A typical block diagram of this type of DIU is shown in Figure II-B-4(2).

(3) Main engine processor. Main engine or rocket engine control will be effected through the main engine processors. Each of these engines is currently envisioned to require the following computers with the indicated characteristics:

- Number of Computers 4
- Memory sizes, each (words) 6K
- Word Length (bits) 16
- Add Time (microseconds) 2

(4) Jet engine processor. Jet engine control will be effected through the jet engine processors. Each of these engines is currently envisioned to require the following computers with the indicated characteristics:

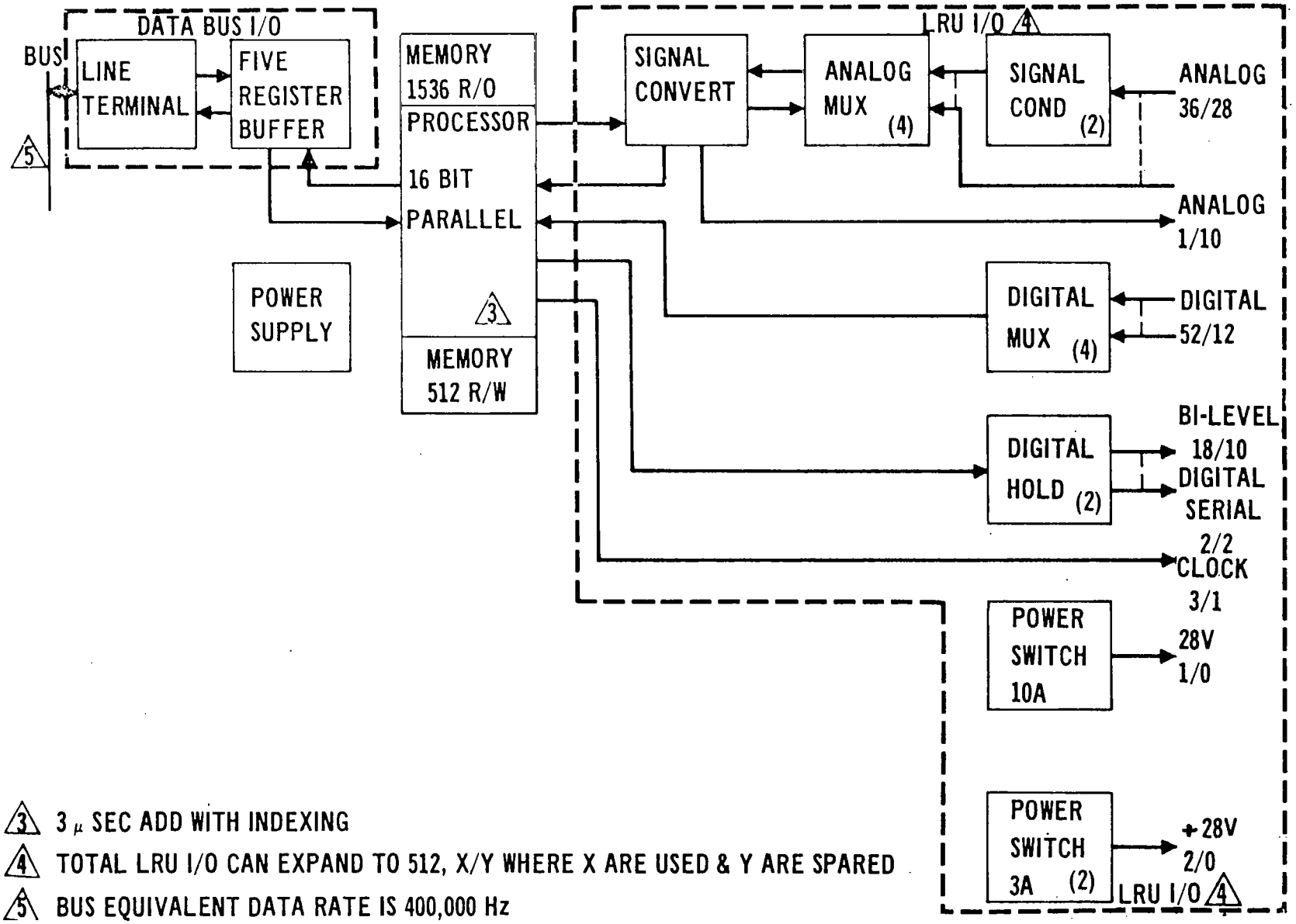
- Number of Computers 4
- Memory Size, each (words) 1.3K
- Word Length (bits) 16
- Add Time (microseconds) 2

(5) Display processor. Display of information is effected through the use of multimode cathode ray tube (CRT) devices. These programmable devices allow the display of only that data pertinent to the current mission phase; all other data is relegated to the status monitor or cautional warning classification.

Each of the five CRT displays are driven by a symbol generator. The display processor is interfaced through a bus network to each symbol generator for display driving. Display driving is also possible from the central computer via the bus network to the symbol generator. Figure II-B-5(1) shows the complete control and display baseline schematic.

Characteristics of the display processor's are as follows:

INERTIAL MEASUREMENT UNIT DIU



3 μ SEC ADD WITH INDEXING

4 TOTAL LRU I/O CAN EXPAND TO 512, X/Y WHERE X ARE USED & Y ARE SPARED

5 BUS EQUIVALENT DATA RATE IS 400,000 Hz

FIGURE II-B-4

CONTROLS & DISPLAYS BASELINE
SYSTEM SCHEMATIC

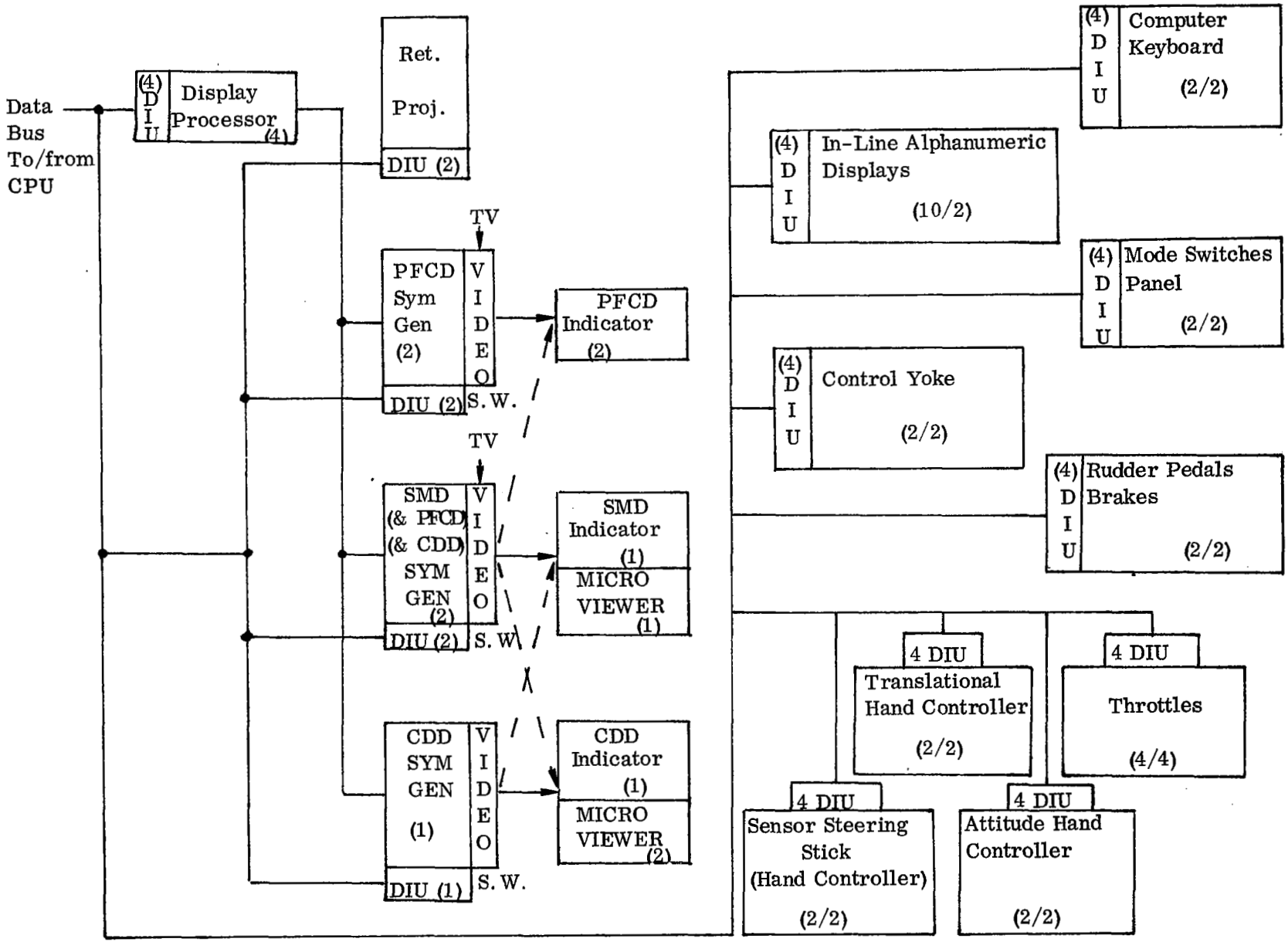


FIGURE II-B-5

- Number of Computers 4
- Memory Size, each (words) 6K
- Word Length (bits) 32
- Add Time (microseconds) 2

Memory is utilized to refresh the CRT's at a rate of 50-60 Hz to prevent flicker. Source data update occurs at a low 1 Hz rate.

b. Bus structure. Space shuttle intercommunication is to be effected utilizing individual hard wires as the transmission medium between black boxes and from subsystem to subsystem. The signal transmission technique chosen is the multiplexed data bus.

The system employs serial digital time division multiplexing (TDM) and is computer controlled using a request/reply data flow control technique. Bi-phase (Manchester) digital coding and alternating current coupling methods are employed. The system timing reference (clock) required for synchronization is transmitted over a separate bus.

Bus-to-subsystem interface is effected through standard Digital Interface Units (DIU's). An example of the interconnection technique and the interconnection redundancy is shown in Figure II-B-6(2). Four buses are run aft for interconnection to all subsystems in this area. An additional four buses are run forward (i.e., toward the cockpit) for interconnection with subsystems located there. Eight bus positions are available on each I/O controller for bus interconnection to the central computer.

Although computer-to-subsystem intercommunication is achieved via a bus network, each of the dedicated processors are afforded an intercomputer channel for additional communication. This capability is provided to enable simultaneous computer-to-computer communications while the bus network is in use for another function. The layout of the communication links is shown in Figure II-B-7(1).

Figure II-B-8(1) outlines the expected data transmission distribution over the orbiter aft bus. Message length in 8-bit bytes is plotted as a function of both number of transmissions per second and number of bytes transmitted per second.

A total traffic estimate for Aft Data Bus No. 1 by MDC/TRW has been placed at approximately 1.08 KB/sec. The possible operational capability range of the complete bus system is 100 KBS-5MBPS.

SIMPLIFIED DATA BUS

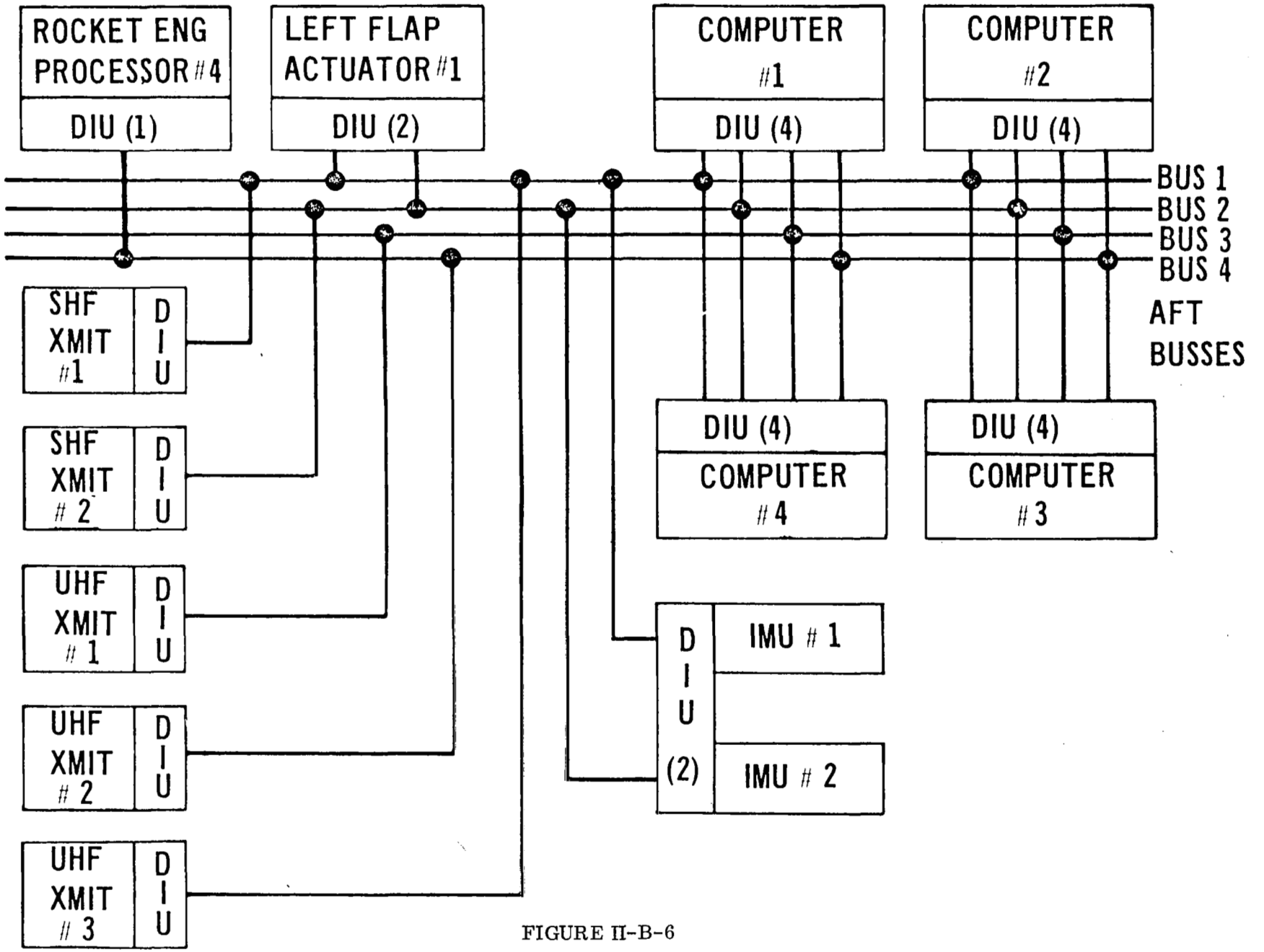
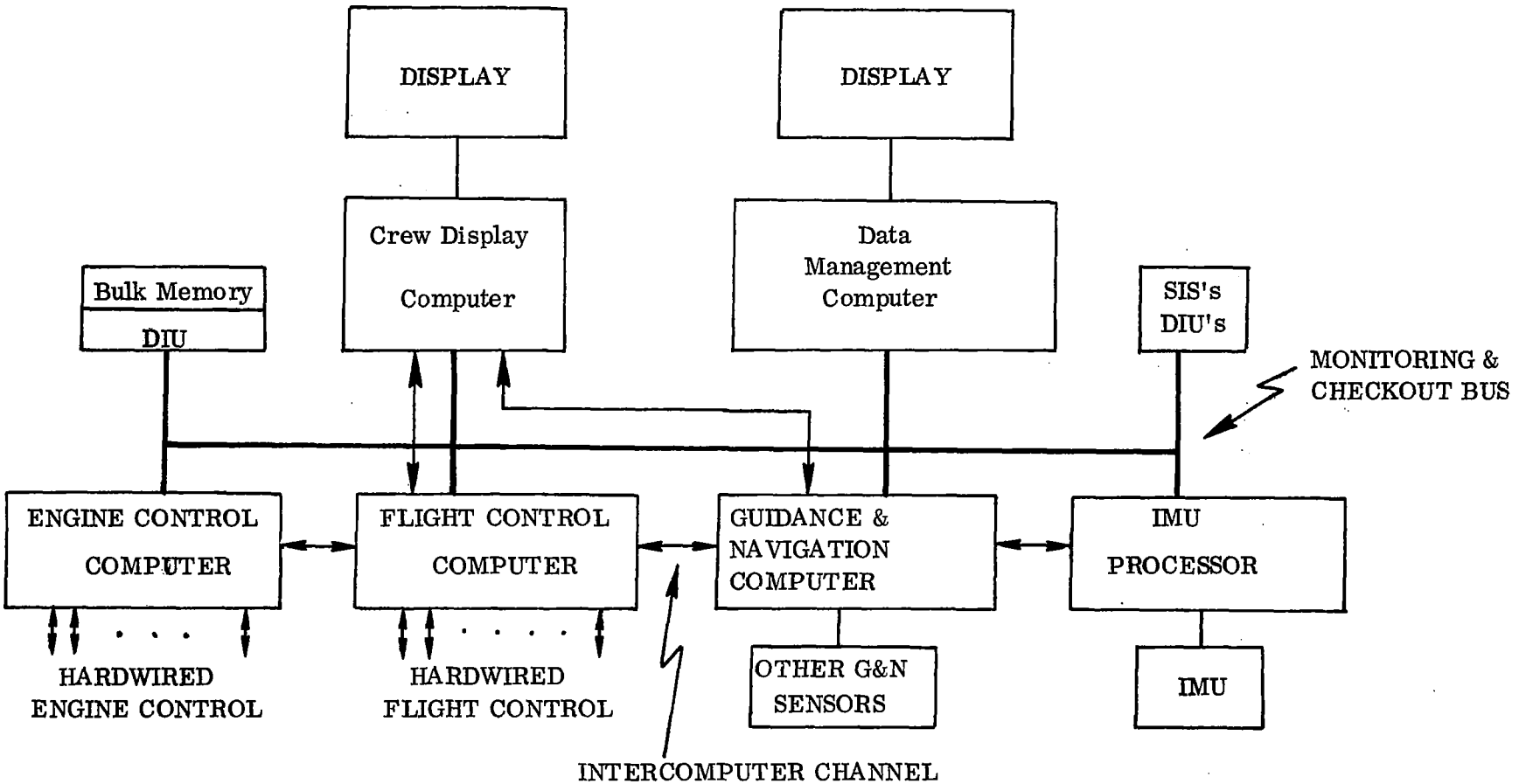


FIGURE II-B-6



DEDICATED SPECIAL COMPUTER CONFIGURATION

FIGURE II-B-7

DATA TRANSMISSION DISTRIBUTION
FOR THE ORBITER AFT BUS

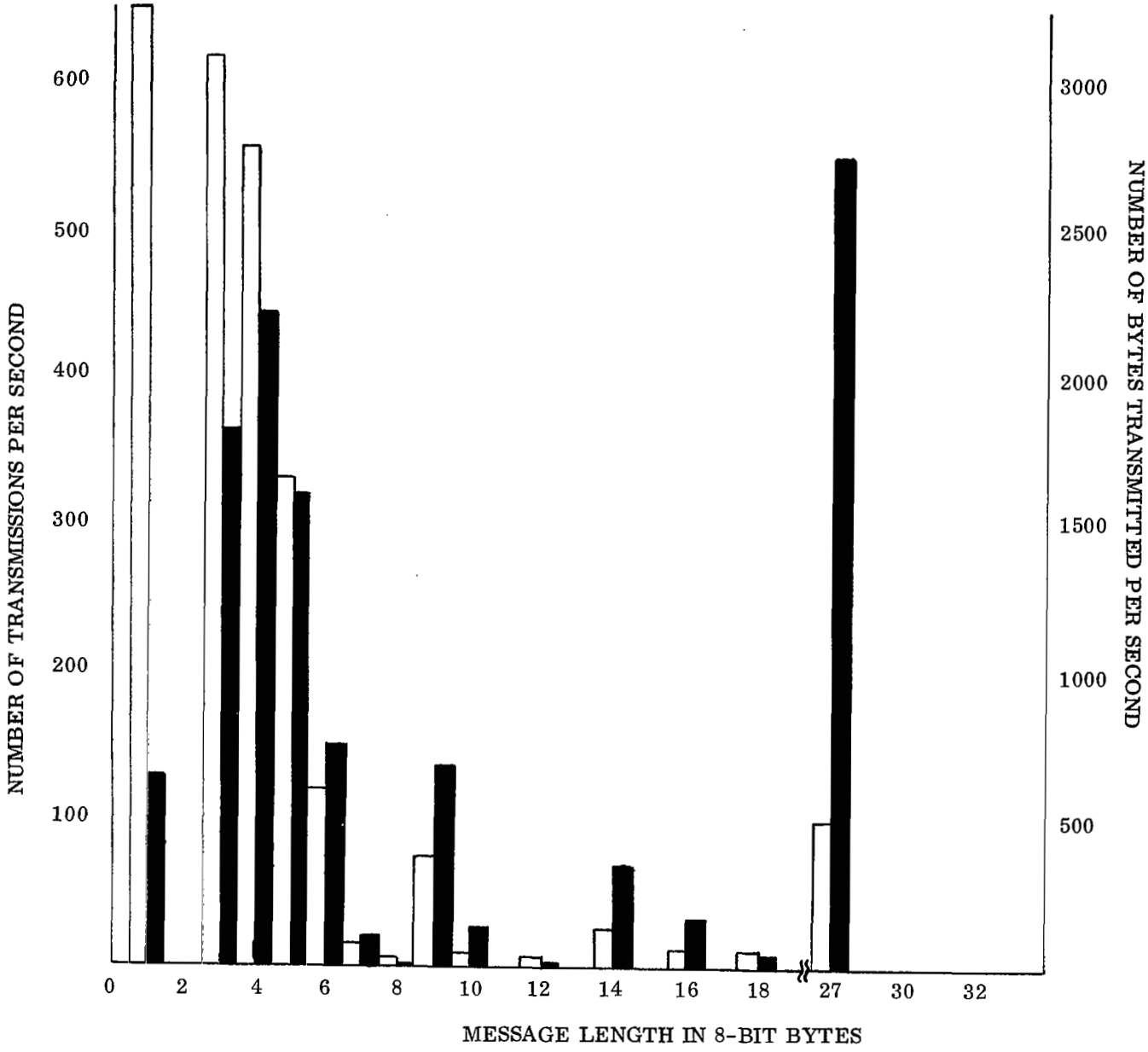


FIGURE II-B-8

The data bus message structure is displayed in Figure II-B-9(1). Although the bus transmission structure is not specifically identified, it is assumed from the vertical parity byte provision that the transmission is parallel by bit up to one byte and then serial by byte.

Bytes are comprised of 8 bits plus one bit for horizontal parity. Data message length varies depending upon the specific DIU in use.

Computer memory requirements for data bus control are as follows:

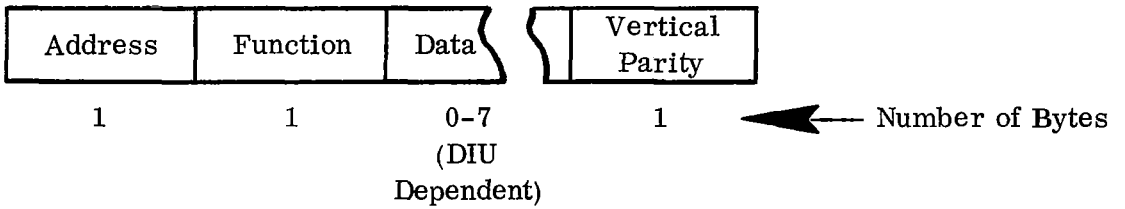
Function	Memory Requirements (Words)
Bus Scheduling	500
Control selection of appropriate I/O program for data bus.	
Modify programs as required to accommodate changing requirements and reconfiguration.	
Initiate IOCU action.	
I/O Drivers	
Command List - List of Data Bus Commands	1000
IOCU Programs - Bus Control Programs	1000
	2500

c. System reconfiguration. Avionics system reconfiguration to meet the failure mode criteria as currently envisioned will be effected through two distinct techniques.

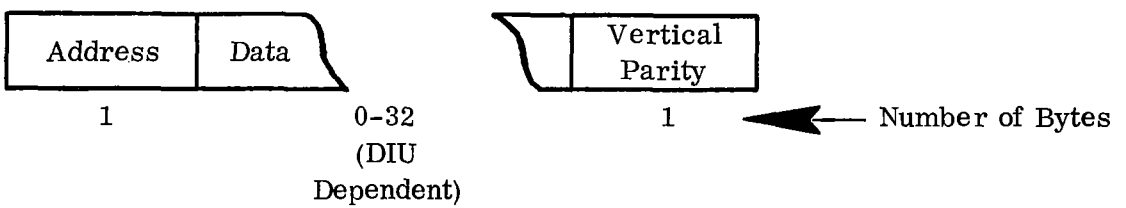
(1) Central computer. The first is for protection of the central computer in event of an LRU failure. LRU's are identified to be the central processing unit, a 16K memory bank, and an I/O control unit. Each of these is quadruply redundant with module selection as a function of the system executive, envisioned to be "hardware" executive in this case. This is depicted schematically in Figure II-B-10(1).

(2) Subsystem reconfiguration. Reconfiguration in the event of failures in the remaining subsystems is effected as a software function

FROM CENTRAL COMPUTER TO DIU'S: Overhead: 3 Bytes



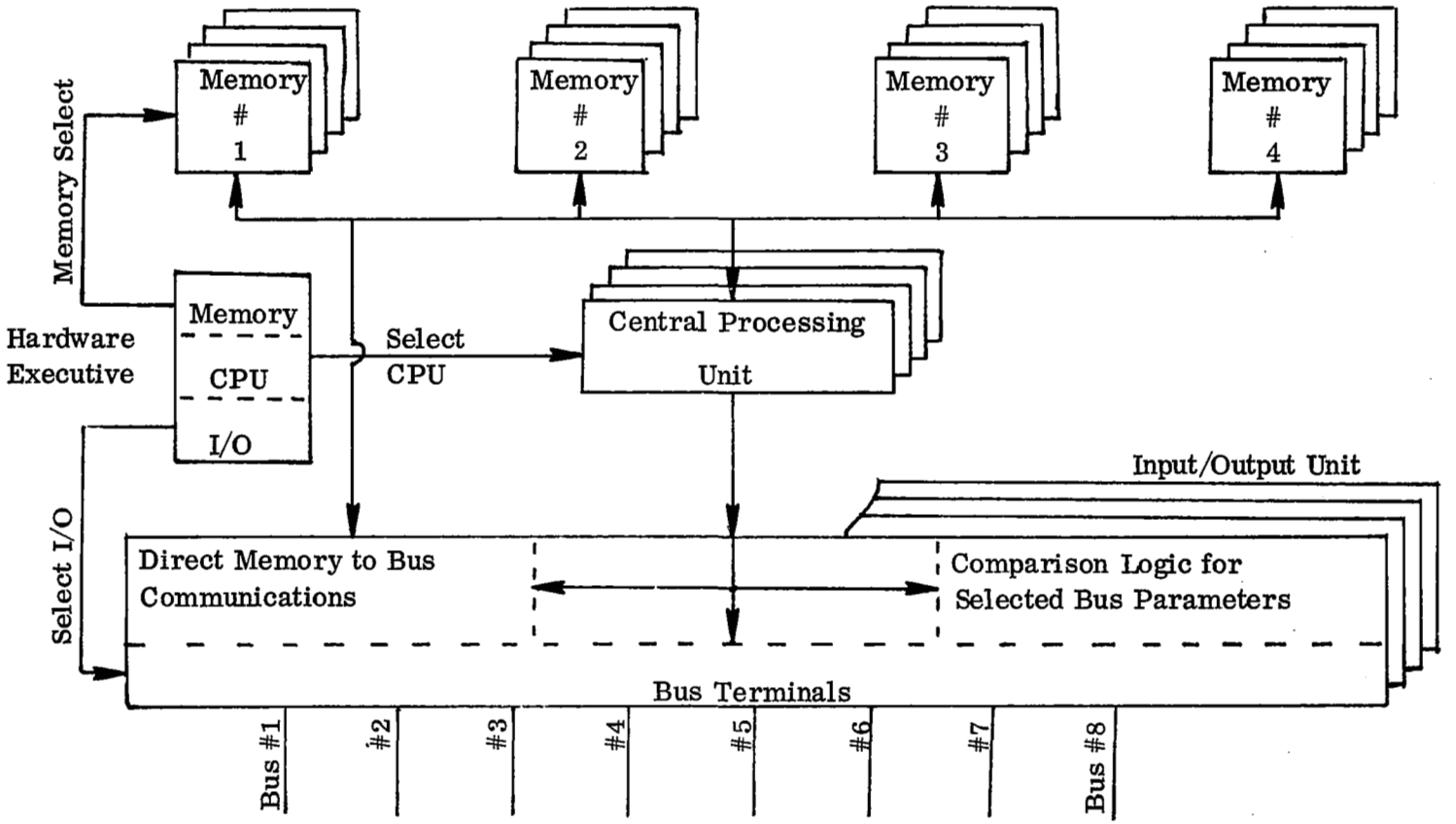
TO CENTRAL COMPUTER FROM DIU'S: Overhead: 2 Bytes



Address: Identification of DIU
Function: Data Assignment or Request
Data: Information to be Communicated
Vertical Parity: Burst Error Detection

DATA BUS MESSAGE FORMATS

FIGURE II-B-9

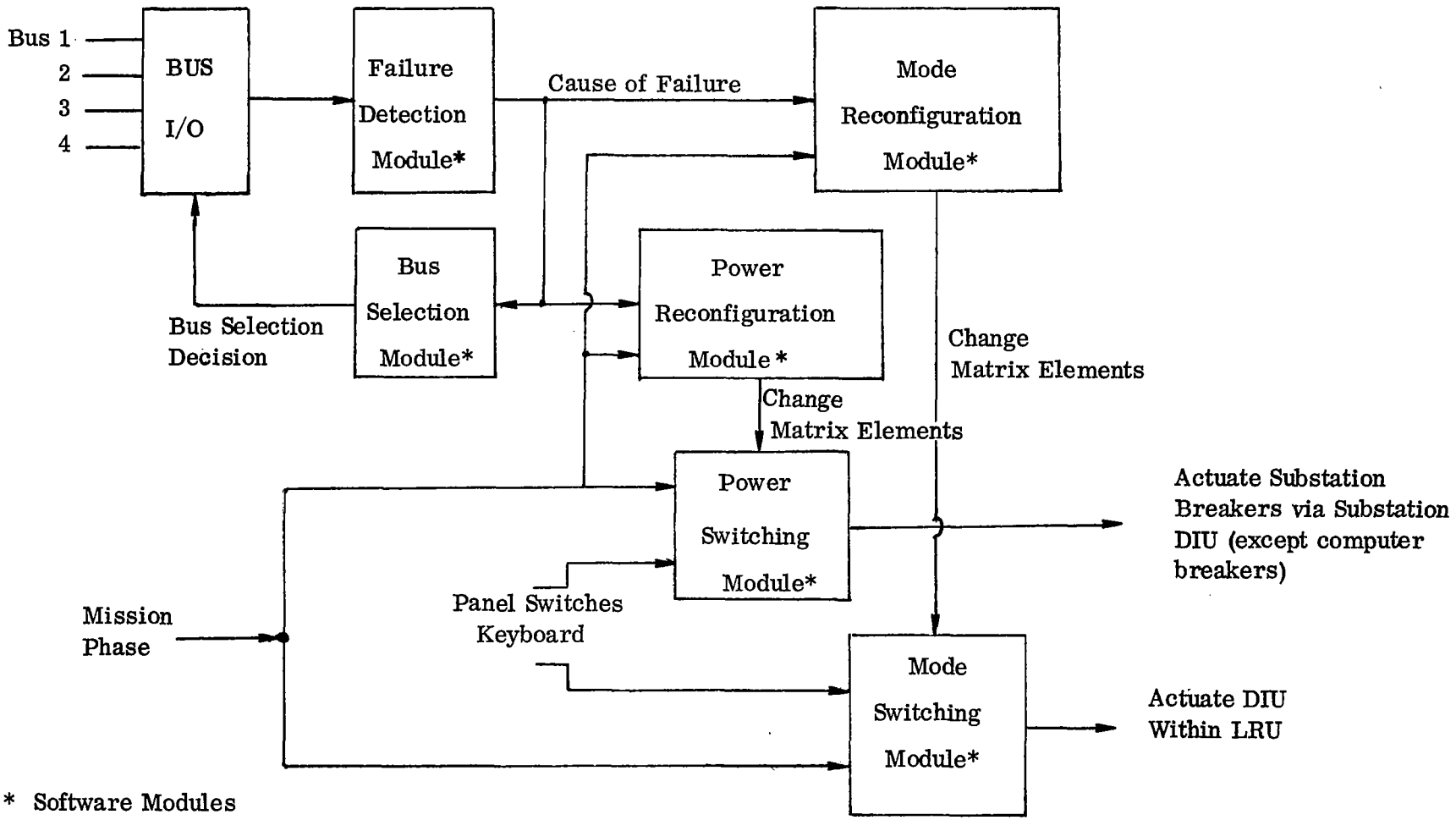


RECONFIGURABLE COMPUTER FUNCTIONAL DIAGRAM

FIGURE II-B-10

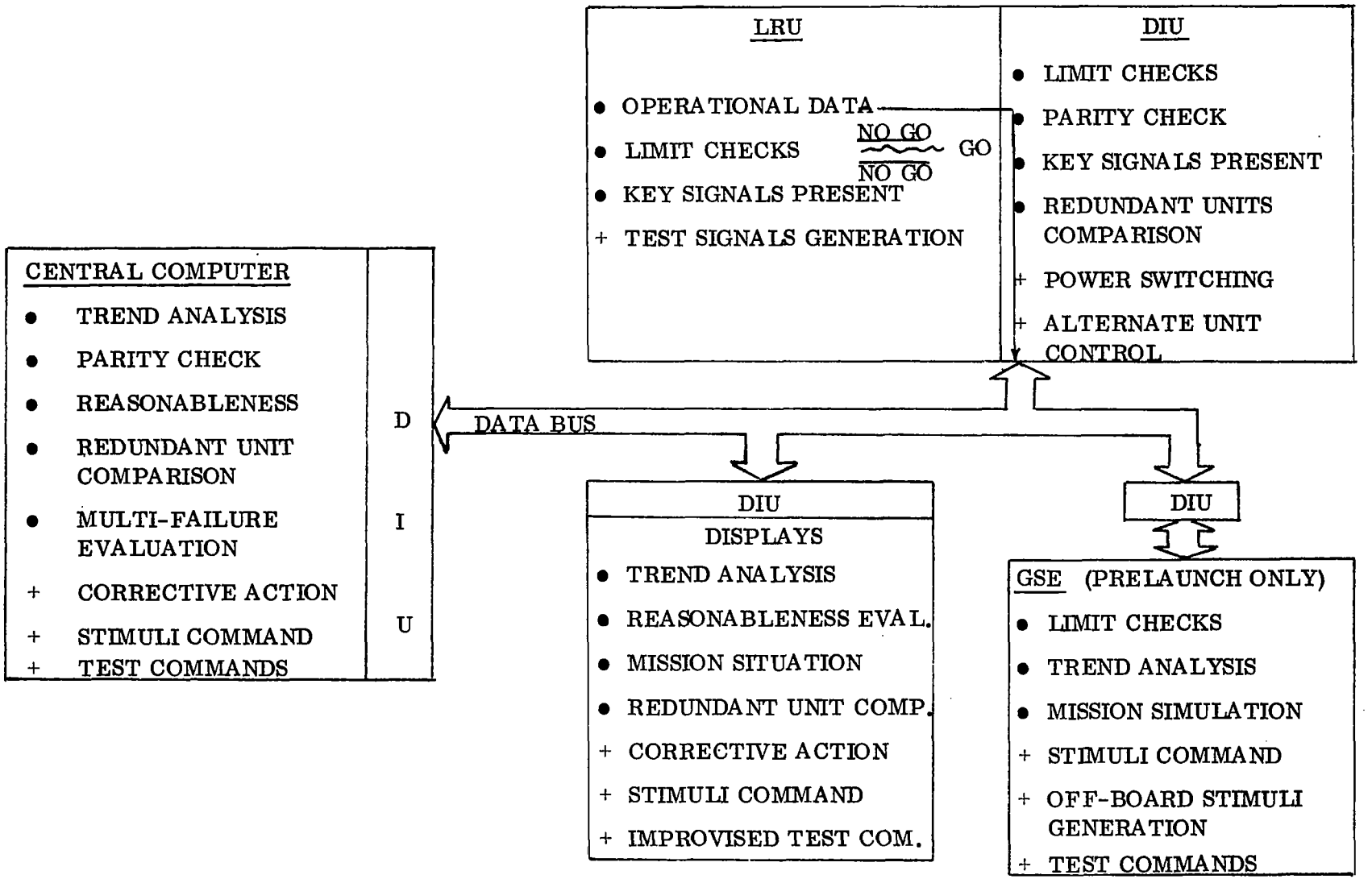
of the central computer. At this point in time it is not certain as to whether this is an application or system program function. Figure II-B-11(1) shows functionally how subsystem reconfiguration will be effected.

Figures II-B-12(1) through II-B-16(1) show checkout functions at various levels and for various subsystems. The significance of the central computer is shown in each case. Pilot/copilot activated reconfiguration capability exists for the central computer. This includes override capability for the executive.



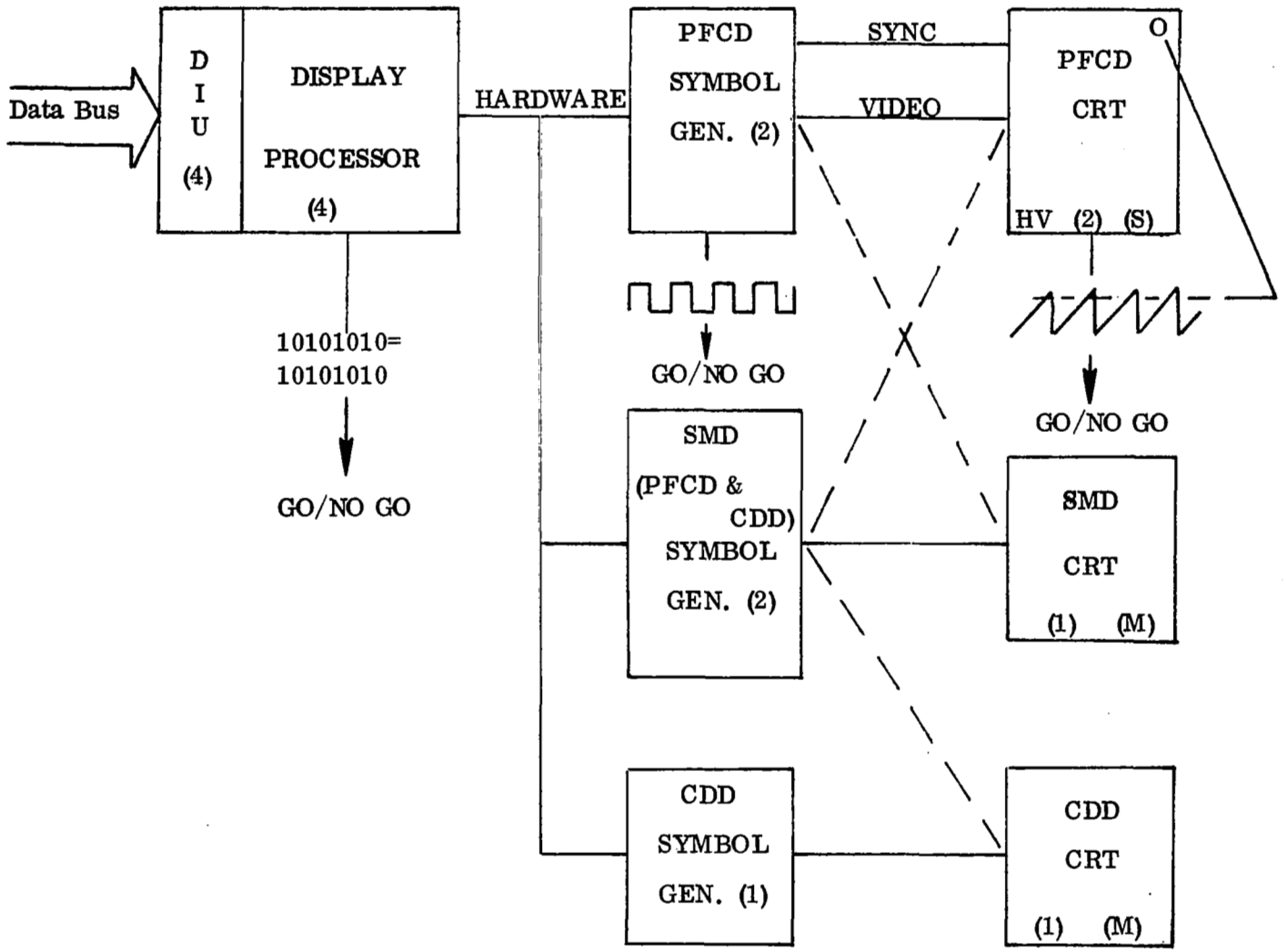
POWER AND MODE RECONFIGURATION OF SUBSYSTEMS

FIGURE II-B-11

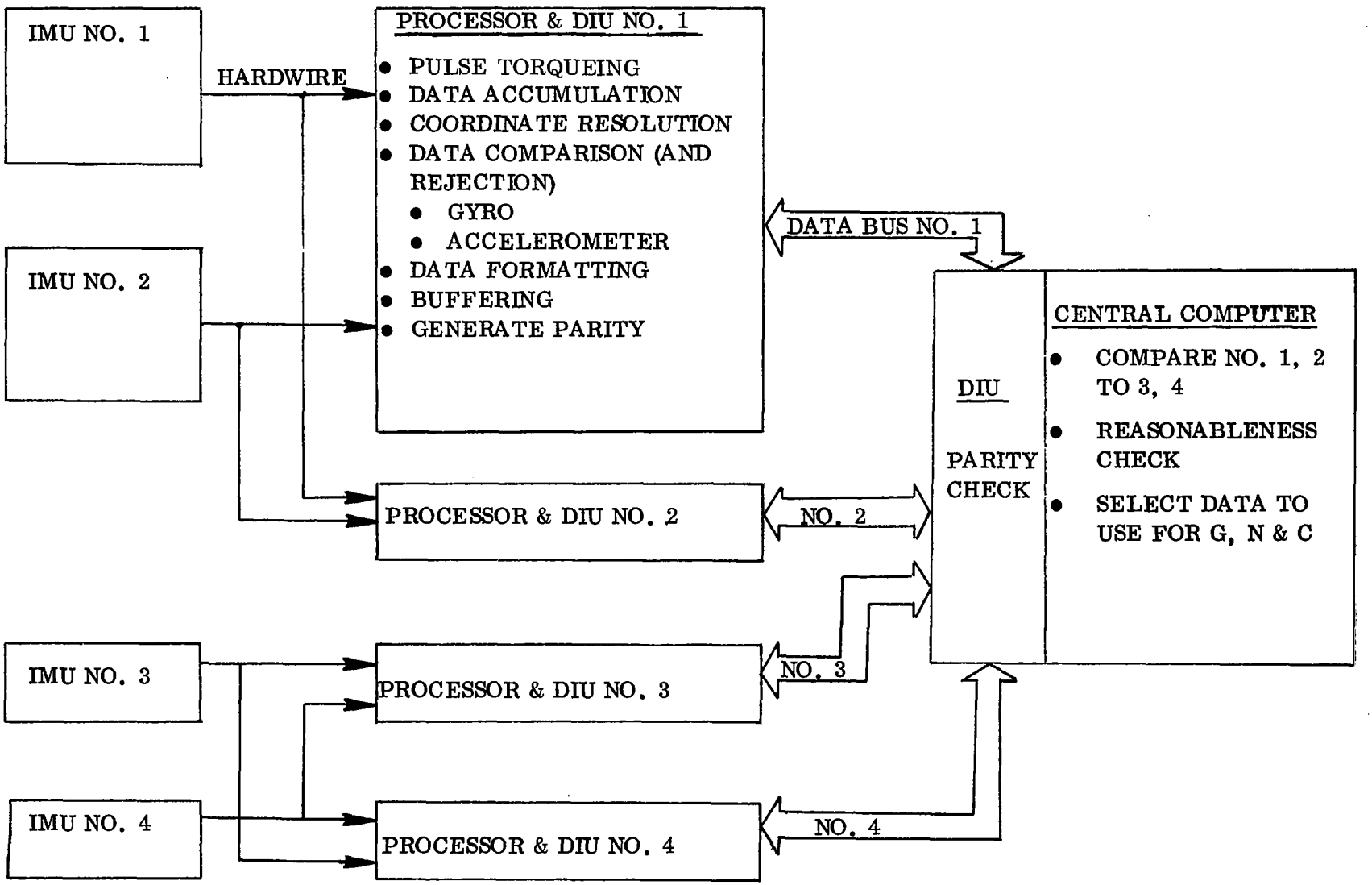


CHECKOUT FUNCTIONS

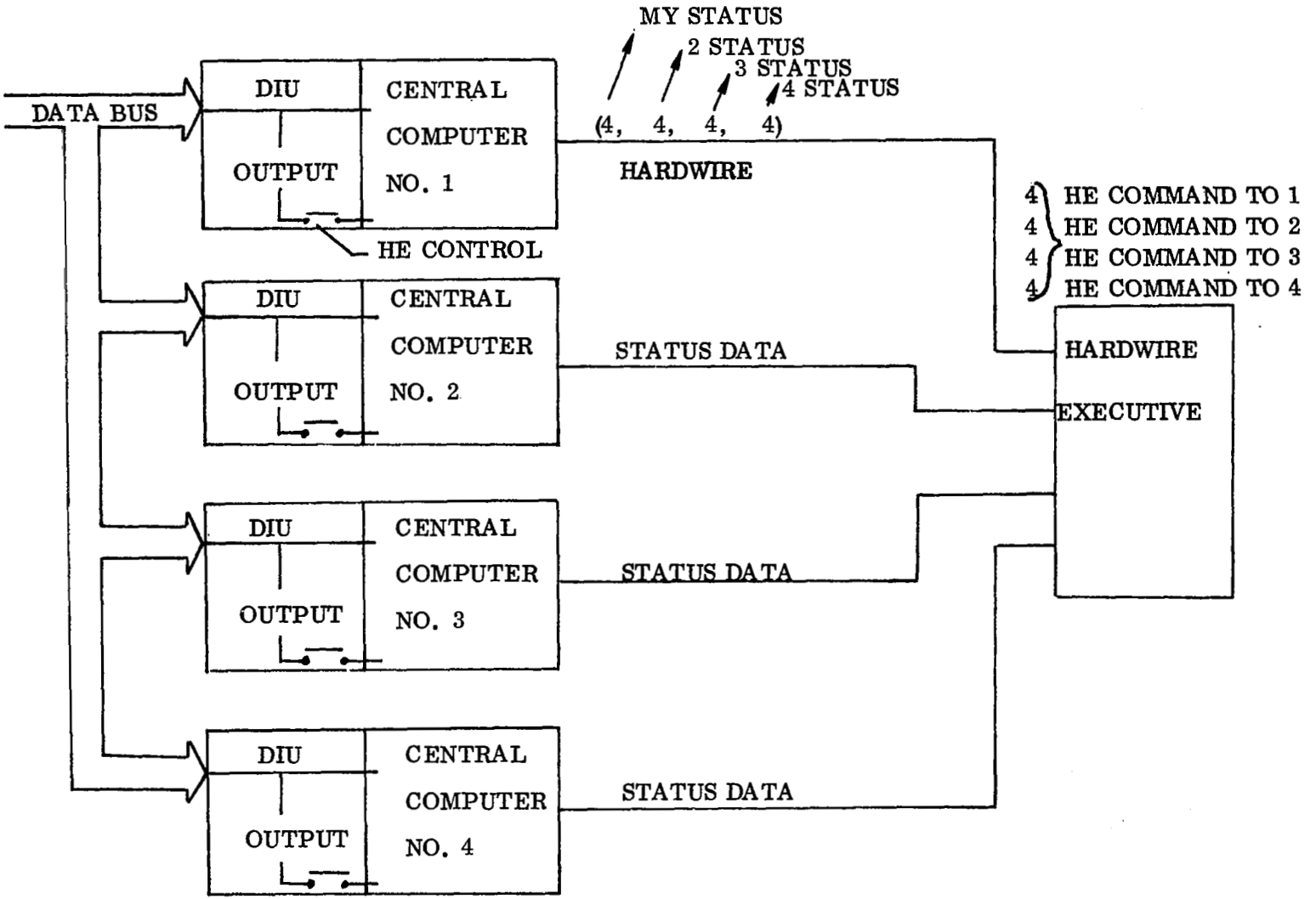
FIGURE II-B-12



DISPLAY CHECKOUT
FIGURE II-B-13

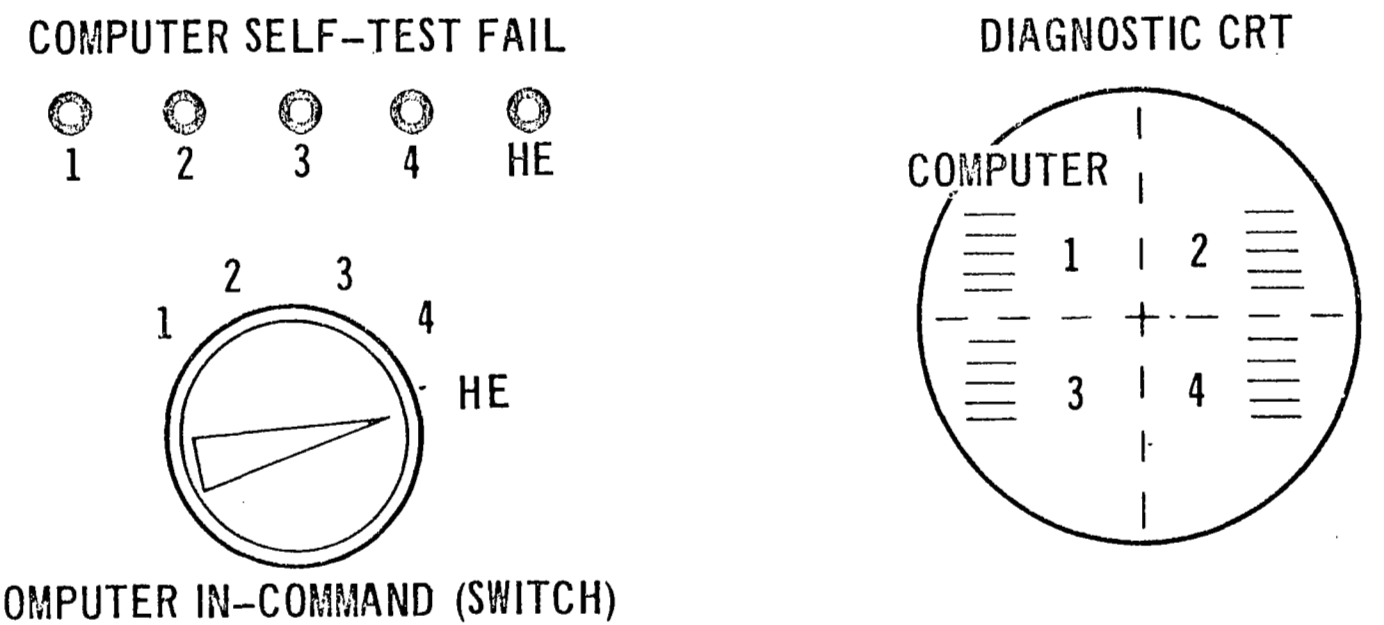


IMU CHECKOUT
FIGURE II-B-14



CENTRAL COMPUTER CHECKOUT

FIGURE II-B-15



1. ALL FOUR COMPUTERS OPERATE AT CRITICAL TIMES. ONE IS IN CONTROL.
2. HARDWARE EXECUTIVE (HE) NORMALLY CONTROLS SWITCHING.
3. CREW CAN DISENGAGE HE AND SELECT ANY COMPUTER MANUALLY.
4. SELF-TEST LIGHTS AND COMPUTER DIAGNOSTICS ASSIST CREW IN COMPUTER EVALUATION.
5. COMPUTERS MONITOR HE ACTION TO INSURE SAFE CONDITION.

COMPUTER SWITCHING

FIGURE II-B-16

C. Executive System Functional Requirements

1. General. Space shuttle Executive System Functional Requirements are derived from several sources:

- Mission requirements analysis,
- Hardware architecture analysis,
- Applications requirements analysis, and
- Combinations of the above.

Mission reliability requirements have significant effect on the monitor system complexity both explicitly and implicitly through hardware redundancy. In fact, this requirement implies multilevel program priority interrupt capabilities. Hardware affects monitor system requirements directly in that support must be provided for each peripheral device present in a computer subsystem. Subsystem complexity will be an indication of the number of possible concurrent activities. The relative importance of these activities with their response time performance requirements will determine the necessity for priority scheduling. This is an example of the interaction between Performance Requirements and Functional Requirements.

Monitor system functional requirements resulting from consideration of applications support requirements are as follows:

- The applications programmer will not have access to the complete instruction repertoire. The system will operate in a privileged mode. Capability must be provided to perform those needed functions which cannot be accomplished by applications due to this restriction;
- To ensure correct scheduling, the monitor must handle all requests for peripheral device usage;
- The monitor system can provide a virtual machine with programming conveniences that reduce applications development effort; and

- The monitor system can provide powerful tools for execution of applications programs by implementing supervisor calls such as TURN ON, DELAY, etc. The applications interface is provided by the Request Processor Module which processes all supervisor calls. Relatively few of the monitor system Functional Requirements are affected by the applications programs, per se. It is the interaction between the application programs primarily that gives rise to the Functional Requirements.

The Functional Requirements are grouped by computer subsystem. Requirements which are common to all subsystems are fulfilled by the Central Computer Complex.

2. Central Computer Complex (CCC) Monitor. The monitor subsystem for the Central Computer Complex must support the following mission functions (3):

- Guidance,
- On-Board Mission Planning,
- Configuration and Sequencing Control,
- Energy Management,
- On-Board Checkout,
- Display Executive Control, and
- Data Base Executive Control.

In order to accomplish this support, the CCC monitor will handle error detection, analysis, alarm, isolation, and recovery for other computer subsystems. For the recovery function, the monitor will maintain tables of reconfigurable subsystems. Upon detection of hard failures, the monitor will accomplish a preplanned reconfiguration to ensure mission integrity.

In addition, the monitor will accept data and task scheduling information from the Uplink Modem and other computer subsystems. It will also transmit data and task schedules to other computer subsystems. The monitor will interface with supervisors of the other computer subsystems in order to provide these task scheduling and data transfer functions.

a. Capabilities. Thorough search of the available space shuttle documentation has indicated that the computer system executive requirements may have many similarities to existing computer applications. Process control applications, for instance, have the following requirements:

- Analog input/output.
- Digital input/output.
- Scientific calculations.
- Computer actuated displays.
- Error logging.
- Initialization.

Message switching applications have the following requirements:

- Message transmit initiation.
- Buffer manipulation.
- Message error detection.
- Error recovery procedures.
- Message-receive initiation/continuation.
- Error logging.
- Initialization.

Real-Time Terminal System applications must satisfy the following requirements:

- Time-slice partitioning.
- Error logging.
- Buffer manipulation.
- Message initiation.

- Error recovery procedures.
- Initialization.

All of the above requirements are present explicitly or implicitly in the space shuttle application. The above systems are characterized by stringent response time requirements, many concurrent activities, varying importance of activities, relatively slow speed peripheral devices, and a high degree of interaction of activities. These same characteristics are evidenced by the space shuttle.

Experience has shown that the following capabilities are necessary in order to satisfy requirements with the above characteristics:

- Priority interrupt structure.
- Priority level independent of priority hardware structure.
- Capability to selectively inhibit interrupts.
- All communication with I/O devices will take place via the supervisor.
- Privileged mode of operation (i.e., privileged instruction repertoire).
- Memory protect facilities for task and data protection, and
- Variable program priority level (dynamically variable in real-time under application, executive, or event occurrence control).

b. Assumptions. The following functional requirements will be assumed for the computer architecture, which will significantly affect monitor requirements.

(1) Portions of the monitor will reside in the main store. Parts of the monitor will possibly reside in bulk and be called into the main store as required. Performance requirements might ordinarily prohibit the delays associated with loading large portions of the monitor as needed. However, systems exist that would likely permit this form of memory hierarchy for executive residence. For example, this concept would be feasible with the IBM

System 360 Models 85 and 195. These computers use a cache execution memory with a large bulk or extended main store. The gross features are:

- Memory reference to an instruction located in cache memory brings in that instruction plus the next seven instructions for execution.

- Memory reference to an instruction not in cache memory brings that instruction, plus the next seven instructions from bulk to cache memory for execution and also loads blocks of instructions into cache memory. Since program instructions tend to cluster, many subsequent memory references will be directed to the relatively fast cache memory. Instructions are fetched in groups, reducing the number of memory references required. Experience has shown that this combination operates at about 80% of the rate that a computer with all cache memory would reach.

(2) The functional relationship between elements of the Executive Control Routines and other elements of the software system is shown in Figure II-C-1.

(3) The monitor system will consist of a kernel with ancillary program modules.

c. Executive control routines (ECR). The ECR group of modules is associated with the functional requirements for resolving priority, resource allocation, and program execution conflicts. The following program modules will comprise the executive control routines portion of the CCC monitor:

(1) Task Scheduler will have responsibility for appending requests for program module execution on the proper priority thread.

(2) CPU Allocator will maintain records of available CPU's and upon request will perform necessary functions to associate an available CPU with an active program module. Upon release, the CPU will be returned to the available pool.

(3) Main Store Allocator (MSA) examines indexed tables to isolate main store space needed by requesting program modules.

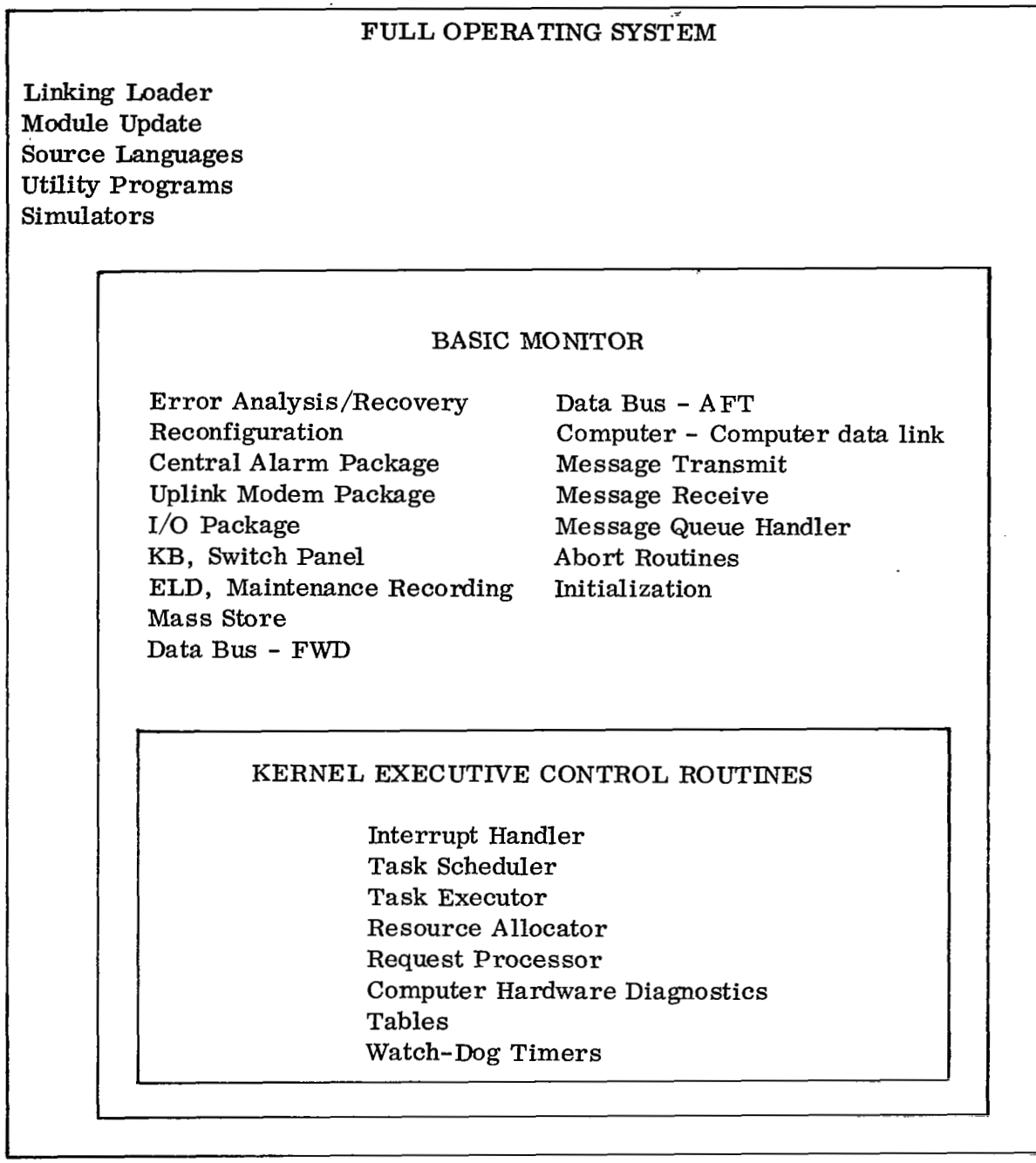


FIGURE II.C.1. Relationship of Kernel to the Remainder of the System

If scratch or high speed store is used, then the MSA will determine demand for high speed store also. Given sufficient available storage, the MSA associates storage with program module and performs the necessary associated accounting.

Upon storage release, the MSA will return storage to the available pool. It is possible that periodic "garbage collecting" may be necessary to provide sufficient areas of contiguous storage. This may imply necessity for dynamic relocation of active program modules.

(4) Task executor (dispatcher). Functional Requirements are the following:

- Determine highest priority module requesting execution.
- Cause resource allocation to be effected.
- Perform initialization of module.
- Cause execution of program module.

(5) Task terminator/suspender. This module must satisfy the following functional requirements:

- Remove program module in question from active priority thread.
- Perform necessary housekeeping operations to maintain thread integrity.
- For task suspension, an appropriate lock (state variable) must be set to ensure that module is not executed. The module, however, is not removed from priority thread. Then, when the lock is removed, the task can be executed.

(6) Request processor. Functional requirements will be:

- Request analysis.
- Parameter passing, possibly by micro-code.
- Passing control to proper request program module based on logic of request analysis.

(7) Main store protect/unprotect. This module must satisfy the following functional requirements:

(a) Perform required hardware manipulations to cause read/write prohibition, if required. Such actions could be the following:

- Test key index data for validity.
- Execute computer instructions to effect inhibition.
- Perform error analysis on key index data.

(b) Similar actions must be taken to remove prohibition.

(c) It is possible that status must be updated so as to maintain records of available unprotected storage.

(d) These functional requirements are generally satisfied by coordinating implementation with the resource allocation modules.

(8) Interrupt handler. This module should accomplish the following Functional Requirements:

- Save required registers and storage contents for interrupted program. This may be done by micro-code.
- Analyze status registers for source of error interrupts. Invoke required error processing modules.
- Analyze status registers for source of abort interrupt. Invoke suitable abort procedure modules.
- Enable or disable preplanned interrupts.
- Determine interrupt priority and make monitor call to task executor at that priority level.
- When all interrupt requests have been serviced, notify task executor (dispatcher).

(9) Abort procedures - TBD.

(10) Real-time clock/interval timer. These modules must satisfy functional requirements to accomplish functions associated with various software clocks that the monitor and (possibly) application programs use. Some of the clocks are the following:

- Diagnostic countdown(s).
- Program module execution(s).
- Program module delay(s).
- Interval timer(s).
- System performance measurements.

(11) Time slice partitioning. Module must meet the following functional requirements:

- Maintain records of elapsed time since initialization of program module.
- Compare elapsed time with requested time.
- Cause elapsed time to approximate requested elapsed time/time period.

(12) System performance measurement. These modules must satisfy the following functional requirements with respect to hardware and software:

(a) Hardware.

- Error fault detection.
- Passive instruction examination.
- Other software triggers.

(b) Software.

- System auditing.

- Event tracing.
- Performance recording and analysis.

These considerations are discussed in detail in Campbell and Heffner (4).

(13) Initialization routines. System initialization routines comprise the collection of programs necessary to meet Functional Requirements to bring the hardware/programming system on-line from a cold start. A cold start implies that no programs are running in the complex and, therefore, all bootstrapping must be done by the initialization programs.

Once system initialization has been performed, other application oriented initialization may be necessary. The act of loading or changing the operating system programs will be considered initialization. "Changing" is included in the definition because the monitor system required for pre-launch checkout and post-flight analysis may be considerably different from the system required to support flight. The following initialization programs will be required:

- System initializer (not part of this study).
- Pre-launch checkout initializer (not part of this study).
- Flight system initializer.
- Post-flight system initializer (not part of this study).
- Possible initialization routines to operate as a function of mission phase.

d. Time line controller/interpreter (TLC). This module will satisfy functional requirements to act as the system manipulator for mission profile related requirements. The monitor system must support manipulation of space shuttle tasks based on the following factors:

(1) Elapsed time.

(2) Time difference from reference time.

(3) Occurrence of arbitrary events.

- Events must be detected.
- Appropriate program modules must be notified of occurrence of specified events.
- Appropriate tasks must be scheduled to accomplish reaction to event occurrence.

e. Central alarm package. This group of programs handles the tasks germane to satisfying functional requirements for recording errors within the computer system, logging on output media, displaying and maintaining a history, and other alarming requirements. The programs will format downlink output, output to the printer, and output to the Error Logging Device (ELD). The modules will maintain a history of alarms active within the system and output this history on demand to the specified output devices. The following program packages (modules) will be required:

- Alarm history maintenance,
- Alarm output,
- Alarm analysis, and
- History output.

f. Input/output package (IOP). This collection of subroutines will satisfy message I/O functional requirements for the following devices: printer, ELD, keyboard inputs. Output messages will be formatted by the user routines.

The IOP will operate in conjunction with the device handler for the specific device. Since the IOP will place the message in an output buffer, the calling program may be rolled-back after the call is made without destroying the message. All output, correct returns, etc., become the responsibility of the IOP. Once the buffer has been loaded, the device handler will have the responsibility for outputting the message.

For keyboard input, the IOP performs error checking on words and message, and handles output error notifications, internal message formatting, analysis of message contents for action or data, and transmission of action requests. The following program modules will be required:

(1) I/O request processor.

(2) I/O initiator(s).

(3) I/O continuator(s).

- I/O error analysis.
- Input message format.
- Input message error notice output.
- Input message decoding/analysis for action or data.
- Input message transmit action requests.

g. Device handlers. Device handlers are collections of program modules concerned with the functional requirements for manipulation of external devices that are usually interrupt driven. The modules are divided into initiator-continuator pairs. The initiator has the responsibility for analysis of the calling sequence, proper response (i.e., setting buffer lengths, etc.), and transmission/receipt of the first character or block. The continuator is an interrupt driven program responsible for send/receipt of character or that part of a message sent/received during an interrupt cycle, error analysis on character, if required, determination of termination conditions, proper return from termination (either normal or error), and housekeeping associated with termination.

Usually, a continuator will require either a predetermined number of characters or an EOM character. The transmission/receipt of characters will continue until one of these conditions is met or until an error is encountered.

Upon meeting termination conditions, the continuator will perform a longitudinal record check for errors. Response to error conditions may differ with the individual continuator.

Initiator/continuator pairs should be provided for the following devices:

- Keyboards.
- Uplink modem.
- Data buses and intercomputer data link.
- Error logging device (ELD).
- Health monitoring system.

- Printer.
- LRU checkout bus.
- Mass storage devices.
- Displays.
- Downlink modem.

h. Error detection, analysis and recovery. Error routines will be concerned with functional requirements for the detection of malfunctions in the internal computer hardware and in peripheral devices attached to the computer, exclusive of the BIT panel, and compensation for these malfunctions to meet the system reliability performance requirement.

In order to accomplish error analysis and recovery in a sophisticated hardware/programming system, considerable study is required to determine the most propitious method of meeting the hardware system performance specifications. In nearly all cases, some combination of hardware/software will be required in determining hardware malfunctions, e.g., the interval timer or periodic interrupt-diagnostic countdown. In general, it is assumed that where transmission of data is involved, hardware parity checks of varying sophistication will be required. This is not to be limited to message checking, but is also to be done as a matter of course on transfers to and from storage. Furthermore, after distinguishing hard failures from transient failures, the usual requirement is for substitution of standby equipment for the LRU that has failed. The subsystem monitor will be notified of an error condition by the presence of an error-generated interrupt which will cause the related handler to relinquish control to the appropriate error processing program module.

(1) Error diagnostic routines. Computer system diagnostic routines will include off-line and on-line computer hardware exercisers with associated output routines. Computer peripherals will be checked using a software watch-dog timer. When the device generates an interrupt response, the timer will be set for a period greater than the expected interrupt period. The interrupt return routine (continuator) will reset the timer. If the timer runs out, the device will be considered to have failed.

(2) Configuration mode control. This will be accomplished by maintaining a current configuration status along with a list of available back-up states, and by coding preplanned sequences to move from the failure state to the desired state upon LRU failure, with accompanying housekeeping functions related to available states and present status.

i. Communication handler. This module must meet functional requirements for transmission and receipt of messages. The following requirements must be satisfied:

- Request processing,
- Data bus selection,
- Destination selection,
- Buffer allocation,
- Error diagnostics,
- Queue maintenance,
- Termination conditions,
- Message statistics,
- Module return execution, and
- Module roll-back in the event of failure.

j. Tables. Current values of the following system descriptors must be maintained:

- Masks,
- Flags (semaphores),
- Common constants,
- Global variables,
- System configuration status,
- Subsystem monitor data,
- Applications program communications data, and
- Branch vectors.

Tabular values must be protected by access control policies.

3. Crew Display and Control Computer (DCC) Subsystem Monitor. The DCC monitor system will satisfy functional requirements similar to requirements discussed previously for the CCC. Since the computers are similar in design to the delineation possible with the information presently available, it seems reasonable to conclude that much of the software utilized in the CCC can also be used in the DCC. For these reasons this section will be organized to reflect the functional requirements derived from the hardware architecture and applications requirements of the DCC subsystem that are different from those of the CCC subsystem.

a. Executive control routines (ECR). Some of the program modules considered necessary for the functional requirements discussed for the CCC will be present in the DCC in a functionally abbreviated form. This arises from the fact that there is a master-slave relationship between the CCC and the DCC in these cases. The CCC exercises control, while the DCC program modules exercise implementation functions. The following are examples of program modules exhibiting the master-slave relationship discussed above:

- Abort procedures.
- System performance measurement routines.
- Initialization routines.

b. Central alarm package. This group of program modules will have functional responsibility to transmit alarm notification to the CCC for further processing. Upon detection of an anomaly the appropriate message will be formatted and transmitted.

c. Device handlers. The following unique device handler modules will be required:

(1) Cathode ray tube (CRT) system.

(a) Message initiator.

(b) Message continuator.

(c) Message position calculations (blanking, etc.).

(d) Image oriented calculations (geometry).

- Beam position.
- Reference orientation.

(e) Cursor positioning.

(f) Message intensity illumination.

(g) Display file manager.

(2) Alphanumeric display.

- Display initiator.
- Display buffer formatter.
- Display image selector.

(3) Mass stores. Mass storage devices are not presently specified for the DCC. Should requirements change, support from initiator/continuator pairs will be required.

d. Summary. We have discussed the functional requirements derived from interaction of hardware architecture and applications requirements. These can be summarized as follows:

- Functional requirements will be similar to those derived for the CCC.
- The method used in the notification of a hard failure is a function of the level of hierarchy. CCC failures are transmitted to the hardware executive for reconfiguration. Other subsystem hard failures are reported to the CCC for reconfiguration.
- Many of the software modules utilized in meeting the Functional Requirements of the CCC can be utilized in the monitor system for the Crew Display and Control Monitor System.

4. Flight Control Subsystem (FCS) Monitor. The FCS monitor will be required to provide general functions as outlined in the discussion of the CCC with respect to the following computer subsystems:

- Rocket engine processors (REP).
- Jet engine processors (JEP).

- Special DIU processors.

The following program modules will exhibit the master-slave symbiosis discussed for the DCC:

- Abort procedures.
- System performance measurement routines.
- Initialization routines.

The FCS special processors enumerated below are envisioned (1) (location and specifications of the DIU's TBD):

Special Processors

	Orbiter	Booster
Rocket Engine	4	22
Jet Engine	8	12
Display and Control	4	4
Special DIU's	16-20	16-20
Total	32-36	54-58

This section is organized to reflect the interaction of hardware architecture and applications that give rise to FCS functional requirements.

a. Executive control routines (ECR). The ECR will satisfy functional requirements similar to requirements discussed in the CCC. Implementation will vary somewhat since the computer architectures involved differ. We anticipate that the following functional modules will be required:

- Task scheduler.
- Task executor.
- Task terminator/suspender.
- Interrupt handler.
- Real-time clock routines.

Modules satisfying functional requirements unique to the CCC, as discussed above, will not be present.

b. Central alarm package. The central alarm package will satisfy functional requirements identical to corresponding requirements presented in the discussion of DCC requirements.

c. Analog input/output package (AIOP). The AIOP will handle communication between analog sensors, output devices and dedicated processors (Special DIU's). Since all dedicated processors are alike, all AIOP's will be functionally similar. The AIOP modules will satisfy the following functional requirements:

- Perform demand I/O upon detection of a request from the astronaut-pilot.
- Provide different I/O frequencies to be defined by further system trade studies.
- Select a particular sensor via an analog MUX.
- Convert from analog value into previously defined engineering units.
- Format message for transmission to CCC.
- Output an analog value to a MUX.
- Convert from engineering units to analog or digital values for output.

Studies of relative computer loading may show that engineering units conversions should be performed in CCC.

The following functional modules will be required:

(1) Analog input.

(a) Scan.

- Initiator will prepare masks and flags as required and examine scan classes for current activity. If a scan is required, it is accomplished via the Analog Scan Continuator.

- Continuator will execute sensor selection and reading via the analog MUX.

(b) Conversion. This will be accomplished by applying mathematical transformations to the readings provided by the Analog Scan Continuator. These transformations are usually linear, quadratic or some similar relation that is relatively easy to evaluate.

(2) Analog output. This module must output a prescribed analog voltage or current to a selected device. Digital to analog conversion will be effected.

d. Digital input/output package (DIOP). The DIOP will perform functions similar to the AIOP functions, but for digital devices. The following functional modules will be required:

(1) Digital input.

(a) Digital MUX. If digital sensors and output devices are connected through a MUX, then the conventional logical method is to operate via the initiator, continuator combination. This procedure is necessary when hardware register settling times are significant and there are different scan frequencies present. The Digital Scan Initiator and Digital Scan Continuator will satisfy functional requirements similar to the analog modules.

(b) Without digital MUX. If digital sensors and output devices are connected directly to the computer hardware, logical requirements become simpler. The Digital Scan routine can be executed at a predetermined frequency to input each value without the complexities associated with manipulating the MUX.

- Pulse information. Input of pulse type information is likely to bypass the Digital MUX. Particular logical requirements are a function of hardware-programming trade-off. Relative processor loading will significantly affect hardware requirements. If studies show the expected processor load to be on the order of 50% of available computing time or more, we can expect to accumulate pulses via hardware and periodically transmit accumulation to the processor for evaluation. If processor loading is relatively light (20-25%), we can expect

the processor to perform the integration, depending on the number of expected integrations required. This is another example of Performance Requirements interacting with Functional Requirements.

(2) Digital output. The effect of the requirement for a digital MUX on Digital Output is identical to that for Digital Input. With these considerations in mind, some possibilities for Digital Output are as follows:

(a) Contact closure. This is the act of setting or resetting a bistable device. Contact closures can be either momentary or latching. Functional Requirements follow:

- Select bistable device,
- Output set or reset command,
- Check status to ensure validity, and
- Notify Central Alarm Package if invalid.

(b) Pulse output. This is the transmission of pulse type information similar to that discussed above.

(c) Loading digital values into registers. This has the following functional requirements:

- Select registers,
- Transmit value from computer storage to registers,
- Test register to ensure validity, and
- Notify Central Alarm Package if invalid.

e. Direct digital control (DDC). Study is required to determine feasibility of employing DDC techniques in the control of the various actuators that will interface with the computer system. Hardware simplification may certainly result at the cost of additional development effort.

DDC accomplishes control of actuators by filtering each error signal with a composite of historical effects. A considerable amount of experience

in control using hardware methods to implement the filtering has been amassed. Identical results can be obtained using programming techniques with a concurrent reduction in hardware complexity and weight.

In effecting DDC, there are programming considerations such as the following:

(1) Controller mode equations. Conventional methods use simple mathematical transformations called Proportional, Reset or Integral, and Rate. A single mode equation would generate an actuator correction signal that is proportional to the error signal.

Actuator Correction = K*E

where "K" is proportionately Constant

"E" is error signal

Transformations will be as complex or simple as required by the control law.

(2) Special transformations. These can be applied as desired. A convenient procedure is to develop an interactions matrix. This matrix will predict the effect of the history of changes in one actuator on all variables. The effect of previous moves on all variables can be used to determine more nearly the necessary correct actuator moves and thereby increase system response without increasing the tendency to become unstable.

(3) Mission profile considerations affect logical operations in that control loops may be required to provide different characteristics as a function of mission phase. Some considerations follow:

- Loop-delete is required to remove a control loop from the loop library.
- Loop-enter provides capability to start up new control loops as required by adding to the active library and utilizing the Loop Activate module. This module provides facility for on-line reconfiguration of control system.
- Change-mode-constants allows alteration of control loop characteristics as required.

- Loop-activate causes a control loop in the loop library to be placed in an active status.

- Loop-deactivate places a loop in the loop library in an inactive status.

- Sequencing functions can be imbedded in the DDC features by insertion of programming modules to effect sequencing as part of the system requirements. The programmed testing and operating system calls necessary to accomplish sequencing functions can be more efficiently imposed as part of the monitor than as part of an applications program. These sequencing functions can be utilized as an interface to the CCC Time-Line Controller modules (e.g., sequence requests will be transmitted from CCC to destination special processor). A special processor monitor will interface with the transmitted sequence request to effect a series of events that represent a desired sequence.

f. Filtering, smoothing and prediction. Mathematical procedures can be employed to remove estimated error components of signals and otherwise pre-process incoming data, possibly in real-time, to transform it into more amenable values or formats for further processing. Some available techniques are:

- Exponential smoothing. This is possibly the most attractive candidate for removing estimated error in data from the point of view of programming simplicity and rapidity of execution. Exponential smoothing transformations are of the following form:

  New value = K *(New Datum) + (1-K) * (Previous New Value)

  where K is the smoothing constant ($0 \leq K \leq 1$). Simulation techniques, using historical data, where possible, are useful in estimating K values.

- Trending. This operation can be performed on data once the estimated error is removed.

- Limit checking. This can be performed against either the smoothed data or the trend data to check violations of magnitudes and rates of change. Notification of anomalies are transmitted to the CCC for further evaluation and recording.

g. Device handlers. Device handling modules meet functional requirements to direct the operation of peripheral I/O units. Generally a device handler will consist of a symbiotic pair of functional modules:

(1) Device initiator. This module satisfies request analysis, parameter transfer, and initialization requirements.

(2) Device continuator. This module satisfies requirements to manipulate peripheral devices using data provided by the Device Initiator. Common requirements are device selection, data transfer to or from a device, storage of information in tables, program accounting associated with termination conditions, and error detection and analysis. Special functions may be required by variations in hardware architecture in addition to the common features.

Device handlers will be required for the following peripheral equipment:

- Analog MUX,

- Digital MUX, and

- Data bus.

h. Communications handler. Communications handlers satisfy functional requirements for the receipt, transmission, recording, error analysis, queue manipulation and other logical aspects of message transmission between special processors and the CCC via a data bus or other link. Communications handlers will interface with a device handler which will manipulate the peripheral hardware in question. Messages will be broken into two classifications:

(1) Transmitted messages. The message transmit initiator will be functionally identical to the Data Bus Initiator in the event that the only communications may be via the data bus. If other communications media are available, for example computer to computer, the Message Transmit

Initiator will be a module meeting the following functional requirements:

- Request analysis,
- Parameter transfer,
- Communication media selection,
- Device initiation, and
- Output message queue handling.

(2) Message acquisition. This requires the following modules:

(a) Message receive initiator. This module satisfies the following functional requirements:

- Buffer assignment.
- Initialization.

If the data bus is the only communications media, the Message Receive functional requirements will be met by the Data Bus Handler.

(b) Message receive continuator. This module satisfies the following functional requirements:

- Input message queue handling,
- Word and message parity check and corrective action (horizontal and longitudinal check),
- Message continuity check for possible lost messages, and
- Message analysis for content to distinguish between whether data or task execution request was transmitted. Sequencing commands can either be transmitted by the CCC, as required, or sequences can be stored in the special processor and sequence execution commands transmitted by the CCC to cause operations of a complete sequence of events.

i. Tables. Functional Requirements are identical to the Functional Requirements presented in the discussion of the CCC with the exception that configuration status tables are not required.

j. Error detection, analysis and recovery. Computer logical operations to meet Functional Requirements to provide mission reliability through error detection, analysis and recovery will operate in several ways:

(1) Computer hardware diagnostic routines are required to exercise computer logic and storage during otherwise idle time and compare actual results with expected results. Malfunction notification will be transmitted to CCC for recording and further evaluation.

(2) Diagnostic countdown techniques are required whenever an event is expected to occur within a known time. The diagnostic countdown can be set to a period greater than the expected elapsed time. If the event occurs, it can trigger resetting of the diagnostic timer. If the diagnostic countdown runs out, then the device responsible for the event is deemed to have failed. This failure notification is transmitted to the CCC for recording and further evaluation.

(3) Configuration mode control will meet functional requirements to be effected by either the subsystem, CCC or hardware executive. It is not clear at this time which is the optimal place. Intuitively, it seems better to have subsystem configuration mode control residing in the CCC and CCC mode control in the hardware executive.

(4) Protect violation functional requirements can be categorized as responses to:

- Attempts to write into storage in which "protection" has been invoked,
- Attempts to execute privileged instructions, and
- Attempts to refer to storage locations outside of module boundaries.

The functional requirements for these situations are presently undefined.

(5) Error recovery routine functional requirements include the following:

- Analysis to separate transient malfunctions from "hard" failures, and

- Graceful degradation in the event of a hard failure.

k. Summary. Considering the above discussion, it is evident that although Functional Requirements for the Flight Control Monitor System will have some similarities to the CCC Monitor System, the hardware architecture and applications requirements give rise to a greater degree of special purpose requirements:

- The FCS Monitor System can be viewed as similar to a typical industrial process control system with additional message switching and reconfigurable capability.

- The hardware architecture is dedicated to sensor analysis and manipulation.

- Sequencing capability must be provided by interaction with the CCC monitor system.

- Data can be pre-processed to reduce computing load on CCC.

- Peripheral devices require a symbiotic pair of functional modules for each class of device.

- Reconfiguration requirements may be satisfied by procedures implemented in the CCC and FCS. Presently, the specific functional requirements have not been determined.

5. Navigation and Sensor Computer Subsystem (NSC) Monitor. The NSC monitor system will be required to provide functions similar to functions described in the discussion of the FCS. These functions pertain to the following computer subsystems:

- IMU processors.

- Special processors.

A schematic diagram of a typical IMU processor is shown in Figure II-B-4. The IMU processor acts as a data concentrator for navigation sensor and calculated data to be utilized by the CCC for guidance calculations.

Data from sensors, such as Radio Altimeters, will be analyzed by special processors associated with DIU's. The NSC monitor will interface with CCC hardware-programming combinations via the data buses and IOCU's. The NSC system will transmit navigation data, status information, and error messages to the CCC system. The NSC system will receive data and task execution and sequencing directives from the CCC system.

Functions associated with the modules itemized below are similar to those discussed for the FCS:

- Executive control routines,
- Central alarm package,
- Analog input/output package,
- Digital input/output package,
- Direct digital control,
- Filtering, smoothing, and prediction,
- Device handlers,
- Communications handlers,
- Tables, and
- Error detection, analysis and recovery.

This monitor package will be concerned primarily with input of sensor data with few anticipated outputs to peripheral controllers.

D. Executive System Performance Requirements

1. General. Performance Requirements are defined as a measure of how well the system must perform in terms of parameters such as timing, accuracy, reliability, and capacity. Performance Requirements may arise from hardware (e.g., where it is desirable to drive a peripheral device at its maximum input or output rates) or from applications (e.g., where tolerance limits require scanning a sensor at a given rate). Convenient points for measuring Performance Requirements are at an interface through which data is transferred between devices. For this reason, and because they represent logical applications groupings, Performance Requirements will be categorized as either system or subsystem.

2. System Performance Requirements. System Performance Requirements are constant throughout the computing complex.

a. Fail operational, fail operational, fail safe is a performance criteria that the entire system must meet. This requirement has a significant effect on programming complexity. It implies:

- Multilevel program module priority interrupt,
- Configuration control, and
- Redundancy within subsystems.

b. Sensor performance requirements are logically identical although particulars may vary through the system. Sensors must be categorized by type (digital or analog), scan period, conversion requirements and other pertinent parameters to establish performance requirements for the program modules responsible for scanning and conversion. Similar requirements are necessary for output devices which must be categorized by parameters such as digital/analog, output period, conversion equations required, output type (pulse), etc. No sensor performance requirements have been determined at this time.

c. Configuration status and control register formats must be designated. Reconfiguration sequences must be defined utilizing the above registers to determine status maintenance and configuration change requirements; these have not yet been defined.

3. Subsystem Performance Requirements. Subsystem Performance Requirements apply to a particular subsystem.

a. Central computer complex (CCC).

(1) Hardware.

- Data bus traffic must be determined. Present estimates place Aft Data Bus traffic at 1.08 kilobits/sec. This estimate combined with Forward Data Bus traffic will help set Performance Requirements for the IOCU's and their related driver programs. Other data requirements must also be considered. As system definition evolves, further estimates will be refined.

- Mass store transfer rate must be established to set an upper bound on the transfer of information to and from main storage. Mass storage size is estimated as 14.7K words/computer, which is extremely low.

- Sensors must be classified by scan frequency; whether they are analog or digital, signal level, conversion factors, smoothing, limit checking, alarms, and other allied requirements.

- Core size requirements must be further refined. Present estimates are 64K, quadruply redundant. Of available core, 38K is estimated as being required with 686 ms/sec peak calculations. These estimates are likely to be optimistic.

- IOCU specifications must be delineated.

- Hardware executive performance requirements must be described.

- Built-in test (BIT) procedures need to be defined (TBD).

(2) Applications-related performance for the CCC has been defined (3) as follows:

- Guidance will establish data rate performance requirements on the Navigation and Sensor subsystem. In addition, output performance requirements will ensure that the output task can be transmitted to the Flight Control subsystem to manipulate the rocket engines, if required.

  Presently, error tolerances have been set for certain mission phases. Error analysis must be performed on the sequence of events from sensor scanning through conversion, calculation, data transmission, conversion to digital, and output to determine the contribution to error of each phase. Performance requirements for each phase can then be set so that total error is less than error tolerance.

- On-Board Mission Planning must be defined with regard to its I/O requirements, frequency of execution and other pertinent parameters to estimate effect on system loading and necessity for establishing related Performance Requirements.

- Configuration and Sequencing Control is discussed in Section II. C. 2. h.

- Energy Management must be defined more clearly to establish related requirements.

- On-Board Checkout Management - TBD.

- Display Executive Control - TBD.

- Data Base Executive Control - TBD.

b. Flight control system (FCS).

(1) Hardware. Consists of eight jet engine processors with 1. 5K words main store each, and four rocket engine processors with 6K words main store each.

- Performance Requirements are concerned with sensor I/O rates as discussed above for the CCC.

- Data bus I/O rates must be defined.

(2) Applications. Execution rates for program modules involved in conversion and control loop calculations with accompanying equations must be defined. Sequencing procedures must be defined for engine startup and shutdown.

c. Navigation and sensor subsystem (NSC).

(1) Hardware. Consists of four IMU 3-Packs with associated special processors. Each special processor has 1536 words R/O memory, 512 words R/W memory.

- Performance Requirements will be established as discussed above for the FCS. In addition, error analysis must be performed as discussed above for the FCS.

- Data bus I/O rates must be established.

(2) Applications. Performance requirements are a function of mission phase. Data are to be provided for guidance calculations in the following phases:

- Launch,

- Orbital insertion,

- Rendezvous,

- Station keeping,

- Entry,

- Booster and orbiter ferry between airports.

Double-precision attitude determination can be made in about 14 ms plus overhead (5). The following are accompanying considerations (1):

- Accuracy requirements will specify update frequency.

- IMU gyro drift is estimated to be 12 sec. of arc/hr.

- Error of update algorithm is a function of sampling period and magnitude of slew angle. This must be considered in requirements for these calculations.

d. Crew display and control subsystem (DCC).

(1) Hardware. Consists of four display processors, 6K words each, connected through the redundant data bus to symbol generators, two input keyboards, a mode switch panel, and various sensors.

- Performance Requirements for data transfer between the Display Processor and the Symbol Generator(s) must be established. Presently, image refreshment is estimated at 1/sec rep rate.

- Mass store requirements and transfer rates must be established.

- Keyboard input frequency and program module requirements must be estimated as part of system load.

- Other sensors must be analyzed as discussed above for the CCC.

(2) Applications. Display support modules must meet performance requirement of one display update each second. The perspective display calculations must satisfy this requirement with respect to execution time; that is, the logic module time plus overhead and other activities cannot exceed one second. The sophistication of the display will be forced to conform to this constraint.

e. Special DIU processors.

(1) Hardware - TBD.

(2) Applications are primarily concerned with sensors and must be analyzed as discussed above for the CCC.

4. Summary. As is noted above, it is seen that performance requirements are defined as measurable standards of achievement for the monitor system.

- There are standards that are invariant through the avionics monitor system. For this reason, those qualities are categorized as System Performance Requirements.

- There are other standards that are peculiar to a subsystem. They are categorized as Subsystem Performance Requirements.

- Considerable effort will be required to complete all performance requirements. Many are undefined to date. The determination of all monitor system performance requirements will entail knowledge of the intricacies of the operation of all of the functional modules of each subsystem and of the interactions between these subsystems. This knowledge is presently deficient.

References

1. Space Shuttle Integrated Avionics. McDonnell Douglas Slide Presentation, McDonnell Douglas Astronautics Company, June 12, 1970.

2. Space Shuttle Integrated Avionics. McDonnell Douglas Slide Presentation, McDonnell Douglas Astronautics Company, February 11, 1970.

3. Integral Launch and Reentry Vehicle Systems, Configuration Design and Subsystems, Volume I, Book 1. NASA CR-66863-1 (MDC E0049), McDonnell Douglas Astronautics Company, November 1969.

4. Campbell, D. J. and Heffner, W. J.: Measurement and Analysis of Large Operating Systems during System Development. Proceedings of FJCC, Volume 33, No. 1, 1968.

5. Hrastar, John: Attitude Control of a Spacecraft with a Strapdown Inertial Reference System and Onboard Computer. NASA TN D-5959, September 1970.

SECTION III. SPACE STATION/BASE

A. General

The purpose of this section is to describe the space station data management system hardware configuration, and to determine and define the performance and functional requirements related to executive systems responsible for control of the multi-computer data management system. The section consists of major divisions devoted to

- Data Management System Description,
- Executive System Functional Requirements, and
- Executive System Performance Requirements.

The first part places particular emphasis on functions, hardware description, and the bus structure. Of the four major computer complexes within the data management system, two are at the first stages of conceptual design and are outlined only roughly at this time. The remaining two are identical and, while they are still conceptual in nature, more advanced from a definition standpoint. The functions of these complexes with respect to the space station mission are delineated clearly. The bussing concept is also outlined in some detail functionally.

The second and third parts are concerned with analysis of the computing hardware architecture and characteristics of space station application software leading to a logical determination of the performance and functional requirements for executive control of the entire data management system. Application functional and performance requirements do not, for the most part, exist at this time necessitating a large degree of a priori insight into the nature of the programs. As a result, experience and intuition combined with largely speculative and conceptual information forms the major rationale for requirements development. Despite the general lack of expected time-line activity, the functional requirements outlined in this section are felt to be not only credible, but also comprehensive.

- Antenna Control. Selection and pointing of antennas.
- Navigation. Calculation of Onboard Orbital Ephemeris with sensor derived update.
- Stabilization/Control. Implementation of control laws, attitude reference transformations, and energy management.
- Rendezvous and Docking. Computations supporting rendezvous and docking operations.
- Reconfiguration. Dynamic modification of subsystem equipment and software configurations.
- Subsystem Consumables Management. Consumable utilization and prediction of replenishment requirements; rescheduling of consumable expenditure to conform to available stores.
- Inventory Control. Subsystem spares status and replenishment requirements, record-keeping.
- Maintenance Information Control. Location and display of subsystem maintenance, setup, and operating procedures.
- Habitability Status/Support of Other Vehicles. Status determination of logistic and experiment modules (when docked) for habitability.
- Checkout. Status, diagnostics, fault isolation for all station, integrated and attached module subsystems.
- Self-Check. Self-diagnosis for error detection and recovery.
- Training. Simulation of infrequently performed station operations, such as docking, to maintain crew proficiency.

b. Experiment (computation) support.

- Experiment Control and Scheduling. Initiation, operation, and termination of experiments according to mission timeline.

B. Data Management System Description

1. General. It is the intent of this section to establish the status of those aspects of the Data Management System (DMS) baseline that directly influence executive system requirements analysis and design. Overall system functions to be performed by the DMS are described. The current conceptual computer hardware configuration is described with specific emphasis on the areas that directly impact upon executive requirements.

2. DMS Functions. Functional requirements consist of those constraints imposed on the data management system in performing the functions allocated to it or arising from crew and program element interface considerations. They are listed below (1).

a. Subsystem (computation) support.

- Subsystem Control and Scheduling. Master control of automated subsystem and experiment operations.
- Display Generation and Control. Control of integrated subsystem displays and the generation of display commands.
- Data Bus Control. Timing and control of data reception and transmission via the data bus.
- Telemetry Control. Selection and scheduling of downlink data transmittal.
- Data Acquisition Formatting. Sequencing, encoding, and scheduling of subsystem data.
- Decommutation. Retrieval and reconstitution of encoded subsystem data to its original or to some other desired form.
- Storage. Control of subsystem data accumulation and retrieval.
- Command. Command validation, storage, and execution for subsystems.
- Data Computation. Redundancy reduction, trend extrapolation, and general data manipulation in support of subsystems.

- Display Generation and Control. Control of experiment displays and generation of display commands.

- Data Acquisition Formatting. Sequencing, encoding, and scheduling of experiment data.

- Decommutation. Retrieval and reconstitution of experiment data to its original or a new form.

- Storage. Control of experiment data accumulation and retrieval.

- Experiment Data Correlation. Association of experiment data with operational data, such as position/velocity coordinates.

- Experiment Calibration. Correlation of sensor calibration and usage data with sensor outputs accompanied by mechanical adjustment of equipment and software parameters.

- Data Compaction. Redundancy reduction and software data compression algorithms.

- Inventory Control. Generation of experiment spares status and replenishment lists.

- Maintenance Information Control. Location and display of maintenance, repair, and operating procedures for experiments.

- Checkout. Status, diagnostics, fault isolation for experiments.

- Self-Check. Self-diagnosis for error detection and recovery.

- Experiment Planning. Short-range (daily or bi-daily planning).

- Crew Health and Proficiency Testing. Analysis of physiological and psychological data in conjunction with laboratory equipment.

c. Data acquisition.

- Signal Conditioning. Conversion of sensor outputs to desired amplitude levels and frequency content; i.e., scaling and frequency to voltage conversion.

- Multiplexing. Combining of multiple data channels into one channel using frequency or time division techniques.

- Analog to Digital Conversion. Encoding of sampled analog data to a selected digital form.

- Formatting. Collection of a prescribed amount of data, along with its associated synchronization words, control words, timing signals, and other required overhead information into proper order for onboard or downlink distribution and processing.

- Limit Checking. Comparison of a measured parameter against prescribed limits.

- Address Decoding. Recognition of instructions contained within a binary-coded control signal.

- Hardware Programming. Synchronization and control of digital logic circuits and events by timing signals (microprogramming).

d. Data distribution.

- Addressing. Command selection of device, word or message, transfer, and acquisition.

- Decoding. Deciphering of a device address, channel address, and instructions.

- Error Detection. Determination of message validity.

- Buffering. Temporary storage of analog-oriented or digital data after acquisition and prior to transfer.

e. Data storage.

- Recording. Transformation of electronic signals to a form or medium allowing their accumulation and retention.

- Retrieval. Location and transformation of recorded data to its original form.

f. Image processing.

- Film Development. Photographic processing.

- Image Transformation. Conversion of imagery to a form suitable for viewing and editing.

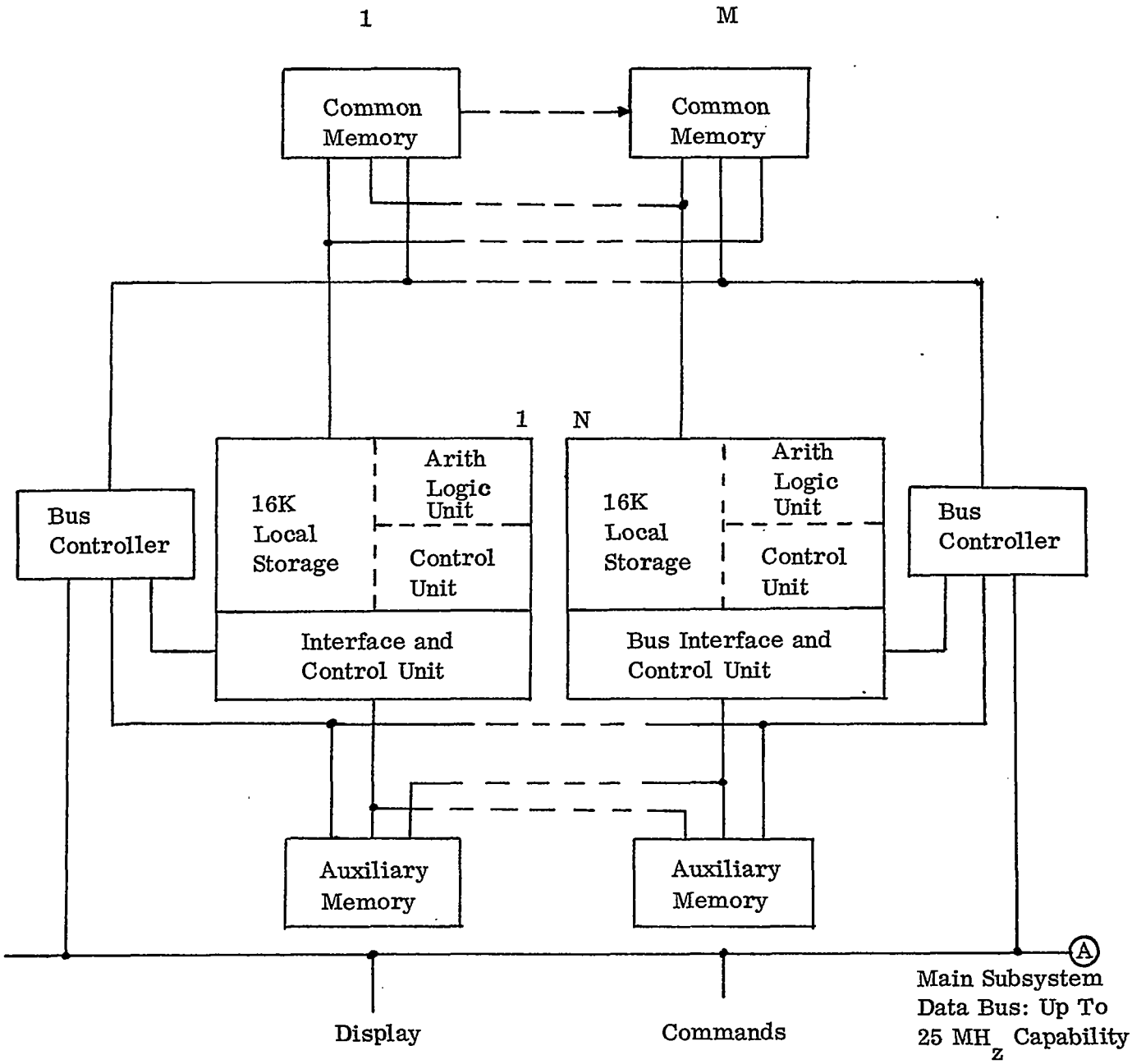
- Image Storage. Storage of video, analog, and digital data in physical form.

- Image Retrieval. Acquisition of records stored on film, microfilm, or tape.

3. DMS Hardware Description. The DMS consists of two independent (DMS and experiment) multiprocessor complexes and a dedicated, hardware-redundant Guidance, Navigation and Control (GNC) computer. A dedicated simplex computer has been included within the biomedical facility. All subsystems are interconnected through a complex data bus network.

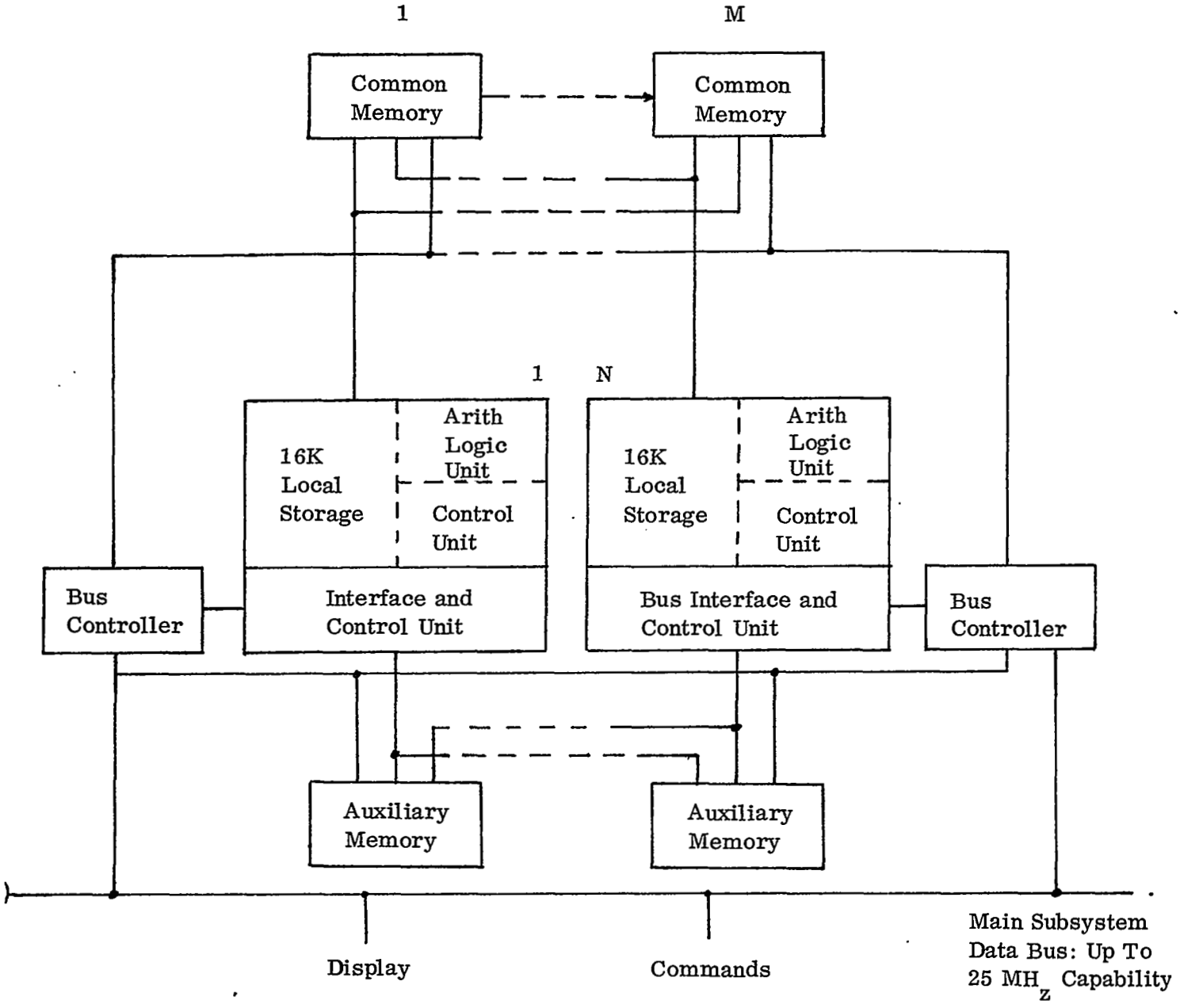
a. Data management systems. Block diagrams of the basic features of the computer complex architecture are shown in Figure III-B-1, parts A and B. A command/control console located on both the central operations and the experiment control decks facilitates direction of computer operations, modification of existing software routines and subsystem mode changes. Portable control units located on other decks provide supplemental crew/subsystem interfacing.

The DMS and experiment computers and their functions are shown, together with main and auxiliary memory, in Figure III-B-1(1). The main memory is arranged in modules which are connected to the processors via a switching unit. The quantity of Arithmetic Logic Units (ALU's) possible for each multiprocessor complex is limited to the number of memory modules ports less the number of I/O Bus Controllers. Under this scheme, any memory module or group of modules can be used by any of the multiple processors. The memory modules will be used to store programs in the active state. Other programs and data will be maintained in mass storage.



DISTRIBUTED MULTIPROCESSOR CONFIGURATION - PART A
(Station Control and Support)

FIGURE III-B-1



DISTRIBUTED MULTIPROCESSOR CONFIGURATION - PART B
(Experiment Control and Support)

FIGURE III-B-1

The mass storage will be a random-access device, but of slower speed than the main memory modules. Other large amounts of data requiring infrequent access will be stored on a bulk data, sequential auxiliary device, such as a tape unit.

Bulk data storage utilizes ultra-high density magnetic tape recording techniques and is configured in a system to meet high data volume storage requirements and relatively slow access speed requirements. It is expected that this storage subsystem configuration will be used primarily for recording of digital data prior to processing on-board or prior to return to earth via logistics vehicles.

Block diagrams showing the relationship between the data management computer complex and the internal subsystems is shown in Figures III-B-2(2) and III-B-3(2). The complex is comprised of two multiprocessors (each system consists of at least two individual processors) and two dedicated processors. The multiprocessors perform station-keeping and experiment control while the dedicated processors perform Guidance and Navigation (G&N) functions (hardware redundant) and support biomedical requirements (simplex).

The station-keeping and experiment control processors comprise a symmetric multiprocessor configuration where each set has two processors, one shared memory set, and common I/O channels.

A dedicated processor implemented in a dual redundant configuration is chosen for the G&N subsystem since it offers the highest degree of availability (a complete backup station is available in the event of failure) for this critical function. Biomedical applications are accomplished with a simplex dedicated processor; i.e., one processor, memory set and I/O set.

Salient features of the DMS are as follows:

(1) Computer.

- DMS Multiprocessor as master executive and computation center.
- Experiment Multiprocessor for processing and sub-executive control.
- GNC Dedicated Dual Processors.
- Dispersed Processors for localized computation and interface simplification.

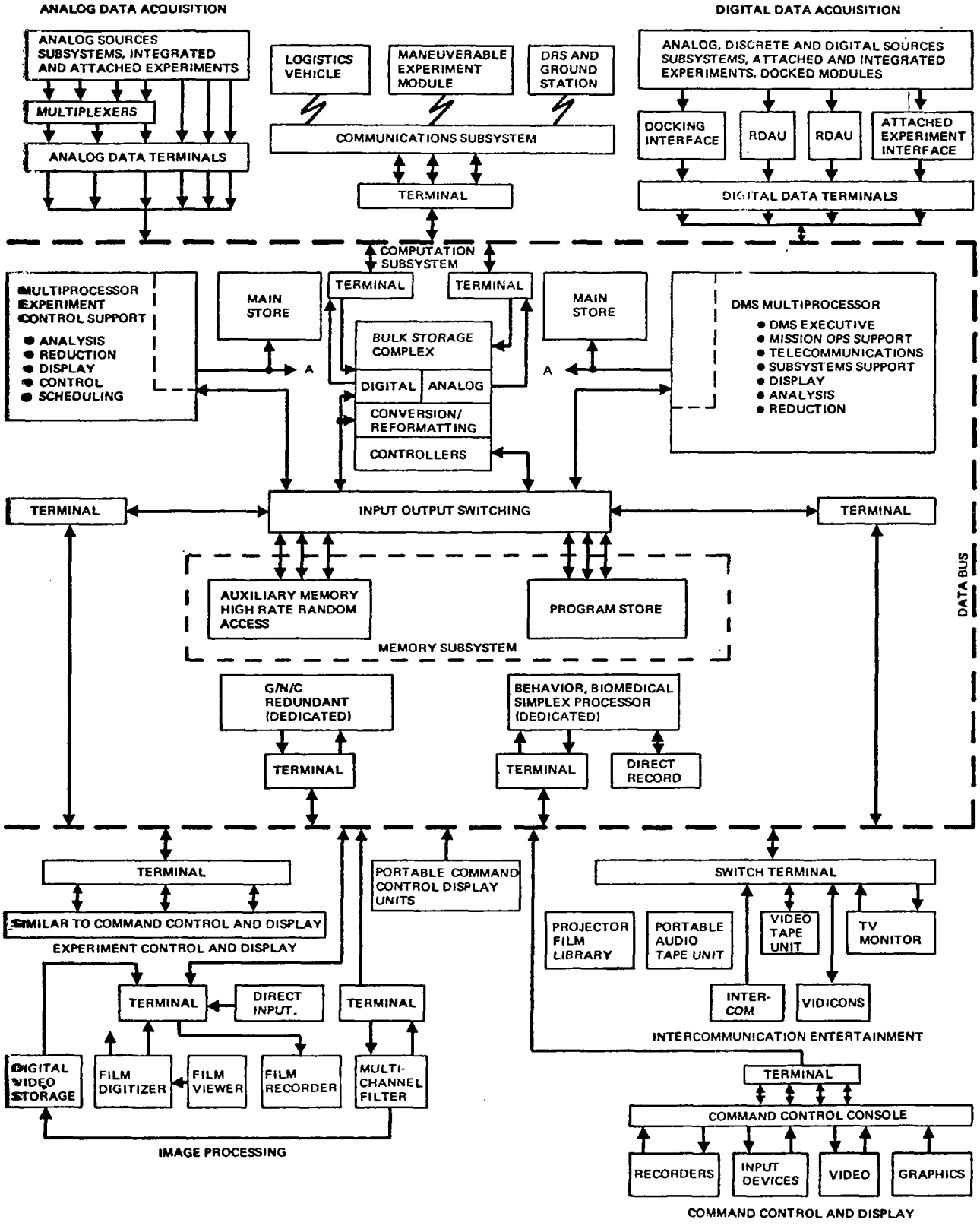


FIGURE III-B-2 DMS BLOCK DIAGRAM

DISTRIBUTED COMPUTER CONFIGURATION

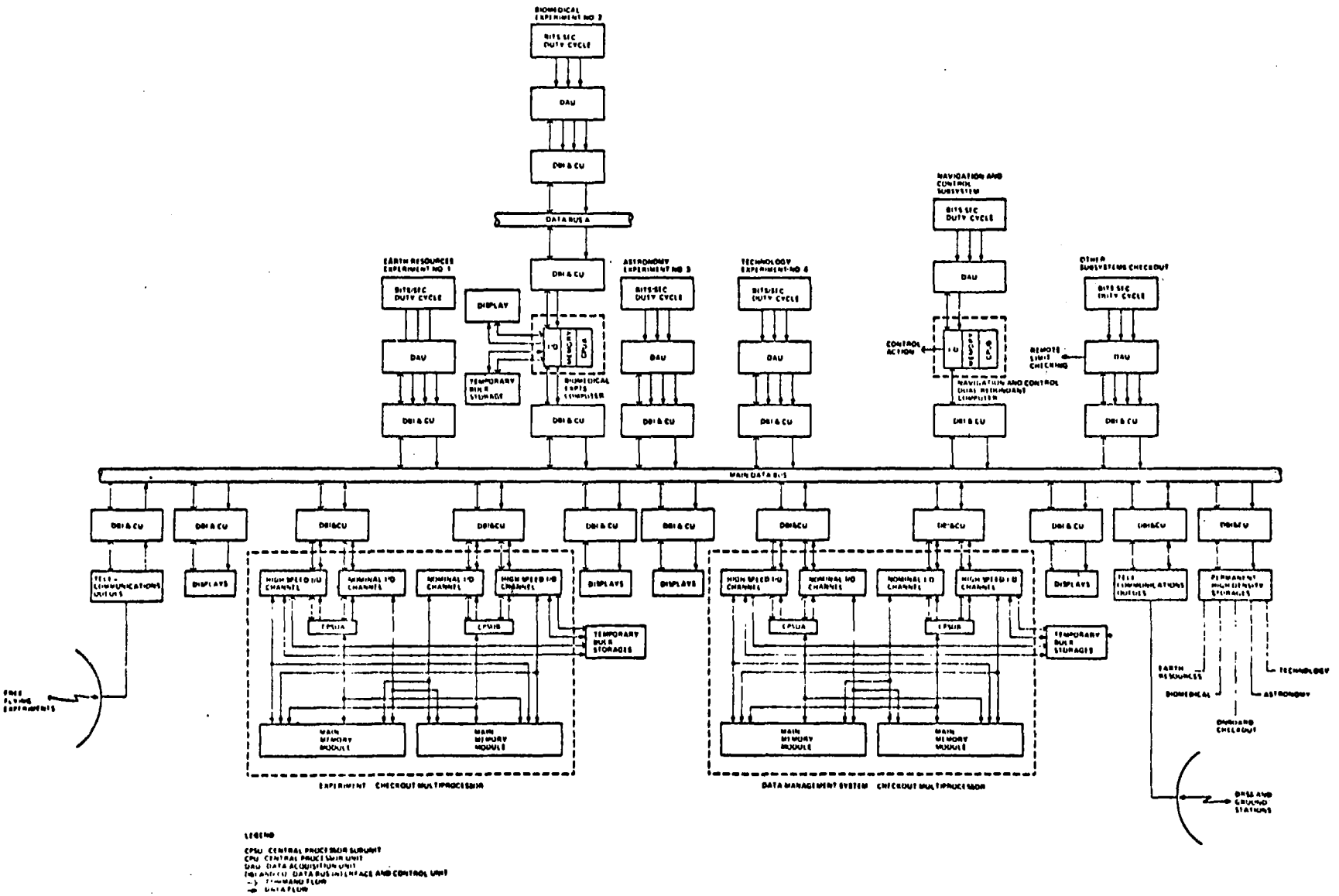


FIGURE III-B-3

(2) Data acquisition.

- Data acquired by remote units; remote units perform A/D conversion, multiplexing, and limit checking.
- Standard terminals used for interface to data bus.
- DMS computers provide control, sequencing and formatting.

(3) Data distribution.

- TDM/FDM Coax Bus. Multiple buses provide redundancy.
- Control provided by polling and selection from Master Station.

(4) Display controls.

- Integrated displays.
- Centralized mission control/operations center.
- Master experiment control center.
- Portable units for remote control/display.
- Controls. Nonmechanical.

(5) Bulk data storage.

- Tape selected on basis of recording density and low volume of medium.

(6) Image processing.

- Minimal system for control and calibration of experiments.

b. Processor and memory description. Certain common ground rules apply to all DMS subsystem configurations as follows:

- All general purpose processors have the same performance capability. Fewer assemblies in implementing the computer subsystem minimizes spare assemblies, maintenance, training and manuals. It also provides maximum levels of fail-safe capability and software interchange.

- A minimum of two auxiliary memory units in critical operations shall be on-line at all time. File tapes, disc units, and 64K memory units will be assigned in pairs to the central processors in station control and management. High density and incremental magnetic tape units may be assigned singly.

- At least two spare units shall be stocked for all units involved in critical operation. Two or more spare units have been included for all central processor, main memory, file tape, or disc units involved in GNC processing or station control and data management processing.

- A monolithic memory has been chosen for the shared main memory because of weight, power and volume considerations. Discs are used for the auxiliary memory and file-tapes for program store. High density tapes are used to record data occurring at high rates primarily from the experiment packages. For low rate data occurring asynchronously, incremental tapes are provided.

The two multiprocessor configurations allow executive control of station operations and experiment operations control. The DMS multiprocessor contains the master executive function that coordinates all operations.

Near autonomous operations of functions assigned to the two processors minimizes data transfer between the two equipment groups. Each multiprocessor requires high rate, parallel data transfer. Transfer across the data bus normally is limited to executive control functions. This enables greater utilization of the available bandwidth of the data bus for data and control transfer between experiments and subsystems.

In special instances, the configuration boundary of one multiprocessor extends into the physical area of the other multiprocessor. This occurs under control of the executive program which reassigns a processor memory or I/O unit. The reassignment results from an error condition or temporary processing overload. Data Bus bandwidth and data access time limit this type of reassignment.

A single simplex computer is dedicated to Biomedical-class experiments on the basis of a large number of specialized measurements and to provide maximum flexibility and freedom for crew interface. Since the function is not flight critical, a simplex machine was chosen.

The Onboard Checkout System (OCS) utilizes the DMS basic operating system. This includes the operational language, compiler, and the utility and supervisory routines necessary to support checkout operations. The major OCS software effort lies in the special application programs unique to checkout.

The central processors will have a standard instruction set supplemented with real-time features. Special macro-instructions improve operation of certain high usage arithmetic (compare or search program routines). Using the operating system set simplifies program checkout and simulation of real-time performance. Processors used for both multiprocessor and dedicated processor functions are identical with the possible exception of microprogram modules. These differences shall not affect the capability for any processor to execute the supervisor and critical station-keeping programs. This design requirement assumes that any processor may be used as a physical and logical replacement for a processor assigned to the Space Station control function. (1)

A common set of processing and storage units is used for all configurations to minimize spare requirements and maintain a high level of fail-soft capability for critical operations.

Characteristics of each unit are summarized below: (3)

(1) General Purpose Processor.

- 150-200 KOPS capability.
- Floating point arithmetic.
- Shared memory bus.

- 8-Channel, Hi-speed I/O interface.
- Word Length - 32 bits, plus check bits.

(2) Main memory.

- Type - monolithic.
- Capacity - 16,384 - 32 bit word length, plus check bits.
- Cycle time - 1 micro sec.
- Access time - .5 micro sec.
- Interface - 6 ports for interconnection to processors (up to 4) and bus controllers.
- Control - random address.

(3) Magnetic disc.

- Capacity - 14,500,000 bytes plus check bits, per disc pack.
- Data rate - 312,000 bytes per second transfer, read or write.
- Access time - 72.4 msec average.
- Control - record addressable.
- Interface - 2 ports for interconnection to processor I/O channels or Bus Controllers.

(4) High-density tape.

- Density - 16,000 bits/in.
- Tape speed - 100 in./sec.
- Tracks - total: 128 (16 byte) plus check bits. Single operation: 16 (2 byte) plus check bits.

- Data Rate - 800,000 words per sec.
- Reel Capacity - 1,000 ft.
- Interface - 2 ports for interconnection to processor I/O Channels or Bus Controllers.

(5) Incremental magnetic tape.

- Density - 1,000 bits/in.
- Tracks - 16 plus check bits.
- Reel Capacity - 2,400 ft.
- Interface - 2 ports for interconnection to processor I/O Channels or Bus Controllers.

(6) Monolithic memory - auxiliary.

- Type - Monolithic.
- Capacity - 64K words, 32 bits plus check bits.
- Cycle Time - 2.0 micro sec.
- Access Time - 1.0 micro sec.
- Interface - 2 ports for interconnection to processor I/O Channels or Bus Controllers.
- Control - Word Addressable.

The computer auxiliary memory provides the capability for reading a variety of stored programs into the computer on an as-needed basis for selected or intermittent computer operations. The programs will have been prepared in advance and kept in storage in anticipation of the above; or new programs will be generated, as required, on the ground and transmitted (via RF link) or carried (via logistic vehicle) to the station.

During system operation, one of the processor modules will contain the executive program and will function as a master controller. It will assign tasks and equipment to the other processors as the computation workload varies. When equipment is not actually operating, it will be placed in a standby condition to conserve power. Periodically, each processor will run self-test routines to detect failures. If a failure is detected, the failed unit will be switched out of the system and replaced by another module. The crew will be notified via the Onboard Checkout System (OCS) so that corrective action can be initiated.

c. Bus structure.

(1) Data distribution. The digital portion of the Data Distribution Subsystem is a data bus consisting of a primary and a redundant coaxial cable for digital transmission through the Space Station. Twisted wire pairs are used for secondary buses.

Important Data Distribution Subsystem characteristics are:

- Digital control of all subsystems.
- TDM/FDM of digital data in a half-duplex mode.
- Standard interface hardware is made feasible by the standard interface word.

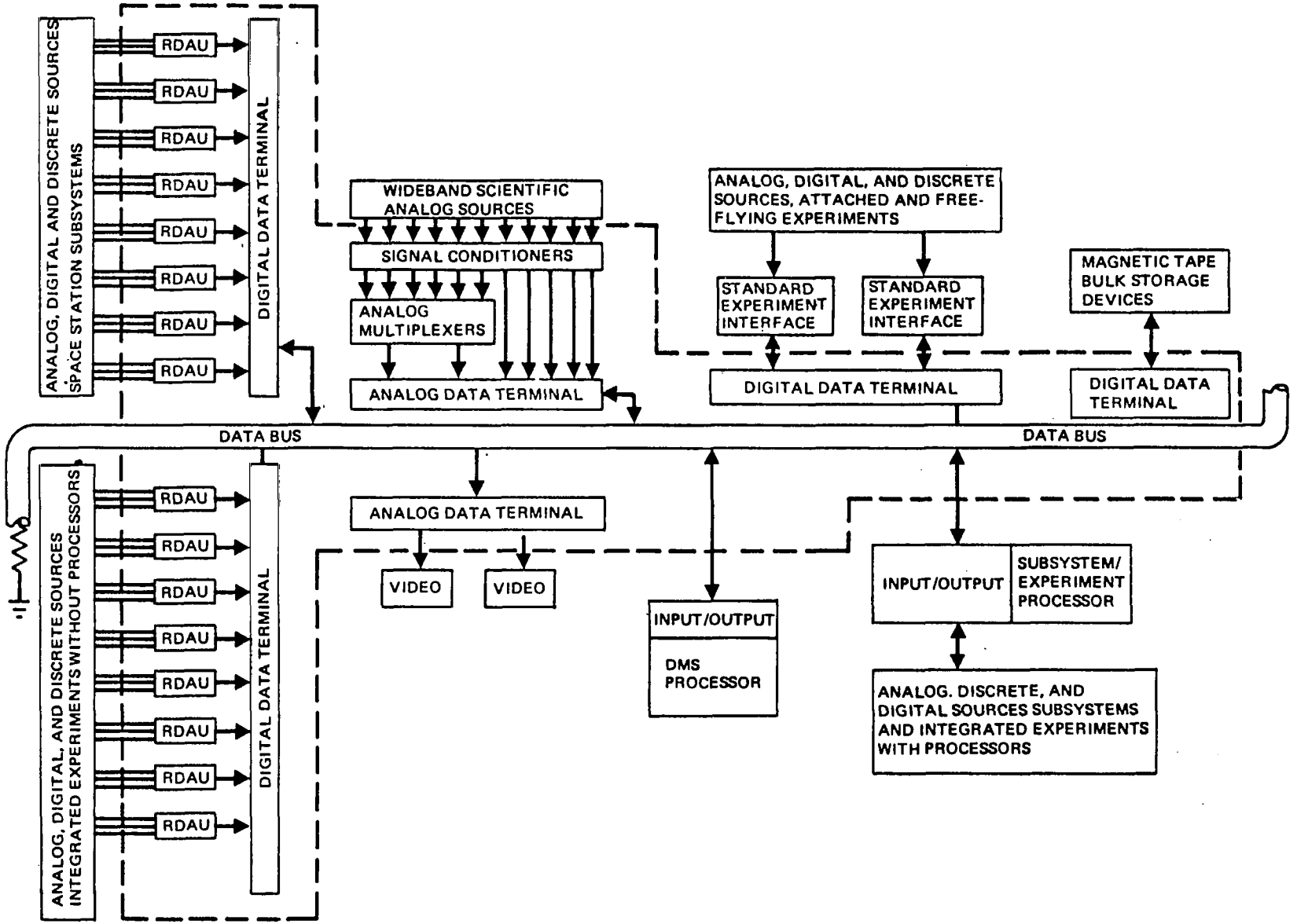
Within the Digital Data Distribution Subsystem, addressing and control is accomplished either under multiprocessor control or by manual control from an astronaut input console. This control is accomplished through the use of a standard interface word. It consists of 32 bits which provide 14 bits for terminal/channel addressing, a read/write bit to ensure that terminals will have access to the line in proper sequence, a 16-bit (half word) data/command word, and a parity bit. The form of the data word may serve to indicate the proper mode and gain for a remote data acquisition unit as commanded by the controller, a word count when a continuous frame of data is to be transmitted by a terminal, or simply acknowledgement by a write device that it has accepted the data and no errors have been found.

In addressing terminals/devices which are to output data, a one word-period delay will ensue, prior to that unit's transmission on the line, to allow time for a second command to be issued to a receiving device to accept data. The transmitting device will then respond with its terminal/channel address followed by data. A receiving device/terminal will wait to acknowledge message receipt until an end of message indication, message word count equal to zero, or an acknowledge request from the bus control unit.

If the addressed terminal cannot respond immediately with the requested data, the terminal originating the response will so inform the multiprocessor by raising a "not ready" bit. In this case, the multiprocessor will interrogate the terminal at some later time to get the desired data. Terminal-to-terminal data transfer is segmented via the use of subcarriers. This allows simultaneous terminal control of subsystem terminals by the DMS operations control computer and experiment terminal control by the experiment control computer. It also allows certain types of data to be transferred in a semi-controlled, uninterrupted data stream and divides the distribution system into somewhat more manageable segments. The net result is a reduction in terminal-to-terminal data rates, response-time requirements, and the size of individual buffer storage devices.

Data, entirely digital in form, is time multiplexed onto a specified carrier frequency (or frequencies) which is, in turn, frequency multiplexed with other carriers onto the common data bus system. The exact configuration of the bus in terms of number of lines, partitioning of data types, and carrier frequency allocation involves consideration of total Space Station data requirements and has not yet been established.

(2) Data acquisition. The Data Acquisition System (DAS) is controlled by the DMS multiprocessor and is interconnected by the data bus. Figure III-B-4(1) shows the selected subsystem concept. All information sources are coupled to the data bus through one of two types of terminals (analog or



DESIGN REFERENCE MODULE-DATA ACQUISITION SUBSYSTEM

FIGURE III-B-4

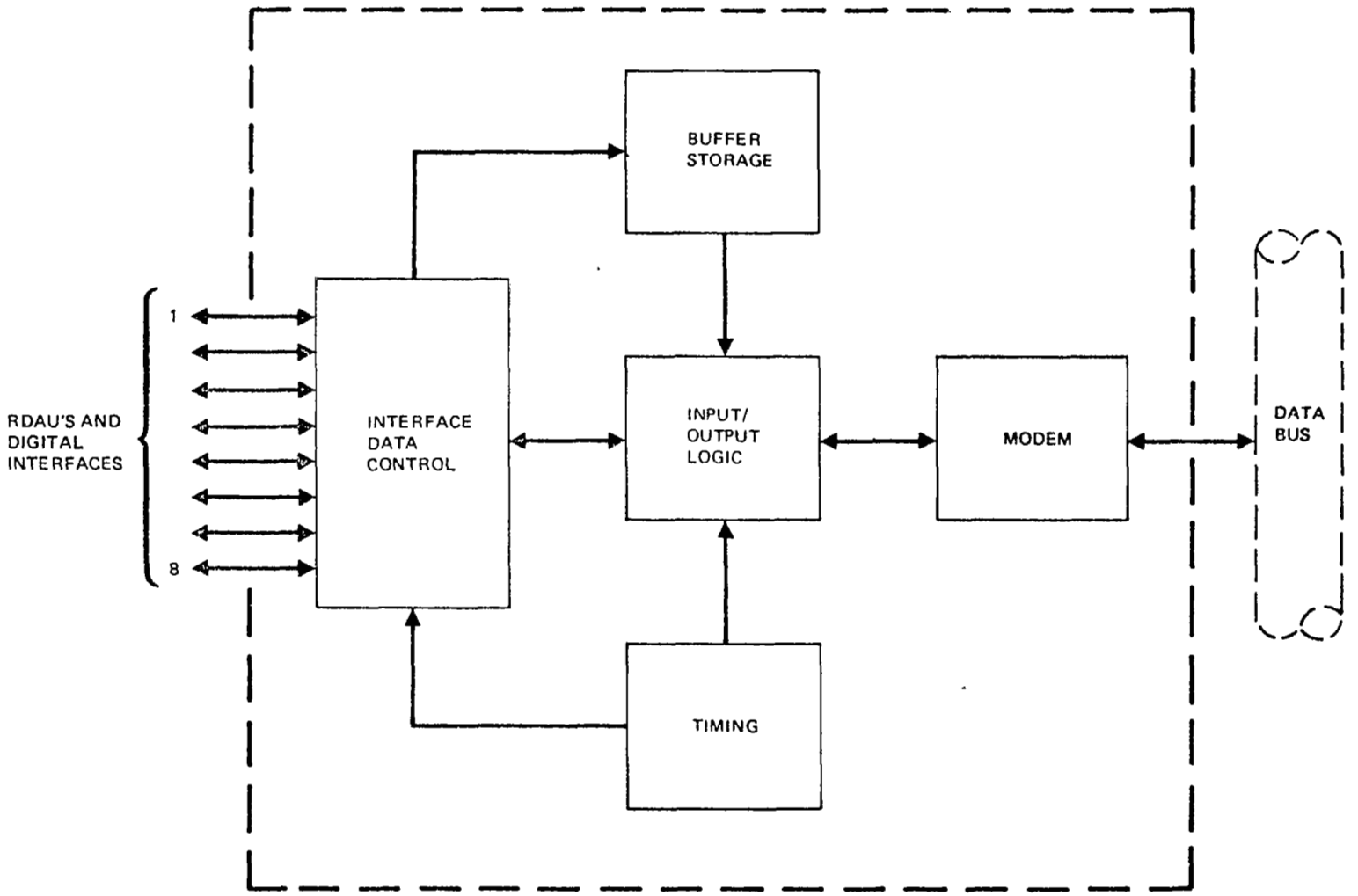
digital) which are addressable by the DMS processor. The standard digital control word is the means of control. Each terminal can handle eight four-wire interfaces. The number of inputs to a digital terminal is expandable to 512 by connecting a Remote Data Acquisition Unit (RDAU) to each input. Each RDAU will accept up to 64 analog and/or 8-bit digital signals and will accept formats transferred from the computers.

The system is activated by a control word from the DMS processor. The addressed terminal will then place its data on the data bus and the DMS processor can either accept it or, through additional control words, route it to another subsystem/experiment.

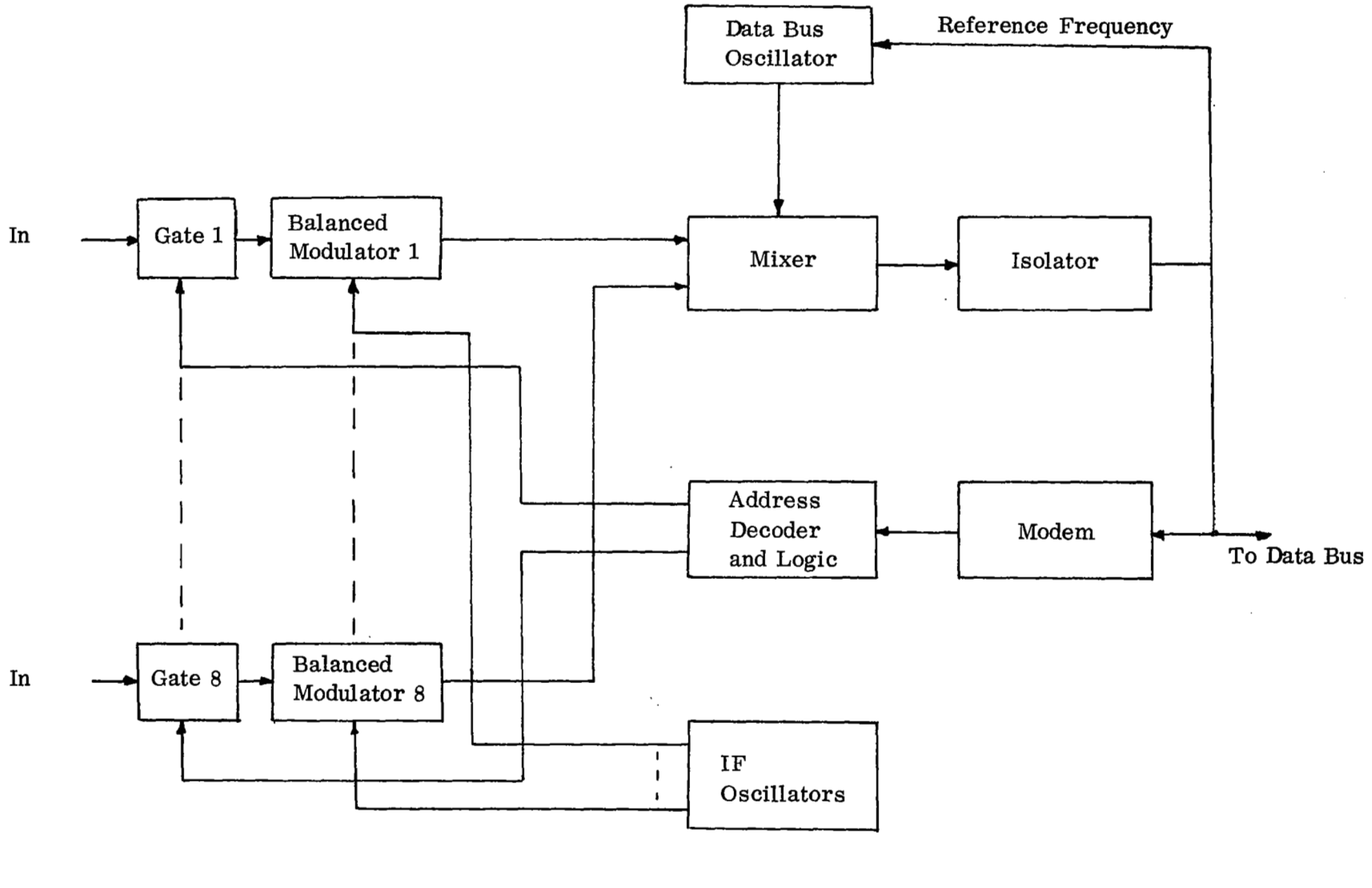
(a) Digital Data Terminal. The device which provides the data bus interface is the digital data terminal and is shown in Figure III-B-5(1). The terminals contain modulator/demodulators (MODEMS), input/output logic, and interface data control logic. They also contain buffer storage to accommodate irregular data rates from interfacing equipment and to allow assembly of data for block transfer onto the bus. This terminal is designed to handle eight standard four-wire interfaces with individual bit rates of one million bits per second (Mbps) or less. Each interface is isolated to prevent propagating a data source failure to another interface. The digital MODEM interfaces with the data bus and isolates the data bus from the terminal electronics to prevent electrical loading in either direction. Clock logic is modular and divides the data bus clock to whatever frequency or frequencies that are required by the individual interfaces. Buffer storage is also modular and may be provided as a single quantity unit for the terminal, or expanded/contracted for any or all interfaces individually. This feature allows storage to be tailored to a particular interface data rate. The DMS processor can efficiently utilize this storage by sending a control word to cause the terminal to sample its inputs, and, at a later time, send another control word to request a data dump.

(b) Analog data terminals. The modular analog terminals operate in an amplitude modulation (AM) configuration (see Figure III-B-6(1). Each input is raised to a different intermediate frequency (IF). The IF signals are then shifted to a frequency spectrum compatible with the data bus. Any or all analog inputs may be selected (via a control word from the DMS processor) through the address decoder and logic for simultaneous output. The analog output is placed directly on the data bus through an isolator.

(c) Remote data acquisition unit. The RDAU (see Figure III-B-7(1)) performs the following five functions:

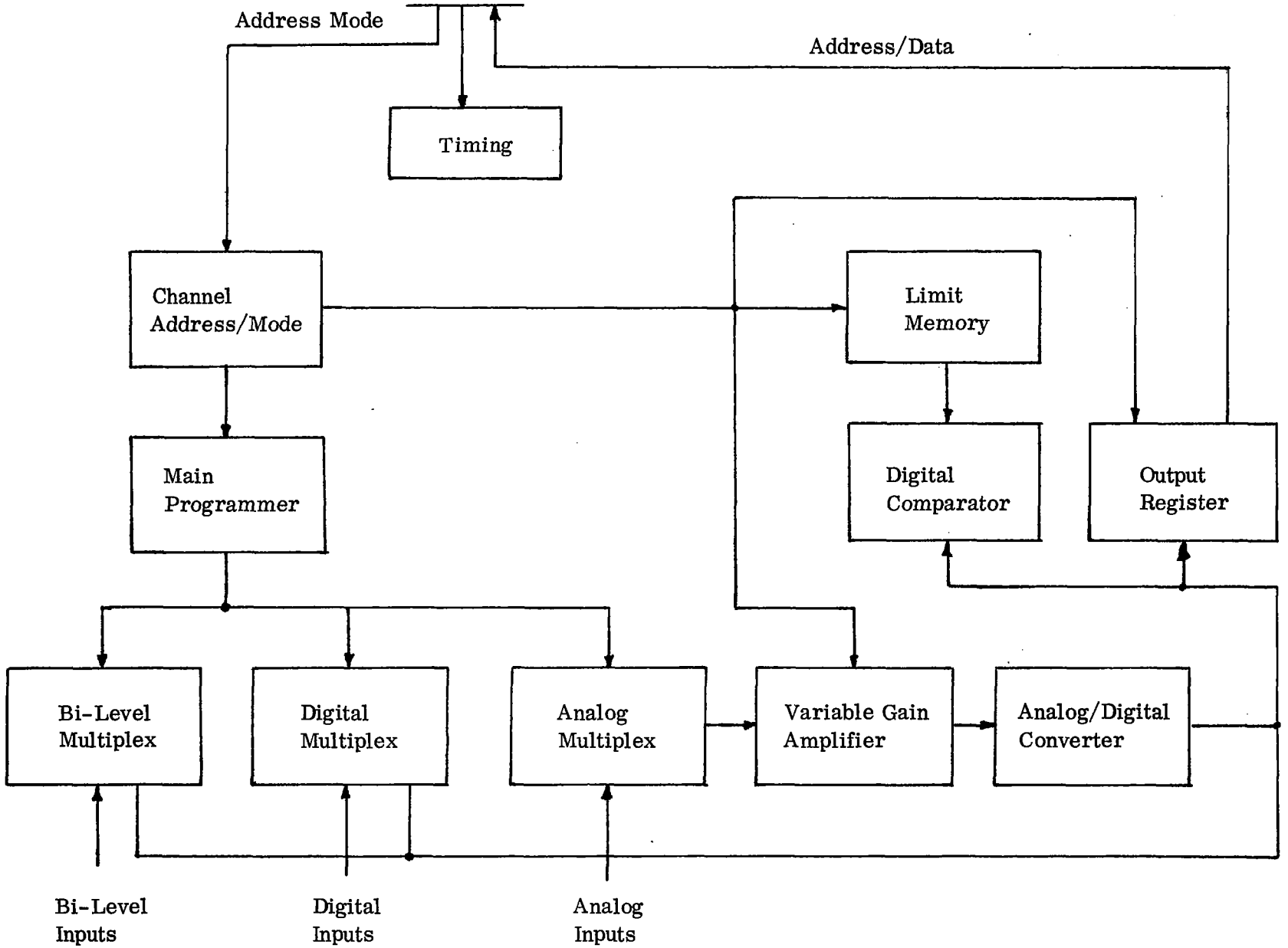


DIGITAL DATA TERMINAL
FIGURE III-B-5



ANALOG TERMINAL

FIGURE III-B-6



REMOTE DATA ACQUISITION UNIT
FIGURE III-B-7

- Signal conditioning,
- Multiplexing,
- A/D conversion,
- Limit checking, and
- Timing.

Signal conditioning is accomplished through a programmable gain amplifier or both a programmable gain amplifier and a preconditioning (ahead of the multiplexer) network.

The 64-channel multiplexer can accommodate selected combinations of analog and digital inputs.

The analog to digital (A/D) conversion bit rate equals the clock rate of the RDAU. The A/D converter output is digitally compared with high and low limits extracted from a self-contained local read/write memory. If the measured parameter exceeds either limit, the return data is flagged so the DMS processor is aware of any failures or out-of-tolerance conditions (check-out functions).

d. System reconfiguration. The computer subsystem is a physically distributed computational configuration. The DMS multiprocessor main memory modules, and auxiliary memory units are on Deck 3. The experiment multiprocessor, main memory modules, and auxiliary memory units are on Deck 1. This distributed organization allows for graceful degradation and a complete backup system to be available in the event that one multiprocessor becomes completely inoperative. This backup capability will be used in the following manner:

- The experiment multiprocessor shall contain a count-down timer (watch dog) which must be reset by the DMS Multiprocessor before the timer reaches zero. There will also be a buffer of critical parameters contained within the experiment processor that will be updated by the DMS computer at the same time as the timer reset. In the event that the DMS computer does not reset time before zero count, the experiment computer takes over the job of the DMS computer (the master executive is always in the experiment processor's main storage) using the table of critical parameter as an initial data base.

- Each auxiliary memory unit group will contain all the programs that are required by that particular multiprocessor. The programs required for main store will read in from auxiliary store (in a nondestruct manner so that a complete program set always resides in auxiliary memory) as needed. Additions or changes to the onboard program set can occur via up-link or crew-initiated action (1).

- Reconfiguration at lower system levels has not been studied in detail at this time. The current thinking is to isolate the fault to a Line Replaceable Unit (LRU) and inform the crew of its failure. The procedure, then, is to replace the failed unit with a stored spare. It is a safe assumption that on-line, or at least dormant, redundancy will exist in critical subsystems such as Guidance, Navigation and Control (GNC) and life support systems where human response would be a limiting factor. At this time there is little lower level detail as to the extent of redundancy to be incorporated for dynamic reconfiguration.

C. Executive System Functional Requirements

1. General. The purpose of this section is to outline the functional requirements of an executive control program for the space station. The preceding sub-section has described the hardware associated with the computer systems as presently envisioned. The following sub-section will relate hardware and space station mission requirements to yield the executive functional requirements.

Executive functional requirements are developed through a five step procedure. The steps are:

- Major on-board space station functions are enumerated,
- The computer application programs that will be developed to implement these functions are illustrated,
- The DMS multiprocessor executive functional requirements are outlined,
- The GNC executive functional requirements are discussed, and
- The Biomedical executive functional requirements are presented.

The functional requirements reflect a meld of performance demands and hardware capabilities. Neither performance nor hardware is well enough defined at this time to draw an isomorphic correspondence between them and the executive functional requirements. However, there is sufficient information to determine major functional requirements and to define features required to support those functions.

The objective of the executive is to control, direct, and serve the application programs in fulfilling the total Space Station mission functions. To uncover the functional requirements of the executive control software, we may block out the functions of the Space Station and the functions of the application programs (these functions have been considered to some detail under NASA contracts (4)). If the functions and features of the computing requirements can thus be established, the functional requirements for executive software control can be defined.

a. On-board functional requirements. The on-board functions to be supported by the computer complex for the Space Station mission are:

(1) Checkout.

(a) Preflight.

- Mode control.
- Subsystem monitoring.
- Status display.
- Data acquisition.
- Calibration.
- Component substitution.
- Stimuli control.
- Diagnostic.
- Fault isolation.
- Failure effect intelligence.
- Limit check.
- Fault/tutorial display.
- Data/status summary.

(b) Operations.

- Mode control.
- Subsystem monitoring.
- Status display.
- Data acquisition.
- Trend analysis.
- Data storage and retrieval.
- Calibration.
- Component substitution.
- Stimuli control.
- Diagnostic.
- Fault isolation.
- Failure effect intelligence.
- Limit check.
- Fault/tutorial display.
- Inventory.
- Data/status summary.

(2) Flight operations.

(a) Manned.

- Subsystem status display.
- Overall subsystem control.
- Activity scheduling.
- Configuration control.

- Personnel/skill scheduling.
- Inventory management.
- Operational procedures.
- Emergency procedures.
- Data acquisition.
- Communications center.

(b) Unmanned.

- Subsystem status display.
- Overall subsystem control.
- Configuration control.
- Inventory management.
- Operational procedures.
- Emergency procedures.
- Data acquisition.

(3) Training.

- Free-flying module operations refresher.
- Rendezvous refresher.
- Reentry refresher.
- Emergency procedure refresher.

(4) Network.

- Scheduling.
- Verification.
- Checkout.

- Acquisition.
- Validation.
- Compression/decompression.
- Formatting.
- Distribution.
- Ground commands.

(5) Experiments.

- Data acquisition.
- Decommutation.
- Format.
- Conversion.
- Reduction.
- Edit.
- Accumulate.
- Storage.
- Retrieval.
- Analysis.
- Archival.
- Summation.
- Distribution.
- Image processing.
- Control.
- Antenna pointing.

(6) Engineering data management.

- Process OCS derived summary data.
- Model subsystem specified performance.
- Performance evaluation.
- Performance enhancement.
- Biomedical monitoring.

b. Application tasks. Application programs to fulfill the onboard functions can be listed in functional groupings. Each function (and explanatory subfunction) is listed according to the major onboard features they supervise. The features by function group are:

(1) Mission control.

(a) Checkout supervisor.

- Onboard checkout backup.
- Onboard checkout evolutionary development.

(b) Flight operations supervisor.

- Subsystem status.
- Configuration control.

(c) GNC supervisor.

- Tracking data processor.
- Numerical integrators.
- Vector analysis.
- Trajectory determination.
- Evaluation calculations.
- Station acquisition.

(d) Training supervisor.

- Space station operation.
- Experiments operation.
- Onboard checkout operation.
- Emergency procedures.
- Ground support.

(2) Functional experiments control. (Experiments supervisor.)

- Storage.
- Archival.
- Analysis.
- Retrieval.
- Image processing.
- Summation.

(3) Engineering evaluation. (Engineering data supervisor.)

- OCS data processing.
- Performance evaluation.
- Performance enhancement.
- Subsystem modeling.
- Test data processing.

(4) Information facility. (Information supervisor.)

- Mission planning.
- Scheduling.
- Logistics/inventory.
- Experiment data distribution.
- Engineering data distribution.

c. Executive support. Executive control programs are required to provide resources to the application programs that satisfy mission functions. There are limited resources, for example ALU's, main memory, channels, and data bus; and these resources must be allocated among the application demands in such a way that all mission functions are satisfactorily supported.

Two policies are noted (and they are basic to executive systems): (1) Uniform identification header will be appended to every task in the system, and (2) the executive will interface all communications external to the task.

(1) A uniform header will contain all the attributes necessary to initiate scheduling of a task on any of the system processors. The header will contain the initial task priority, core requirements, execution time estimates, facilities required, dead-line, start-time, and other pertinent information. Pertinent information will include, for example, the type of microcode to be executed (if available in the hardware). Thus, a common-high-level language is encouraged which produces code executable on several computers optimized by the microcode most suitable for the application.

(2) The executive will provide all interfaces to the task. Tasks are not dependent upon individual computer system resources and are free to access other external subsystem resources. To the scheduler each of the subsystems (propulsion for example), whether demanding or responding to requests, is a resource to be allocated. All programs can be operated through a common set of procedures regardless of their individual natures. The uniform header will permit task-data sets to surrender personality, but maintain coded attributes and traits.

The configuration discussion (III. B) describes four identifiable computer complexes. The biomedical subsystem contains a computer dedicated to biomedical experiment functions. The GNC subsystem contains a dual-redundant computer that is dedicated to GNC functions. The DMS computer is a multiprocessor handling data management. The second multiprocessor is identified with experiments and onboard checkout. All computer ALU's have access to the same subsystems via identical data bus interfaces.

The multiprocessors specified for data management and for experiment/checkout are identical. All interfaces are identical and all subsystems are within reach of each ALU of each multiprocessor. This is not to say that advanced concepts will not be implemented. On the contrary, it is quite possible that several very different object procedures might be implemented in microcode and executed on the same processor concurrently.

The DMS executive will supply the required resources to the application programs and direct the overall computer system scheduling activities. The ALU's, the interfaces, and the subsystems are resources to be managed. The interface code and command procedures will be independent of which multiprocessor is directing the task.

A single baseline executive is required for the multiprocessors. This executive will be implemented in each multiprocessor. The header identification for each task will identify the control programs to be used. For example, certain experiments may require unique I/O handlers. These special handlers would be brought into a segment overlay portion of the executive and executed. Any special subsystem functional requirements would be attended to by the handler tailored to those functions. To proceed a step further in our example, suppose that a multiprocessor failure has occurred and that DMS functions and experiments are being conducted on one multiprocessor. Further, let us postulate that different bulk storage access methods are used by the competing tasks. The header identification can be used to see that each task is linked to the proper access method. Thus, the backup computer has the capability of applying code that is common to the two systems and code that is peculiar to either of the systems.

Intuitively, it appears that a single executive for the two multiprocessing systems will be less expensive to develop than two independent multiprocessing/multiprogramming systems. The baseline executive, further, will have a higher degree of confidence testing and validation. The overall product will be more resistant to obsolesence due to its wider applicability and adaptive nature.

The dual redundant computer described (III. B) will be dedicated to GNC functions. The dual-redundancy will be entirely for backup. Multiprocessing will not be available.

The GNC executive will receive commands from the DMS executive. The commands may originate as astronaut manual control, stored mission timeline, or commands telemetered from the ground mission control. The prime GNC executive function will be the sequencing of application programs to execute the manuvers identified by commands, self-testing, and subsystem fail-safe support.

The Biomedical executive has a limited number of functions to support. However, a great deal of flexibility is desired in the area of manual initiation, modification, and addition of task software. Most of the presently considered biomedical functions require an astronaut in the loop. The GNC executive is concerned with acquiring sensor data for the application programs and providing displays and manual capabilities for the astronaut.

Each of the executives will be discussed to present their functional requirements. The DMS Executive will be adaptive for DMS multiprocessor or experiments/checkout multiprocessor. A GNC Executive functional requirements is developed for the dual redundant computer. The biomedical subsystem is supported by a Biomedical Executive functional requirements description.

2. DMS Executive. We have reviewed the space station functions, the application software functions, the hardware configurations, the combination of functions to be performed, software tasks to be supported, and hardware required executive features that, in turn, specify the required functions. The features of the DMS Executive for multiprocessors and their functional requirements are:

a. System control.

(1) Scheduling.

- Task course scheduling. Course scheduling occurs when a new task is initiated, a change in task attributes occurs, or there is a change in hardware or software state.

- Device inventory. Device inventory builds and maintains configuration maps and tables of

facilities available for assignment and facilities assigned to each task.

- Command interpreter. The command interpreter checks format of headers (tasks) and input control stream. Decodes for scheduler.

- Dynamic allocator. The dynamic allocator assigns main memory and ALU time to the tasks being processed.

- Dispatcher. The dispatcher dispatches ALUs to tasks.

(2) Task-to-task calls. Inter-task communication is made via supervisor calls.

(3) Parallel task execution. This refers to the capability of a running task to initiate the execution of another task.

(4) Program timed request control.

- Mission timeline decode and interpretation.

- Real-time clock services will be required.

- Interval timers and watch dog timers will support several functions.

(5) Priority interrupt processing. Interrupts are closely identified with scheduling functions. The hardwired priority levels may determine servicing order. The nature of the function will determine the priority level assignment.

(6) Task termination/suspension.

- Task termination. Completed tasks are removed, files closed, main storage returned to the dynamic allocator, and facilities released to the device inventory routine.

- Task suspension. The function is required to temporarily suspend individual task processing. This possibly includes the capability of rolling-out portions of a task onto mass storage (to free main memory for a high priority user).

(7) Mass storage allocation. Scratch and permanent files are assigned space as files are opened or expand. Mass storage handlers are required (access method) to interface users. Multiple access methods may be required to support various functions (experiment vs. display requirements, for example).

(8) Fast buffer assignment. Some portion of high speed memory is required for temporary buffers. These buffers are seized and quickly released by the fast buffer assignment module.

(9) Overlay manipulation. A comprehensive overlay structure will be required due to the size of application tasks in a multiprocessing structure. A segment directory will be supported.

(10) Memory protection. The security of program files, tasks, and data is basic to the reliability and safety of the entire system. The hardware design will constrain the procedures necessary for maintaining protection. For example, memory organization protection keys can be hardware implemented; memory protection passwords can be software implemented.

(11) Request processor. A module to process all supervisor calls. It decodes requests for services and links to the executive routine specified. For example, an acronym such as EXIT would generate a trap or interrupt to the request processor which would link to the termination procedure.

b. Input/output control services (IOCS). Input/output requests will be processed by the supervisor. This may be accomplished by macro call or micro coding. The functions required are:

(1) Request queueing. If the requested service is not available, it is added to a priority queue.

(2) I/O handler. The I/O handler establishes the connection between the device and the channel.

- Initiator. The initiator determines that the task can legally access the information and satisfies additional constraints such as buffer area requests.
- Continuator. The continuator supports the data transfer with error recognition, recovery procedures, and termination procedures.

(3) Data bus control. The data bus scheduler will poll or capture (via interrupt) the common data bus. The scheduler will implement a control algorithm to maximize the effective use of the bus.

c. System monitoring. Functional requirements include support to output, record, and maintain status of anomalies detected by the checking procedures. The following specific functional requirements must be satisfied.

(1) Alarm history maintenance. Status tables must be initiated and updated. These tables will indicate:

- Present alarms in system.
- Present alarm notification/recognition.

(2) Alarm output control. This requirement will be met by logical procedures to:

- Determine alarms requiring output.
- Alarm message formatting.
- Alarm message output.

(3) Alarm history recording. This requirement indicates the following:

- Upon demand, the tables of system alarms will be interrogated.
- Alarm history output messages will be formatted.
- Alarm history message will be output.

d. Computer system performance monitoring and logging. Functional requirements are satisfied by a combination of hardware and software. Meeting the functional requirements can be critical to system design.

Hardware and software triggers must be provided to gather the following statistics:

- Number of executive requests.
- Number of bus transfers.
- Average bus data volume.
- Percent ALU utilization.
- Percent main memory utilization.
- Error frequency count.
- Failure count.

e. Error recognition. Functional requirements will be satisfied by a combination of hardware and software techniques. Requirements can be categorized as follows:

(1) Hardware. Requirements include:

- Fault interrupts.
- Parity generated interrupts.
- Illegal attempts to execute privileged instructions.
- Protect violations.
- Overflow/underflow.
- Other techniques TBD.

(2) Software. Requirements include:

- Longitudinal record check for parity.
- Testing access keys for protected files.
- Testing format of system calls.

- "Busy" returns.
- Analysis of data formats where possible.
- Diagnostic countdown will be required to detect malfunctions such as endless loops or stalled peripherals.

f. Failure detection and recovery. Functional requirements will be met as follows:

(1) Fault interrupts. Where possible hardware faults will generate an interrupt. The monitor system will be responsible for directing execution of the appropriate module to perform recovery procedures.

(2) Limit checks. The monitor system will cause limit checks to be performed on critical variables. Limit violations will indicate malfunction and appropriate recovery routines will be invoked.

(3) Errors. As is discussed above, there are error detection procedures. Some of the errors are generated by hardware failures. Where this is the case, error detection and failure detection are identical.

(4) Transient failures. Requirements are that transient failures must be separated from "hard" failures. Different recovery action is taken in each case. For the case of transient errors, the appropriate recovery action may be merely to record the fact that the failure has occurred. For hard failures, recovery procedures involve graceful degradation or reduced system capability until the failed item is replaced.

(5) Restart after failure. Recovery requirements include:

- Initialization. For certain classes of failures, this is the only alternative. It is a cold start initialization of the failed sub-system or system.
- Checkpoint. This functional requirements is met by taking snapshots of the system during operation. That is, periodically, critical parameters are sampled and stored. In the event of failure, the system is reconstructed by returning to the previous checkpoint and restarting using values of critical parameters at that time.

3. GNC Executive. A system of executive control programs will be developed to support the GNC application functions. The executive software will include the following functions:

a. System control function. The GNC control function is limited to scheduling made up of several control modules:

- Task coarse scheduling. Makes a new task or module available for execution. Entries to the coarse scheduler may be by interrupt, time control or DMS command.

- Facilities inventory. Maintains the status of all subsystems interfaced or required by GNC functions.

- Dynamic memory allocator. Assigns memory to application tasks. Prepares switch list to direct ALU time.

b. Task-to-task calls. Interfaces interaction between application programs. For example, updating ephemeris tables from new calculations or from ground transmissions.

c. Program timed request control. Support for real-time clock and mission timeline demands.

d. Priority interrupt processing. Interpretation of interrupts and signalling affected modules.

e. Task termination/suspension. Wrapping-up completed programs (return resources to facility inventory, memory to the memory allocator). This is also required in order to temporarily defer a program's execution.

f. Overlay manipulator. Some of the larger application functions may require overlays (trajectory calculations, for example).

g. Request processor. Decodes and schedules routines to satisfy inputs from the DMS or ground control.

h. I/O operation and control services.

- I/O request scheduling. Conflicting requests for data bus and subsystems will be queued until the resource can be allocated.

- I/O control. Modules will be required to initiate I/O transfers and to support transfers to completion.

- I/O device handlers. The lowest level interface handlers will be required to perform all hardware-related functions (such as error recognition and retry).

- Data link control. Due to the common bus technique, the GNC system could directly utilize the data link or it could use the DMS services for this function.

- Communication handling. The executive will provide communication interface to all other systems and subsystems.

i. System monitoring. The configuration will monitor itself and record any alarms. In the event of a hard failure, the failing redundant processor will be determined and replaced. In the event of complete loss during critical maneuvers, a multiprocessor will be interrupted for backup.

j. Performance monitoring. This function requires maintenance of at least the following:

- Number of executive requests.

- Number of bus transfers.

- Average bus data volume.

- Percent of ALU utilization.

- Percent main memory utilization.

- Error frequency count.

- Failure count.

k. Error recognition and recovery.

(1) Program error detection.

- Incorrect I/O request.

- Busy return.

- Incorrect format.
- Non-convergent process.
- Error recovery sequences.
- Parity error recovery sequence.

(2) Failure detection and recovery.

- Hardware diagnostics.
- Fault isolation.
- Spare switching.
- Degraded mode operation.
- Restart sequence.

1. Configuration control. Units of the configuration will be removed by failure or for maintenance. The executive will maintain knowledge of the viable configuration.

4. Biomedical Executive. A dedicated computer is provided for control of biomedical data acquisition, control and display. The Biomedical Supervisor will require real-time capabilities, but not all of the sophisticated operating system devices employed in the multi-processors. Almost all of the experiments and analyses require manual intervention to set up and perform. That is, most require an astronaut as part of the experiment (5).

The Biomedical Supervisor must provide support to satisfy the following functional requirements.

- On-board checkout.
- Data acquisition.
- Data management.
- Monitor system services.

As is seen above, the functional requirements dictate a real-time management information and control system. This system class is characterized by:

- Stringent response time requirements.
- Many concurrent activities.
- Varying importance of activities.
- Slow speed (relatively) peripheral devices.
- High degree of interaction of activities.

Experience has shown that the following capabilities are necessary to satisfy requirements with the above characteristics:

- Priority interrupt structure.
- Priority level independent of priority hardware structure.
- Capability to selectively inhibit interrupts.
- All communication with I/O devices via the supervisor.

- Privileged mode of operation (i.e., privileged instruction repertoire).
- Memory protect facilities for task and data protection, and
- Variable program priority level (dynamically variable in real-time under application, executive, or event occurrence control).

Functional requirements are derived from the following sources:

a. Hardware.

(1) Multiplexer control. Functional requirements are to handle the interface with sensors and output devices.

The following requirements exist:

(a) Analog/digital scan.

- Sensor selection.
- Input value.
- Convert to digital, if required.
- Limit check.
- Alarm anomalies.

(b) Analog/digital output.

- Select output device.
- Transmit output value after converting to digital value, if required.
- Output checking and alarming may be required.

(2) Mass store control. Functional requirements are:

- Analyze requests.

- Determine addresses in main store and bulk store.
- Allocate storage, if required.
- Perform transfer.
- Perform rudimentary error check.
- Perform appropriate exit.

(3) Display control. Functional requirements are:

- Initiation.
- Update.
- Formatting.
- Possible calculations TBD.

(4) Data bus control. Functional requirements are to provide support for receipt and transmission of messages via the data bus. The following specifics are required:

(a) Message receipt initiation.

- Buffer assignment.
- History table initiation for message.
- Error check.

(b) Message receive continuation.

- Error check.
- Data storage.
- Message termination control.
- Maintain count of messages received.

(c) Message transmit.

- Initiate control codes.
- Continue upon receipt of acknowledgment.
- Accounting procedures for message termination.
- Maintain control count of messages sent.

(5) Operator interface control. Functional requirements are:

- Detect operator request.
- Analyze request.
- Respond to request analysis.

(6) Input/output control. Functional requirements are:

(a) I/O initiation.

- Request analysis.
- Buffer assignment.
- Formatting.
- Device selection.
- Transmission of initial character.

(b) I/O continuation.

- Character output.
- Logic associated with termination conditions.
- Special criteria may be required for message analysis.

(7) Resource allocation. Functional requirements are:

(a) Competing demands. Resolution of request conflicts for the following resources:

- Mass store.
- Main store.
- Data bus.
- Displays.
- Magnetic tapes, if present.
- Other peripheral devices.

(b) Accounting for available resources.

(8) Interrupt service. Functional requirements are:

- Save critical registers and main storage for interrupted program.
- Analyze cause of interrupt.
- Invoke appropriate servicing modules.
- Restore pre-interrupt conditions.

(9) Error detection, analysis and recovery. Functional requirements are:

- Determine experiment status.
- Isolate malfunction.
- Determine remedial action.
- Cause remedial action to be displayed.

(10) Alarming. Functional requirements are:

- Maintain tables of alarms present and output.

- Format alarm messages.
- Cause output of alarm messages.
- Record alarm acknowledgment.

b. Applications support. The following functional requirements are derived from the interaction of the hardware architecture and the applications requirements placed upon the hardware:

(1) Experiment support. Functional requirements are:

(a) Human interaction capability.

- Display functions.
- Input console functions.

(b) Data processing capability. Large volumes and rates of data are expected for some biomedical functions. Supporting application functions include:

- Information attrition.
- Data reduction.
- Mass store functions.
- Magnetic tape functions.

(2) File management. Functional requirements are:

- Address segment by name.
- Storage allocation by system.

(3) Inventory calculations. Executive functional requirements are satisfied by considerations already discussed (file management and human interaction/control).

(4) Task execution. Functional requirements are:

- Priority request queue control.

- Complete resource allocation and linkage.
- Initialize program module.
- Cause module execution.

(5) Task scheduling. Functional requirements are:

- Append requesting module to appropriate priority thread.
- Perform necessary housekeeping to maintain thread integrity.

(6) Task termination/suspension. Functional requirements are:

- Remove module from request thread.
- Perform appropriate housekeeping to maintain thread integrity.
- Program control is relinquished for task suspension, but the module is not removed from the request thread.

(7) Queue maintenance. Functional requirements are:

(a) Lists.

- Initialize.
- Append, insert or remove items from list.
- Perform logical operations associated with termination.

(b) Buffers.

- Initialize, i.e., determine size and location.

- Read/write data into buffer.
- Perform logic associated with termination.

(c) Tables. Current values of selected system parameters such as tuning constants, masks, semaphores, assignments, etc., must be maintained.

(8) Management information. Functional requirements for support have been satisfied by previous discussion. Actual management information system is an applications function and is not studied in this report.

(9) Summary. As is seen above, the functional requirements for the Biomedical Executive are similar to functional requirements for a real-time management information and control system.

(a) The monitor must provide support for:

- High data rate experimentation.
- Data call-up services.
- Astronaut computer system interaction.

(b) Functional requirements for support of these functions are derived from the following considerations:

- Hardware.
- Applications requirements-hardware interaction.

D. Executive System Performance Requirements

Executive system performance requirements are not well defined at this time. However, some performance constraints can be found. There are two sources of performance requirements; internal and external functions. The final measure of performance is the capability to support the required tasks within the constraints of accuracy, timing, control, flexibility, and reliability. The most often stressed performance requirement concerns reliability and configuration control. At least one level of fail operational with the second failure fail-soft for mission critical elements are firm performance standards.

1. System (Internal) Performance. The Software Requirements Document (4) specifies system performance as the capability of processing information and displaying/transferring data to satisfy system functions. Performance requirements for internal functions should take the form of specifications for individual elements:

- Available resource assignments (i.e., ALU time, main memory, bulk store).
- Accuracy.
- Input/Output rates and volumes.
- Frequency of execution.
- Duration of execution.

The sum of these individual performance measures should be the overall system performance requirements. System performance is measured also in a larger sense for:

- Responsiveness.
- Interface difficulty.
- Flexibility.
- Maintainability.
- Reliability.
- Modularity.
- Overload handling capacity.

2. Subsystem (External) Performance. Subsystems (biomedical, experiment, GNC, etc.) have received a great deal of study since they are building blocks that interface the system with the real world. Our concern is not with subsystem performance per se, but with performance forcing factors at the executive interface.

Some of the factors that impact heavily upon performance have been estimated. For example, data rates and subsystem interfaces produce loads upon executive control programs. A minimum performance requirement is thus established that executive services be capable of satisfying these demands.

The executive functions of hardware/software interface, resource allocation, checkout and application program supervision are performance dependent upon several factors. Some of these factors are: (1) interface functions, (2) subsystem control and data rates, and (3) execution speed.

a. Supervisor - hardware interface. Major interface functions are identified in Table III.D.1 as extracted (4) from Volume III, Software Requirements Document. The interface points between functions/sensors/subsystems and on-board executive supervisors are supported by I/O handlers and interface routines. Data captured from the interface handlers is made available to the on-board supervisors (as marked).

Digital data interfaces are further defined (6) in Table III.D.2 excerpted from Appendix A, Data Acquisition Subsystem Trade Study. The sources, data rates, data descriptions, and the quantity of each are listed.

b. Subsystem control and data rates. Considerable work has been done toward definition of data requirements (collection and storage) for experiments. A great deal of historical information is available on GNC data requirements. The following (6) is a summary of the available current hardware performance requirements which will impact the executive software design.

(1) Data acquisition.

- Signal conditioning. Conditioning of input signals ranging from 5 volts to 20 millivolts, 4 gains (1 to 250) programmable by command.
- Multiplexing. Sequential and random access of channel inputs.
- Formatting. Capable of operating under software control.

	Interface Functions	Onboard Supervisor: Checkout	Experiments	GNC	Flight operations
A.	Vehicle Sensors				
	1. Space Station Performance	X			X
	2. Space Station Status	X			X
	3. Experiments Data		X		
	4. Navigation			X	
	5. Acceleration			X	
	6. Rendezvous			X	
	7. Tracking			X	
	8. Balance			X	X
B.	Space Station Communications				
	1. Displays	X	X	X	X
	2. Indicators	X	X	X	X
	3. Switch/Keyboard Inputs	X	X	X	X
	4. Printer	X	X	X	X
	5. Terminals	X	X	X	X
C.	Vehicle Controls				
	1. Balance				X
	2. Subsystem	X	X		X
	3. Configuration				X
	4. Maneuvering			X	
	5. Target Acquisition		X	X	X
	6. Stimuli	X			
D.	Instrument Landing System (ILS)			X	X
E.	Automatic Landing System (ALS)	X		X	X
F.	Free Flying Module (FFM)	X	X	X	X
G.	Personnel	X	X	X	X
H.	Ground Control Uplink/Downlink	X	X	X	X

INTERFACE IDENTIFICATION

Table III. D. 1

QTY	DATA RATE	SOURCE	DATA DESCRIPTION
13	5.1 Kbps	Remote Data Acquisition Units (RDAU)	Subsystem mission Data
64	5.1 Kbps	RADU	Subsystem checkout data for Space Station Subsystems without dedicated processors.
1	0.76 Kbps	Subsystem Processor	Subsystem checkout data for Space Station Subsystems with dedicated processors.
6	167.03 Kbps	Integrated Experiment Processors	Scientific experiment data from Integrated experiments with dedicated processors.
10	5.60 Kbps	Integrated Experiments	Scientific experiment data from integrated experiments without dedicated processors.
10	5.1 Kbps	RDAU	Checkout data from integrated experiments without dedicated processors.
12	10 Kbps	Command Receiver	Digital commands to or from MSFN, orbital tug, DRSS, ALS/ILS. FF module (4), attached module (3), remote sub-satellites (2).
10	50 Kbps	Transponders	Tracking information to or from MSFN, orbital tug, DRSS, ALS/ILS, FF module (4), remote sub-satellites (2).
11	1000 Kbps	RF receivers or docking ports	Digital wide band data to or from MSFN, DRSS, FF experiment modules (4), attached modules (3), and remote sub-satellites (2).
2	50 Kbps	RF receivers	Digital narrow band data from orbital tug, ALS/ILS
1	5 Kbps		Astronaut EVA.

DIGITAL DATA INTERFACES TO DATA TERMINALS

Table III. D. 2

- Analog-to-digital. 8-bit accuracy.

- Limit checking. Capable of parallel bit-by-bit comparison of 8-bit digital words. Able to generate out-of-limit flag upon non-comparison of any given word.

- Address decoding. Provision for decoding terminal channel and word addresses.

- Timing. Programmable clock rates.

- Buffering. Modularly expandable. Storage periods (capacity divided by input data rate) in excess of bus polling periods or interrupt driving intervals.

(2) Data distribution.

- Addressing. Ability to address up to 256 independent devices. Transfer data at composite rate greater than 50 megabits per second. Ability to address up to 64 channels within a terminal. Provisions to include information transfer between data terminals under computer control. Provision to allow information transfer between data terminal and computer under computer control. Manual interrupt should acquire data bus in less than 1 second.

- Decoding. Control word to consist of 32 bits. Data words to consist of 16 bits.

- Error detection. Minimum receiver signal-to-noise ratio (S/N) of 13 db for analog data channels. Minimum receiver S/N of 29 db for video channels. Elemental bit error rate of 10^{-4} to 10^{-6}. Probability of undetected error less than 10^{-18}. Message blocks to be rejected if the bit error rate is exceeded within the message period.

(3) Data storage.

- Digital recording. Bit packing densities of 16×10^3 bits/inch/track. 28 tracks/recorder. Data capacity of 10^{10} bits minimum per recorder. Estimates are for state-of-the-art devices for 90 day storage of 7.97×10^9 bits per day.

- Video recording. Frequency response 4.5 MH_Z -3 db. Record time 3 hours minimum.

- Digital data retrieval. Maximum time required to locate a given starting address label to average 10 seconds.

- Image processing. Control and calibrate experiments. The following applications may be beyond practical physical limits (power, weight, and volume): image viewing, selection, and editing; manual enhancement, selection and filtering.

- Subsystem (computation) support. Continuous computation support shall be provided in the event of component failures. Capability shall decrease gradually in the event of failures. Redundant subsystem control will be physically separated by one space station deck. Capability shall be provided to replace the programs stored in volatile program store by programs from auxiliary storage. Capability for quick and delayed time program changes by ground personnel. Provision for selection of operating modes and data readouts by crewman. Access to DMS computer from both control and remote control console. Uplink capability of 120 kilobits per second digital data. Downlink capability of 120 kilobits per second plus up to 10 megabits per second on a time-shared basis.

- Digital data (total peak rates). 805 sources of mission data; peak rate 60 KBS. 3296 sources of experiment data; peak rate 22 MBS. 4450 sources of checkout data; peak rate 175 KBS.

- Digital data (total average rates). 804 sources of mission data 60 KBS. 3296 sources of experiment data 11.5 MBS. Since GNC data normally is processed within the GNC subsystem computer, checkout data will average approximately 115 KBS.

The data acquisition requirements have been summarized (6) by subsystem, Table III.D.3 (extracted from Appendix A, Space Station DMS Subsystem Requirements). The table illustrates loading factors that specify control program performance at subsystem interfaces for number of test points and for average data rates. These do not represent subsystem performance or total data acquired, but only that data acquired by the DMS.

c. Execution Speed. Speed of execution, main memory requirements, and auxiliary storage requirements have been estimated (5) for some major functions. These may not be performance requirements, rather performance estimates. However, they illustrate a speed to storage relationship that may be of design value when trading-off speed vs. memory optimization techniques (Table III.D.4, as taken from Appendix A1, Space Station DMS Subsystem Requirements).

The collection of available estimates that constrain performance will be of value in the executive design phase. These estimates will greatly assist in trading-off design concepts including data management, queue control and scheduling procedures.

SPACE STATION SUBSYSTEM DATA ACQUISITION REQUIREMENTS

Subsystem Name	Checkout Data Requirements		Mission Data Requirements	
	Number of Test Points	Data Rate bit/sec (Average Rate)	Number of Test Points	Data Rate bit/sec
Guidance and Navigation	500	60.5×10^3	95	7.6×10^3
Power	1000	16.89×10^3	200	
Reactor 350				
Solar Array and Batteries 450				
Distribution 200				
Data Management System	750	13.58×10^3	75	6×10^3
Communication	425	4.648×10^3	50	4×10^3
EC/LS	355	4.99×10^3	90	7.2×10^3
Structures	190	2.334×10^3	20	1.5×10^3
Attitude Control	470	6.1×10^3	85	6.8×10^3
Propulsion	550	66.73×10^3	100	8×10^3
Crew System/Docking	305		35	2.8×10^3
Total	4550	175×10^3	805	60×10^3

Table III. D. 3

FUNCTION	MAX SPEED (KOPS/SEC)	MAIN STORAGE (INSTRUCTIONS AND DATA) (K WORDS)	AUXILIARY STORAGE (K WORDS)
SUBSYSTEMS			
MASTER EXECUTIVE	60	20	--
ONBOARD CHECKOUT	65	20	58
NAVIGATION AND CONTROL	200	48	--
SUBSYSTEM CONTROL AND PROCESSING	100	30	--
MISSION SUPPORT	5	5	402
CREW SUPPORT	2	4	24
EXPERIMENTS			
EXPERIMENT SUB-EXECUTIVE	60	20	--
EXPERIMENT CHECKOUT	35	16	17
EXPERIMENT CONTROL AND PROCESSING	600	80	6,000
EXPERIMENT SUPPORT	5	5	350
TOTAL	1,132	681	6,851

COMPUTER SUBSYSTEM PERFORMANCE REQUIREMENTS (6)

Table III. D. 4

REFERENCES

1. Space Station Definition, Vol. V, Subsystems, Book 4, Electronics. MSFC-DRL-160 Line Item 8 (MDC GO605), McDonnell Douglas Astronautics Company, July 1970.

2. Space Station Phase B Study Results, GPSS/360 Data Flow Simulation Model. IBM Presentation, IBM Corporation, August 12, 1970.

3. Space Station Definition, Vol. V, Subsystems, Book 3, Information Management. MSFC-DRL-160 Line Item 8 (MDC GO605), McDonnell Douglas Astronautics Company, July 1970.

4. Space Station Development Definition, Vol. III, Software Requirement Document. MSFC-DRL-160 Line Item 18 (MDC GO544), McDonnell Douglas Astronautics Company, May 1970.

5. Space Station Data Management System Subsystem Requirements, Appendix 1. IBM Preliminary Documentation of Space Station Requirements, IBM Corporation, 1969.

6. Space Station Definition, Vol. V, Subsystems, Book 4, Electronics, Appendixes A through D. MSFC-DRL-160 Line Item 8 (MDC GO605), McDonnell Douglas Astronautics Company, July 1970.

SECTION IV. EXECUTIVE DESIGN REQUIREMENTS

A. General

This section discusses guidelines, objectives, and standards that will form the methodological baseline for executive systems functional designs to be developed in accordance with the report. The purpose of specifying design requirements is to insure that certain design objectives not directly related to system function or performance are met. In order to establish a framework of clarity with regard to the approach taken here and elsewhere in this report, it is necessary that a good understanding of the objectives of the design effort prevail.

The primary purpose of the design is to provide a functional description of the shuttle and station executive systems in sufficient detail so as to enable subsequent development of a comprehensive deterministic model and digital simulator for these systems. The design therefore is directed not so much toward forming the basis for further system development (e.g., detail design), although it certainly will satisfy this need to a large extent, as it is directed toward forming the basis for the understanding so necessary to simulator realism. As a result, the design will likely, on the one hand, be more tutorial than absolutely necessary and may not outline, on the other hand, details that could not be modeled feasibly even if they were outlined.

With these thoughts in mind, care has been taken to insure that certain information will be provided in the overall design documentation and that this information will be presented in a way that will simplify the extraction of executive features that are of major concern to a simulation effort.

The topics of concern in this design requirements specification are:

- Procedure Design,
- Data Design,
- Level of Design Detail, and
- Design Evaluation and Comparison Methodology.

Procedure, in the context of the specification, is taken to mean a logical algorithm, or set of precisely defined rules, for performing an

arbitrary function. While the prevailing assumption is that procedures will be fabricated or implemented in the form of computer programs, they are certainly not restricted to this interpretation since a digital logic circuit or logic control store would suffice and, in many cases, actually be preferable.

While system data are frequently given little or no concern, history supports the fact that this is a gross error of omission. Indeed, given a set of functionally sound procedures that constitutes an executive, the stability and credibility of the executive is completely determined by the nature and structure of data. The specification therefore addresses the treatment of data in a design effort.

"Level of software design detail" is not a concept that can be specified numerically or with precision as is the case with digital logic circuit design. For instance, a logic designer has a clear image of the required level of detail if he is to work to the "gate level." No such definitive boundary exists in the software realm. There is a distinct boundary imposed at the "instruction level," but even this familiar boundary is becoming defocused by trends in microprogramming. The specification technique used in this report is to outline a priori functions and modules to be included in the design.

One of the major concerns of the design effort will be to present information in such a way as to facilitate subsequent evaluation of the design through analysis of the "paper" design to augment the planned simulation effort. To guide this aspect of the design, certain evaluation criteria are defined to insure that the design documentation enhances evaluation. Having defined criteria for measurement of values, it is possible to contrast two designs by comparison of the measures. A methodology for this comparison is discussed as the final design requirement specification.

B. Procedure Design Requirements

The report is concerned here with outlining a set of procedure attributes to be determined during design, a standard format for the description of all procedures, and a set of a priori system parameters which can be measured and recorded numerically by the cognizant procedures in order to evaluate system performance.

1. Attribute Specification. Form 1 is provided as a preliminary vehicle for concisely recording selected attributes of procedures. The attributes included in the form are defined as follows.

1. Name:
2. Size Estimates:
    a. Code:
    b. Local Data:
3. Relative Statement Type:
    a. Computational:
    b. Logical:
4. Execution Condition:
    a. Frequency:
    b. Number of Operations:
5. Complexity:
    a. Number of Loops:
    b. Number of Paths:
    c. Number of Blocks:
    d. Block Exit Density:
6. Type:
    a. Simple:
    b. Function:
        (1) Integer:
        (2) Real:
        (3) Boolean:
        (4) Complex:
        (5) Special:
7. Class:
    a. Reentrant:
        (1) Recursive:
        (2) Non-Recursive:
    b. Non-Reentrant:
8. Priority:
9. Development Time:
10. Residence Requirements:
11. Events Causing Execution:
    a. Procedures that Reference:
    b. Interrupt:
    c. Queue Processing:
    d. Direct Call:

PROCEDURE ATTRIBUTE
DOCUMENTATION FORMAT

Form 1

Figure IV. B. 1

a. Name. Each program is assigned a unique alphanumeric name. The name will be sufficient to locate and define the module in libraries, internal storage, or external storage and is formed by the rules for acronym definition (shown in Form 2).

b. Size estimates. Estimates are provided here for number of lines of code and space required for local data. Since no specific word-lengths in bits are assumed, each entry provides a qualifier that can be used to develop a conversion factor. All sizes are given in decimal, and lines of code are qualified as either assembly code (ASM) or problem oriented language (POL) code, such as FORTRAN, BASIC, etc.

c. Relative statement type. The number of each different instruction class is estimated (logical or computational).

d. Execution condition. The frequency of execution (number of times per second) and the number of operations per execution (number of instructions executed per call).

e. Complexity. The complexity of a procedure is measured by the number of loops, number of paths, number of flow diagram blocks, and block exit density. These factors impact design intimacy, complication of coding, and thoroughness of checkout. The exit density calculation will be a valuable measure of complexity since it relates decisions and paths at module level.

(1) Number of loops. The enumeration of repetitively executed sequences of instructions. Loop level is multiplied by the number of loops at each level (1, 2, ...) to account for nestedness.

(2) Number of paths. The number of transitions that can be made among procedure subdivisions. This is the same as the total number of exits from all blocks.

(3) Number of blocks. The total number of flow diagram blocks.

(4) Block exit density. Value of (2) divided by value of (3).

f. Type. This attribute is specified by the usage. A procedure is either a simple procedure or a function.

(1) Simple. This refers to a procedure that accepts input parameters and returns output parameters only through use of a formal parameter list or through global variables.

(2) Function. A procedure which accepts input through a formal parameter list and returns a single output through a hardware register is a function. It may return an output that is in the machine format of an integer, real, boolean, complex, or special variable.

g. Class. Class is determined by the structure of a procedure and may be non-reentrant/non-recursive (normal), reentrant, or recursive.

(1) Reentrant. This is a procedure that can be called repeatedly at any stage of execution and properly complete each call. This implies that the program module may not be dynamically modified and does not store intermediate results locally.

(2) Recursive. A procedure which calls itself (through the use of push-down lists) is said to be recursive. The call may be direct or indirect through another procedure. A recursive procedure is not strictly reentrant since it can not be called at any point in its execution. It is however reentrant in the sense that the recursion is accomplished by repeated calls to (reenters at the beginning) itself.

h. Priority. An integer from 0 to n (where 0 is the highest priority) that indicates the estimated relative importance or urgency for execution of this procedure.

i. Development time. The estimated man-months required for total programming implementation (requirements analysis through checkout, including documentation).

j. Residence requirements. Main memory allocation requirements for procedures, transient areas, overlays, and bulk storage considerations will be estimated; an indication of storage requirements when the program is in a dormant state is included.

k. Events causing execution. An indication of what event(s) must occur in order to cause execution of the procedure is given. This should expose transition routes among procedures and show the initiating conditions and mechanisms for procedure connectivity. Examples are direct call by another procedure, indirect call by queuing, and interrupt activation. A list of the procedures that reference this procedure is given.

2. Documentation Standards. Form 2 is provided as a standard for describing all modules within the design. It is shown as a prototype to make it self-explanatory.

3. Performance Measurement. A list of a priori parameters is given below. Throughout the design, this list will dictate the inclusion in certain modules of design specifications that add the function of sampling or measurement. The general concept will include retrieval from storage of

Procedure Identifier: An upper case alphanumeric mnemonic identifier of unlimited length is provided as a name. An acronym for this identifier will follow the name and be enclosed in parentheses. The acronym is formed by following a set of rules until the name has been reduced to six (6) or fewer characters according to the rules. The rules are:

1. If the name is six or fewer characters long, the acronym is the name.
2. Otherwise, beginning at the rightmost character in the name and working leftward, delete vowels until six characters remain (in which case the acronym becomes the six remaining characters) or more than six remain. In the latter case, again working from right to left, delete characters until six remain. the acronym is then these remaining characters. In forming the acronym, leading vowels are treated as consonants.

   Example: INTeRRuPt (INTRRP)

Purpose: This is self-explanatory. A brief description sufficient to indicate the role of the procedure and its relation to other procedures is given.

Approach: The design philosophy and algorithms for performing the intended functions are explained. This includes a prose functional description of control considerations, flow diagrams, and a description of (local) data that are used but not described elsewhere. Flow diagramming follows these rules:

1. Control Flow - Solid line with arrow.
2. Data Access/Store - Dash line with arrow.
3. Procedure Entry Point - Call-by-name is indicated by a bubble; a keyed (interrupt/queue) entry is indicated by a double line with arrow.
4. Off-Page Connector - Acronym.alphanumeric.

Local data, i.e., data defined as part of the body of the procedure, are described here. Global data descriptions and descriptions of other data defined external to the body of the procedure are provided on separate sheets according to a specified format and are merely referenced in the approach.

A discussion is given of special language features that will enhance program development or execution by reducing core requirements, execution time, coding, or linkage complexity. A list of useful macro instructions or frequently used procedures (hardware or software) that will enhance program preparation and/or execution will be given.

External Procedures Referenced: The names of all procedures referenced by this procedure are itemized here as a linear list of names separated by commas. Called procedures that execute concurrently (possibly) have their names underlined.

External Data Referenced: The name(s) of all external data referenced by this procedure is itemized here as a list. A statement is included to show if a data entry is added, deleted or changed.

FUNCTIONAL PROCEDURE DESCRIPTION FORMAT

Form 2

Figure IV.B.2

the most recent value of the parameters followed by action necessary to cause them to be recorded on a mass storage device or transmitted to ground for subsequent analysis. The list is not necessarily exhaustive and may be modified during design. The task name and an event identifier are assumed to be appended to all recording actions.

a. Task activation time. This is the (real) time at which the system first becomes aware that a task must be executed.

b. Task memory request time. This time is recorded each time a request for memory is made either by or on behalf of a task.

c. Task memory assign time. This time is recorded each time a request for memory for the task is granted. This is paired with its corresponding request time.

d. Task memory request size. The size of a task memory request is recorded at the time of the request and is associated with the request.

e. Task memory release time. When memory is released for a task, the time and size of the release are recorded.

f. Task queued for execution time. The time when a task is queued, after activation, for assignment of processor time.

g. Task first execution time. For each task activation, the time when the task is first assigned processor time is recorded. Processor identification is also recorded.

h. Task idle time. Each time a task ceases to use an assigned processor, but does not terminate in some sense, the event and time are recorded. The deassigned processor is also recorded.

i. Task reassigned a processor. Each time an idle task is reassigned a processor the time and processor are recorded.

j. Task termination time. Time is recorded when a task terminates in some sense. The deassigned processor is also recorded.

k. Supervisor request time. Each supervisor request will be identified and the request time recorded. The servicing processor will be recorded.

l. Interrupt time. Each interrupt that occurs will be identified and the time recorded. The nature of certain high frequency interrupts may preclude such measurement under some conditions. In this case, the number

of occurrences will be counted by incrementing an associated counter that can be sampled over relatively large time intervals.

m. Task priority. Task priority will be recorded at activation time and each time thereafter when a change occurs.

n. I/O buffer. Each input/output supervisor request will cause originating task name, buffer size, source device, destination device, and transmission media (channel, bus, etc.) to be recorded.

C. Data Design Requirements

The need to emphasize data usage and structure policies has been discussed and should be clear. This subsection provides a discussion of those points of interest felt to be most pertinent to data documentation. The class of data to which these comments are directed is what can be referred to as external data. That is, data which are defined to reside external to the definition constituting a procedure body. Data defined strictly local to a procedure, and therefore referenced only by that procedure, are discussed on Form 2. Attributes and documentation are discussed.

1. External Data Attribute Specification. Form 3 exhibits a preliminary list of attributes associated with external data. The list is defined as follows:

a. Name. A name uniquely identifying the data. It is formed in a manner similar to that of a procedure name.

b. Size estimates. See "Size Estimate" discussion found on Form 4.

c. Procedures that reference. This is a list of names of procedures that reference this named data structure.

d. Reference frequency. This provides an estimate of the number of times the data structure is referenced in a specified interval. The type of reference is indicated by breakdown of the estimate into categories showing changes, additions (expansion), and deletions (compression). The estimates are dependent on the execution frequency and logic of those procedures that reference the data.

e. Structure type. The logical structure of the data is indicated by stating whether it is simple (linear unconnected single variables),

1. Name:

2. Size Estimates:

3. Procedures that Reference:

4. Reference Frequency:

    a. Element Changes:

    b. Element Additions:

    c. Element Deletions:

5. Structure Type:

    a. Simple:

    b. Array:

    c. List:

    d. String:

    e. File:

    f. Record:

    g. Memory Map:

EXTERNAL DATA ATTRIBUTE DOCUMENTATION FORMAT

Form 3

Figure IV. C. 1

array, list, string, file, record, or memory map. Other types may prove of interest as design progresses.

2. External Data Documentation Standards. Form 4 delineates a prototype documentation format that is self explanatory.

D. Level of Design Detail

Level of design is implied intuitively by delineation, below, of executive functions to be addressed by the design effort and a modular organization list for the real time flight executive system. The functions and modules are a priori in nature having been derived from a cursory analysis based on experience. They are not intended to be exhaustive.

1. Executive Functions.

   a. System control.

      (1) Scheduling.

         - Device scheduling
         - Task loading and relocation
         - Main memory allocation (dynamic)
         - Task swapping (roll-out/in)
         - Dynamic dispatching (ALU)

      (2) Task-to-task call linkage.

      (3) Parallel task execution.

      (4) Program timed request control

         - Mission time-line control/interpretation
         - Watchdog timing
         - Real-time clock/interval timing
         - Subsystem stall alarms

Data Identifier: A mnemonic name with specifications similar to those given for naming procedures is provided.

Purpose: A brief description of the reason for specifying these data is provided. This should include a discussion of usage. Important simple variables will be collected together in a single description when their logical relationships warrant. This format is reserved primarily, however, for descriptions of data structures such as files, records, memory maps, stacks, queues and packed word components.

Structure: A prose description of the specific structure of the elements of data and the reason for choice of this structure, unless no other choice is pertinent, is given. A diagram showing the relations among components of the data is included along with a description of each component. Structure can be simple, array, list, string, etc.

Size Estimate: The size of each component is estimated in number of bits. In the case of components whose size is dictated exclusively by hardware register sizes or logic functions, no size is given, but the commonly accepted register name (e.g., accumulator, index, etc.) or logical function (integer arithmetic, floating point arithmetic, character manipulation, etc.) is indicated. A formula is also given which indicates the total structure size in number of bits as a function of register and logical function sizes.

FUNCTIONAL EXTERNAL DATA DESCRIPTION FORMAT

Form 4

Figure IV.C.2

(5) Interrupt processing.

(6) Task termination/suspension.

(7) Mass storage allocation (file control).

(8) Fast buffer assignment.

(9) Overlay (segment) manipulation.

(10) Memory protection.

(11) Request (supervisor call) processor.

(12) Device inventory.

b. I/O operation and control.

(1) I/O request scheduling.

- Request initiation
- Request queueing
- Device and channel allocation
- Channel or processor program generation

(2) I/O device handling.

- Keyboard
- Teletype
- CRT systems
- Digital contact closures
- A/D and D/A interface and multiplexing
- Error log and recovery

(3) I/O data transfer monitoring.

(4) I/O system status monitoring.

(5) I/O system status updating.

(6) Symbolic I/O assignment.

(7) Data movement control.

- CPU to subsystem (CPU or other device)
- Up/Down link telemetry
- CPU to/from bus

(8) Communication handling.

- Message transmit
- Message receive
- Message queueing

c. System monitoring.

(1) Anomaly/error alarm monitoring.

- Alarm output control
- Alarm history recording

(2) Computer system performance monitoring and logging.

- Executive requests utilization and timing
- Number of bus transfers
- Average bus data volume
- Arithmetic logic unit utilization
- Main memory utilization

- Error frequency count maintenance
- Failure count monitoring
- Queue lengths
- Channel utilization
- Device utilization
- Interrupt frequency

d. Support programming.

(1) On-line error correction.

(2) Program path reconstruction (interrupt history dump).

(3) Contiguous listing of machine errors.

(4) Program parameter modification.

(5) Imbedded selective trace routines.

e. Error recognition and recovery.

(1) Program error detection.

- Incorrect I/O requests
- Requests for equipments already in use (busy return or queuing)
- Incorrect data formats
- Incorrect instruction or formats
- Unstable iterative or non-convergent process

(2) Error recovery/action sequencing.

(3) Parity error recovery sequencing.

f. Failure detection and recovery.

(1) Hardware diagnostic routine execution.

(2) Diagnostic countdown.

(3) Failure recovery.

- Spare switching (reconfiguration)
- Degraded mode operation

(4) Restart after equipment failure.

(5) Configuration control.

2. Executive Modules.

a. Master control.

(1) Mission phase control.

- Phase termination
- Phase initiation

(2) Configuration mode control.

- Error detection/isolation
- Error recovery

(3) Astronaut/pilot control.

(4) Central interrupt control.

(5) Mission schedule control.

- Schedule interpreter
- Schedule editor
- Subsystem supervisor control

(6) Device handlers.

(7) Device availability control.

(8) Device diagnostics.

(9) Request processor.

(10) Message initiation.

(11) CPU-CPU data and signal control.

(12) CPU-bus handler.

b. Task scheduler.

(1) Priority recognizer.

(2) Device availability queue control.

(3) Main storage availability queue control.

(4) ALU time queue control.

(5) ALU time dispatcher.

c. Main storage control.

(1) Task relocation control.

(2) Task swap control.

(3) Main storage allocation control.

(4) Main storage release control.

(5) Main storage protection control.

d. Mass storage control.

(1) File directory control.

(2) File allocation control.

(3) File release control.

(4) File protection control.

(5) File processor control.

e. I/O communication control.

(1) I/O supervisor.

(2) Input control.

(3) Output control.

(4) Command/control output director.

(5) Display manager.

E. Evaluation and Comparison Methodology

Future manned space flights will require a higher degree of autonomy and expanded reliance upon on-board computers than has been exhibited by past and current systems. The associated increased computer responsibilities and capabilities require an executive operating system that will enhance every facet of the mission. This subsection discusses methods for evaluating and thereby comparing these operating system designs. Because of the extreme cost of implementing operating systems to an acceptable level of reliability and performance, it is desirable to be able to evaluate their design prior to commitment of funds and manpower. Analytical (relatively inaccurate) and deterministic (more accurate) simulation techniques have proven their value in forecasting performance. However, no suitable methodology exists for evaluating the features and scope of a design that exists in a documented form only. For this reason, the systems under discussion here are "paper" systems. That is, they are design documents (flowcharts and descriptions) that are conceptual in nature. It is quite true that concepts are difficult to compare. Many features of operating systems are complex, non-deterministic, and avoid measurement even on checked-out operating systems. However, paper operating systems can be evaluated to a degree and this report will describe a procedure for doing so.

A preliminary list of evaluation criteria has been developed as part of the study. The list is exhaustive in that all operating system features which reflect on total mission performance can be represented. The criteria and each definition will be refined as the study progresses to enable more comprehensive system evaluations.

A comparative analysis produces guidelines and measures for determining the relative value of executive software and computer hardware configurations. The hardware configuration has undeniable impact on the executive software. For example, software complexity may be greatly influenced by the processor communication/isolation differences basic to the integrated vs. federated computer hardware approaches. Every attempt is made to keep the comparison criteria independent of computer configuration to prevent introduction of hardware biases into the evaluation. Gross architectural features (such as number of independent computers in a system), however, prohibit total independence.

The analysis involved definition of comparison criteria, identification of comparable system features, and the investigation of the design documents. The design documents are assumed to include flowcharts, prose descriptions, and programming estimates for successive levels of software definition. Ideally, the systems to be compared should exist at the same level of definition for each level of evaluation.

Individual module descriptions are to be analyzed and compared on the basis of prototype or generic analytical and subjective criteria. For example, comparison of the number of instructions, the number of nested loops, and the number of submodules (such as device handlers) is a relative enumeration technique. Estimates of certain values, such as program size, can be based on recorded values for similar operational modules. Even though these historical values may pertain to very different hardware, their relative or ratioed values can be useful indicators.

1. Evaluation. The following list itemizes system features which are for the most part intuitive in nature:

- Modularity
- Complexity
- Flexibility
- Storage availability
- Development effort
- Storage requirements
- System integrity
- Reconfigurability

- Maintainability
- Performance

In order to understand the applicability of these terms to an executive system and its functional design specification, we must discuss their meaning as applied to a software system. Before proceeding with this discussion, the term "architectural" must be defined with respect to software.

"Architecture" is taken to be that logical configuration or organization of computer programs and hardware as it appears to the user of the configuration.

The word "logical" is underlined in order to emphasize the fact that, to the user, the system may appear to exhibit a very different organization than it actually has in the view of the designer - whom we assume has the most realistic and factual vantage point.

We now define the intuitive features, making frequent reference to the concept of architecture. Bear in mind that architecture is relativistic in nature and depends on the point of view from which the organization is examined. Figure IV.E.1 summarizes these features.

a. Modularity. To the software designer, his system would be highly modular if it was made up of a very large number of essentially autonomous subprograms. His ability to alter small portions of the system would be enhanced to a large extent. We might compare an extremely modular software system with a digital logic design based on discrete components. However, in the same way that discrete element circuits have a high "overhead" in the form of large rise times, long delays, and high packaging weights and volumes, highly modular software systems suffer from high "overhead" in the form of time to link procedures, and storage for the code to accomplish these "housekeeping" chores.

Obviously, in the same way that digital designers are trending away from discrete components and toward large scale integrated (LSI) circuits because of the speed and mass/volume advantages, the software designer must do the same. However, while the digital designer is focusing his design efforts on packaging widely used high-level modules such as memory modules, multipliers, decoders, etc., we must acknowledge that his efforts cannot be applied to "one of a kind" hardware devices that do not use the modules. The LSI packaging concept applies in cases where the module-level function is well understood; and considerable uncertainty still exists as to what functions to modularize!

SYSTEM EVALUATION FEATURES	BEST SYSTEM VALUE
Reconfigurability	"High"
System Integrity	"Yes"
Average Functional Subroutine Modularity	Lowest
Average Functional Macroinstruction Modularity	Lowest
Complexity	Lowest
Flexibility	Highest
Storage Requirements	Lowest
Storage Availability	Highest
Development Effort	Lowest

SYSTEM FEATURE SUMMARY

Figure IV. E. 1

Naturally the same may be said about software modularization as we trend toward packaging larger scale software modules. In the digital logic world, the logic modules become the building blocks for the system architect. The same is true in the case of software. The architect in each case must place himself at the vantage point of the user in order to design a system that fits the user's needs.

The level of system software modularity has matured to a point comparable to the integrated circuit of the early 1960's. The system software analogy to LSI will not be feasible until the inherent flexibility of microprogramming is applied to the definition of high-level instructions tailored to enhance software system design. Through this medium the software architect and designer can realize the packing density increase and time delay decrease so necessary for the achievement of a high degree of user-oriented software modularity.

Consider now the measurement of modularity of a system. Clearly it is not desirable simply to have a large number of procedures, since this trends toward a "discrete element" design approach. It is furthermore undesirable to have the total system expressed as a single procedure. The latter is true because some degree of breadboarding will be required coupled with flexibility to alter subfunctions without manipulating the entire system. (If mission requirements could be rigorously specified without the possibility of future deviations, the necessary requirement for flexibility would be diminished and we might justify the development of special software fabrication tools to map a fully-checked out breadboard system into one single procedure with extremely low inherent overhead. This is not likely to be feasible in the first generation reusable shuttle or space station.)

The "proper" degree of modularity seems to depend on the conclusiveness with which we can define the function of the modules. The functions of the major modules listed previously constitute a collection of well understood functions. It is, therefore, on the basis of these functions that we will define "modularity".

Two levels of modularity for each function are considered to be important. The lowest level is the number of macroinstructions required by a design to define the function, and the highest level is taken to be the number of procedures - or subroutines - required by a design to define the function. We, therefore, will have two measures of modularity for each of the following functions:

- System control,
- I/O operation and control,
- System monitoring,
- Support programming,
- Error recognition and recovery, and
- Failure detection and recovery.

Average system functional modularities for each of the two levels are measures of system modularity and are the average of each modularity over the above functions.

Notice that as we become more adept at defining higher level system sub-functions that are used in a high number of instances, these sub-functions tend to become subroutines. This causes the subroutine modularity measure to increase, indicating a relatively detrimental trend because of the associated linkage overhead increase. At the same time, however, the macro-instruction modularity measure must decrease tending to restore the balance somewhat. The above trend is definitely desirable even though system overhead increases. The reason, of course, is that well-established subroutines become candidates for implementation as microsequences in control store or conventional logic modules in system hardware. This results in a reduction in subroutine modularity. This trend might be referred to as "software to hardware bootstrapping".

b. Complexity. Complexity, like modularity, is defined with respect to a system's architecture and, therefore, differs from vantage point to vantage point. System complexity is defined to be the total number of commands and supervisor requests provided by the system to the user. A system design that can provide the flexibility to support all requirements with a small number of commands and requests is obviously less complex to the user than one requiring a larger number because it is easier to use and interface with.

c. Flexibility. This system feature also takes on different definitions depending on the point of view. From the user's point of view it is identical to complexity by definition. This illustrates a set of conflicting user desires since he at once wants a highly flexible system and yet needs simplicity. We define flexibility to be the same as complexity to remain consistent with the architecture concept. We should recognize that an assumption is being made here; we assume that, while a less complex system

may support the same set of requirements as one that is more complex, the more complex system offers a wider selection of ways to satisfy the requirements and is therefore more flexible.

d. Storage Availability. This is taken to be the size, in main memory words, of the memory space in a given hardware configuration that is available for accommodation of user programs. This is admittedly a configuration-dependent definition. However, consistent with the concept of taking the user's viewpoint of architecture, this is a meaningful definition.

e. Development Effort. Development effort is defined to be the number of computational statements divided by twelve (12) plus the number of logical statements divided by eight (8). See Form 1, Item 3. The numerics are weighting factors taken to mean that computational statements can be developed at the rate of twelve (12) per day and logical statements can be developed at eight (8) per day.

f. Storage Requirement. This measure of system design "goodness" is defined to be the size, in main memory words, of the memory space required to accommodate the complete executive system.

g. System Integrity. This intuitive system architecture feature is a measure of how well the system responds, from the user's viewpoint, to errors in his programs and failures in the hardware. Since future space systems will be designed specifically to meet failure mode recovery performance requirements that are established independently of a particular user, we will concern ourselves here only with the response of the system to user program errors.

System integrity then reduces to consideration of two classes of user-induced errors; errors resulting from improper use of the system and errors resulting from instability in the user's programs.

Examples of improper use of the system include such things as: failure to request allocation of a device before attempting to access it, failure to return storage that has been allocated, and improper sequencing of requests for service when a specified ordering must be observed. Instability in the user's program can occur as a result of an improbable set of parameter values that cause looping or non-convergence in an iterative procedure.

Presumably, the first class of user-induced errors will be eliminated during component testing and total system checkout. This is the burden of testing and quality assurance activities during development. However, it is clear that no numerical value can be placed on the system's ability to resist

or detect and recover from these two classes of errors. Typically, the design will include exhaustive provisions for guarding against them. Cross-checks on the validity and sequencing of all operations requested of the system by either command or supervisor calls, and setting and checking watch-dog timers and stall alarms is commonplace. If features such as these are found in a design, we indicate system integrity by acknowledgement ("yes"). This requires that the logical aspects of interactions among functions be thoroughly reviewed. Failure to take into account all contingencies will be indicated by lack of acknowledgement ("no").

h. Reconfigurability. This system feature is measured by whether the user must be aware, in his program design, of different architectural configurations. If, subsequent to a system failure, the user must show awareness, in any functional way, of a change in architecture, we say that the system exhibits a "low" degree of reconfigurability. If, on the other hand, system failures do not impact on the user's program logic, we say the system exhibits a "high" degree of reconfigurability. This definition does not take into account the possibility that the system may not be able to reconfigure and may, therefore, fail "hard." The justification for this lack of accounting comes from the mission performance requirement that space systems must be able to sustain failures through reconfiguration or hardware multiple redundancy. Adequate reconfigurability or hardware redundancy is, therefore, tacitly assumed.

i. Maintainability. System maintainability is a measure of the effect on system architecture resulting from improvements, expansions, corrections, or other changes to the system. If user program logic, format, or organization must be altered as a result of system maintenance activities, we say that the system exhibits a "low" degree of maintainability; it is "high" otherwise. This feature is dependent to a great extent on the design aspects of system support activities and programs. Since this report does not address this subject, we exclude maintainability temporarily from the list of evaluation features.

j. Performance. This system feature is also excluded temporarily from the evaluation criteria because it is indeterminate on the basis of design analysis only.

2. Comparison. The literature (1), (2) has not yielded information relating to the comparison of complex real time executive system design. This was to be expected since the field of computing sciences is still relatively immature.

However, a preliminary method that is based on the attributes and features discussed previously has been developed. This comparison methodology consists, firstly, of filling out the chart suggested by Figure IV. E. 2 where the attributes of all system procedures are recorded. Next, the total of each of the attributes taken over all procedures is transferred to the appropriate column shown in Figure IV. E. 3 for each of two systems. The features shown in Figure IV. E. 1 are also evaluated for each system and recorded on Figure IV. E. 3. Finally, the total size of external data is recorded based on some arbitrary value for any variables required to obtain a constant value.

Figure IV. E. 3, thus completed, brings the major features and attributes of each system together in sharp contrast. Further comparison must be made by subjective comparison based on the application of weighting factors. This aspect of comparison is felt to be beyond the scope of the report.

Procedure Name	Procedure Attributes																			
	Code size	Local Data Size	# Computational Statements	# Logical Statements	Execution Frequency	# Operations	# Loops	# Paths	# Blocks	Block Exit Density	Type Simple	Type Function	Class Reentrant	Class Recursive	Priority	Development Time	Residence Requirements	# Procedures that Reference	Activated by Interrupt	Activated by Queue	Activated by Direct Call

System Procedure Attributes

Figure IV. E. 2

SUMMARY OF PROCEDURE ATTRIBUTES	SYSTEM A	SYSTEM B
Code Size		
Local Data Size		
# Computational Statements		
# Logical Statements		
Execution Frequency		
# Operations		
# Loops		
# Paths		
# Blocks		
Block Exit Density		
Type Simple		
Type Function		
Class Reentrant		
Class Recursive		
Priority		
Development Time		
Residence Requirements		
# Procedures that Reference		
Activated by Interrupt		
Activated by Queue		
Activated by Direct Call		
SYSTEM EVALUATION FEATURES		
Reconfigurability		
Integrity		
Subroutine Modularity		
Macroinstruction Modularity		
Complexity		
Flexibility		
Storage Availability		
Development Effort		
External Data Size		

System Comparison Summary

Figure IV. E. 3

REFERENCES

1. Budd, A. E.: A Method for the Evaluation of Software Executive Operating or Monitor Systems. The MITRE Corporation, Bedford, Massachusetts, September 1967.

2. Anderson, M. D. and Marek, V. J.: Evaluation of Aerospace Computer Architecture. AIAA Paper No. 68-836, AIAA Guidance, Control, and Flight Dynamics Conference, Pasadena, California, August 12-14, 1968.

BIBLIOGRAPHY

1. Adelman, A. and Kemp, A. J.: Space Station Information Management. Institute of Electrical and Electronics Engineers (EASCON) Conference Presentation, Washington, D.C., October 29-31, 1969.

2. Advanced Topics in Systems Programming. University of Michigan Engineering Summer Conferences, June 16-27, 1969.

3. Anderson, M. D. and Marek, V. J.: Evaluation of Aerospace Computer Architecture. AIAA Paper No. 68-836, AIAA Guidance Control, and Flight Dynamics Conference, Pasadena, California, August 12-14, 1968.

4. Budd, A. E.: A Method for the Evaluation of Software Executive, Operating or Monitor Systems. The MITRE Corporation, Bedford, Massachusetts, September 1967.

5. Campbell, P. J. and Heffner, W. J.: Measurement and Analysis of Large Operating Systems During System Development. Proceedings of FJCC, Vol. 33, No. 1, 1968.

6. Chestnut, Harold: Systems, Engineering Tools. John Wiley and Sons, Inc., New York, New York, 1966.

7. Comparative Operating Systems, A Symposium. Association of Computing Machinery, Cleveland/Akron Chapter, Brandon Systems Press, Inc., 1969.

8. Computer Subsystem Trade Study, Appendix 3, IBM Preliminary Documentation of Space Station Requirements, IBM Corporation, 1969.

9. Data Acquisition Subsystem Trade Study, Appendix 4, IBM Preliminary Documentation of Space Station Requirements. IBM Corporation, 1969.

10. Data Distribution Subsystem Trade Study, Appendix 5, IBM Preliminary Documentation of Space Station Requirements. IBM Corporation, 1969.

11. Data Management System Requirements, Section 4.1, IBM Preliminary Documentation of Space Station Requirements. IBM Corporation, 1969.

12. Data Management System Station Technology Assessment, Appendix 9, IBM Preliminary Documentation of Space Station Requirements. IBM Corporation, 1969.

13. Data Storage Subsystem Trade Study, Appendix 8, IBM Preliminary Documentation of Space Station Requirements. IBM Corporation, 1969.

14. Digesu, F. E.: A Conceptual Design of Space Shuttle Integrated Avionics Systems. NASA TM-X-53987, February 3, 1970.

15. Experiments Requirements, Appendix 2, IBM Preliminary Documentation of Space Station Requirements. IBM Corporation, 1969.

16. Final Report, Integral Launch and Reentry Vehicle, Executive Summary; Special Studies, Vol, III, Part A (Sections 1-3 and Appendix A); and Part B (Sections 4-5 and Appendixes B, C, and D). Lockheed Missiles and Space Corporation, A959837, December 22, 1969.

17. Frasier, C. W. and Gardiner, R. A.: Space Shuttle Vehicle Electronics. NASA MSC Report Presented at the Space Shuttle Vehicle Symposium, Washington, D. C., October 17, 1969.

18. Guidelines and Constraints Document, Space Station Program, Phase B. MSFC Document No. SE-001-009-2H, Space Station Task Team, National Aeronautics and Space Administration, Marshall Space Flight Center, June 12, 1970.

19. Hokom, R. A.: Executive Program Control for Spaceborne Multiprocessors. Autonetics, A Division of North American Aviation, Inc., 1968.

20. Hrastar, John: Attitude Control of a Spacecraft with a Strapdown Inertial Reference System and Onboard Computer. NASA TN D-5959, September, 1970.

21. Image Processing System Trade Study, Appendix 6, IBM Preliminary Documentation of Space Station Requirements. IBM Corporation, 1969.

22. Information Management Study, Final Report, Overall Summary Results. MSC 00197 (MTR-1524, Vol. 1), The MITRE Corporation, June 30, 1970.

23. Integral Launch and Reentry Vehicle Systems, Configuration Design and Subsystems, Vol. I, Book 1. NASA CR-66863-1 (MDC E0049), McDonnell Douglas Astronautics Corporation, November, 1969.

24. Integral Launch and Retrieval Vehicle Systems, Executive Summary. NASA CR-66862 (MDC E0049), McDonnell Douglas Astronautics Corporation, November, 1969.

25. Integrated Avionics for Space Shuttle Vehicles, McDonnell Douglas Slide Presentation. McDonnell Douglas Astronautics Corporation, September 25, 1969.

26. Integrated Avionics Software. McDonnell Douglas Slide Presentation, McDonnell Douglas Astronautics Corporation, September, 1969.

27. Integrated Avionics Systems Trade-Offs. McDonnell Douglas Slide Presentation, McDonnell Douglas Astronautics Corporation, February 11, 1970.

28. Martin, James: Design of Real-Time Computer Systems. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1967.

29. Martin, James: Programming Real-Time Computer Systems. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1965.

30. Master Executive Control Techniques for an Advanced Avionics Digital Computer System. Naval Air Systems Command Project No. AIR-5333F4-69-1 Final Report, Honeywell Document No. 14206-FR, Honeywell, Systems Research Division, Research Department.

31. Mendelson, Myron J. and England, A. W.: The SDS Sigma 7, A Real-Time Time-Sharing Computer. Proceedings of the Fall Joint Computer Conference (FJCC), 1966.

32. Miller, W. F.: Computation and Control in Complex Experiments. Presentation at IBM Scientific Computing Symposium for Man-Machine Communication, Yorktown Heights, New York, May 3-5, 1965.

33. Preliminary Performance Specifications, Vol. II and III, Product Configuration Requirements. MSFC DRL-160 Line Item 19 (MDC GO633), McDonnell Douglas Astronautics Company, July 1970.

34. Preliminary System Design Data, Vol. 1, Space Station Preliminary Design. MSFC-DRL-160 Line Item 13 (MDC GO634), McDonnell Douglas Astronautics Company, July 1970.

35. Project Gemini Final Report Summary. NAS9-996, IBM Corporation.

36. Sholta, Louise: Digital Processing - A System Orientation. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1963.

37. Space Master Phase A Final Report. MCR-69-36, Martin Marietta Corporation, Denver Division, December 1969.

38. Space Shuttle Final Technical Report, Vol. I, II, III, IV, V, VI, VII, VIII, IX, X. General Dynamics Report No. GDC-DCB69-046, General Dynamics Corporation, Convair Division, October 31, 1969.

39. Space Shuttle Integrated Avionics. McDonnell Douglas Slide Presentation, McDonnell Douglas Astronautics Company, June 12, 1970.

40. Space Shuttle Integrated Electronics. IBM 69-M43-0003 Slide Presentation, IBM Corporation, September 29, 1969.

41. Space Shuttle Main Engine Statement of Work, Phase B. Appendix to Procurement Request DCN 1-0-21-00001, NASA, George C. Marshall Space Flight Center, February 16, 1970.

42. Space Shuttle System Program Definition Phase B, Statement of Work. Enclosure No. 4 to RFP No. 10-8423, National Aeronautics and Space Administration, Office of Manned Space Flight, Washington, D.C., February 1970.

43. Space Station Data Management System Subsystem Requirements, Appendix 1. IBM Preliminary Documentation of Space Station Requirements, IBM Corporation, 1969.

44. Space Station Definition, Vol. V, Subsystems, Book 4, Electronics, Appendixes A through D. MSFC-DRL-160 Line Item 8 (MDC GO605), McDonnell Douglas Astronautics Company, July 1970.

45. Space Station Program Definition, Summary. MSFC-DRL-160 Line Item 8 (MDC GO605), McDonnell Douglas Astronautics Corporation, August 1970.

46. Space Station Definition, Vol. V, Subsystems, Book 4, Electronics. MSFC-DRL-160 Line Item 8 (MDC GO605), McDonnell Douglas Astronautics Company, July 1970.

47. Space Station Definition, Vol. V, Subsystems, Book 3, Information Management. MSFC-DRL-160 Line Item 8 (MDC GO605), McDonnell Douglas Astronautics Company, July 1970.

48. Space Station Development Definition, Vol. III, Software Requirement Document. MSFC-DRL-160 Line Item 18 (MDC GO544) McDonnell Douglas Astronautics Company, May 1970.

49. Space Station Phase B Study Results, GPSS/360 Data Flow Simulation Model. IBM Presentation, IBM Corporation, August 12, 1970.

50. Standardized Interface Study, Appendix 7. IBM Preliminary Documentation of Space Station Requirements, IBM Corporation, 1969.

51. UNIVAC 1108 Operating System Exec. 8. UNIVAC Division of Sperry Rand Corporation, August 23, 1968.

52. Vocabulary for Information Processing. UNIVAC - Sperry Rand Corporation, Philadelphia, Pennsylvania, October, 1968.

NASA-Langley, 1971 —— 6