IN-17
128407
P- 66

NASA Technical Memorandum 103869

Space Station Freedom Data Management System Growth and Evolution Report

R. Bartlett, G. Davis, T. L. Grant, J. Gibson, R. Hedges, M. J. Johnson, Y. K. Liu, A. Patterson-Hine, N. Sliwa, H. Sowizral, and J. Yan

September 1992

National Aeronautics and Space Administration

NASA Technical Memorandum 103869

Space Station Freedom Data Management System Growth and Evolution Report

T. L. Grant and J. Yan, Ames Research Center, Moffett Field, California

September 1992

NASA

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035-1000

The Study Team

The Data Management System (DMS) analysis team consists of civil servants and contractors at NASA Ames Research Center, Information Sciences Division.

Intelligent Systems Technology Branch, Ames Research Center, Moffett Field, CA
Gloria Davis
Terry Grant
Bob Hedges
Y. K. Liu
Dr. Ann Patterson-Hine
Nancy Sliwa

Sterling Federal Systems, Inc., Palo Alto, CA
Dr. Jerry Yan

Research Institute for Advanced Computational Science, Moffett Field, CA
Dr. Marjory Johnson
Dr. Henry Sowizral

RECOM Software, Inc., San Jose, CA
Dr. James Gibson
Gary Fariss

Digital Equipment Corporation, Moffett Field, CA
Roger Bartlett

Prof. Joanne Bechta Dugan provided the reliability analysis of the DMS network reported in appendix B; she used the HARP code and the work was sponsored by Langley Research Center (LaRC). Members of the Failure Tolerance/Redundancy Management Working Group provided network failure information and preliminary models that aided our study of the failure tolerance of the DMS network.

We also acknowledge the following individuals for reviewing early versions of the manuscripts: Gregg Swietek (NASA Headquarters), Mike Pasciuto (NASA), Donald Woods (McDonnell Douglas Space Systems Company (MDSSC)) and George Ganoe (NASA LaRC).

Table of Contents

Page

The Study Team iii

EXECUTIVE SUMMARY

1. Introduction xiii

2. Purpose and Scope of the Report xiii

3. Background xiii

4. Baseline Evaluation xiii
    4.1. System Performance xiv
        4.1.1 Network performance xiv
        4.1.2 Processor performance xiv
        4.1.3 Software services xv
    4.2. System Dependability xv

5. Significant Findings xvi
    5.1. Processor-Related Findings xvi
    5.2. Network- and System-Related Findings xvi

6. Recommendations xvi
    6.1. Further Tests xvii
    6.2. Short-Term Growth xvii
    6.3. Long-Term Evolution xvii

7. Summary xvii

SPACE STATION FREEDOM DATA MANAGEMENT SYSTEM GROWTH AND EVOLUTION REPORT

1. Introduction 1
    1.1. The Role of the Data Management System 1
    1.2. Purpose of this Report 1
        1.2.1. Baseline evaluation 1
        1.2.2. Directions and mechanisms for growth and evolution 2

2. Requirements for DMS Growth and Evolution 2
    2.1. Integration of Subsystem Elements and Operations 2
    2.2. Instrumentation and Sensor Reconfiguration 2
    2.3. Subsystem Status Monitoring and Fault Detection 2
    2.4. Onboard, Automated Element Test and Verification 2
    2.5. Inventory Management Requirements 3
    2.6. Onboard Training Needs 3
    2.7. Payloads and Payload Operations 3
    2.8. EVA/IVA Requirements for Maintenance 3
    2.9 Advanced Automation Capabilities 3

3. Design Overview 3
    3.1. Assumptions 3
    3.2. The Data Management System Overview 4
    3.3. Hardware Component Description 4
        3.3.1. DMS optical network 5
        3.3.2. Network interface unit 7
        3.3.3. Ring concentrator and FDDI network media 7
        3.3.4. Standard data processor 8

3.3.5. 1553B local bus 8
3.3.6. The controller multiplexer-demultiplexer 10
3.3.7. Mass storage unit 10
3.3.8. Multi-purpose application console 11
3.3.9. Gateway 11
3.3.10. Time generation unit and time distribution bus 11
3.3.11. Emergency monitoring and distribution system 11
3.3.12. High rate link and patch panel 11
3.4. System Software Description 12
3.4.1. The Lynx operating system 12
3.4.2. Ada runtime environment 13
3.4.3. Network operating system 13

4. Evaluation of the DMS Baseline Architecture 13
4.1. System Performance 13
4.1.1. Evaluation approaches and criteria 13
4.1.2. Preliminary results 14
4.1.2.1. The DMS optical network 14
4.1.2.2. SDP performance and loading on the 1553B 14
4.1.2.3. SDP system bus: Multibus II 15
4.1.2.4. EDP/RAM bus: MicroChannel 15
4.1.2.5. Network operating system 15
4.1.2.6. Multiplexer demultiplexer 15
4.1.2.7. Primary/secondary storage 16
4.1.2.8. Runtime object data base performance on the DMS kit 16
4.2. System Dependability 18
4.3. Limitations on Growth and Evolution 18
4.3.1. Design uncertainty 18
4.3.2. Design limits 19

5. Recommendations for DMS Growth and Evolution 20
5.1. Recommendations for Further Tests 20
5.1.1. System level tests 20
5.1.2. Lynx operating system 21
5.1.3. Other tests 22
5.2. Recommendations for Short-Term Growth 22
5.2.1. Connectivity 22
5.2.1.1. Sensor/effector interface (MDMs/1553B) 23
5.2.1.2. Optical network between systems 23
5.2.1.3. Recommendation for connectivity 23
5.2.2. SDP performance 23
5.2.2.1. 386 performance and instruction set architecture 23
5.2.2.2. Upgrading SDP using Intel 486 24
5.2.2.3. Upgrading SDP using Intel i860 24
5.3. Recommendations for Long-Term Evolution 24
5.3.1. Toward a functionally distributed system architecture 24
5.3.2. Operating system/Ada runtime environment 25
5.3.3. A scenario for growth and evolution 25

Appendices

Page

A. References and Contacts 31
A.1. References 31
A.2. Government Contacts 31

B. ARC Simulation Report 32
B.1. Abstract 32
B.2. Introduction 32
B.3. Model 32
B.3.1. DMS elements 32
B.3.1.1. Application EDP 33
B.3.1.2. Multibus II backplane 33
B.3.1.3. Network interface unit 34
B.3.1.4. MicroChannel 34
B.3.1.5. Network interface adaptor 34
B.3.1.6. Core network 34
B.3.1.7. Bus interface unit 34
B.3.1.8. Local bus 34
B.3.1.9. Multiplexer/demultiplexer 35
B.3.2. Workload model 35
B.3.2.1. Job scheduling 35
B.3.2.2. Job models 35
B.3.2.2.1. Node SDP application software and derived object generation 35
B.3.2.2.2. Node SDP 5 core network loading summary 35
B.3.2.2.3. Node MDM application function allocation 35
B.3.2.2.4. Node MDM I/O processing activity 35
B.3.2.2.5. Node MDM local bus traffic loading 36
B.4. Results 36
B.4.1. APP-EDP analysis 36
B.4.2. MDM analysis 36
B.4.3. Other components 37
B.5. Conclusions 37

C. Advanced Automation for Space Station Freedom 37
C.1. The Environmental Control Life Support System 37
C.2. Power Management and Distribution 38
C.3. Advanced Automation Network Monitoring System 39
C.3. The Flight Telerobotic Servicer 39

D. Integrated Design Constraints 40
D.1. The Connectivity Architecture 41
D.2. Memory Hierarchy 42
D.3. Instruction Set Architecture 42
D.4. Processor Modularity 43
D.5. Special Facilities 43
D.6. Processor Architecture 44
D.7. Chip Technology 44
D.8. Chip Robustness 44
D.9. Openness 44
D.10. Software Architecture 45

E. Fault Tolerant/Redundancy Management Analysis 45
E.1. Evaluation Approaches and Criteria 45
E.1.1. Evaluation criteria 45
E.2. Failure Tolerance/Redundancy Management Process 46
E.3. Modeling Techniques 46
E.3.1. Model limitations 46
E.3.2. Redundancy analysis of the token ring network 46
E.3.3. Reliability analysis of the token ring network 47

F. Acronyms 49

Figures and Tables

Page

Figure 3-1. Data management system configuration at PMC. 5
Figure 3-2. Data management system configuration at AC. 6
Figure 3-3. The network interface unit. 8
Figure 3-4. The standard data processor. 9
Figure 3-5. The embedded data processor. 9
Figure 3-6. Schematic diagram of a sample SDP. 10
Figure 3-7. Controller multiplexer-demultiplexer functional block diagram. 11
Figure 3-8. MPAC model overview with MSU on network. 12
Figure 4-1. Centrifuge facility/SSF interface scheme. 17
Figure 4-2. Typical path of optical signal showing multiple feed-through points. 19
Figure 5-1. Scenario for inserting advanced SDPs into the DMS. 26

Table 1-1. The current shuttle general purpose computer xv
Table 3-1. Software design review documents 6
Table 3-2. Power and weight summary 7
Table 3-3. NOS latency requirements 7
Table 4-1. Performance of the 386 with cache enabled and disabled 15
Table 4-2. Average time for 10 reads of an 80-byte object on DMS kit 18
Table 5-1. SSF/DFRF avionics subsystem analogues 20

Figure B-1. SDP-5 simulation elements. 33
Figure D-1. Integrated design constraints for computer systems. 40
Figure E-1. Simplified schematic diagram of the DMS at PMC. 47
Figure E-2. Expanded view of site 2. 47
Figure E-3. Reconfiguration example for site 8 failure. 48

Table B-1. Application processor characteristics 33
Table B-2. Multibus II characteristics 33
Table B-3. NIU characteristics 34
Table B-4. Microchannel characteristics 34
Table B-5. FDDI network characteristics 34
Table B-6. Local bus characteristics 34
Table B-7. RODB characteristics 35
Table B-8. Results of SDP-5 simulation 36
Table B-9. Application-EDP Analysis 36
Table B-10. Breakdown of MDM-1 utilization 36
Table B-11. Breakdown of MDM-4 utilization 37

EXECUTIVE SUMMARY

1. Introduction

This report presents the results of a 6-month study performed at the request of the Advanced Development Manager of the Advanced Programs Branch of Space Station Engineering (Code MT) at NASA Headquarters. The evaluation team consists of civil service personnel and contractors at the NASA Ames Research Center (ARC). The study was completed at the end of calendar year 1990. Interim memos were issued as appropriate and preliminary results have been discussed with the Johnson Space Center (JSC) Work Package-2 (WP-2) civil service and contractor teams.

2. Purpose and Scope of the Report

This report has two objectives:

1. To carry out preliminary evaluations and recommend further analysis to assess the performance, dependability and growth capabilities of the current Space Station Data Management System design

2. To propose directions and mechanisms for evolution and technology insertion

The Space Station Program and the design process for the Data Management System (DMS) are large and complex undertakings. Consequently the evaluation team chose to focus on a few significant areas to achieve a reasonable level of depth. The areas chosen were processors, networks, system architecture and fault tolerance. The choice of these areas was influenced by an existing base of expertise and modeling tools that could support the relatively short study.

The configuration evaluated represents the DMS baseline of December 1990. A later report will consider the baseline established in March 1991 after the scrub activity in early 1991 mandated by the Congressional budget reductions. Documentation of problems using the older baseline is still significant because many of the underlying problems will continue to exist in the revised configuration.

3. Background

The Space Station DMS faces several challenges unique in the history of manned- and unmanned-spacecraft computer systems. These include a 30-year life and evolution of the Station as a science platform and potentially as a transportation node for manned missions to the solar system. This implies evolving requirements for both core (Station management) and payload users. The Station represents a facility in space for multiple users instead of a flight vehicle with limited capability for long-term science users. The DMS must be adaptable to changes in mission over a 30-year lifetime and must allow the insertion of new technology as it becomes available. It must also maintain performance and safety margins throughout its life.

The DMS is the infrastructure on Space Station Freedom (SSF) responsible for integrating information onboard. It allows integrated data processing and communications for both the core functions and the payloads. The DMS accommodates crew visibility (monitoring status performance characteristics) and control over the subsystems through the Multi-Purpose Application Consoles (MPAC), and it provides applications software processing for all onboard subsystems. An interface with the Communications and Tracking System that permits access to the radio frequency (RF) digital data links extends the DMS data support role to the ground. The DMS concept is based on a local area network of loosely coupled data processing stations. Each data processing station is called a Standard Data Processor (SDP) and is based on an 80386-microprocessor-class PC computer. SDPs are connected to one or more local buses that permit monitoring and control of the sensors and effectors used to create a unique subsystem functional entity. Access to other subsystems is done through a global fiber distributed data interface (FDDI) network. The FDDI is an optical data bus network sized to allow substantial increases in data and information traffic flow as the Station evolves.

The overview of the DMS presented in this report was perturbed several times by ongoing scrub activities that caused continual changes in the design during the early design stages. It is based on the charts and reports presented at the Preliminary Design Review (PDR) 3 (refs. 1-6), plus documentation produced by various Resource Board decisions and directives. Because of the pace of changes and the lag in documentation, recent changes (mostly reductions in scope) in the design of the SSF and DMS are not reflected in this report. A follow-on report will address the modified SSF DMS configuration.

4. Baseline Evaluation

The evaluation process was challenging. Scrub activities resulted in constant changes in the DMS configuration, which made simulation and modeling difficult. Test bed experiments were hindered by the lack of component prototypes such as the "DMS Kit." The DMS performs overlapping, real-time functions, and the modular nature of the design provides many options for matching resources to the functions. Therefore, any performance measure must carefully define the usage scenario, the hardware and software configuration, and the function(s) to be

PAGE XII INTENTIONALLY BLANK

PRECEDING PAGE BLANK NOT FILMED

measured. Thus, it is anticipated that these early analyses may be controversial and sometimes inaccurate. For growth and evolution capability studies, the worst-case scenarios associated with system management for the SDPs and the networks seemed to be the most time critical, so these were the focus of our preliminary analysis.

To achieve the DMS goals the designers have chosen a number of approaches that are novel to the spacecraft computing environment:

Industry standard buses and communication protocols

Commercial processors (Intel 80386)

High-level language standard (Ada)

Modern, commercially prevalent operating system (UNIX variant)

Megabits of processor memory with expansion capability

Hundreds of megabits of nonvolatile storage on disk

Program standard development tools

Distributed architecture with provisions for additional processors

On-orbit replacement of computers

Backplane architecture that allows special-purpose processors, memory, etc., to be plugged in later

Multiplexers with spare slots for additional input/output (I/O) and custom design boards

4.1. System Performance

It is difficult to determine the adequacy of the system's performance as a global entity without corresponding performance requirements for comparison, but we were unable to find any requirements. In the absence of firm specifications, the best that could be done was a detailed look at several of the areas that might be potential barriers to evolution or be performance bottlenecks in the baseline configuration.

Since the DMS is a loosely coupled group of processors, it seemed obvious that network traffic might be an area of concern. The baseline design is an FDDI optical network with a 100-megabits-per-second maximum throughput. Computational power, I/O, and backplane characteristics of each embedded data processor (EDP) were examined in detail to see if the performance of the flight version was as advertised by the vendor. System and software services such as the mass storage units (MSU) and runtime object database (RODB) performance were also evaluated. Relative comparisons were made with equivalent Shuttle systems to establish a data point.

The DMS is actually a small subset of the total computational power on the Station. In general, functions that require coordination between subsystems or interaction with the crew or ground are allocated to DMS processors. The vast majority of the low-level, high-speed, closed-loop control of Station systems will be performed at the orbital replaceable unit (ORU) or multiplexer-demultiplexer (MDM) levels. MDM designs were not mature enough during the review period to make an intelligent assessment, but they will be included in future efforts.

4.1.1 Network performance– The DMS Contract End Item (CEI) specification does not require a deterministic message-passing capability that can be provided under the FDDI synchronous mode. The network interface unit (NIU) design implementation has not been described in sufficient detail. Therefore, it cannot be determined to what degree it adheres to the FDDI standard. The terminology for the ring concentrators (RC) conflicts with the FDDI standard; there are no requirements for FDDI RCs because all ORUs are "class A" stations, meaning that they have dual attachment. There are two risks involved in making modifications to this commercial standard: (1) there is very little test data or operating experience with the unmodified standard (none in a flight environment), so the revisions are being done without a solid foundation, and (2) once the changes are made, NASA becomes the sole owner of a unique design and cannot take advantage of changes produced in the commercial world.

4.1.2 Processor performance– ARC DMS and traffic model simulations appear to agree qualitatively with the preliminary simulation results produced by the McDonnell Douglas Space Systems Company (MDSSC). Each study utilized the baseline configurations and workloads documented in reference 7. Overloading of both the application EDP and the local 1553B bus seems to be easily accomplished with baseline unit assumptions and the best available sensor and effector traffic models. An experimental evaluation of the performance of the 80386 was also carried out. Comparative tests were run on a commercial IBM PC model-PS/2 with cache memory disabled (the flight version of the 386 card in the DMS does not have cache). Dhrystone and Whetstone benchmark programs were used to perform the benchmark analysis. The programs were compiled using Gnu C on the LynxOS, which is the closest commercial equivalent to the flight system IBM is developing. Results indicated a significant loss in performance without the cache; 3.1 MIPS for the flight CPU versus 4.0 MIPS for a commercial PC product. The work package-2 (WP-2) prime contractor and the DMS subcontractors have both acknowledged the validity of these findings.

WP-2 selected a backplane connection technique based on the Multibus II product. This backplane scheme is attracting support as an industry standard from many vendors of commercial data and signal processing systems. The Multibus II standard supports both uni- and multiprocessing and allows up to 21 communicating agents, which is well beyond the projected needs of the Station DMS. The maximum theoretical throughput of Multibus II is 40 MB per second, which is greater than the FDDI capacity. Therefore, it is surmised that the FDDI global ring will become a limiting factor to system performance before the Multibus backplane.

Commercial support of IBM's MicroChannel bus is an important consideration to the life-cycle cost of the DMS. A lack of multiple-vendor support for the MicroChannel and the IBM PS/2 in the future could become a barrier to effective DMS logistics and evolution. The MicroChannel seems as good a choice as any other bus that is available now, with this single exception.

4.1.3 Software services– The Program Definition and Requirements document (PDR) requires that multiple copies of programs be kept on board in the event of a system failure that prevents access to one of the hard disk MSUs.

It is not certain how redundancy of data and programs is achieved with the current proposed configuration. In the case of a failure of one SDP, it is uncertain how redundant RODB data is kept for recovery purposes or what its impact on data traffic is. Also, if the SDP or multipurpose application console (MPAC) reboot mechanism relies on a disk, a boot program should reside on the booted system on a battery-backed-up RAM device or a ROM device. It might be better for reliability and performance under today's technology to have a single centralized disk farm with a high-speed controller device that supports disk mirroring. The proposed baseline configuration may have problems with initialization and reconfiguration delays because the MSUs are shared across the network and SDP operating systems will need to load from them.

The performance of the RODB was evaluated recently on the DMS Test Bed at JSC. WP-2 contractors at JSC determined that in the best case, with the information available locally and no application load, it took 0.43 seconds to access a single data object. This is significantly slower than what would be required if the application used the DMS according to the DMS Critical Design Review (CDR). Assuming the RODB could be optimized by 2 orders of magnitude (or 100-fold), remote RODB access would still take 14 milliseconds. Translated into a rate, the RODB could handle somewhere between 73 and 233 read requests per second. According to the DMS software CDR, a 100-fold speed increase would still be insufficient.

As a comparison, one DMS SDP is compared to the Shuttle's General Purpose Computer upgraded version. Remember that there are multiple SDPs distributed throughout the Station.

Table 1-1. The current shuttle general purpose computer

	Shuttle GPC	DMS SDP
Memory	0.4 MB	4 MB
MIPS	1	3
Nonvolatile storage	34 MB (Tape)	270 MB
Language	HAL-S	Ada
Bus maximum rate	1 megabit per second	100 megabits per second

The comparison with the Shuttle is a good, although not perfect, one. The on-orbit portion of the Shuttle's mission is the most benign from a real-time-response point of view; it is only this environment with which the DMS must contend. Ascent and descent response times drove the design of the Shuttle system (triple and quadruple hardware redundancy with software comparison of output commands). For the Station, the DMS is only required to be single-failure tolerant.

4.2. System Dependability

The failure tolerance and redundancy management analysis for SSF is focused on the functional requirements of the integrated DMS. This analysis is based on a system modeling technique called digraphing, which is currently being performed as part of the design review process.

The digraph models for several critical functions are currently being constructed and reviewed. The DMS requirements are considered in this process so that the design can be verified to meet the two-failure tolerance requirement for overall Station safety.

The failure modes and effects analysis was documented in MDC H4563. The focus was on two main modes of failure: (1) failure of the ring concentrator (RC) because of power supply failure, and (2) network media failures caused by breaks in the fiber and/or connectors. If it is possible for an RC to fail as a unit, then FDDI standard reconfiguration procedures after such a failure will result in isolation of all nodes connected to it. This condition has not been fully analyzed at this time. This does not mean that the DMS is only single-failure tolerant to this mode of failure, however, since there exists a second network path that can provide access to the isolated processors. Also, the subsystem function assigned to a particular node will continue to operate even if the global monitoring and control capabilities are lost. Therefore, the failure tolerance level of the DMS varies with the type of failure. These and other failure scenarios should be run on test beds to determine both the sensitivity of the system to various types of transient and permanent failures and the operational impacts of the cold standby processor configuration for redundancy.

It has not been determined by ARC whether the safety of the Station is jeopardized by the failure of more than one SDP. The time required to bring up a cold backup processor might be 15-30 min. If a second failure occurred during this period, the effect on the crew or the facility is unknown and highly dependent on the failure scenario. These are additional reasons for test bed evaluations at both JSC and ARC.

5. Significant Findings

Although the studies and analysis for the DMS are far from being complete, there have been some significant findings during the initial 6-month activity. As is usually the case, there have also been a number of areas identified for further evaluation; they are listed in section 6 of this report.

5.1. Processor-Related Findings

1. The EDP has 3.1 MIPS capacity instead of the advertised 4.0 MIPS. This has been acknowledged by the WP-2 DMS subcontractor. This has a potentially significant impact when the number of SDPs is reduced and multiple functional programs have to run on each SDP.

2. Even the reduced EDP capability of 3.1 MIPS may be further reduced when the full impacts of the POSIX kernel and LynxOS are measured. Very early tests indicated that the effective application software capacity may be down to 2.6 MIPS. This estimate needs to be confirmed with additional testing at ARC as system software is developed.

3. The upgrade path to a 80486 processor may not be viable. The 486 has a cache memory but no parity or error detection/correction code. Therefore, cache may have to be disabled or an off-chip cache designed especially for the Station application. These have significant long-term impacts on both hardware and software systems. Commercially available compilers and tools for the 486 processor will presume an on-chip cache and must be modified for the unique NASA configuration.

5.2. Network- and System-Related Findings

1. The Multibus II backplane appears adequate through permanent manned configuration. Projected growth requirements based on data collected in PDR and various management meetings indicate that the backplane selected for the SDP is adequate for the core functions. Growth in the payload support may become an issue in the future and must be closely monitored.

2. Network simulations at ARC and JSC/MDSSC match qualitatively. This successful comparison demonstrates that NASA can now conduct independent parallel studies and analyses of the DMS network and traffic patterns between the various processing nodes.

3. There are detailed DMS real-time requirements. During the review of the DMS a search through the various requirements and specifications revealed that there were no definitive requirements for real-time operation. There are specifications for the DMS in terms of CPU speed, memory size, data bus speed and packet sizes, but no numbers for subsystem operation. The presumption seems to have been that the SDP will be able to satisfy subsystem performance requirements. A relative comparison of the Station SDP and the Shuttle GPC indicates that the presumption is probably correct; however, this needs to be verified in test bed simulations.

6. Recommendations

During the 30-year mission life of SSF, significant growth in user needs as well as computational technologies will take place. Therefore, the DMS should be designed from the outset to accommodate changes in functionality, performance and technology. It should allow both near-term growth and long-term evolution. However, the problem is that system-wide performance requirements are not well quantified, even for initial users, primarily because no one has the experience to identify driving scenarios in which the needed end-to-end responses can be defined. These requirements could be quantified, starting with analogies to a similar ground-based data management system, and developed through test bed experiments with functional equivalents that simulate critical Space Station operating scenarios.

The DMS evaluation team recommends that NASA be prepared from the outset to (1) conduct more analyses to determine whether the current DMS software and hardware can meet established and projected requirements, (2) develop near-term plans for augmenting the payload support capability with minimal impact on the rest of the system, and (3) develop a long-term policy and mechanism for technology insertion.

The component development for SSF has been ongoing for many years. Presumably, the requirements for these components were derived from overall program requirements. As the design is scrubbed, the financial and scheduling pressures on the program force a mix-and-match use of existing component designs. Unfortunately, the design requirements can no longer be traced back to the original program requirements. Ensuring traceability requires thoughtful analysis and considerable resources. What exists for the DMS now is an Architecture Control Document (ACD) and a CEI Specification that have little connection to any definable performance requirement for the DMS system.

It is recommended that a detailed comparison of the ACD and CEI specification be made with the PRD. In addition, a validity check of the final requirements should be performed by analysis and lab simulations. These tests should include development of worst-case usage scenarios (e.g., reboost or Shuttle docking) that drive DMS response time, network traffic capacity, and latency; they should also include failure tolerance and reconfiguration requirements.

6.1. Further Tests

A variety of tests should be conducted to assess the viability of both present and evolutionary hardware and software configuration options. These include real-time (development *and* embedded) operating system mixtures on present 386 processors with possible successor CISC (e.g., 486) and RISC processors. The real-time environmental requirements should also be experimentally assessed. Local area network protocol comparisons should also be conducted.

Many features of LynxOS still need to be evaluated and/or fine tuned for DMS. These include memory locking features, default quantum size, task "dispatch latency," and implementation of a file system, as well as the ability of LynxOS to deal with the real-time functions projected for SSF.

6.2. Short-Term Growth

Local bus connectivity should be improved, both in total capacity and in ease of adaptation to general interelement communication. Early development of the 802.4H protocol interface and distribution of the fiber optic media to all hardware racks is recommended. Furthermore, an alternative NIU and/or network design study is recommended to reduce both development and life-cycle risk.

For meeting near-term performance needs, staying within Intel 80X86 instruction set architecture (ISA) may be a reasonable option because of the cost and time required for revalidating DMS software. Therefore, the 486 seems to be the most natural candidate for upgrading the SDP. However, there are still concerns about the use of the 486 for the DMS, related to the 486 performance with the on-chip cache disabled. The Intel i860 also looks promising as a (RISC) successor or payload processor, but further tests have to be carried out to determine the cost of porting code across different ISAs and communicating in a heterogeneous computer environment.

6.3. Long-Term Evolution

The evolutionary goal for the DMS should be a system that is distributed functionally, as well as physically. The provision of separate networks for core and payload applications as the DMS evolves would allow the use of state-of-the-art technologies for the payload network, while more conservative technologies could be used for the life-critical and safety-critical core applications.

Based on initial simulation and experimental results at MDSSC, ARC and JSC Test Bed, it seems possible that the combination of 386 hardware and LynxOS software will be insufficient to meet the constraints of electrical power as well as computational needs. During the projected 30-year life span of the SSF, the DMS operating system may have to evolve to support parallel computers and portable multiprocessor operating systems.

From a passive (read only) tap to the DMS FDDI core ring, actual mission data could be fed to advanced architectural modules, performing mirrored functions using source-compatible applications. This noninvasive, mirrored capability would assist in the verification and validation process of DMS upgrades before switching to newer and faster equipment. A second ring (or crossbar) network with passive taps to the core ring would also allow for newer, faster, less power-hungry processor and software enhancements, as well as for more advanced fiber transceivers and protocols, over time.

7. Summary

The Information Sciences Division at the NASA Ames Research Center has completed a 6-month study of portions of the Space Station Freedom Data Management System (DMS). This study looked at the present capabilities and future growth potential of the DMS, and the results are documented in this report. Issues have been raised that were discussed with the appropriate Johnson Space Center (JSC) management and Work Package-2 contractor organizations. Areas requiring additional study have been identified and suggestions for long-term upgrades have been proposed. This activity has allowed the Ames personnel to develop a rapport with the JSC civil service and contractor teams that does permit an independent check and balance technique for the DMS.

SPACE STATION FREEDOM DATA MANAGEMENT SYSTEM GROWTH AND EVOLUTION REPORT

1. Introduction

Space Station Freedom (SSF) will represent a new generation of NASA flight missions—a departure from the traditional modes of operation. At one end of the spectrum, NASA missions, such as those conducted by the Shuttle, have been somewhat repetitive and of short duration. This has led to an operational practice of performing repairs and upgrades on the ground between missions during vehicle refurbishment. Changing scientific objectives for different missions were accommodated by replacing equipment and payloads on the ground. At the other extreme, NASA has launched and operated numerous long-duration unmanned spacecraft. Relying on highly redundant hardware design and the flexibility afforded by reprogrammable software systems, the vast majority of these missions have succeeded and demonstrated a degree of mission adaptability.

Space Station Freedom is a radical departure from either way of doing business. It will have a lifespan exceeding that of the longest lived deep-spacecraft, and will need to adapt its mission and science objectives to meet the changing needs of the space and science communities. It is essential that the ability to adapt be built into the Station design before the first element launch.

The beating heart and central nervous system of SSF is its Data Management System (DMS). The assembly, initial capability and evolutionary growth of the entire Station is critically dependent on the capabilities of the DMS, both its initial configuration and its ability to grow to support the changing needs of SSF during its 30-year life.

1.1. The Role of the Data Management System

The DMS is a key element of SSF. The challenging role of the DMS is to support 30 years of continually changing data processing and networking needs, using a design based on today's proven technology.

The DMS will support the day-to-day needs of various on-orbit avionic systems such as the Operations Management System (OMS), Communications and Tracking System (CTS), Thermal Control System (TCS), Environmental Control and Life Support System (ECLSS), Electrical Power System (EPS) and Guidance, Navigation and Control System (GN&C). The crew will use the DMS to interact with core subsystems and payloads. At Permanent Manned Configuration (PMC), it should be capable of supporting additional automation and robotics (A&R) functions to minimize intra- and extravehicular activities (IVA and EVA).

The SSF program also includes the ground control facilities, crew training and payload operation centers. It uses other NASA capabilities such as the Tracking and Data Relay Satellite System (TDRSS) and the ground wide-area data network (WAN). The DMS responds to mission control commands and experimenters through these communication links.

In order to support productive, safe operations in a changing environment, operations management should evolve to keep pace with changing science and engineering objectives; thus the requirements for DMS will also have to evolve. The top-level Program Requirements Documents (PRD) outline a strategy for dealing with this continual growth by including in the onboard operations requirements not only mission operations, but also planning, maintenance and training. This implies that during increments of growth, the DMS should support the immediate mission operations and the integration, testing and verification of a new core system (hardware and software), and should assist in maintaining all within safe operating envelopes.

1.2. Purpose of this Report

This report was prepared with two objectives in mind:

1. To carry out preliminary evaluations and recommend further tests to assess the performance, dependability and growth capabilities of the current DMS design

2. To propose directions and mechanisms for evolution and for technology insertion

1.2.1. Baseline evaluation– In order to minimize program risk and development costs from the outset, NASA has selected a specific set of design elements to optimize the use of common hardware and provide industry standard component capability for both core and payload users (ref. 8). These elements were, at the time of selection, considered modern; widely-used; commercial, off-the-shelf (COTS) industry/technology standards. Although individual processor and networking components are widely used in today's commercial marketplace, they have not been integrated into a working system for a real-time, spaceborne, fault-tolerant environment.

The primary baseline evaluation question, then, is whether the present DMS configuration can fulfill its role at each stage of the program, from construction to Permanently Manned Configuration (PMC) and Assembly Complete (AC) configurations. Furthermore, NASA should quickly determine if these components will accommodate sufficient expansibility to deal with the fault-tolerance, performance and functional evolution needs that will increase throughout SSF's 30-year life span.

Three major aspects of the DMS were evaluated: performance, dependability and limitations. First, the DMS's ability to meet real-time processing and communication deadlines, and current and potential bottlenecks, were evaluated. Second, to meet SSF's requirements for failure tolerance, DMS reconfiguration ability in response to node failure was analyzed. Third, architectural issues that hinder or prohibit technology insertion were identified. These preliminary analyses pointed to further tests that should be carried out as hardware and software prototypes become available.

1.2.2. Directions and mechanisms for growth and evolution– The present design of DMS hardware and software components should be perceived as only the first rung in a 30-year-long ladder of improvements in SSF computational capabilities. These capabilities should be designed, from the outset, to grow with the expanding needs and technological developments during the full lifespan of SSF. The DMS items that should evolve during SSF's lifetime are identified in section 2 of this report; the choice of items was based on the baseline evaluation. Candidates for growth, their associated costs and impact on performance and dependability will also be discussed. These recommendations will be further translated into (1) design accommodations required for DMS evolution, and (2) a scenario for DMS evolution and technology insertion.

2. Requirements for DMS Growth and Evolution

The 30-year life of the SSF will see many changes. With the baseline design of the DMS as a foundation, evolutionary systems will be added to provide and enable incorporation of additional functions and advanced capabilities. A number of candidate functions were identified in reference 9. This section reflects a number of findings of that report and has been influenced by other SSF studies. Supporting requirements presented herein are based on a study carried out at ARC (ref. 10). Appendix C presents excerpts from this study.

2.1 Integration of Subsystem Elements and Operations

The current DMS hardware and software architecture treats each subsystem as a relatively self-sufficient unit; together, they form a loose collection of systems. This design maintains a basic level of independence of elements and operations but tends to preclude further integration in the future. The incorporation of new technology in tightly coupled distributed computers, parallel computing, and pooled sparing would allow more efficient use of existing computer resources and would minimize weight and power requirements, both short-term and long-term. The DMS architecture must be flexible enough to incorporate such technology as it matures.

2.2. Instrumentation and Sensor Reconfiguration

The optimal placement of sensors used for control and diagnosis on complex structures like SSF is still an area of research. It is expected that the optimal configuration will change after experience with controlling and monitoring SSF has accumulated for a few years. This is especially true in systems for which there are plans for future automation. For example, a monitoring system using model-based reasoning may use fewer sensors placed at different locations and still reliably yield the same results as a conventional system. The architecture of the DMS must be sufficiently flexible to allow adjustments in the number and use of the sensors and effectors in each of the distributed systems.

2.3. Subsystem Status Monitoring and Fault Detection

The ability of the OMS application software to monitor subsystems and provide fault detection, isolation, and recovery (FDIR) assistance is determined by the design of hierarchical levels of fault containment from the orbital replacement unit (ORU) level to the integrated system level. Functional alternatives must be incorporated into the hardware and software design that will allow dynamic monitoring and reconfiguration when failures occur. Reliable intersystem communication of status information is imperative for fault management at the systems level. This includes timely communication of subsystem status even during reconfiguration of the optical network.

2.4. Onboard, Automated Element Test and Verification

Space Station Freedom will have a maintenance and repair logistics control problem not faced by previous NASA spacecraft. It will carry onboard a set of spare parts that must be in good working order at all times. Locating and testing this equipment could consume a major portion of crew IVA and EVA time. In addition, revision data on each spare part should be closely tracked to ensure the part's compatibility with new kinds of parts on SSF. Although this can be done for ground-based systems, the data should also be checked frequently onboard. Automating this task is well within current technology, given that sufficient computing power is available onboard.

2.5. Inventory Management Requirements

Inventory management will be significantly different for SSF than it has been for the Space Shuttles. Every item, consumables and nonconsumables, should be tracked for both usage and change in location. Current plans are crew intensive and require manual tracking and data recording. This function could be automated with current computer technology and programming methodology.

2.6. Onboard Training Needs

The training situation will be markedly different from NASA's current experience in manned space flight. At present, astronauts are extensively trained for every contingency possible on a short-duration mission. For SSF, the length of stay will be too long, and the contingencies too varied, to permit exhaustive saturation training. Astronauts should have training-assistance tools onboard to keep their skills sharp and to provide prompting when a seldom-used procedure is involved. These requirements for in-flight training will increase significantly as the number of crew members increases and as the payloads, missions, and configuration change over time. Irrespective of the complexity of the system,[1] the DMS should be able to provide sufficient computing power to accommodate these needs as specific requirements for onboard training are identified.

2.7. Payloads and Payload Operations

The science needs of SSF will evolve over time. The requirements to support growing and changing science demands must be recognized and facilitated in the SSF design. The primary requirement for science is the capability to collect and pass on to the principle investigators massive amounts of data. Traditionally, raw (unprocessed) data is simply telemetered down. Such is the design for SSF. With no capabilities allocated for additional onboard processing of data, valuable science data will eventually either flood the investigators or be discarded because of communication bandwidth and onboard storage limitations. Furthermore, many onboard housekeeping systems are also to be monitored and controlled by telemetry and commands sent through TDRSS. All this causes a contention for the downlink that has already been recognized by other space projects. The DMS should also be able to accommodate the removal and installation of payloads, any change in payload handling procedures, and increased data processing and storage needs. The DMS should have the capacity to eventually support significant payload processing onboard, or valuable data may be lost.

[1] The complexity of the system could range from a straightforward advisory expert system with information on each system element, to an interactive system that provides facilities for online teaching and testing of astronaut skills.

2.8. EVA/IVA Requirements for Maintenance

The external maintenance task team (EMTT) (i.e., the Fisher-Price Report) established that the extravehicular maintenance requirements of SSF over its lifetime could alone easily overwhelm the available EVA resources (ref. 11). Two interdependent means of preventing such an oversubscription have been recommended:

1. Increase the use of telerobotics for external activities

2. Increase the level of automation for many housekeeping and bookkeeping functions currently assigned to the crew

Although a wide variety of time- and labor-saving activities are within today's software technology capability, the initial configuration of the DMS does not implement all of them. The DMS should be expandable to allow a higher level of automated assistance to the crew in performing IVA housekeeping and subsystem maintenance tasks.

2.9 Advanced Automation Capabilities

Many of the advanced automation capabilities being developed to support various activities on SSF focus on some type of FDIR. Three advanced automation projects targeted for implementation on the SSF are Automated ECLSS, Power Management and Distribution (PMAD), and Advanced Automation Network Monitoring System (AANMS). Information about each of these programs is included in Appendix C. Since the computational capabilities requested for these programs far exceed the capabilities allocated by the DMS, end functionality was scaled back and a future need for additional capability was noted (ref. 10).

3. Design Overview

3.1. Assumptions

The overview of the DMS presented in this section is based on the information presented at PDR3 (refs. 1-6), plus documentation produced by various Program board decisions and directives. Because of the pace of changes and the lag in documentation, recent changes (mostly reductions in scope) in the design are not reflected in this report. The configuration evaluated represents the DMS baseline of December 1990. A later report will consider the baseline established after the first quarter of 1991 scrub activity.

3.2. The Data Management System Overview

The DMS is the subsystem in SSF responsible for integrating information onboard. It provides integrated data processing and communications for both the core functions and the payloads. Via the multi-purpose application consoles (MPAC), the DMS provides the crew with visibility and control for operating modes (nominal and abnormal). It also provides an interface to virtually all other subsystems onboard, including the CTS, which links the digital data to the ground. The DMS concept is based on a local area network of loosely coupled data-processing stations. Growth and evolution is based on giving the DMS optical network an initial high capacity plus the ability to add computing elements incrementally subject to the available weight, power, and volume of the evolving SSF.

As shown in figure 3-1, the baseline hardware design (for PMC) nominally consists of 17 standard data processors (SDP), 2 mass storage units (MSU), 7 MPACs, 63 multiplexer-demultiplexers (MDM), and their communication links: the DMS Optical Network, local buses, and high-rate links (HRL) (only to provide more payload telemetry capacity to the ground). Each functional component is built into one of a small number of chassis that are ORUs with two standard backplane buses (Multibus II and MicroChannel), standard interfaces to networks and buses as appropriate, and standard connections to power input and thermal sinks. The communication networks and buses in the DMS conform to popular standards appropriate to their rates: 1553B (1 Mbps), 802.4 (10 Mbps), FDDI (100 Mbps), and the passive fiber-optic cable and patch panel for HRLs. The DMS also includes the Emergency Monitor and Distribution System (EMADS), which is purposely independent of the DMS network, and the Time Generation Unit (TGU) and Time Distribution Bus (TDB).

The AC version of the hardware design is shown in figure 3-2; it will have the same physical interfaces that are installed and tested at PMC. The primary hardware differences will be the number of active functional components. The network and buses are reconfigurable starting at PMC using one of two patch panels. The media for the HRL, FDDI, and 802.4 links will all be installed and tested at PMC and will be all the same type of fiber-optic connectors and cable. All sensors for AC will also be installed and tested at PMC unless they reside in a deferred ORU.

One of the new ORUs proposed for AC is the Bus Network Interface Unit (BNIU) to connect payload MDMs via 802.4H to the DMS optical network. However, a review item disposition (RID) is pending to move such capability to the PMC phase, so it may be available earlier. In any case, no change is planned for the core MDMs. Another change proposed for AC is to double the total throughput capability of the DMS optical network by splitting the PMC configuration into two separate rings: one for core systems and one for payloads. Each would have their own gateway to the ground via CTS, but inter-ring command and control would be provided via a redundant bridge unit. A third change proposed for AC is the addition of a link from the TGU to a Global Positioning System Receiver in the CTS.

The design for both PMC and AC configurations are specified in detail in the DMS Architecture Control Document (ACD) (ref. 4) and the Contract End Item (CEI) Specification (ref. 5).

The baseline software, aside from individual applications, has many modules that further define the character and capabilities of the DMS. The Computer Software Configuration Items (CSCI) include the following:

1. Operating System/Ada Run-Time Environment (OS/Ada RTE)
2. Network Operating System (NOS)
3. Standard Services (STSV)
4. User Support Environment (USE)
5. Data Storage And Retrieval (DSAR)
6. System Management (SM)
7. Master Object Data Base (MODB)
8. Operations Management Application (OMA)
9. Interface Software

The documents that detail the functional requirements of each item are listed in table 3-1.

3.3. Hardware Component Description

Table 3-2 lists the major components of the DMS, and a brief description follows. A more detailed description is provided in the CEI specification (ref. 5).

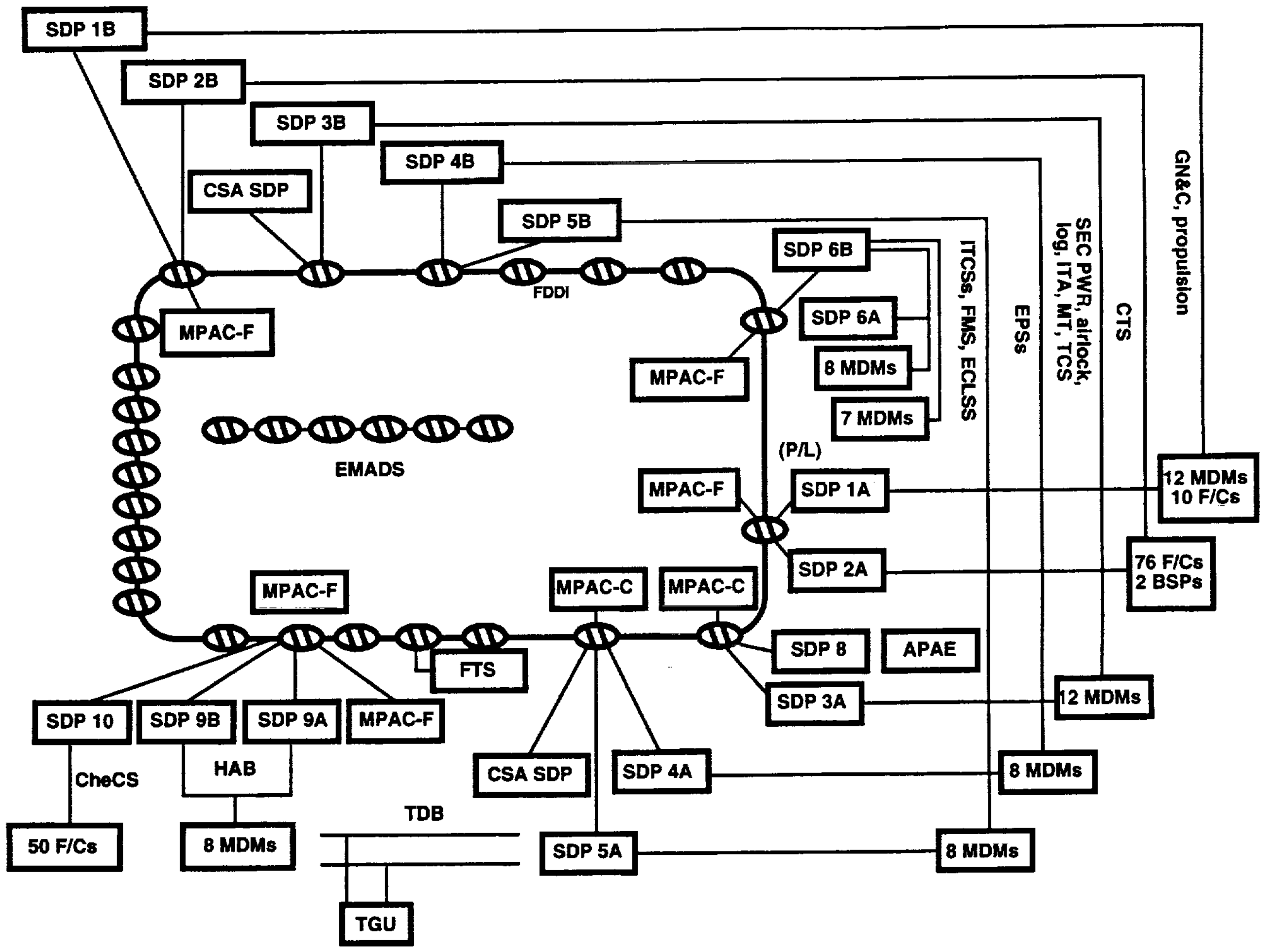


Figure 3-1. Data management system configuration at PMC.

3.3.1. DMS optical network– The backbone network for the DMS uses the FDDI protocol[2] to connect its components. FDDI is a fiber-optic, dual-redundant, counterrotational, token-based ring network with a channel data rate of 100 Mbps. In a token-based ring network, a collection of stations or nodes are connected by one-way transmission links to form a closed loop. Each node transmits or repeats data to its immediate downstream neighbor, so that information passes around the ring in a circular fashion. A single token or special bit pattern circulates among the nodes. When a node has data to transmit, it removes the token from the circulating pattern. The node then transmits one or more frames while it holds the token. When it is finished transmitting, the node releases the token by transmitting it to the next node on the ring. A set of timers internal to the node determine how long it can transmit before releasing the token.

The advantage of a token-based protocol is that only one node attempts to transmit at a time, and there is no interference between nodes contending for the network channel. Therefore utilization of a heavily loaded network can approach 100%. One disadvantage is that if the token becomes lost because of a transmission error or faulty node, no node can transmit. The protocol should be able to recover from this situation.

The FDDI specification includes two separate fiber-optic cables for redundancy. The second ring is used if the primary ring fails. Packets circulate in opposite directions on the two rings so that if both rings fail at the same node, one long ring can be constructed by having the nodes on

[2] FDDI conforms to the OSI reference model of the ISO, which subdivides a network system into seven layers. Each layer offers a set of services to the layers above it and utilizes the services of the layers below it to perform its functions. Each layer communicates with its peer or corresponding layer on another node by sending or receiving a protocol data unit (PDU). The FDDI standard deals mainly with the physical layer and the lower half of the data link layer, the medium access control sublayer. It also deals with the Station Management functions for the physical and link layers; these functions are critical to robust reconfiguration under failure conditions.

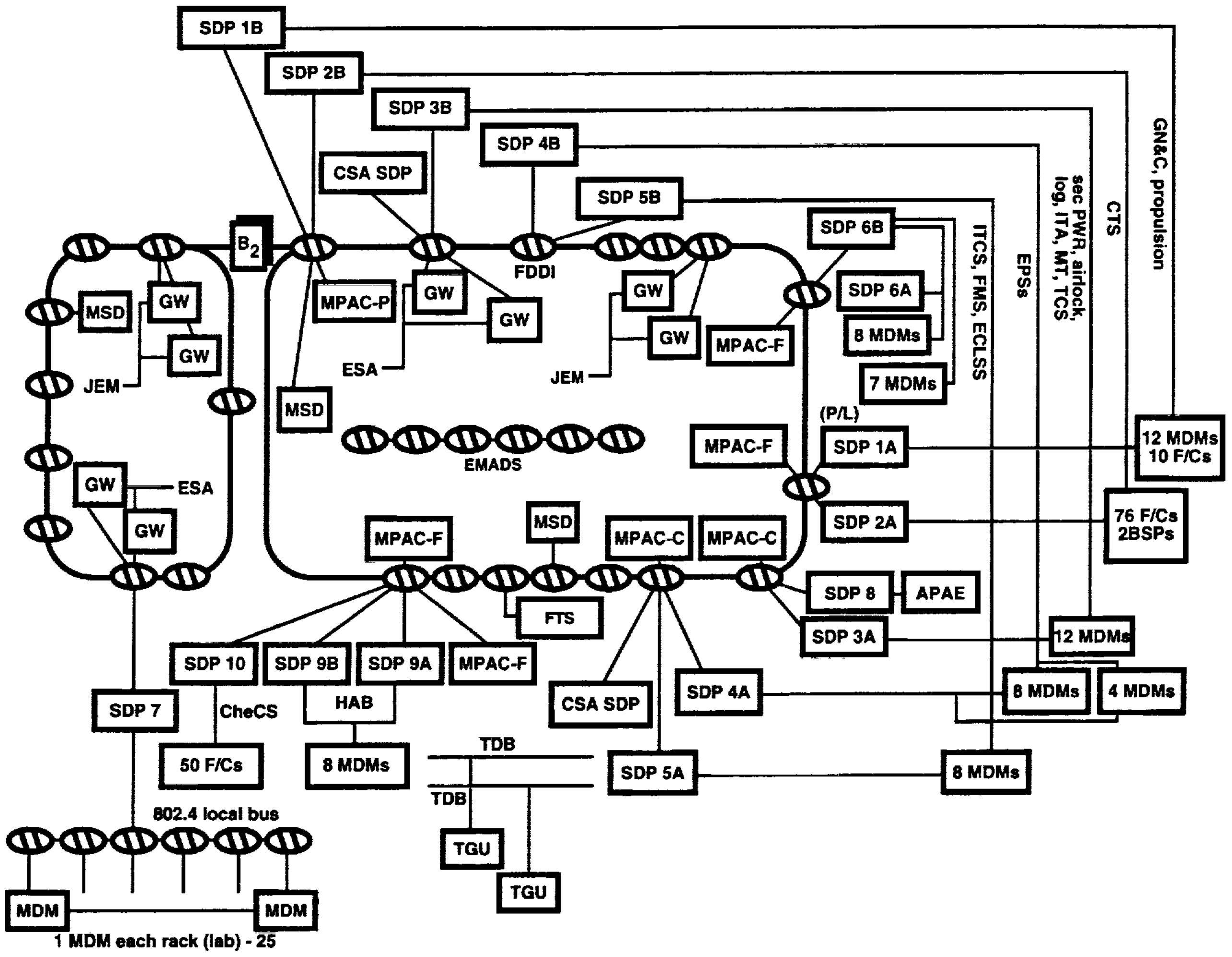


Figure 3-2. Data management system configuration at AC.

Table 3-1. Software design review documents

CSCI	Software Requirements Specification (SRS)	Software Preliminary Design	Software Test Specifications
OS/Ada RTE	MDC H4189	MDC H4352	MDC H4545
NOS	MDC H4188	—	MDC H4548
STSV	MDC H4191	MDC H4350	MDC H4547
USE	MDC H4192	MDC H4348	MDC H4544
DSAR	MDC H4187	MDC H4353	MDC H4549
SM	MDC H4190	MDC H4351	MDC H4546
MODB	MDC H4481	MDC H4354	MDC H4550
OMA	MDC H4245	MDC H4722	—

Table 3-2. Power and weight summary[a]

Component	Power (W)	Weight (lb)
SDP	130	37
EDP-A	28	2
EDP-N	20	1.5
NIA	35[b]	2.8
MSU	160	20
RC	7	25
TGU	50	37
GW	65[c]	40
1553B	20.2	TBD
MDM[d]	35	26.1
LLA	3.8	1.7
HLA	11.0	1.7
AIO	4.7	1.4
DIO	23.5	2.0
SDO	2.5	1.7
SDIO	5.8	1.3
MDM USER cards:		
1553	7.5	1.3
for I/O bus	1.5	
for USER bus	1.5	
MPAC-C	900	
Electronic Chassis	250	189
Display Console	650	
MPAC-F	1015	
Electronic Chassis	265	199
Display Console	750	

[a]Assumes 12 Meg DRAM on SDP/MPAC's
[b]With maximum memory
[c]Without Foreign IF card
[d]With chassis, power supply, and controller only

either side of the failure logically join the two cables by doubling back packets received from one cable onto the other.

FDDI permits several levels of service class with different priorities for use on the ring. The classes are synchronous, asynchronous, and immediate. Immediate messages have the highest priority and are used only in extraordinary circumstances by the Station Management function. A node need not wait to capture the token to transmit an immediate frame. Synchronous messages have the next highest priority and are designed to support guaranteed access times for periodic data, such as times required for core telemetry. Asynchronous messages have lower priority and are transmitted as time allows. Up to eight different levels of asynchronous messages may be defined, with different priority levels and bandwidth allocations defined for each. The DMS as specified does not support synchronous messages. This has serious implications for system-level fault tolerance, as will be detailed in section 4.3.1.

3.3.2. Network interface unit– The network interface unit (NIU) is shown in figure 3-3; it is an interface device between the high-speed optical network (FDDI) and the ORU host components (SDP, MSU, GateWay, MPAC, etc.). On the host side, the interface is a Multibus II backplane. A network interface adapter (NIA) card with dual optical transmitter/receivers connects with the redundant ring concentrators (RCs) that form the rest of the DMS optical network. The interface adapter receives time code on a separate electrical connection from the TDB. The NIU contains an embedded data processor (EDP) to provide, in addition to the time service, the full ISO/OSI layered NOS services for applications software on the host. The total power needed for the NIA plus the EDP is estimated at 55 watts.

The NIU is required to provide bidirectional throughput of 5 Mbps for full NOS services assuming 2048-byte messages, and 10 Mbps with 10% of the traffic requiring full services and 90% requiring only link (FDDI) service. As shown in table 3-3, it is also required to provide four priorities of message traffic of average and 95% delay under "normal conditions" with grade #2 service (possible errors and dropouts). In addition, the NIU should perform self-initialization on power-up, built-in tests for the DMS system management, and station management functions for the FDDI.

Table 3-3. NOS latency requirements (ref. 5, pp. 3-256)

Traffic priority	Mean delay (ms)	95% delay (ms)
Background	< 80	< 150
Normal	< 50	< 80
Expedited	< 20	< 25
Emergency	< 20	< 25

3.3.3. Ring concentrator and FDDI network media– The ring concentrators[3] (RCs) and FDDI network media provide the physical-layer connections for the DMS optical network, which has its link-layer protocol implemented via the NIUs in the ORUs. Together, the RCs and

[3] The term "ring concentrator" is a misnomer with respect to the FDDI protocol and should be changed to avoid confusion in failure analysis studies. RCs are a unique ORU aggregate of bypass switches and optical regenerators. Because of the unique design, the Station Management function cannot isolate a failure to one RC.

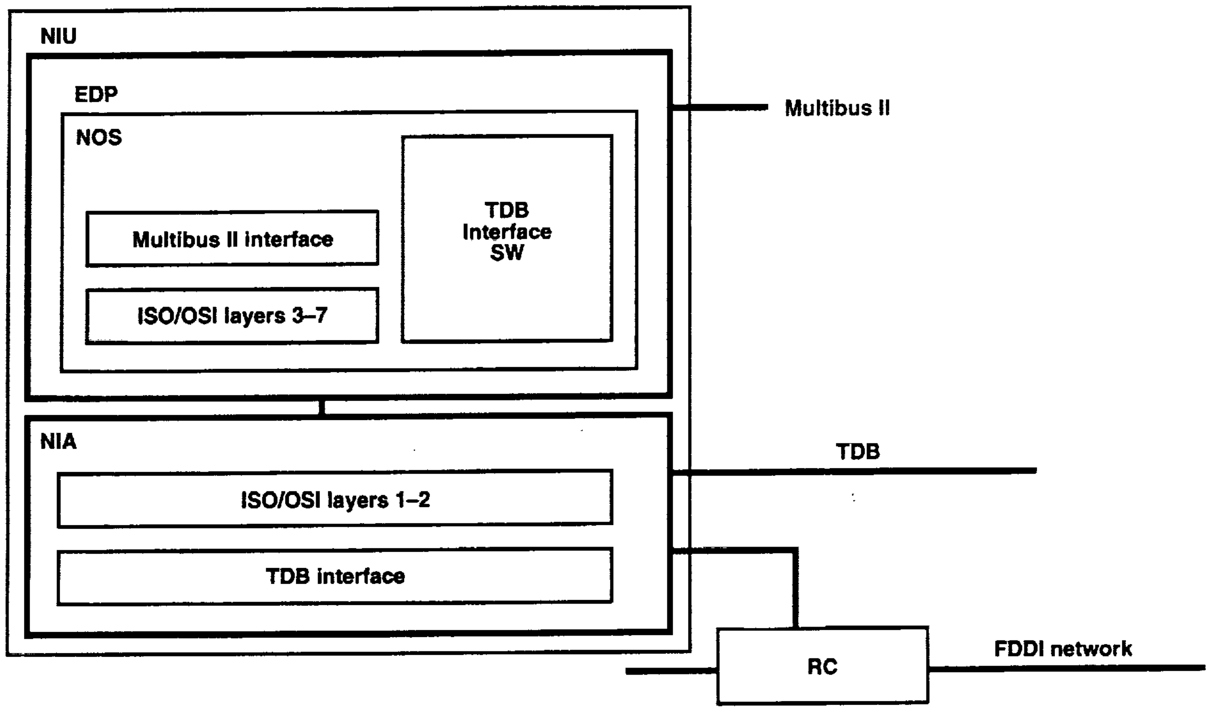


Figure 3-3. The network interface unit.

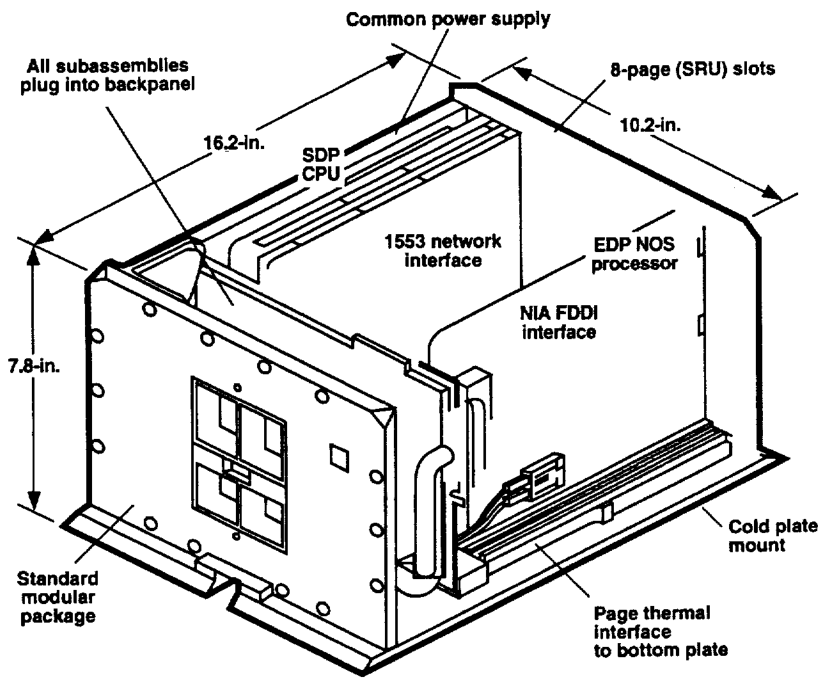
media form the topology of the FDDI dual-redundant, counterrotating rings that support the instantaneous digital rate of 100 Mbps. The RCs provide the optical switching to connect ORUs to the rings when they are "good" and to bypass them otherwise. The RCs also provide active regenerators where needed to overcome connector and switch losses and to maintain optical signal strength within the limits specified. The RCs do not provide any built-in test capability for fault detection, isolation, and status reporting; instead, they rely on station management in the NIUs for that function. The optical medium is radiation-hardened, multimode fiber, 100/140 microns, with a graded index of refraction. The modal bandwidth is 200 MHz*km (implies 2000 MHz at 100 meters).

3.3.4. Standard data processor– The SDP is a general purpose computer that provides data processing and temporary storage for SSF systems and payloads. Various configurations are possible via different numbers of EDPs, different memory sizes, and different types and numbers of interface cards on an eight-slot chassis (shown in fig. 3-4) with a Multibus II (IEEE 1296) standard backplane. The present EDP (shown in fig. 3-5) is 386-based, with 3.1 million instructions per second (MIPS) and 4 megabytes (MB) of random access memory (RAM) for OS/Ada RTE, standard services, and applications software. As shown in figure 3-6, the SDP can communicate with other ORUs via an NIU to the DMS optical network, and some configurations can communicate with system and payload components via one or more bus interface units (BIUs) to a 1553B local bus (or a 802.4 local bus at AC). It can process time (via the NIU) to a precision of 10 μsec, and will provide a watchdog timer and built-in tests to support health and safety monitoring.

3.3.5. 1553B local bus– The MIL-STD 1553B data bus is a serial multidrop configuration constructed with shielded twisted-pair cable, transformer-coupled to the drivers/receivers at each drop. The data rate is 1 Mbps, self-clocked using Manchester II encoding. Three types of entities may exist on the bus: a bus controller, a remote terminal, and a bus monitor. Only 32 entities may exist on one bus (normally mostly remote terminals). The bus protocol is command/response, with only the bus controller able to initiate a command.

Under the command of the bus controller, data frames may be transferred between the bus controller and a remote terminal and also between remote terminals. The bus controller allows up to 12 μsec for a remote terminal to respond to a command and at least 4 μsec separate messages (intermessage gap time).

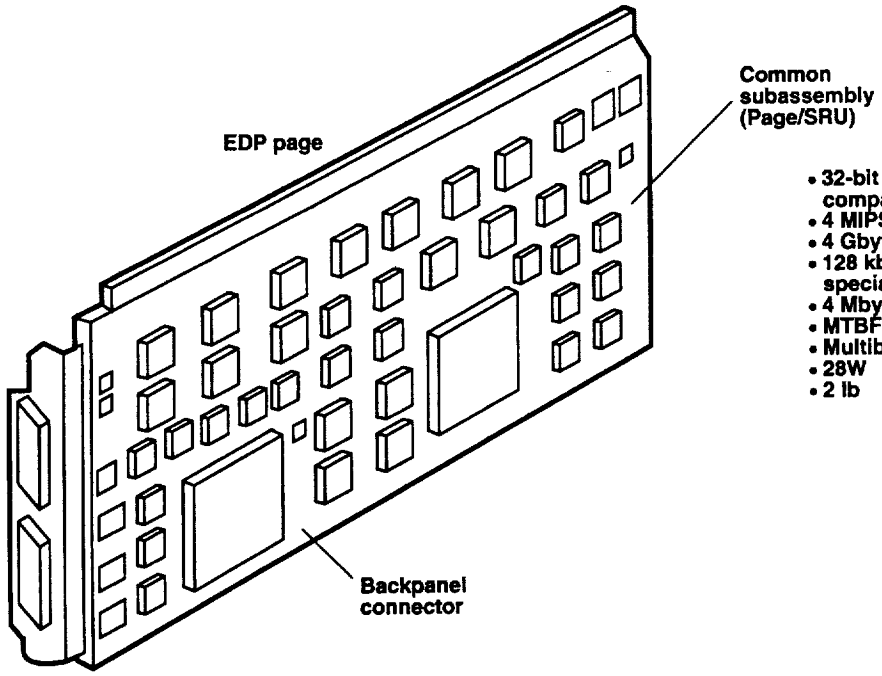
Because of the sync bits, parity bit, and command/response protocol, the maximum throughput of 1553B is 748 kbps for the maximum 32-word (64-byte) messages. The throughput drops sharply with decreasing message size, dropping to 210 kbps for one-word messages. The



- 2 EPDs (CPU plus NOS)
- FDDI network interface
- MIL-STD-1553B bus interface
- MTBF 32,000 hours
- 95% built-in-test coverage
- Radiation tolerant
- VHSIC class parts
- 129W
- 34 lb
- 0.6 ft^3

(Current SDP uses 4 of 8 slots)

Figure 3-4. The standard data processor.



- 32-bit 80386 ISA for ground/onboard compatibility
- 4 MIPS
- 4 Gbytes linear addressing
- 128 kbytes ROM for start-up and special functions
- 4 Mbytes DRAM with ECC and scrub
- MTBF 168,000 hours
- Multibus II and MCA backpanel interface
- 28W
- 2 lb

Figure 3-5. The embedded data processor.

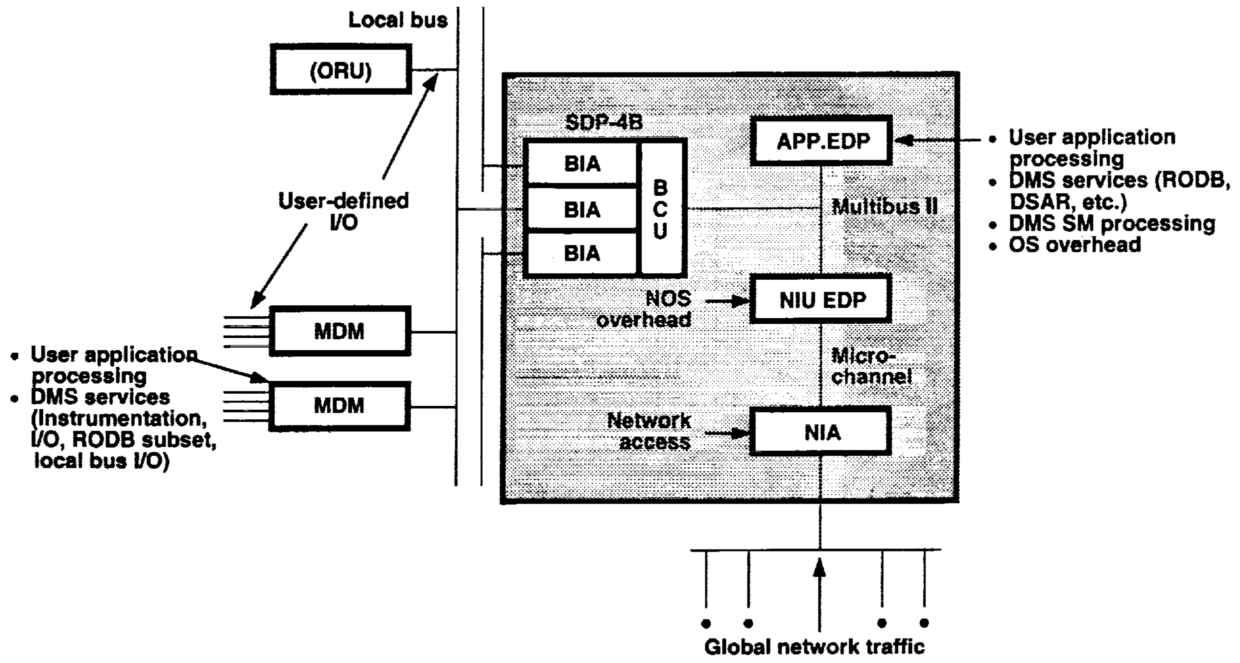


Figure 3-6. Schematic diagram of a sample SDP (SDP-4B).

latency is small and well defined, and it is relatively easy to define faults, but the bus controller (in the SDP) is a single point of failure for the whole bus.

3.3.6. The controller multiplexer-demultiplexer– The MDM serves as both an interface to various inputs and outputs (I/O) with the DMS and as a general controller for both the DMS and its external users (see fig. 3-7). I/O can be via standard sensor/effector cards or via unique devices of core systems or payloads on the SSF. The MDM digitally sends and receives all data and commands to other DMS-connected entities via the MIL-STD 1553B local bus. It uses the local bus to communicate to its host SDP and via its host to other DMS components. The form of the signal sent to sensors/effectors or to unique devices is determined by the I/O cards or user cards in each instance. Thus a large degree of (module) flexibility is available as long as the I/O cards or user cards conform to space qualification standards and to their respective buses.

The MDM processor functions as an I/O controller via firmware provided by the DMS; it also executes user application software (UAS). An I/O database that automatically links to the runtime object database (RODB) in the host SDP is included in UAS services. The MDM provides general purpose, real-time I/O control services including optional time tagging of input data, multiplexing, bit packing, demultiplexing, and built-in tests for health status reporting.

The MDM uses a 386sx processor, with 0.7 MIPS, 1.25 MB of RAM, and 0.5 MB of electronically erasable programmable read-only memory (EEPROM) available to users, and the 1553B local bus. An IEEE 802.4-compatible MDM is being considered for AC, but for now this more flexible MDM bus is deferred. MDMs will be available in three chassis sizes: the largest has 15 I/O card slots, one 386sx card slot, and one (386sx) controller; the smallest has only 3 I/O card slots along with the 386sx slot and controller. See table 3-2 for I/O card type, weight, and power. (Power ranges from 72 to 31 w/unit.)

3.3.7. Mass storage unit– The MSU is a secondary memory device (magnetic hard disk) for bulk storage. The MSU provides the processing, memory, and I/O capabilities to service commands requesting the sending or retrieval of information to and from its storage and to transfer the information to and from a designated destination. The MSU can communicate with other ORUs via an NIU to the DMS optical network. It provides processing capability equal to an SDP and provides nonvolatile read/write random access to 250 MB of formatted memory. Data storage and retrieval via the MSU is limited by the access latency and transfer rate of its NIU to the DMS optical network (see section 3.3.2). For crew-initiated access, disk storage has been added to two of the MPACs to speed loading of applications. (Local disk storage has an access time of 12 msec and an I/O rate of 32 MB/sec.)

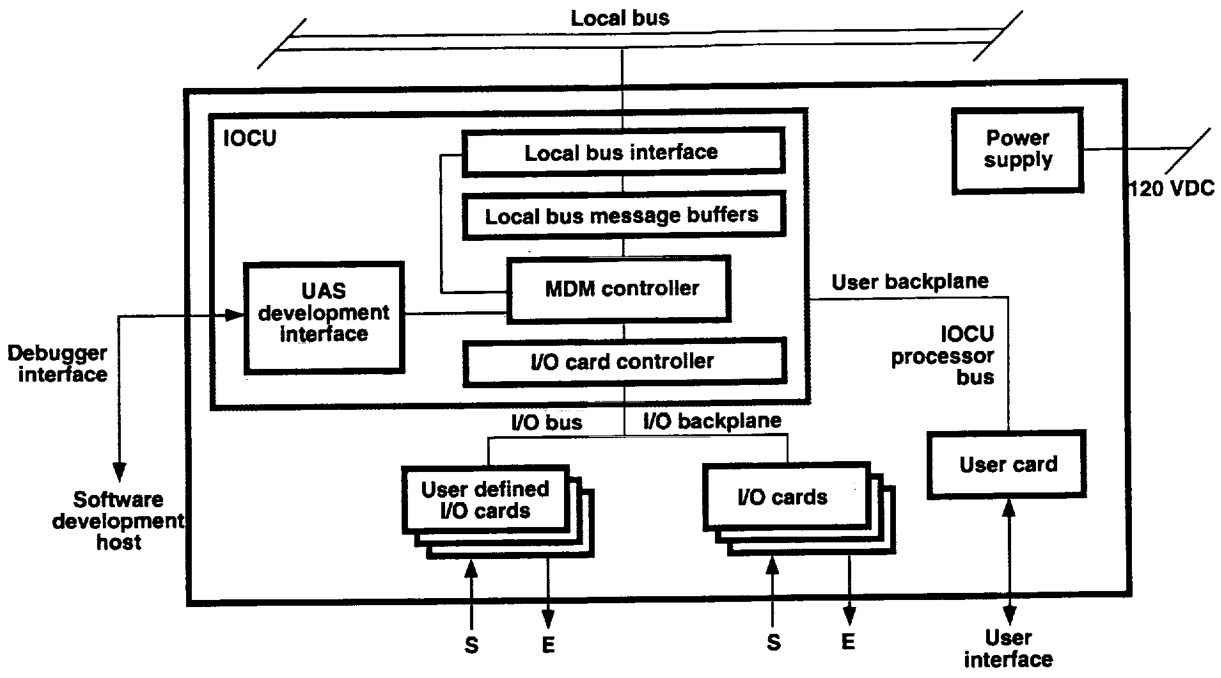


Figure 3-7. Controller multiplexer-demultiplexer functional block diagram.

3.3.8. Multi-purpose application console– The MPAC (fig. 3-8) is provided in three configurations: fixed (MPAC-F), cupola (MPAC-C), and orbiter (MPAC-O). The MPAC-F and MPAC-C can communicate with other ORUs via an NIU to the DMS optical network. The MPAC-O uses the BIU 1553B interface to communicate with other ORUs. MPACs provide processing capability equivalent to an SDP. Each has card-level processing units specifically for operator interface control and for graphic, text, and video display. MPACs interface with a keyboard, a trackball, and a hand controller. (MPAC-F and MPAC-O have left- and right-hand controllers.)

3.3.9. Gateway– The GW is an ORU interface device that provides connectivity between the U.S. DMS optical network and a non-U.S. optical network or CTS for which protocol conversion is required. The GW performs two-way information transfer at a combined rate of 10 Mbps for any traffic mix. The protocol conversion is performed to a common interface standard at the network layer (#3) of the ISO/OSI reference model. The GW also provides an interface and processing for the precision time and frequency information, and provides built-in tests for fault detection, isolation, and status reporting.

3.3.10. Time generation unit and time distribution bus– The TGU and TDB provide a precision time and frequency source for all subsystems and payloads. The TGU receives and processes time code from the C&T reference and the Global Position System receiver (at AC) to generate a baseline time and frequency reference data stream. It also provides built-in tests for fault detection, isolation, and status reporting. The TDB is an independent bus using twisted/shielded pair wiring (Electronic Industries Association std RS-422A) that connect ORUs to the time and frequency digital data. It is designed to preserve a time accuracy of 1 μsec.

3.3.11. Emergency monitoring and distribution system– The EMADS is an independent function for monitoring and annunciating SSF emergency conditions that could threaten the life of the crew. EMADS is a hardware-only system for backup to the DMS caution and warning system. It includes sensors for monitoring the emergency conditions plus annunciation hardware for each condition and connecting links. At present the conditions defined are fire/smoke and rapid depressurization. Scarring is provided for atmospheric contamination. A manual test is provided on each annunciator panel to test (1) the station-wide hardware, (2) the power supply, and (3) the lamp and tone generator for each annunciator.

3.3.12. High rate link and patch panel– The HRL media and patch panel uses fiber optics to send point-to-point digital data between payloads, onboard payload processors, etc., and the CTS. The patch panel ORU provides no less than 100 ports compatible with the HRL media and provides for manual reconfiguration of links for half-duplex data transmission via passive fiber optics. It is designed to support rates higher than those that can be handled by the DMS. Instantaneous rates of at least 100 Mbps will be supported between any two points. (The

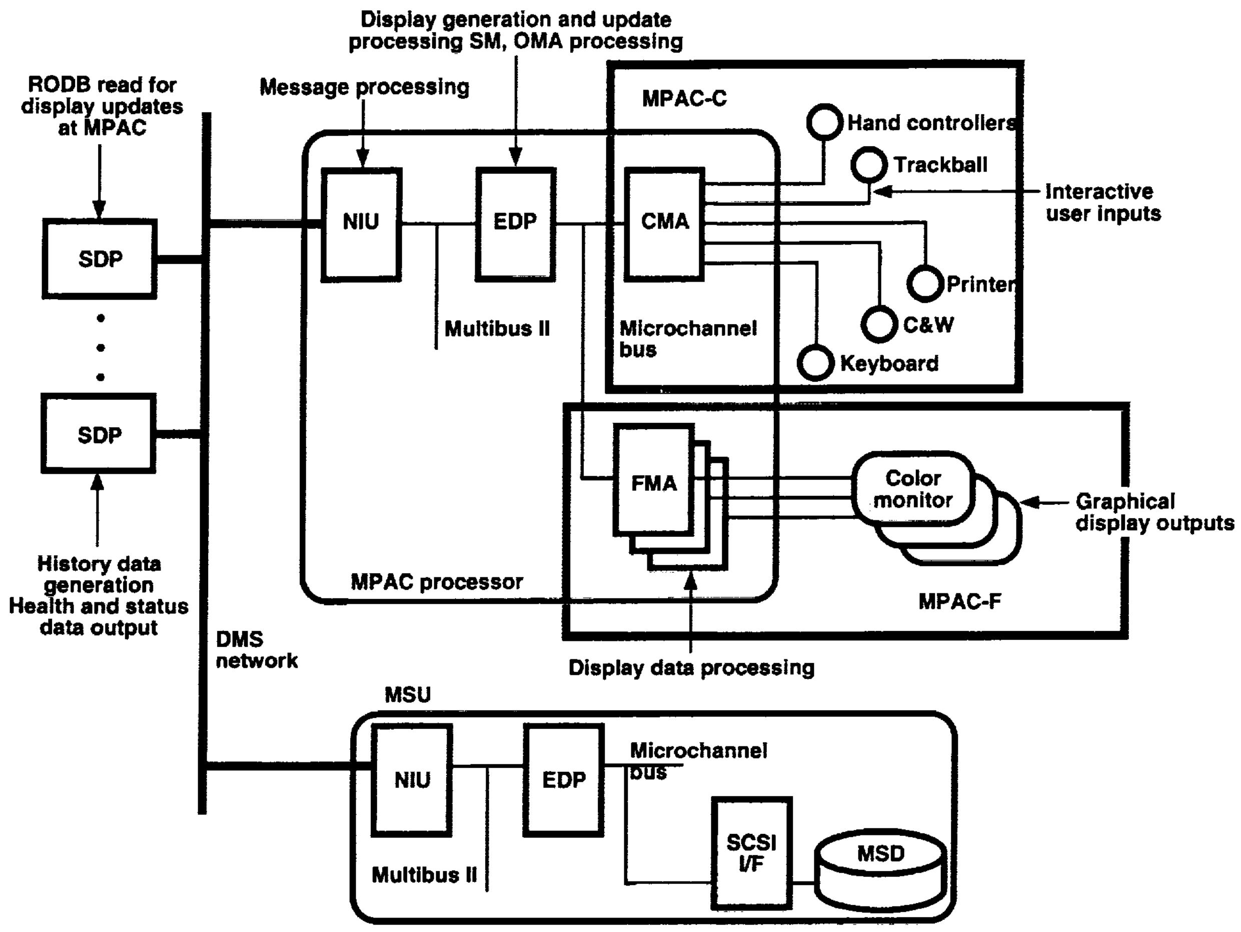


Figure 3-8. MPAC model overview with MSU on network.

media will have the same physical characteristics as the FDDI media.) No less than 8 ports will connect to the CTS, and at least 90 ports are distributed to various payload locations throughout SSF.

3.4. System Software Description

The requirements and preliminary design of these CSCIs have been published by International Business Machines (IBM) and McDonnell Douglas Space Systems Company (MDSSC) (as listed in table 3-1). The commercial software that will support each CSCI, however, has not been identified, except the operating system. This section of the report therefore concentrates on the Lynx real-time operating system (LynxOS), which was selected to be the DMS EDP operating system. The Ada RTE and Network Operating System (NOS) will also be discussed, because their selection has far-reaching implications regarding evolution.

3.4.1. The Lynx operating system– LynxOS is a Unix-based operating system developed by Lynx Real-Time Systems, Inc., located in Los Gatos, California. LynxOS was designed in a "clean room" environment and is not subject to American Telephone & Telegraph (AT&T) licensing terms and royalties. The primary features of LynxOS include (1) real-time performance, (2) real-time scheduling, (3) ROMable kernel, (4) memory locking, (5) Portable Operating System Interface for Computer Environments (POSIX) compliancy, and (6) contiguous file facilities.

Exact specifications regarding the multifunction real-time operating requirements for SSF have not yet solidified. In general, a real-time operating system is one that has some ability to respond to asynchronous, external events, delivering a required level of service within a bounded response time. This required level of service implies the ability to support time-critical SSF application programs. Unlike general purpose systems, real-time systems are designed around worst-case latency and determinacy (or predictability of response). A bounded response time means that both the average and the worst-case response times are deterministic. Lynx, Inc., claims that the average response time for a 20-MHz, Intel 386DX–based computer is 50 μsec, and the worst-case response time is 250 μsec.

Standard Unix is designed as a general purpose, time-sharing, multi-user operating system. It alters priorities dynamically to enforce equitable access to system resources and to optimize time-sharing performance. The priorities of LynxOS tasks, on the other hand, are absolute and under user control. The LynxOS kernel does not alter priorities unless instructed to do so. A priority in LynxOS is a number from 0 (lowest) to 31 (highest), with a default priority of 17. This differs from AT&T System V, where "nice values" range in the opposite direction from 39 (lowest) to 0 (highest), with a default value of 20. When an event occurs to cause LynxOS to reschedule, the highest priority task is run until completed. If there is more than one task with the same highest priority, then the tasks are scheduled round-robin with a configurable time quantum.

Lynx Real-Time Systems, Inc., claims that, depending on the configuration, the LynxOS kernel requires 130 to 160 kbytes of memory for code and initialization data, and can easily be placed in ROM. This feature is important for remote real-time sensors or communication nodes. To ensure real-time response and rapid interprocess communication, LynxOS also allows tasks to be locked in memory so that they will not be swapped out to disk.

3.4.2. Ada run-time environment– The Ada RTE may be used to solve some of the problems with the POSIX kernel. The final selection of an Ada compiler is still pending. DDC-I and Verdix appear to be good candidates for this, but POSIX and the Ada RTE are so dependent on each other that the final selection will have to rely on the choice for the OS kernel. Also, the pending IEEE standards for real-time extensions and POSIX/Ada should be considered if the goal is complete standardization. The hardware will also play an important role in this decision.

The Ada RTE that is chosen should offer a significant number of products for development of user interface and communications software. Tools should be acquired to make this easier, together with documentation and examples for writing device drivers. There may be considerable lag time before these tools are available for interfacing between the Ada RTE and POSIX or the LynxOS.

3.4.3. Network operating system– The proposed DMS optical network is based on the FDDI link protocol standard, so the NOS should be compatible with FDDI. The subcontractor for the NOS has been selected, but its detailed design or proposed characteristics have not been available for review. Details on the justification for using a language other than Ada should be examined to determine the necessity for this waiver. The network operating system will be dependent on the NIA hardware and on the I/O bus selected. Whether Ada or assembler is used for this will naturally affect the ease of evolving the NOS.

4. Evaluation of the DMS Baseline Architecture

4.1. System Performance

4.1.1. Evaluation approaches and criteria– The primary problem facing baseline evaluation stems from the fact that the DMS architectural freeze decisions have to be made before the DMS test bed kit components and modules can be fully developed and integrated, i.e., before *actual* tests on the full DMS hardware and software complement can be performed.[4] System simulations will have to be used at this juncture to evaluate present DMS design. With constant change request (CR) activities, RID, and scrub activities affecting nearly all DMS components, the interpreted requirements and design assumptions underlying these simulations could easily become uncertain and even misleading. Furthermore, real systems are notorious for exhibiting hidden flaws that cannot be uncovered using simulation. An additional problem is that the actual avionics software (e.g., TCS, ECLSS, EPS, GN&C, etc.) has not yet been completely designed or coded.

The DMS performs many overlapping real-time functions, and the modular nature of the design provides many options for matching resources to functions. Tests for performance must carefully define the usage scenario, the hardware and software configuration, and the functions measured. Thus it is expected that some of the early analyses and results will be controversial. The system CEI specification has very few explicit quantitative performance requirements at the system level (see ref. 5, table 3.2.1-1). Early analysis should focus on the worst-case scenarios for a baseline configuration; furthermore, obtaining agreement between the users and developers on the details is as important as the list of assumptions and

[4] Discussions with IBM Federal Systems Division, Houston, made it clear that "Functional Equivalent Units" (i.e., non–MIL-Spec, non–Space Qualified) kit versions will not be fully integrated and available until mid-1992. For the flight-qualified version of the DMS we will have to wait until mid-1993. Test beds are of considerable value when they can provide enough data to make key hardware, software, and application design decisions. In this case, test bed equipment will arrive only *after* the key DMS design freeze decisions are made.

the numeric results. For growth and evolution capability studies, the worst-case scenarios associated with system management for the SDPs and the networks seem to be the most time critical, so these were the focus our preliminary analysis. The latency in receipt of remote status data under a failure detection, isolation, and recovery scenario is an example used in many early simulation analyses at the NASA Ames Research Center (ARC) Advanced Architecture Test Bed.

Two approaches are currently used at ARC for evaluating DMS performance, simulation models, and test bed experiments. Three simulation tools are being applied in-house:

1. Local Area Network Extensible Simulator (LANES) link protocol simulation, which has a high-fidelity model of the FDDI protocol and a flexible run-time parameterization of configuration, interfaces, and workload

2. Distributed Processing/Network Simulator (DPNS), which has a higher level modeling capability than LANES for setting configurations as networks and links with various processor, memory, and sensor/effector options at each node, and a separate workload description method for assigning a hierarchy of parallel and serial jobs to the resources

3. An Experimentation Environment for Concurrent Systems (AXE) workload programming tool (ref. 12) for dynamic resource allocation studies, which allows the most flexibility in defining configuration and workload granularity.

Development of multiple models are planned with these tools to study critical performance and design evolution issues

4.1.2. Preliminary results–

4.1.2.1. The DMS optical network: The DMS optical network uses the FDDI protocol. However, because the NIU design implementation has not been described in sufficient detail, it cannot be determined to what degree it adheres to the FDDI standard. Technical papers have provided characterizations of the protocol performance (including some based on ARC analysis and simulations). Because the protocol is complex and has been only partially implemented in commercial interfaces to date, its subtleties can easily be misunderstood. For example, the DMS CEI specification does not require a deterministic message-passing capability that can be provided under the FDDI synchronous mode; a PDR1 and PDR3 RID has been submitted to flag this weakness. Also, the terminology for the RCs conflicts with the FDDI standard (see section 3.3.3 footnote). Although prototype DMS kits are referenced as having an FDDI network, they have neither the redundancy nor the station management of the standard, and will not have those features in the official DMS kit releases of 1991 either.

The intent for the DMS is for all stations to be Class A stations, i.e., all stations are attached to both the primary and the secondary ring. However, the physical connections within the standard data processors are not described in enough detail to allow evaluation of "wraparound capabilities" of the system. The standard FDDI physical connection pairs a receiver on one ring with a transmitter on the other ring. There are two such physical entities (called PHY) in each Class A station, according to the FDDI standard. It is not clear that the NIA for the SDP has been designed according to these specifications. In addition, some station management objectives that are outside the FDDI standard are being defined in an attempt to satisfy the system requirement for critical operations with two failures. At this point, the development is at risk because modifications to a commercial standard are being made without any actual test data or operating experience. It is possible for such modifications to invalidate all claims of performance and reliability that are based on analyses of the standard and to invalidate portions of the standard that deal with configuration management. Insufficient detail concerning the DMS optical network has been provided for us to evaluate the impact of modifications to the FDDI standard that have been proposed for the DMS.

4.1.2.2. SDP performance and loading on the 1553B: Preliminary simulations of DMS and traffic models have been run by MDSSC using some of the baseline configurations and workloads; they were reported in reference 7. Although a thorough review of the cases is still pending, ARC simulation models appear to agree qualitatively with the MDSSC results. Both the application EDP and the local 1553B bus were readily overloaded under baseline unit assumptions and under some sensor and effector traffic models (see appendix B).

An experimental evaluation of the performance of the 386 was also carried out. An IBM PS/2 Model 70-A21 computer, which has 25-MHz 386 and 387 processors and a 64-kbyte external cache was used for the performance evaluation (ref. 13). Comparative tests were run with the cache disabled, because the current EDP specification does not mention using any cache for its computer system (ref. 14). The instruction mix and relative usage weights for the 386 MIPS Instruction Benchmark were specified in the Configuration Item Development Specification for the EDP (ref. 14), but the benchmark program was not available for this testing. Therefore, the Dhrystone and Whetstone benchmark programs were compiled using Gnu C on the LynxOS. The results, shown in table 4-1, showed a significant loss in performance without the cache, but these results are only relative and cannot be directly compared with the requirement as specified in reference 4.

Table 4-1. Performance of the 386 with cache enabled and disabled

Benchmark	Dhrystone	Whetstone
386/387 with 64-kbyte cache	7205 Dhrystone/ second	1330 K-Whetstone/ second
386/387 with cache disabled	3970 Dhrystone/ second	1085 K-Whetstone/ second

4.1.2.3. SDP system bus: Multibus II: The system bus for the standard data processor is used for local interprocessor communication between the EDP-4 processor and the EDP-1 processor for the NOS. It will also be used for other backplane devices, such as memory expansion and the MicroChannel bus controller. The proposed system bus is the Multibus II. This bus standard has advantages because it supports multiprocessing and allows up to 21 communicating agents. The Multibus II backplane is attracting support from multiple vendors for commercial data and signal processing systems and supports a heterogeneous mix of loosely coupled processors.

The maximum theoretical throughput of the Multibus II is 40 MB/sec, but if multiple agents will be present on the bus, test bed simulations and actual scenarios should be used to determine whether the Multibus II will be a bottleneck. For long-term evolution, FutureBus+ might be a viable upgrade candidate. Based on initial indications of support from industry and other government agencies, the FutureBus+ standard will probably provide a broad customer base and strong vendor support in the future. However, qualified versions of the FutureBus+ are not available now, and commercial versions are not expected to be available until 1991. Therefore, a study of the feasibility of making such a bus transition is recommended.

4.1.2.4. EDP/RAM bus: MicroChannel: In addition to being the embedded processor system bus connecting the 386 CPU to RAM, the IBM proprietary MicroChannel bus has been proposed for the interface from the embedded 386 and the NIA to the FDDI token ring and the MPAC display device. This makes commercial support of the MicroChannel bus an important consideration to the life-cycle cost of the DMS. Since the IBM PS/2 uses this bus for I/O, development of interface hardware and software for the network interface unit and the MPAC displays can be expected in the near term. However, general commercial support for MicroChannel is currently mixed. A lack of vendor support for the MicroChannel and the IBM PS/2 in the future could become a barrier to DMS evolution. For example, if IBM should opt to drop MicroChannel in its next personal computer product, NASA can expect higher maintenance costs and evolutionary restrictions.

Ignoring these considerations, from a performance standpoint the MicroChannel seems as good a choice as any other bus that is available now. Given a choice of the extended industry standard architecture bus, NuBus, FutureBus+, and Multibus II, the "MicroChannel [should still] be retained as the local bus" (ref. 15).

4.1.2.5. Network operating system: The life-cycle cost of the NOS software also depends on the selection of the EDP/RAM bus (MicroChannel). The selection of the MicroChannel as it relates to the evolution of DMS software is valid provided that the device drivers for the FDDI network interface controller are commercially supported.

According to schedule, the program will not get the first NOS software for network interface tests until August 1991. However, as shown in table 3-3, a brief review of the requirements in the CEI Specification (ref. 5, p. 3-256) shows a serious weakness in the NIU performance of timely fault management; the only latency specifications are for nominal conditions and even then, only at a 95% probability. If an anomaly occurs in the DMS optical network, there is no specified time to propagate the status to other stations. This could make detection and isolation of faults a very uncertain process at best.

4.1.2.6. Multiplexer-demultiplexer: The evolutionary constraints on the MDM are similar to those on the SDP because they both use Intel's 386 as their baseline processor. A more detailed analysis of MDM performance is premature because of design and implementation uncertainties. However, some top-level constraints associated with the MDM can be identified through the perspective of the Centrifuge Facility Project at ARC and a review of the specifications in the DMS and MDM CEI documents:

1. Sensor/effector service, as specified, requires a sample rate of only 10 Hz.

2. Throughput per MDM is limited to 22.4 kbps via UAS and 96 kbps via standard I/O cards.

3. Although throughput is 10 Mbps at the host SDP-to-DMS network interface, the RODB constraint is not

defined, and limitations 1 and 2 may lead payload designers to go around the MDM and the system RODB with custom components.

More details of the payload interface options can be found in reference 8; four options are outlined in response to the Space Station Advisory Committee finding that the payload interface is unclear, however the interfaces are described as "still needing to mature" (ref. 8). The findings and responses from the meeting are interpreted as showing the need for more systems integration development and more planning for future missions. In particular,

1. The cost for payloads to use DMS "commonality" elements is undefined.

2. The 802.4 bus and MDM interface, scrubbed in 1989, is needed by some payloads.

3. Even though the HRL/patch panel for telemetry is a costly and minimally defined interface, 70% of the payloads are being planned for its use.

Figure 4-1 is an overview of the Centrifuge Facility payload interface scheme to deal with the DMS and OMA constraints. It is a good example of payload interface issues because it is a long-lived payload with only moderate telemetry needs (no single high rate, total aggregate rate 5-15 Mbps), and it is required by OMA health-and-safety and command needs to connect to the DMS. One would expect all its data processing and communications to fit within the DMS, but instead, a dual interface with a custom HRL link is planned because the MDM/1553 interface is too constraining, the 802.4 solution is too uncertain, and migrating between them is too costly.

The current prevailing view (or model) of a payload instrument also contributes to the dual interface; past experience is with instruments for which all data is sent as telemetry to the ground. Although future instruments should be projecting the use of internal processing and automation to reduce telemetry, today's model of requirements is fuzzy at best, and a high design risk.

4.1.2.7. Primary/secondary storage: The current baseline DMS configuration is a loosely coupled mass of independent processors, especially with respect to mass storage. This is a workable architecture as long as things like automated central control, system management, reconfiguration, and fault management are not required. However, these features are part of the DMS proposal and they would require at least an evolutionary path to a truly distributed system with high communications bandwidth and elimination of bottlenecks. In a distributed configuration, fully redundant primary and secondary storage systems are at least desirable and in the case of fault management, particularly necessary. The CEI Specification requires redundant copies of all critical software and data.

The DMS baseline architecture includes a total of four 250-MB disk drives. Two of these are connected to MSUs that transfer data and programs across the FDDI network to the various subsystems. The other two disk drives are attached to the two MPAC-F systems. It is not certain how redundancy of data and programs is achieved with this configuration, but it is assumed that the two disks on the FDDI ring will maintain redundant copies of the software to keep these systems alive. In a like manner, it is assumed that the MPAC systems will have redundant copies of the software kept on the disks attached to the two MPAC-Fs.

The RODB is kept in RAM according to the DMS design, and the telemetry data is transferred directly to the ground via the CTS, so that leaves much of the disk space allocated to the SDPs and MPACs available for software binaries, swap and page files, and temporary data storage. However, it is uncertain how redundant RODB data is kept for recovery purposes if one SDP fails. Also, if the SDP or MPAC reboot mechanism relies on the disk, a boot program should reside on the booted system on a battery-backed-up RAM or a ROM device to carry on network file transfer protocol until the NOS boots up.

It would be much better for reliability and performance under today's technology to have a single centralized disk farm with a high-speed controller device that supports disk mirroring (shadowing). This is a common method used by several vendors to maintain reliable access to data and programs. The proposed baseline configuration runs the risk of having unacceptable initialization and reconfiguration delays because the MSUs are shared across the network and SDP operating systems will need to load from them.

4.1.2.8. Runtime object data base performance on the DMS kit: The MODB, on the ground, is the repository for all controlled SSF data objects. The RODB, onboard the SSF, contains a subset of the MODB objects for use by all DMS applications. This seems like a good method to control onboard intersystem data flow, but it does not address the implied data-flow performance requirements vis-a-vis the capability of the DMS. The following analysis attempts a preliminary evaluation of the RODB capability in relation to typical requirements.

The DMS Critical Design Review (CDR) lists 11 restrictions that "must be followed if an application is to correctly operate in a DMS environment" (ref. 16). Three of those restrictions apply directly to how applications should use the RODB:

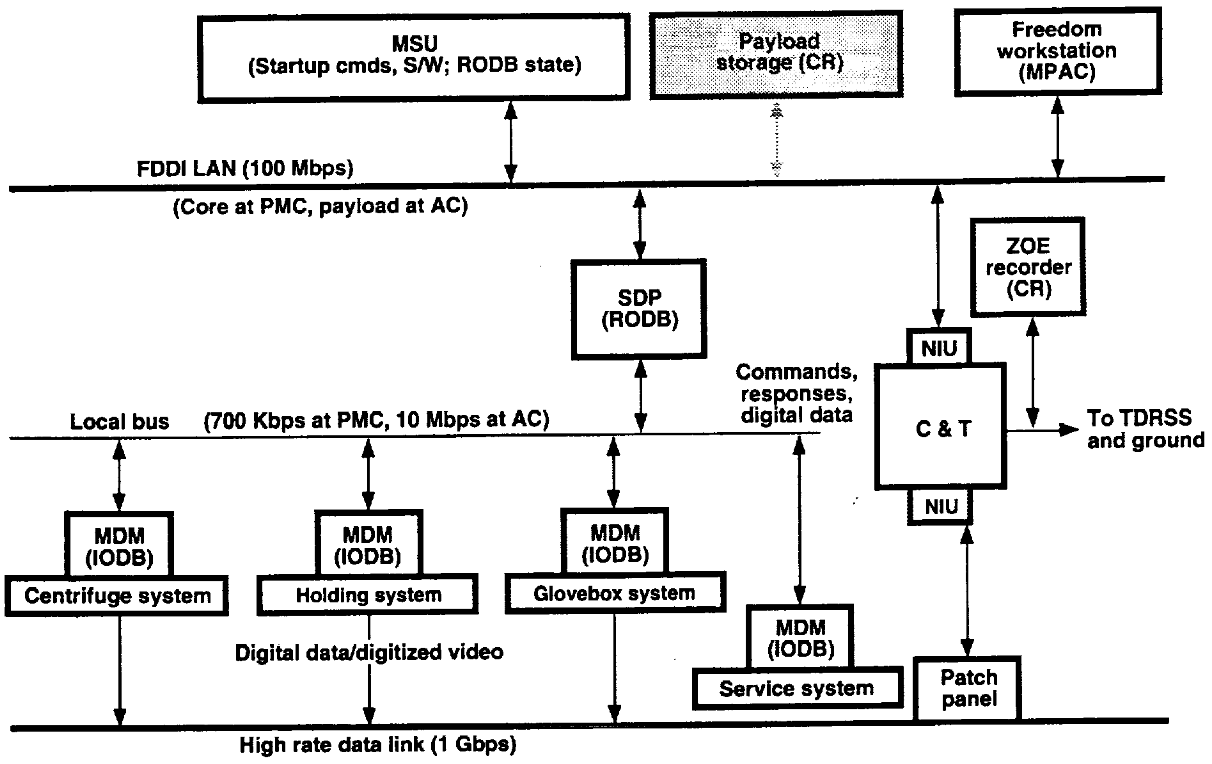

MSU (Startup cmds, S/W; RODB state)
Payload storage (CR)
Freedom workstation (MPAC)
FDDI LAN (100 Mbps)
(Core at PMC, payload at AC)
SDP (RODB)
ZOE recorder (CR)
NIU
Commands, responses, digital data
C & T
To TDRSS and ground
Local bus (700 Kbps at PMC, 10 Mbps at AC)
MDM (IODB)
Centrifuge system
Holding system
Glovebox system
Service system
Digital data/digitized video
Patch panel
High rate data link (1 Gbps)

Figure 4-1. Centrifuge facility/SSF interface scheme (CRs in the figure are changes requested for payload accommodation).

"3) Displays are independent of applications. All data to be displayed is written to the RODB. Display generators can read the data from the RODB at their discretion.

"4) All data shared by programs is shared through the RODB.

"7) All program-to-program communication is provided through RODB actions and attributes. This is true for both intra- and inter-node communications."

The DMS uses the RODB to decouple one application from another. Because applications only know about the data they read or write into RODB, they do not need to know what or how other applications place data into the RODB. This interapplication independence allows system designers and users to view an application as an ORU, with the RODB playing the role of the data bus.

This ability to view applications as ORUs is conceptually useful although it makes the RODB central to almost all computation and sensor data, since all applications should communicate via the RODB. If the ability of the RODB to handle this traffic is too constrained, it may become a bottleneck.

The performance of RODB has been evaluated recently at the DMS Test Bed at Johnson Space Center (JSC). The construction of this test bed is based on IBM's prototype DMS kit, which consists of an MPAC, an SDP, and an MDM simulator that runs on an IBM processor. The MPAC is connected to the SDP via an FDDI network. The SDP is connected to the MDM simulator via a 1553B data bus. RODB runs on the DMS kit under IBM's AIX operating system version 1.1 (as opposed to LynxOS). Also available are a simulated DMS application (ECLSS) and a simulator that acts like sensors and actuators to provide a simulated MDM. The ECLSS simulation resides and executes on the MPAC, the RODB on the SDP, and the simulated MDM on a third processor.

A simple test program was written at JSC to measure RODB access time. The program repetitively reads an 80-byte object 10 times and averages those 10 reads over 100 executions. The test program resided and executed on the same SDP processor that contained the RODB. In half the cases, the data object was available in memory, and the RODB retrieved the data object from the MDM for rest of the cases. These tests were also done with and without the ECLSS application running on the MPAC. Table 4-2 summarizes the results.

In the best case, with the information available locally and no application load, it took 0.43 sec to access a single data object. This is significantly slower than what would be

Table 4-2. Average time for 10 reads of an 80-byte object on DMS kit

Times for 10 reads	No ECLSS (sec)	ECLSS running (sec)
Data available locally	4.29	8.39
Data accessed remotely (via 1553)	13.59	29.49

required if the application actually used the DMS according to the DMS CDR. Admittedly, the numbers are taken from a prototype DMS kit; however, they do point to a potential problem area. Assuming RODB could be optimized by 2 orders of magnitude (or 100-fold), the best-case access time would drop to 4 msec. The remote RODB access time would still be 14 msec. Translated into a rate, the RODB could handle between 73 and 233 read requests per second. If the RODB is to serve as central a function as described in the DMS software CDR, the suggested performance from a 100-fold speed increase would still be insufficient.

A simple scenario illustrates the need for higher performance in the RODB: Assume that a crew member requests that certain information be displayed at an MPAC. The associated display program would retrieve the necessary information from the RODB, format it appropriately, and issue X-window commands to display the data. If we assume that the display retrieves 40 values from the RODB, we can compute two different elapsed retrieval times.

1. In the first scenario, the application program accesses each object individually by sending a request to the RODB via the FDDI network. RODB request messages would have a network latency of 50 msec, an access would require 4 msec, and the return message would require an additional 50 msec for a total of 104 msec. Forty such request cycles would add up to over 4 sec elapsed time. This does not include the time to format, create, and display the information.

2. The second scenario optimizes the RODB access by predefining an RODB display object that has indirect pointers to the 40 required parameters. In this situation, 100 msec is required for the round-trip message latency plus an additional 164 msec for accessing the display object; the associated 40 objects would need a total of 260 msec for this display operation.

A DMS loaded down with more applications could easily extend these times into the multisecond range, which is unacceptable from a human factors perspective. Other factors such as process scheduling delays and RODB loading caused by other applications can significantly affect these numbers. In fact, the above data indicates that merely running one other application (ECLSS) would double the time needed to retrieve the display parameters.

4.2. System Dependability

The failure tolerance and redundancy management (FT/RM) analysis for SSF is focused on the functional requirements of the DMS integrated systems. This analysis is based on a system modeling technique called Digraph Matrix Analysis, which is currently being performed as part of the design review process. Appendix E describes the FT/RM process and two system modeling techniques, and gives an example of the use of each technique. The results of these investigations are useful in determining single- and double-points of failure in the DMS, as well as in predicting the performance of various system configurations under failure conditions. ARC participates in an FT/RM working group that meets approximately every two months to review the status of SSF FT/RM analysis.

4.3. Limitations on Growth and Evolution

Based on the design information available and the preliminary evaluation results described in appendix D, it appears that two general limitations on growth exist: (1) poorly integrated real-time performance requirements in the current design, and (2) the performance limitations of bus and network protocol and interface standards selected for the DMS.

4.3.1. Design uncertainty– The DMS design is a distributed set of loosely coupled, real-time, embedded processors. No system specification defines the worst-case latency for the real-time interactions (whether intended, unavoidable, or failure-related) between systems or from systems to payloads. Limited system engineering analysis to date has rationalized this design by assuming that all subsystem interactions will be quasi-static and will result in low intersystem activity at all processor nodes. The RODB exemplifies this uncertainty: no explicit response time requirements are included in the DMS CEI. Since no real-world experiments have been done on such a system (with RODB-based coordination) to date (except for the recent prototype DMS kit measurements described in section 4.1.2.8), the performance has not been questioned. Potential integration problems caused by slow system management response, unpredictable real-time instabilities, and uncontrolled subsystem interactions have not yet been addressed, and will not be visible in the current development schedule until after CDR. Any problems discovered with system integration will make reconfiguration for growth a difficult and costly process. Onboard, real-time, integration test and verification (IT&V) is supported by DMS (ref. 5, p. 4-5), but IT&V is an open-ended requirement; it is easy to do for minor software upgrades, but nearly impossible to do for a new, high-performance, embedded processor without explicit intersystem response-time specifications.

Experimental performance tests with DMS baseline design components are not possible via kits for another 8–12 months (there are no high-fidelity functional equivalent units (FEU) for FDDI NIA, or NOS, no 1553 Bus Interface Adapter (BIA) and MDMs, and no LynxOS in the prototype kits).

The DMS optical network cannot support the critical communications that will be required for fault management because the latencies (using FDDI) are specified only at the 95% probability, under normal conditions (without failures). Furthermore, reconfiguration latencies are implementation dependent and are only loosely specified for DMS.

4.3.2. Design limits– End-to-end communications relies on ground wide-area network and metropolitan communications protocols that are in a state of rapid change (ref. 19). However, the DMS design does not provide for this major technology shift. (1) No provision is made to add optical fiber to all ORUs that currently use copper data links. (2) The optical fiber used seems to be capable of at least a 1-Gbps data rate based on the modal bandwidth, but it is only specified to support the 100 Mbps of FDDI. (3) No system integration methods are specified for inserting new systemwide technologies (e.g., reliable, fully distributed real-time systems, order-of-magnitude speedup in new processors, or order-of-magnitude increases in mass storage).

The local bus capacity of 700 kbps to core MDMs leaves little margin for growth. The approximately 60 MDMs in the baseline design represent most of the user-programmable processing power, but they have the least powerful connectivity to each other and to the crew or ground. This limits their capability and increases the life-cycle cost of developing and maintaining application programs on them.

Referring to the descriptions of the NIU (section 3.3.2), the RC (section 3.3.3), and the SDP (section 3.3.4) with their attendant figures, one can see that the NIU dominates the SDP in complexity and requires 55 w out of a total of 130 w. Unfortunately the implementation provides only 5 Mbps maximum for full OSI services and has no guaranteed real-time access latency. The application EDP provides less than 3 MIPS and 4 MB for user code and requires 28 w. Thus the SDP design represents a method that has a limited ability to support future added payload or processing requirements, even compared to current workstation technology. Payload users are already considering the creation of their own processors and using the HRL for telemetry, even though they face a considerable development cost to do so. (See figure 4-1 for an example and section 5.2 for recommendations.)

Mass Storage can only be added to the optical network, so NIU performance limits mass storage performance. (The mass storage disk fits into a standard chassis and uses a controller on a Multibus II backplane, but the interfaces and protocol for such additions are not defined.)

The need for on-orbit assembly resulted in multiple feed-through connections of fiber optics in the present design, as shown in figure 4-2. This creates a significant loss of the optical signal, which eventually could lead to the use

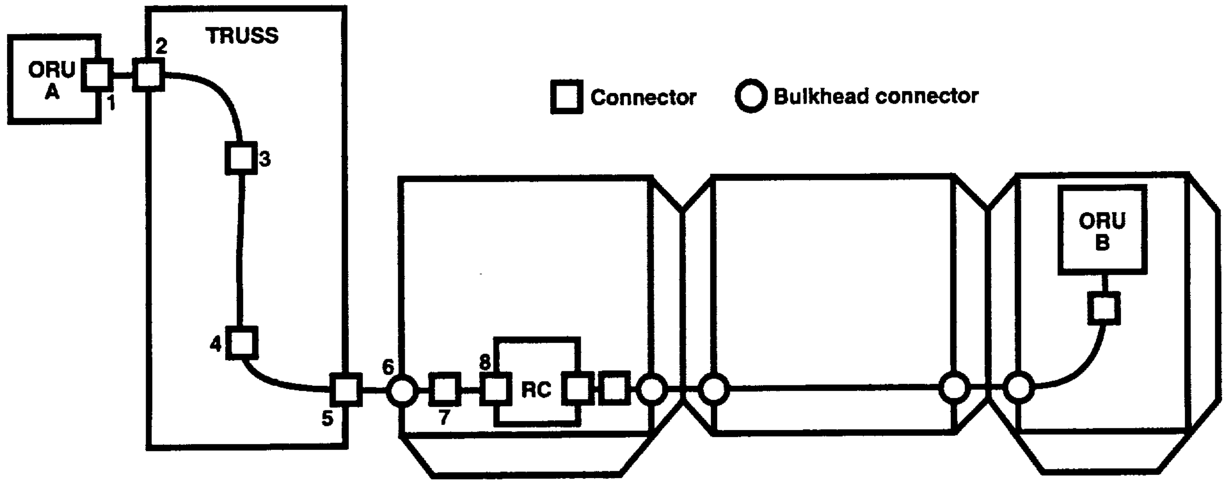


Figure 4-2. Typical path of optical signal showing multiple feed-through points.

of optical regenerators. These in turn limit data bandwidth, add to DMS power consumption, and reduce the overall optical link reliability. Therefore, the final limit for future growth is the connectivity provided by initial cables, connectors, and regenerators installed and tested at PMC. (See section 5.2.1.3 for recommendations.)

5. Recommendations for DMS Growth and Evolution

During the 30-year mission life of SSF, significant growth in user needs as well as computational technologies will take place. Therefore, the DMS should be designed from the outset to accommodate changes in functionality, performance, and technology. It should allow both near-term growth and long-term evolution. The term "growth," in this section, represents hardware and software changes that can be applied to the DMS quickly to meet a specific mission need. The term "evolution" refers to a more global and long-term process that enables developing technologies to be incorporated into the DMS. The ability to transfer technological innovations as they become available will affect not only SSF, but other future NASA programs and initiatives as well.

As discussed in section 4, the current capabilities of the DMS might not be able to meet performance and reliability requirements from the outset. NASA should be prepared to have (1) more tests to determine whether the current DMS software and hardware can meet current requirements, (2) near-term plans for augmenting DMS's capability with minimal impact on the rest of the system, and (3) a long-term policy and mechanism for technology insertion.

5.1. Recommendations for Further Tests

5.1.1. System level tests– Test bed experiments at ARC involving a collection of workstations based on Reduced Instruction Set Computer (RISC) and Complex Instruction Set Computer (CISC) processors are in progress. No commitment has been made to purchase a prototype DMS kit because of ARC's ongoing liaison with the DMS developers at JSC and MDSSC, the advanced nature of the evolution studies, and the immaturity of available prototypes. ARC test bed plans do include experiments on the DMS kits in the next fiscal year, when the first development-quality operating system, NOS, and network protocol implementations are scheduled to be inserted in the kits.

A variety of tests will be conducted to assess the viability of present and evolutionary hardware and software configuration options. These include real-time operating system mixtures on present 386 processors, with possible successor CISC (e.g., 486) and RISC processors. The real-time environment should also be assessed before the actual SSF avionic codes are designed, developed, and integrated. One option is to initially use Ames/Dryden Flight Research Facility (DFRF) aeronautic applications now running on advanced aircraft. Those functions may closely approximate many of the key upcoming SSF avionic applications, as shown in table 5-1.

Table 5-1. SSF/DFRF avionics subsystem analogues

SSF Avionic Applications	Available DFRF Avionic Analogues
Thermal Control System (TCS)	Thermal Management
Fluid Management System (FMS)	Hydraulics Management
Electrical Power System (EPS)	Power Management
Guidance, Navigation, and Control (GN&C)	Vehicle Controls
Propulsion	Vehicle Propulsion
Operations Management Application (OMA)	Mission Planning
Communications and Tracking	Navigation Control, Comm.
Environmental Control and Life Support System (ECLSS)	Life Support
Crew Health Care Systems (CHeCS)	Life Support

Although these applications are not exact analogues and most are written in Fortran or C, they are available now in validated code and specifications. Evaluating them on LynxOS, 386/486 PS2 platforms will provide insights into issues involving expected versus actual real-time performance issues. If there are sufficient resources, conversion of selected DFRF applications to Ada might generate indications of language and compiler real-time development issues.

Local area network protocol comparisons will also be made, for example comparing relative overheads between the OSI standard, standard Transmission Control Protocol/Internet Protocol (TCP/IP), "optimized" (Jacobsen) TCP/IP, and the IEEE "eXpress Transfer Protocol" (XTP) from Protocol Engines, Santa Barbara. XTP has been selected as the Navy Safenet II standard protocol and offers significant real-time speed (<1 msec end-to-end) and reliable delivery of control messages.

5.1.2. Lynx operating system– As we have learned from the Space Shuttle program and many other space projects, a software upgrade and enhancement costs much more than a hardware upgrade. The ability of the DMS to handle software growth will play a crucial role in the success of DMS evolution.

Operating system (OS) software is the foundation of the DMS software because most of the other software components are built on top of the OS. The selection of LynxOS for the DMS OS seems to be a good choice; it provides a real-time capability (see section 3.4.1.) and the advantages of a standard UNIX-based operating system. One of the most important advantages for DMS is the portability of a UNIX OS. The relative ease of porting an application from one processor platform to another is important for DMS because of the long lifespan of the SSF and the need to prevent obsolescence in the DMS. The following paragraphs discuss some features of LynxOS that need to be evaluated and/or fine-tuned for DMS.

1. Interaction between typical UNIX processes is not optimized because they mostly represent individual users. In contrast, all the tasks in a real-time system typically cooperate in a time-sensitive domain to perform a single integrated function. There is more communication between tasks, even at interrupt levels. This capability will become crucial for SSF fault recovery, vehicle health monitoring, and vehicle health management functions. Strict control of response timing, interrupt handling, and task switching will also be required to maintain high confidence levels during system Validation and Verification (V&V). The combination of LynxOS, the 386-based EDP/NIU (SDPs), MDM processors, and the FDDI/1553B network interfaces should be simulated and evaluated in a test bed to assess its ability to deal with the full synchrony of real-time functions projected for SSF.

2. Many real-time processes need to be locked into memory so that memory paging does not occur during time-critical events. In "standard" UNIX, there is no ability to lock pages of memory. Calls or other mechanisms that support the ability to lock specific areas of memory on an address or length basis are strongly desired and are the subject of much present discussion in the IEEE POSIX real-time subgroup meetings. For compatibility reasons, additional features that allow for locking of text and data segments of a process should also be provided. The memory locking features of LynxOS should be tested.

3. Another very common real-time requirement is the need to poll. For example, a process may have to be awakened periodically to take a temperature measurement. Quick action may be required if the temperature strays beyond established bounds. The default quantum size for each process in LynxOS version 1.2 is 64 clock ticks, which translates to 0.64 sec. This value is much higher than a standard UNIX, which is typically around 0.1 sec. The default quantum size in LynxOS can be changed without modification to the LynxOS source program, and a new kernel can be created by the "make install" command after the quantum size is changed. The quantum size needs to be evaluated and fine-tuned for DMS.

4. The LynxOS ability to wait on multiple events should also be tested. The standard UNIX "select(2)" routine does not work well with standard UNIX semaphores and message queues because the latter two do not use the file name space. The particular mechanisms that Lynx, Inc. has devised for waiting on multiple events should be evaluated.

5. Real-time control systems are often run via semaphores that should be accessible by drivers and processes alike. The System V UNIX semaphore is complicated, slow, and does not allow driver access. POSIX has defined a simple binary semaphore mechanism. LynxOS performance should be fully evaluated in this region.

6. In addition to the scheduling algorithm, task "dispatch latency" is also critical to real-time applications. Dispatch latency is the time interval between the return of a driver from an interrupt to the time the first instruction is executed in the highest priority program in the system.

7. Regular files in a standard UNIX file system are implemented as logically contiguous byte streams. Internally, however, they are simply lists of blocks in no particular order. The file system caching mechanism, though useful for speeding ordinary accesses, limits throughput because of relatively small transfer sizes. The time needed to access individual blocks in a regular file may not be constant, because other internal "pointer" blocks may need to be fetched. For these reasons, throughput through the file system may be limited to a value somewhat lower than the ideal disk transfer rates. LynxOS's implementation of the file system should be tested to ensure the timely retrieval of critical data.

8. The raw disk interface transfers directly to and from user memory, and so allows speeds approaching optimal disk transfer speed. The important restriction when writing to a raw disk device is that transfer should be in increments of 512 bytes, and applications should use large transfer sizes if possible. In addition to regular disk files, LynxOS provides a utility and a system call to create a contiguous file. However, creation of a contiguous file takes time. The kernel should search the free block list for a run of contiguous blocks. For DMS applications, it is

recommended that a buffer be created in advance and reused because of the long creation time for a contiguous file. It should be noted that LynxOS does not prohibit users from creating a very large contiguous file. This capability may need to be constrained to a certain file size for payload applications.

All of the the features listed here are scheduled for tests in the ARC Advanced Architecture Test Bed this year in coordination with the JSC Work Package-2 developers and other government, university, and industry participants.

The U.S. Navy Next Generation Computer Resource Program (NGCR) and Space and Naval Warfare Program (SPAWAR) offices, for example, have found that the present POSIX definition of real-time facilities are still inadequate for their more "hard-core" real-time applications and have gone far further in defining their own real-time operating system guidelines. ARC has begun an effort to work with the Navy in these and other Ada/Language and communication/ network/backplane arenas.

Preliminary tests at ARC have established a baseline of latencies for sending datagram-type messages using commercial RISC- and CISC-based workstations. The tests have illustrated the difficulties of porting measurements between dissimilar machines even though they can all claim POSIX compatibility; however, successful repeatable results have been generated. The tests clearly illustrated the differences in time measurement granularity (time granularity ranges from 3.9 to 20 msec) and uncovered a network driver bug in one workstation. On the more powerful workstations tested, an overhead delay of approximately 4 msec (average) was observed for the send/receive loop between two stations. The tests of PS/2 with LynxOS are not complete as yet, but it appears to have approximately the same average performance. With standard UNIX the maximum latencies observable can be quite large, but the LynxOS overhead is claimed to be small; these values are also scheduled to be measured at the ARC Test Bed.

Finally, IEEE POSIX standards include (1) P1003.1 for system interface, (2) P1003.2 for shell and utilities, (3) P1003.3 for verification testing, (4) P1003.4 for real-time extension, (5) P1003.5 for Ada language binding, (6) P1003.6 for trusted system extension, and(7) P1003.0 for open system guidelines. P1003.1 is the only one that has been completed and published by IEEE. All other standards are still in draft versions.

P1003.1 covers the basic system interfaces, such as process primitives, process environment, files and directories, input and output primitives, and device-specific functions. But P1003.1 does not cover other features essential for real-time applications, such as priority scheduling, interprocess communication, and high-resolution timers. In addition, the future successes of Ada language bindings to POSIX will be important for mission-critical applications. NASA needs to influence the decision of the P1003.4 and P1003.5 working groups to achieve the Agency's best interest. JSC software groups are involved in this process.

5.1.3. Other tests– Although the choice of the FDDI protocol for the optical network seems to be a good one for projected performance and reliability reasons, the implementation for the DMS does not include a specific performance level. This raises concerns about the final DMS FDDI characteristics. Such system-level development risks will not be tested until after CDR, according to the current schedule. Some issues associated with FDDI implementations will be testable on commercial hardware in fiscal year 1991, and ARC has plans to begin such tests and supporting simulations early in 1991 on an extended FDDI test bed (subject to programmatic support). Technical coordination will be maintained with the DMS developers and with the Naval Ocean Systems Center, San Diego, where a major FDDI test bed exists to perform Navy Safenet II tests.

5.2. Recommendations for Short-Term Growth

5.2.1. Connectivity– Beyond AC, additional functional components will be added to the DMS. Successful insertion of these components depends on the scars provided by the initial design as well as on the capacity of the network and bus connections. The modular baseline design provides a good starting point (i.e., scars) for adding hardware components. SDP successors, bridges, gateways, or MPACs can be added via RCs, as needed. BIUs may also be added to connect the 802.4 bus (currently deferred until AC) for enhanced sensor/effector connectivity.

Inside the major ORUs (MDMs excepted), a Multibus II backplane bus provides high-performance connectivity between embedded processors, memory, and interface adapters. However, the choice of the 1553B standard for the local bus and the design of the NIU interface to the DMS optical network may limit the speed with which critical messages can be exchanged between ORUs over the network. There are two problems:

1. The Bus Interface Adapter (BIA) to the local bus connecting to the MDMs uses 1553B protocol which has a maximum throughput of 700 kbps. This connection is a bottleneck and limits the growth of MDMs and the

standard connectivity capability to payloads and core subsystems.

2. The NIU to the optical network is designed to have a separate bus (MicroChannel) from a standard 386-based controlling embedded processor to the network interface. This design limits the data flow rate between applications and the network because of the memory-copying requirement between isolated data buffers. (See section 3.3.2 for quantitative values.)

5.2.1.1. Sensor/effector interface (MDMs/1553B): Sensor/effector growth is difficult to quantify because it depends on the connectivity in and around the MDMs, whose designs have not yet been completely specified (see section 4.1.2.5). Inside the MDM, an I/O bus provides a connection to a variety of standard I/O cards, and an embedded 386SX-based processor uses a separate bus for a custom, user-designed card. More than 60 MDMs are considered baseline with the low-risk, but limited-capacity and limited-adaptability 1553B local bus connection (see section 3.3.5). An alternate local bus is proposed that would have about six times the capacity via optical media and the IEEE 802.4H standard protocol, but its qualification is presently deferred. Thus no current I/O requirements can be based on its use.

5.2.1.2. Optical network between systems: The optical media selected for the network has a capacity for growth to at least 10 times the FDDI protocol rate (to approximately 1 Gbps), and the FDDI channel rate of 100 Mbps provides some near-term capacity for growth. However, the NIUs resident in each ORU are specified to support only 1/10 the capacity of the FDDI network. This limitation has little impact on the projected data flow requirements for PMC, but could be a bottleneck to the high-performance and robust automated data-flow requirements anticipated for improved system monitoring and FDIR applications. The NIU design is based on the same 386-based embedded processor as the SDP application processor. This reduces initial qualification cost and risk, but limits throughput growth for future requirements. Power allocated to the NIA (table 3-2) seems high for such limited performance and suggests the need for a technology change at some point in SSF's mission.

This NIU implementation will have a high life-cycle cost, even over the first 10 years of operation, because it will either require an early changeout across most of the network, or become the design driver (and limiting factor) for other system upgrades. Since integrated operation performance and reliability requirements are still under development, and the experience with FDDI interface adapters is very limited, the NIU could become a bottleneck for key interfaces in the baseline DMS (e.g., the Communications and Tracking interface).

5.2.1.3. Recommendation for connectivity: In summary, two recommendations are made to improve connectivity:

1. Local bus connectivity should be improved, both in total capacity and in ease of adaptation to general interelement communication. Early development of the 802.4H protocol interface and distribution of the fiber optic media to all hardware racks is recommended. Making the 802.4H local bus available early provides a low-risk method to support requirement changes during development and alternative technologies at AC. In addition, it satisfies the top-level requirement to provide growth and evolution flexibility for both payload and core systems.

2. An alternate NIU and/or network design study is recommended to reduce both development and life-cycle risk. The NIU appears to be a major risk to growth as well as to initial system integration because of complexity and power. The fact that it is functionally inline with all real-time intersubsystem performance and reliability issues but is scheduled late in the development is additional reason for concern.

5.2.2. SDP performance–

5.2.2.1. 386 performance and instruction set architecture: The Intel 386DX microprocessor was selected to be the main processor for the DMS SDP almost four years ago. By AC, in 1999, the 386 chipset will be 13 years old, and the 486 chipset, nearly 10. Even at present, the relative performance of the 386 and 486 are low compared to other processors available today. Although many COTS software products are in the X86 inventory, only a small percentage of these relate directly to embedded, Ada-based, fault-tolerant, real-time application areas. The software product match for such disciplines is far more prevalent among the high-performance (RISC) workstation processors. Furthermore, some initial simulations of SSF application workloads have shown a requirement for at least 8 MIPS, and this still results in relatively large queue (ref. 7, pp. 5-23).

Increasing the processor performance and capability has become a necessity. However, performance is only one criterion when selecting a processor as the baseline for the DMS. Selecting a processor implies adopting its instruction set architecture (ISA), on which all of the system software will be implemented. Replacing processors with incompatible ISA implies an extremely costly, time-consuming process of software V&V. Therefore, selecting an ISA that has potential for growth and strong commercial support will help ensure that SDP processors can be upgraded easily in the Growth phase of the SSF.

5.2.2.2. Upgrading SDP using Intel 486: For meeting near-term performance needs, staying within Intel 80X86 ISA may be a reasonable option because of the cost and time required for revalidating DMS software. Therefore, the 486 seems to be a natural candidate for upgrading the SDP for several reasons:

1. It is 100% binary compatible with the 386, and the 486 ISA is a superset of the 386 ISA. (It contains six more instructions than the 386.)

2. It has higher performance. In laboratory benchmark tests carried out at ARC, the floating point performance of the 486 (with an 8-Kbyte internal on-chip cache) was about 3 times higher and the integer performance about 2 times higher than the 386 (with a 64-Kbyte external cache). The 486 bus is also faster than the 386 bus.

3. It consumes less or equal power. The 486 consumes less power than the combination of the 386DX and the 387DX. The system board of the 486 also consumes less power than the 386 board.

4. It has higher integration. The on-chip integration of the 486 includes a floating point unit, an 8 Kbyte on-chip cache, and a memory management unit.

5. It provides multiprocessor support. The 486 supports multiprocessor instructions, cache consistency protocols, second-level cache, and other multiprocessor support hooks.

However, there two main concerns in regard to the use of the 486 for the DMS:

1. The 486 has an on-chip, 8 Kbyte internal cache for instructions and data. The cache does not have an error detection and correction (EDAC) function and is therefore not suitable for space applications. Even though the internal cache of the 486 can be disabled by software control, the impact on performance is not yet determined. A larger, second-level, parity-checked external cache may also be necessary to improve the performance.

2. Work Package-2 is required to deliver a space-qualified processor for DMS in 1992. Intel has no plans to produce a space-qualified 486 by then.[5]

5.2.2.3. Upgrading SDP using Intel i860: DMS is expected to support applications that require fast floating-point operations. Although the 486 provides an on-chip floating-point processing unit, its performance can be further enhanced by adding coprocessors. Among commercially available coprocessors compatible to 386/486, the Wizard Adapter,[6] based on the Intel i860, provides impressive performance potential for several reasons:

1. The i860 is a 64-bit, RISC-based, high-performance processor. It is currently available at 33 MHz (41 MIPS) and 40 MHz (50 MIPS).

2. The i860 can be used as a stand-alone CPU. It has been optimized for floating-point and graphics operations.

3. The i860 chip includes an integer core unit, a floating-point unit, a 3-D graphics unit, an 8-Kbyte data cache, a 4-Kbyte instruction cache, and a memory management unit.

4. Intel plans to have the i860 meet the MIL-STD-883C Class B military qualification in the second quarter of 1991. It takes about one additional year to process a space-qualified chip after its military-qualified version is available.

5.3. Recommendations for Long-Term Evolution

5.3.1. Toward a functionally distributed system architecture– The use of distributed systems for ground-based applications is widespread. Primary advantages of distributed systems over centralized systems include enhanced reliability, flexibility, and resource sharing. Incorporating distributed technologies in the DMS was a wise decision for the SSF Program.

However, the full benefits of incorporating distributed systems will only be achieved in the evolutionary DMS. The current baseline for the DMS is a mix of distributed and centralized technologies. Although the FDDI backbone connects the various subsystems, its capacity and its functionality are underutilized. Dedicated links are

[5] According to Intel's schedule, a MIL-STD-883C Class B, military-qualified 486 will be available in the second quarter of 1992, and a Class S, space-qualified 486 in the second quarter of 1993.

[6] The IBM Wizard Adapter card contains an Intel i860, 2 Mbytes of memory, and three IBM chips to interface with the host CPU (IBM PS/2 Model 70 or 80; 386 or 486) through the MicroChannel. The card can run IBM OS/2 or AIX OS operating systems. It can do numeric-intensive portions of 386/486 jobs, or it can run complete jobs independently from, but under the control of, the host 386/486. An Intel i860 Software Development Tool-kit, including a C compiler, assembler, linker, and library is available. All code for the Wizard Adapter must be recompiled using the Tool-kit. Tasks run best on the adapter if their I/O requirements are limited. The Wizard Adapter may have potential for the DMS. A Wizard Adapter running with the 386 may be rated at about 30 MIPS, which is superior to most RISC processors and the 486. Although a coprocessor card will consume more power, its increased performance may enable reduction of DMS SDPs. Researchers at ARC plan to measure the performance of representative workloads on a 386DX/486 with a Wizard Adapter.

provided where there is concern that the backbone network will not be able to support a function. This design is due in part to the division of responsibilities among work packages. It does permit individual work packages to develop and verify their application programs on independent nodes and still provide some assurances that the system can be integrated later in the development cycle.

The current baseline for the DMS networking infrastructure combines state-of-the-art technology (i.e., FDDI) with old technology (i.e., MIL-STD-1553). Even though the FDDI backbone will be grossly underutilized in the initial configuration, its presence will facilitate evolution of the DMS to a truly distributed system. Evolving from FDDI-I to FDDI-II and then to FDDI-FO is a path that is likely to be followed within the commercial world as well.

The evolutionary goal for the DMS should be a system that is distributed functionally, as well as physically. Provision of separate networks for core and payload applications as the DMS evolves will allow the use of state-of-the art technologies for the payload network while more conservative technologies are used for the life-critical and safety-critical core applications. Ideally the payload network should provide an infrastructure that is reminiscent of the scientist's ground-based laboratory. High-speed transmission links should handle all kinds of traffic, audio and video as well as digital. Use of dedicated links should be minimized; large numbers of special-purpose point-to-point links can be difficult to maintain, besides soaking up valuable power, weight, and volume resources. Another essential feature of the evolutionary DMS is the ability to process scientific data in space, so as to substantially reduce the amount of data that needs to be transmitted to the ground. If the payload portion of the DMS mimics ground-based systems, then SSF will be able to incorporate new technologies as they are developed.

The current DMS architecture is not a truly distributed system, but the evolution of the DMS will necessarily require more distribution of resources. As the resources become more and more distributed, the DMS optical network will become increasingly important.

5.3.2. Operating system/Ada runtime environment– Based on initial simulation and experimental results at MDSSC, ARC, and JSC test beds, it seems possible that the combination of 386 hardware and LynxOS software will be insufficient to meet both electrical power constraints and long term computational needs. The power and weight problems continually generate "scrub" activity in one direction, whereas growing computational performance needs constantly motivate an ever-growing number of CRs and RIDs. Should this scenario become the case, three logical options would remain:

1. Keep the 386 hardware and move to a more efficient, less (memory) resource-sapping, ROMable real-time OS

2. Keep LynxOS and try to transition it to a faster microprocessor chipset (e.g., the 486)

3. Change to faster hardware and software components, keeping only the basic fiber-optic network connectivity media the same

The driving premise of the JSC/SSF/SATWG[7] Avionics "Open Software Architecture" definition is its emphasis on *source-level* portability. Many processors with performance levels far beyond the 386 now exist and offer real-time operating systems (including UNIX-based ones). Multiple chipsets, especially RISC microprocessors, are being cast in Class S component technologies on today's fabrication lines. Modern microprocessors offer Ada compilers, some using advanced Ada Tasking capabilities in real-time domains that have more stringent requirements than the DMS. Some workstations available today offer FDDI interconnectivity, and advanced lab models are demonstrating fiber-optic bandwidths at the 1-Gbaud (100-125 MB/sec) level.

During the projected 30-year life span of the SSF, it is possible that parallel processing will become more available. Intel has predicted that the next generation of microprocessors in the 1990s may have multiple CPUs in a single chip. In order to take advantage of this trend, the DMS OS may have to evolve to support parallel computers. Several UNIX-based multiprocessor OSs have been developed in the past, such as Dynix, which runs on the Sequent Balance computer; Symunix, which runs on Ultracomputer of New York University; UNICOS, which runs on Cray supercomputers; and Mach from Carnegie Mellon University, which supports a diverse mix of platforms.

Mach is the most portable multiprocessor OS. It has been ported to the Sequent Balance, Encore Multimax, and BBN Butterfly, but it also runs on single processors, such as Sun 3 workstations and NeXT computers. It offers "lightweight processes," called threads, that do not carry the full state of a UNIX process. Creating a thread is less expensive and faster than creating a new process. Research Scientists at ARC have been studying Mach, UNICOS, and other multiprocessor OSs. The study results may be beneficial to DMS OS decisions in the long run.

5.3.3. A scenario for growth and evolution– Figure 5-1 illustrates a possible path toward developing spaceborne, mission-adaptive architectures and a method by which these efforts could relate to the SSF DMS. From the passive (read only) tap to the DMS FDDI core ring, actual

[7] Strategic Avionics Technology Working Group

mission data could be fed to advanced architectural modules, performing mirrored functions using source-compatible (Ada) applications. This noninvasive, mirrored capability would assist in the V&V process of DMS upgrades before function cutover to newer and faster equipment.

The left side of figure 5-1 shows a process for evaluating various advanced hardware, software, and connectivity scenarios. The basic element of this process is a study of advanced connectivity technologies, spanning the entire intersystem (fiber-optic network media and protocols) and intrasystem (board-level) domains. The result of these studies can then be testbedded using either backplanes such as FutureBus+, or advanced fiber-optic crossbars, or both. Once this connectivity platform is in place, many different CISC, MISC, and RISC processor and coprocessor combinations can be evaluated alongside other advanced peripheral, intersystem network and communication (I/O) board facilities. On such test beds, these advanced concepts can be evaluated with other non-NASA government and industry sectors. A parallel effort should then force all evaluated components through the "Long-Range Architectural Scalability Filter" described in appendix D.

Perhaps the most critical issue in DMS evolution is its ability to upgrade transparently and noninvasively. To ensure this, having concluded that the multimode fiber-optic medium is capable of far greater bandwidth capacity than the original FDDI/OSI design, there should be a second ring (or crossbar) network configured in the race-ways at launch with passive (read only) taps to the core FDDI ring. Over time, this would allow for newer, faster, less power-hungry processor and software enhancements, plus advanced fiber transceivers and protocols. New gigabit switching options should also be evaluated for the post-AC time frame. These would allow the integration of processor products regardless of the particular backplane (or backplaneless) architecture used.

Should the i386 processor chipset be found inadequate in the future, there should be a backup plan. The assumed progression has been in the direction of the i486 and its compatible successors. Appendix D, however, shows that this path, being high-density CISC, is a long and difficult one. RISC technologies exhibit significant assets for the long term and should be studied for their ability to be quickly included in post-AC configurations and for their long-term scalability. Natural candidates include Sun SPARC (eight vendors), MIPS R4000/R6000 (five vendors + USAF JIAWG support for R2000/R3000 RISC chips currently available), Intel i860, and the Motorola 88000 family. All of these have gained some level of international acceptance in the embedded, Ada, realtime OS, and radiation-hardened single-event-upset-tolerant arenas. Complimentary studies should be conducted on radiation and SEU hardness of the candidate technologies, comparing the gate/transistor density characteristics of the

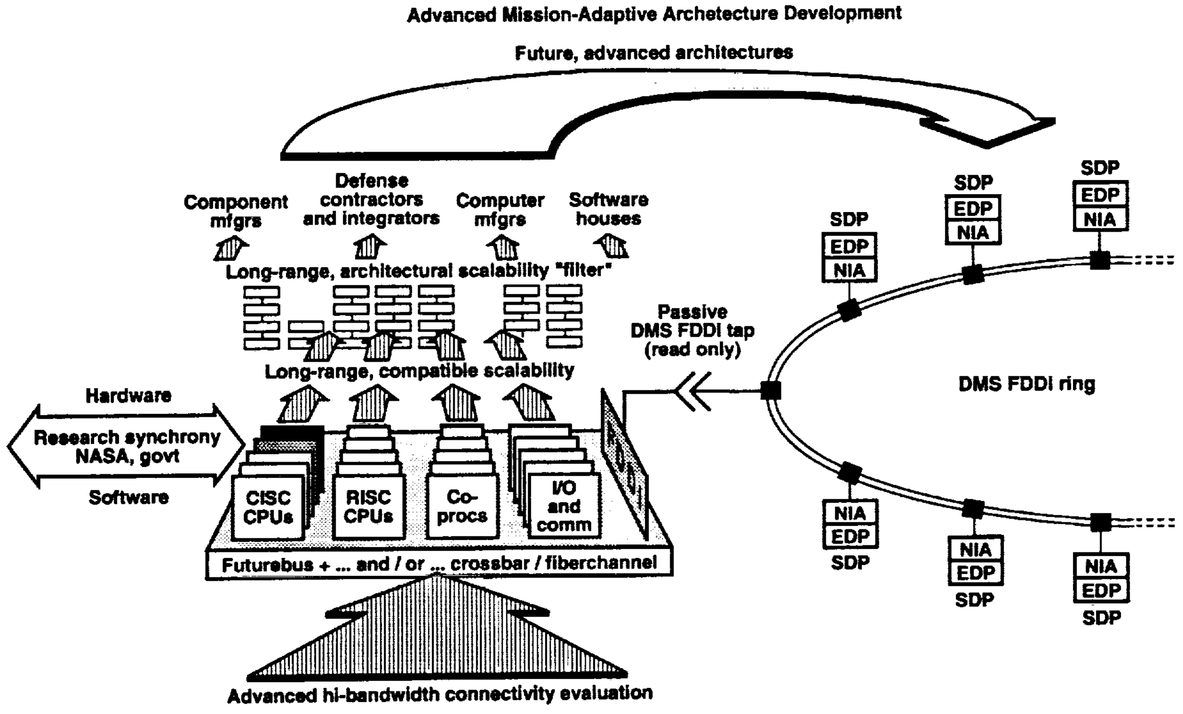

Advanced Mission-Adaptive Archetecture Development
Future, advanced architectures
Component mfgrs
Defense contractors and integrators
Computer mfgrs
Software houses
Long-range, architectural scalability "filter"
Long-range, compatible scalability
Hardware
Research synchrony NASA, govt
Software
CISC CPUs
RISC CPUs
Co-procs
I/O and comm
Futurebus + ... and / or ... crossbar / fiberchannel
Advanced hi-bandwidth connectivity evaluation
Passive DMS FDDI tap (read only)
DMS FDDI ring
SDP
EDP
NIA

Figure 5-1. Scenario for inserting advanced SDPs into the DMS.

technologies and candidate processor density requirements. This would identify the most "naturally evolvable" processor candidates. Further, each candidate processor should be evaluated from an "openness" standpoint in regard to life-cycle costs.

Finally, during the evaluation process for all components and modules, NASA should strive for an integration of people and perspectives at the leading edges of technology. The challenge is to shift from *reactively* understanding new technology to pro-actively shaping it.

APPENDICES

A. References and Contacts

A.1. References

Relevant documents used to support our study, conclusions and design assumptions

1. Space Station Freedom Program Requirements Document, SSP 3000, rev. C, Space Station Freedom, NASA, Washington, D.C., Oct. 1990.
2. Program Definition Requirements Document, SSP 30000, sec. 3, rev. K, Space Station Freedom Program Office, Reston, VA, June 1991.
3. Project Requirements Document, JSC 31000, vol. 3, rev. F, Johnson Space Center, Houston, TX.
4. DMS Architecture Control Document, SSP 30261, rev. C, Johnson Space Center, Houston, TX.
5. Configuration End Item Specification for DMS, SP-M-001, McDonnell Douglas Space Systems Company, Huntington Beach, CA, Feb. 1990.
6. Baseline Operations Plan, JSC 32083, Johnson Space Center, Houston, TX, April 1990.
7. System Engineering and Integration Trade Studies: DMS End-to-End Simulation Analysis Report, MDC H4875, McDonnell Douglas Space Systems Company, Huntington Beach, CA, April 1990.
8. "DMS Presentation to Space Station Advisory Committee," Aug. 14, 1990.
9. Weeks, David: Space Station Freedom automation and robotics: An assessment of the potential for increased productivity. Advanced Development Program, Strategic Plans and Programs Div. Office of Space Station, NASA HQ, Dec. 1989
10. Davis, Gloria: Computational needs survey of NASA automation and robotics missions. NASA TM-103860, May 1991.
11. Fisher, William F.; and Price, Charles R.: External maintenance task team report. Johnson Space Center, Houston, TX, July 1990.
12. Yan, J. C.: AXE—An experimentation environment for concurrent systems. IEEE Software, May 1990.
13. Liu, Y. K.: Analysis of Intel 386 and i486 microprocessors for Space Station Freedom data management system. NASA TM-103862, May 1991.
14. Configuration Item Development Specification for the Embedded Data Processor, MDC H4534, McDonnell Douglas Space Systems Company, Huntington Beach, CA, Sept. 1989.
15. Fariss 89 "A Discussion of Options for Replacement of the IBM MicroChannel," informal Recom, Inc., Report, Ames Research Center, Moffett Field, CA.
16. Software User's Guide (Data Management System) (DR SY-40.1), MDC H4542 Resubmittal 2, McDonnell Douglas Space Systems Company, Huntington Beach, CA, Sept. 1990.
17. JSC 31000, vol. 3, rev. E, Johnson Space Center, Houston, TX.
18. Failure Tolerance and Redundancy Management Design Guide for WP-2, MDC H4865, McDonnell Douglas Space Systems Company, Huntington Beach, CA.
19. Rudin, Harry; and Williams, Robin: Faster, more efficient, streamlined protocols. Guest editorial and special issue, IEEE Communications Magazine, vol. 27, no. 6, USA, June 1989.
20. Schwetman, Herb: CSIM Reference Manual. Microelectronics and Computer Technology Corporation, Revision 14, Austin, TX, March 1990.
21. Failure Modes and Effects Analysis (FMEA) for the Data Management System. (DR SA-6.1), MDC H4563A MDSSC, Huntington Beach, CA, Mar. 1990.

A.2. Government Contacts

Many discussions and meetings with other Government Departments and Agencies have shown common needs and a desire to cooperate. Multiple government organizations have similar ground, aeronautic, space, real-time, rad-hard and space-qualified requirements. Some are leaders in advanced computer designs and standards for these application areas. Below is a list of Governmental bodies and initial hardware and software technologies being considered:

• **U. S. Navy:** NGCR, SPAWAR and SAFENET projects—distributed systems for shipboard

• **Fiber Optic**: "Scalable Coherent Interface" and "IEEE FiberChannel"

• **Fiber Optic Crossbar**: Carnegie Mellon/Network Systems and Ancor (with DoE)

• **Fiber Protocols (Safenet II standard):** eXpress Transfer Protocol (XTP)

• **Advanced Backplanes:** IEEE FutureBus+ (Multibus II, and VME successor)

• **FutureBus+ Processor Boards (in construction):**

  • MIPS R3000, MILVAX, Motorola 88100, i80486, AMD 29000, Motorola 68030
  • Advanced RTOS' + POSIX realtime extensions
  • Realtime Ada Tasking Analyses
  • Submarine Inventory Tracking Applications (These relate closely to SSF issues.)

• **U. S. Air Force** (RH-32 and JIAWG programs):

  • RH-32 Processors: MIPS and i860/960
  • Laser Space/Ground Communications

• **DARPA** (ISTO projects, ARC Advanced Test Bed):

  • Chipsets: i860, i960, iWARP
  • Systems: Touchstone, Nectar, Encore Multimax (i860 parallel), MCC 88100
  • Hi-resolution Workstations & Fiber-Optic Communication

• **IEEE** High Performance Peripheral Interface (HPPI):

  • DC-to-1 Gigabaud/100 MByte/sec Single & Multi-Mode Fiber
  • Ancor Space Switch (Omega-Net-like Crossbar)
  • Multiple Gigabaud Fiber-Optics to Standard Backplane interface boards
  • Visualization in Scientific Computing (ViSC)

B. ARC Simulation Report

B.1. Abstract

A simulation of SDP 5 of the SSF DMS was implemented using the CSIM package (ref. 20). The workload model was taken from an MDSSC report (ref. 7) describing another simulation performed using Network II.5. The results of the two simulations were compared. Qualitative results were in general agreement, with both studies predicting a very high usage for the Application EDP and MDM processors, but low usage for the other elements. The MDSSC report did not contain enough information about the workings of the DMS to allow an accurate simulation to be implemented based on the report alone. Some of the missing information is described, and ambiguities in the description are noted. The results of a static analysis of the MDSSC workload is presented and shows good agreement with the current study, so any discrepancies may be attributed to differences in the processing model.

B.2. Introduction

A simulation of the SSF DMS SDP 5 was carried out at ARC to help validate and understand another simulation performed by E. Y. Omori at MDSSC (ref. 7)—hereafter referred to as the MDSSC report. The results of two simulations: SDP 6 (Lab) and SDP 5 were described. The two SDPs differ in the number of MDMs to which they interface and the number of local busses used to connect to the MDMs. This study only looked at SDP 5, as it was the less complex system of the two.

The simulation was implemented in CSIM, a discrete event simulation package obtained from the Microelectronics and Computer Technology Corporation (ref. 20). The package consisted of a library of subroutines that allowed a discrete-event simulation to be implemented in the C language. In a simulation, C functions may be defined as independent processes, which use facilities, simulate processing time with hold statements, wait for declared events, or communicate with each other by sending messages via mailboxes. Statistics on facility utilization are kept automatically.

Because of the limitations of this simulation, as discussed below, only utilization levels for the various components of the SDP were obtained. The detail of the simulation model was insufficient to obtain meaningful queue lengths, latencies, or response times.

B.3. Model

In the CSIM simulation, physical elements of the SDP were defined as facilities. The workload was defined as a set of processes that utilize these facilities to carry out the functions of the DMS. These elements are described in more detail in the sections below.

B.3.1. DMS elements– The basic elements included in the CSIM simulation are shown in figure B-1. All of these units were simulated as facilities, allowing usage levels to be automatically recorded. Processes were also defined for the BIU, the NIU, and the NIA; these units provide paths for data flow in two directions and need to respond to requests from other units for the transmission of data.

The main source of information concerning the operational characteristics of each of the entities in the simulation was in System Engineering and Integration Trade

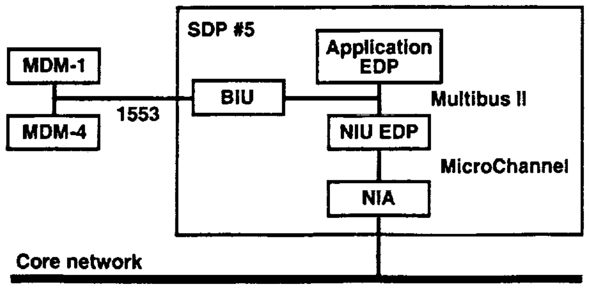


Figure B-1. SDP 5 simulation elements.

Studies (ref. 7, table 4.4-1, page 4-27). Each of the elements in the simulation are described in the paragraphs below.

B.3.1.1. Application EDP: The application processor described in MDSSC Report (ref. 7, table 4.4-1) has the following characteristics:

Table B-1. Application processor characteristics

EDP instruction rate	3.1 MIPS[a]
System node management overhead	5%
Default time slice for executing processes of equal priority	100 msec
OS processing time to generate/accept NOS request	500 microsec
Process switch time	250 microsec
Internal time update process	1000 instructions
Time data size for distribution	14 bytes

[a]This includes 7% EDP refresh. The significance of the phrase "includes 7% EDP refresh" is unclear in reference 1; this could mean that 3.1 MIPS are available for the execution of application programs or that only 93% of this value, or 2.88 MIPS are available.

The ARC model consists of a 3.1 MIPS application EDP, (see fig. B1). Two overhead load factors of the EDP mentioned in the MDSSC Report (ref. 7) were:

1. SM overhead of 5%, (table 4.4-1)

2. OS/Ada run-time environment (OS/Ada RTE) overhead of 15% (page 6-2).

These factors, when included, effectively lower the available processing rate of the EDP to 2.48 MIPS. These load factors were simulated by defining two processes that execute 10 times per sec and occupy the Application-EDP for an appropriate amount of time, totaling 5% and 15% of a 3.1 MIPS processor.

The time slicing value of 100 msec was included in the simulation, as CSIM provides for a round-robin, priority-based scheduling with time-slicing. However, since the CSIM model does not provide for an overhead for switching between processes, the switching time of 250 μsec was not included. Thus, the value of the time-slice parameter does not affect the utilization figures for the EDP, as no overhead in switching between processes is incurred in the simulation. If included, this factor could increase usage of the EDP by a maximum of 0.25%, since the overhead would be incurred 10 times per second at most.

An important question concerning the process switch time is whether this overhead is incurred when a process is started. Since the workload consists of many short functions, this decision has a significant effect upon the utilization of the EDP. Inclusion of a process switch time for each execution of a job occurrence increased the load on the Application-EDP by about 10%. The overhead of 500 μsec for generating or accepting an NOS request is included in the simulation. It is assumed that this overhead applies to the application EDP and not the NIU EDP.

The internal time update process (ref. 7) was not included in the simulation because, although the number of instructions and size of the data message required were given, the frequency of occurrence of the update was not. Therefore, this function could not be modeled. An update occurring once per second would increase utilization of the EDP by 0.03% and the local bus by 0.016%.

B.3.1.2. Multibus II backplane: Table 4.4-2 in the MDSSC Report (ref. 7) contains the following entry for the Multibus II backplane that connects the Application-EDP, the NIU, and the BIU:

Table B-2. Multibus II characteristics

Multibus II backplane transfer rate, transfer frame size, overhead	32 MB/sec = 256 Mbits/sec 36 signal lines (32 data, 4 parity) 32 bit address followed by multiple (up to 7 total) 32-bit data signals

The backplane was modeled as a facility with a transfer rate of 256 Mbits/sec, a frame size of 224 bits (seven words), and an overhead of 32 bits per frame. The access model used was a round-robin with a time-slice of 1 microsec.

B.3.1.3. Network interface unit (NIU): Table 4.4-1 (ref. 7) contains entries that may pertain to the NIU:

Table B-3. NIU characteristics

NOS direct access service data overhead	40 bytes = 320 bits
NIU 2 byte packet throughput	Direct Access Latency = 3.24 msec + 1.7 msec *(#msg in queue) Non-Direct Access Latency = 9.5 msec + 2.8 msec *(# msg in queue)

The meanings of these entries are unclear, and no further information about the processing model of the NIU is contained in the text. Page 6-12 describes the functions of the NIU, but gives no information on its processing load. Some of the information in table 4.4-1 (ref. 7) was apparently obtained from an internal unpublished report on network modeling that was not available for this study. For these reasons, utilization levels of the NIU could not be obtained. The NIU was modeled as a passive routing device that acted as an interface between the Application-EDP and the core network without taking any processing time.

B.3.1.4. MicroChannel: The MicroChannel connecting the NIU and the NIA is described this way in the MDSSC Repport (ref. 7):

Table B-4. Microchannel characteristics

Microchannel data transfer rate, transfer frame size, overhead	13.3 MB/sec instantaneous transfer rate = 106.67 Mbps 8 bits per word (1 byte) 4 bytes/cycle transfer size (block size) 11% idle/89 % work = 12.36% overhead = ~4 bits overhead per block = ~0.037079 μsec

The microchannel was modeled with a transfer rate of 106,670 Kbits/sec and an overhead of 12.36% per transfer.

B.3.1.5. Network interface adaptor (NIA): No information about the NIA is given in table 4.4-1 (ref. 7), nor is any given in the text. The NIA was therefore modeled as a passive device that accepts data from the microchannel and transmits it onto the network while taking no processing time. No utilization figures for the NIA were obtained.

B.3.1.6. Core network: Table 4.4-1 (ref. 7) describes the characteristics of the FDDI protocol as follows:

Table B-5. FDDI network characteristics

Packet size	2048 bytes

Overhead	MAC = 28 bytes(~13.76%)(spec) + LLC = 10 bytes
	Total = 38 bytes = 304 bits

The transfer rate of the network is not given in the MDSSC Report, but it is known from other sources to be 100 Mbits/sec. The network was thus modeled as having a rate of 100,000,000 bits/sec, a frame size of 16384 bits (2048 bytes), and an overhead of 304 bits/frame. A target token rotation time of 0.006 sec was used to simulate the delay in capturing the token. This number was derived by allocating the maximum of 500 microsec for node latency for each of 12 active nodes on the network. This value does not affect the utilization of the network.

B.3.1.7. Bus interface unit: The BIU includes a Bus Control Unit (BCU) and a BIA for controlling the 1553 local bus and connecting the Application-EDP to the MDMs. Since no details of the BIU processing were described in the MDSSC Report, utilization figures for the BIU could not be obtained. The BIU was modeled as a passive routing device that transfers data between the backplane and the local bus.

B.3.1.8. Local bus: Table 4.4-1 (ref. 7) describes the 1553 local bus this way:

Table B-6. Local bus characteristics

Packet size	16 data bits/word with 4 bits Max of 28 words per frame overhead
Overhead	Message overhead of 4 words (commands, status, etc.) plus message 4 μsec gap time, plus 12 μsec RT response time.

The MDSSC Report (ref. 7) does not state the transfer rate of the 1553 bus, but it is known from other sources that the bus uses a 1 MHz clock rate for a transfer rate of 1 Mbit/sec. Table 4.4-1 (ref. 7) does give a value for the BIA transformer to hybrid interface transfer rate as 0.7 Mbps, but it is not clear if this is the rate for bits transmitted over the bus or if it is the effective data transfer rate after subtracting overhead bits. The maximum effective data transfer rate can be calculated as

0.74 Mbps, so these two numbers could be the same. The stated transfer rate of the BIA was ignored.

The bus was modeled as a facility with a transfer rate of 1 Mbit/sec, a frame size of 448 bits, an overhead of 4 bits/word and 40 bits/frame, and a time-slice for access of 616 microsec. In addition, a latency of 16 microsec for bus access was used for the gap times.

B.3.1.9. Multiplexer/demultiplexer: Table 4.4-1 (ref. 7) does not describe the characteristics of the MDMs (there are two connected to SDP 5. However, text on page 6-3 (ref. 7) describes the MDMs as containing a processor to run application programs, and table 5.3.1-1, on page 5-23 (ref. 7) describing the component variations run for SDP 6 implies that the processor speed of the baseline MDMs is 1 MIPS. Therefore, the two MDMs for SDP 5 were modeled as processors that can run application programs at a 1 MIPS rate.

B.3.2. Workload model– The workload model used by MDSSC Report is described in detail in their report. Tables 6.2-1 and 6.2-2 (ref. 7) give the specifications for more than a hundred different functional job streams that are to be executed on the Application EDP. Tables 6.2-3, 6.2-4, and 6.2-5 give the same information for the MDM processors.

In the CSIM simulation the workload presented in these tables was implemented by assigning each line of the tables to a separate process that was scheduled at the specified periodic rate. Each process utilized its processor to execute the number of instructions given in the table. Some of the processes then initiated a data transfer over the appropriate device.

The workload models also include utilization of the Run-Time Object Data Base (RODB). Table 4.4-1 (ref. 7) gives the following access times for the RODB:

Table B-7. RODB characteristics

Data conditioning, conversion, limit check	100 μsec/input + one of the following: 50 μsec/number or 30 μsec/discrete
Data base read/write latency for local application	100 μsec overhead + 25 μsec/object read or write
Data output: data preparation for remote application across network (source data collection process)	100 μsec overhead + 50 μsec/object = 150 μsec total

B.3.2.1. Job scheduling: All of the jobs in the CSIM simulation used the same method of job scheduling. Each line in the five tables specifying functions with their associated loads and frequencies was a separate process. The first occurrence of each job was calculated by taking a random number of uniform distribution between zero and the scheduling period. Each subsequent occurrence was scheduled at the previous time plus the job period. At the end of each occurrence, the process would wait until the next scheduled time. If that time had already passed, the job would not wait, but execute immediately. Some jobs having a period greater than the maximum simulation time (two min) did not execute because the time of the first occurrence of the job was past the end of the simulation.

B.3.2.2. Job models: Each of the five functional tables for SDP 5 in the MDSSC Report required a slightly different processing model. The models are discussed below.

B.3.2.2.1. Node SDP application software and derived object generation. These jobs are listed in table 6.2-1 (ref. 7). Most of the jobs involve execution of a number of instructions at a certain frequency. When these jobs run, they utilize the Application-EDP for the length of time required to execute the instructions, then wait for the next scheduled time. A small number of these jobs also involve access to the RODB. For these jobs, the Application-EDP is also used to condition the RODB objects: 100 microsec per occurrence plus 50 microsec for each object, with the numbers of objects for each job given in table 6.2-1. The message size given in table 6.2-1 is ignored, since it is not known how this affects the processing.

B.3.2.2.2. Node SDP 5 core network loading summary. These jobs, described in table 6.2-2 (ref. 7), involve transmitting data over the core network. The number of instructions required to prepare the output messages is given, as is the total size of the message. The Application-EDP is used to execute the number of instructions given. It is then used to prepare the RODB objects, although whether this should be done is not clear. The data is then sent via the backplane to the NIU for transmittal on the network.

B.3.2.2.3. Node MDM application function allocation. These jobs, described in table 6.2-3 (ref. 7), involve instruction execution only on one of the MDM processors. There are 22 jobs listed, with the first 11 assigned to MDM-1 and the remaining 11 assigned to MDM-4. There was no I/O associated with the jobs.

B.3.2.2.4. Node MDM I/O processing activity. These jobs, described in table 6.2-4 (ref. 7), involve interfacing with physical devices connected to the MDMs: reading sensors and controlling effectors. The number of instructions required to prepare for the I/O are given, and this is all that is simulated for these jobs. Since the sensor and effector devices are not part of this simulation, the actual amounts of I/O data involved were ignored.

B.3.2.2.5. Node MDM local bus traffic loading. These jobs, described in table 6.2-5 (ref. 7), involve transmitting data from the MDM to the Application-EDP via the local bus, the BIU, and the backplane. The resulting activity by the Application-EDP to store the data into the RODB was not modeled, but could be easily added to the simulation.

B.4. Results

Average utilization values for the components of SDP 5 were obtained from a two-minute simulation run. The results of this simulation and those reported by MDSSC are shown in table B-8. As indicated by the 100% utilization figure listed for both simulations for the Application EDP, this component is overloaded. The figures for the other components do not compare very well, but considering the many assumptions that were made in implementing this simulation model, the discrepancies may easily be explained.

A breakdown of overall usage value into contributions from the various elements of the workload and the different functions required by those elements was obtained by selectively enabling and disabling parts of the simulation. This analysis was done using 10-sec simulations. In addition, to confirm the utilization results from the simulation, a static analysis on the workload was performed to calculate the expected average utilization of the Application-EDP and MDMs.

Table B-8. Results of SDP-5 simulation

	% Utilization	
Component	CSIM	MDSSC
Application EDP	100.0	100.0
Backplane	0.0	0.2
Local bus	6.0	3.5
MDM-1 processor	86.3	44.3
MDM-4 processor	92.0	47.1
MicroChannel	0.0	0.4
Core network	0.1	5.1

B.4.1. Application-EDP analysis– The results of the breakdown analysis for the Application-EDP processor are shown in table B-9, which presents the contributions to the Application-EDP load from the functional elements of the workload presented in the MDSSC Report (ref. 7). The utilization contributions were derived from the static analysis of the workload functions and from simulation runs in which the parts of the workload were selectively enabled. The results in all cases are in good agreement between simulation and static analysis.

Table B-9. Application-EDP Analysis

		% Utilization	
Workload element	Source	Simulation	Static
Application function	Tbl 6.2-1	58.1	58.0
Derived object generation	Tbl 6.2-1	2.8	2.8
Core network object generation	Tbl 6.2-2	31.7	31.7
Core network message creation	Tbl 6.2-2	20.2	20.3
System management	Tbl 4.4-1	5.0	5.0
OS/Ada RTE	page 6-12	15.0	15.0
	Total:	132.80	132.80

B.4.2. MDM analysis– The breakdown of utilization of the MDMs by workload functional elements as described in tables 6.2-3, 6.2-4, and 6.2-5 (ref. 7) are shown for MDM-1 in table B-10 and for MDM-4 in table B-11.

Here as well, the results from the simulation and from the static analysis are in agreement. The MDSSC Report (ref. 7), however, shows a significantly lower utilization for both MDMs—about 50% of the usage shown in table B-11. This would indicate that the processing model used here for the simulation and static analysis is not the same as used by MDSSC, but there is no further information concerning their model.

Table B-10. Breakdown of MDM-1 utilization

		% Utilization	
Workload element	Source	Simulation	Static
Application function	Tbl 6.2-3	39.0	39.0
I/O processing activity	Tbl 6.2-4	9.8	9.8
Local bus traffic loading	Tbl 6.2-5	37.5	37.6
	Total:	86.3	86.4

Table B-11. Breakdown of MDM-4 utilization

Workload element	Source	% Utilization Simulation	Static
Application function	Tbl 6.2-3	42.0	42.0
I/O processing activity	Tbl 6.2-4	10.3	10.3
Local bus traffic loading	Tbl 6.2-5	39.8	39.8
	Total:	92.1	92.1

B.4.3. Other components– The utilization levels of the other components included in this simulation were all low—less than 10% of capacity in all cases. The agreement between the two simulations is not very close in these cases, however. The disagreement for the Core Network and MicroChannel can be explained because the MDSSC simulation included loading of the network by SDP nodes other than SDP 5, and this was not included in the CSIM simulation. The disagreement between load figures for the 1553 local bus are similar in nature to the MDM usage figures, and the difference may be due to the same cause since the usage of the MDM and the local bus are closely related.

B.5. Conclusions

The CSIM simulation being described in this document is in general qualitative agreement with the MDSSC simulation in pointing out potential bottlenecks in the DMS: the Application EDP and, to a lesser degree, the MDM processors. The exact utilization levels do not agree precisely, particularly for the MDM processors, which disagree by a factor of two. A discrepancy of this magnitude is not unexpected given the lack of detailed knowledge concerning the processing model of the DMS available for the current study. If more information about the processing model used in the MDSSC Report were made available, the causes of the discrepancies could be investigated.

The amount of specific detail available for describing the workload permits a static analysis of the workload model to accurately predict the utilization levels experienced in the simulation. For this reason, the utilization figures alone could have been established without the simulation. Of course, other performance factors, including response times, delays, and maximum queue lengths, can be predicted with an accurate simulation. However, the current detail level of this study makes such data questionable. Better models of processor and operating system execution would be required to extract accurate values from this simulation.

C. Advanced Automation for SSF

The requirements of four examples of system functions that critically depend on the capability of the DMS are discussed here: the ECLSS, the PMAD, the AANMS, and the Flight Telerobotic Servicer (FTS).

C.1. The Environmental Control Life Support System (ECLSS)

The goal of ECLSS is to utilize closed loop life support technology to minimize resupply and return logistic penalties. Maintaining respirable atmosphere, potable and hygiene water, and emergency operation are its primary duties. The availability of on-orbit computational resources will impact the overall level of automation, as well as the number of sensors and effectors. Complete closure of the ECLSS loop is optimal, but has been deferred until later in the SSF program lifetime.

ECLSS is designed to operate in a range of automation capability—from having a crew to provide corrective and preventative maintenance, manual overrides, non-critical fault isolation, and redundancy management to full automation. Onboard computers are responsible for performing most process control functions, interactions with other systems, fault detection functions, and criticality 1 ORU fault isolation. If proper manual overrides are inhibited by the crew, the ECLSS can perform emergency reconfiguration or take appropriate response to an emergency, such as suppressing a fire. Considering the life critical nature of the ECLSS, most automated functions that support it are critical.

Designed to be primarily automated on-orbit, the breakdown between ground and space processing has not been completely defined. However, the effect of the power, weight, and cost enforcements is leaving little choice to the system designers. The currently allocated resources for each subsystem have been reduced so much that only real time critical functions are to be housed onboard. All functions that can be achieved on-ground must be done so. This is also with the intent that eventual upgrades will take place and some of these functions will migrate back on board SSF.

The current on board ECLSS supervisory system design, using soft real time, utilizes a total of 1 SDP worth of processing power and memory size (4 MIPS/4 MB). That is, although several SDPs are shared among subsystems, the total power and memory capacity total 1 SDP. The hard real time system is distributed among approximately 20 MDMs and the composite required capability is equivalent to 10 to 12 fully loaded MDMs. The fault sensors are directly connected to the MDMs. It is only the supervisor section that interfaces with the DMS. This is for information storage, crew updates, and further data analysis.

The reason for automation is that more can be achieved with less. Although the current design of the ECLSS ensures enough sensors are in place to achieve what is critical, there are no leak detection sensors. To compensate for this, the advanced automation program will analyze the characteristics of failures downstream in the model base and then detect the leak origin or approximate vicinity.

The automated ECLSS consists of rule-based fault-detection algorithms for baseline on orbit ECLSS regenerative systems, with initial operational capability model-based diagnosing on the ground. The model based diagnosis systems will move to flight software when computer resources on board permit. The design of the ECLSS automation system is that the rule-based fault detection algorithms be on board and complemented with model based diagnosis autonomy in the ECLSS ground support center. The primary limitation to achieving this is a lack of computational resources in orbit and on the ground. The computational resources of SSF must be upgraded to allow migration of high fidelity automatic fault diagnosis software on board. The on board fault detection algorithm will meet the Ada task interface.

Water drinkability would be rapidly assured through automation of the water quality monitor output. This requires real time processing of chemical and/or microbial analysis in the life support control system. On board processing would require fast symbolic and/or neural net processing.

For the current lab system, the advanced automation ECLSS is using Sun Risc, which is eventually being ported to Sun 386i and SunOS using KATE, ART/Ada, Lisp and TAE+. The software is a mix of 80% symbolic and 20% numeric code where 500 KB is the size for the detection and 4 MB for the diagnosis. In-orbit applications have been limited to fault detection because the model-based fault diagnosis requires more processing capabilities than are available. The primary impact of this is that fault reasoning is deferred to the ground. A large sacrifice in fault tolerance of the system, the turnaround time from detecting a fault to its isolation and recovery would be delayed by twice the amount of time it takes to transmit over the TDRSS link. In a critical situation, such as a gas leak in the respirable air in the HAB/LAB, this extra time could be intolerable.

The major software hooks and hardware scars necessary for evolution to a more autonomous ECLSS have been identified. The advanced subsystem FDIR requires component sensors to be available from the RODB within 1 sec, allowing the subsystem's control loop latency of 5 to 10 sec. This provides real time fault detection and fault preventive reconfiguration to use 3 to 8 sec, with communication to the subassembly monitoring process taking 2 sec. A software process location transparency (dynamic memory allocation) is also called for. This was explicitly removed from baseline. The automation efficiency would be increased by the use of model-based reasoning tools like KATE and ART/Ada for early design knowledge capture. It is recommended that these tools be added to the SSF software support environment. This model-based reasoning approach to subsystem FDIR would allow minimal use of explicit leak detection sensors by identifying leaks using the baseline process control sensors. The result would be the same, yet would require fewer on board resources.

For the current laboratory system, which is a subset of the initial design, 4 MIPS with 4 MB are sufficient. By nature of the problem, it is a slow system and it works. The turnaround time on sensor sample analysis is, at worst case, minutes, thus there is enough time to wait for commands from the ground based control center. With increased use of the station, the demands on the system will grow. This also does not account for on board fault handling beyond detection. There is not enough capability available to meet this growth. To support timely inclusion of this automation technology, symbolic processing in a real time environment, dynamic memory allocation and much more memory are necessary.

C.2. Power Management and Distribution (PMAD)

Another system targeted for evolutionary upgrades with increased automation is the PMAD system. The PMAD Automation Evolution project is promoting the operation of a highly autonomous, user-supportive PMAD system for the SSF HAB/LAB modules with a fully integrated user override capability. Automated aspects of the system include immediate power system safeguarding, short term load shedding, limit checking and reporting, and redundant load switching. Also automated are schedule implementation, fault detection, fault isolation, fault diagnosis, scheduling load prioritization, and dynamic scheduling. The critical functions of the PMAD system are implementation of normal operation, FDIR, scheduling, and load prioritization. Though initially targeted to support the SSF from the ground, it will eventually be installed on board.

The current laboratory system uses Common Lisp and Common Lisp Object System on the Symbolics and Motorola 68010 computers. To better reflect the end environment of the SSF, it is being ported to Solbourne 5/501 and Q-max 386. Initial objectives of the project can be met with hardware currently used, but as the hardware system is redefined and trimmed to more closely resemble resources available for use with SSF, the sacrifices are imminent. More computing power at the symbolic level is needed. With more than 50 thousand lines of code, of which 45% are symbolic, dynamic memory allocation must be allowed. To meet the current processing requirements, a total of 20 VAX MIPS is required for the Solbourne, 2 MIPS for the 386 and about 5 MIPS for the Symbolics. Also necessary is 0.5 MB memory at the controllers and 2 to 3 MB at higher levels. If a symbolic processing machine is not available, the capabilities it provides must be, otherwise a new system design is necessary.

C.3. Advanced Automation Network Monitoring System

The Advanced Automation Methodology Project is the development of an engineering methodology for advanced automation systems for use in the SSF program guidelines. The two independent systems in this project are the AANMS and the Recovery Procedure Selection Application. Upon completion, each system will be integrated into existing SSF test beds.

This involves the development of a network monitoring system capable of intelligent fault monitoring of FDDI-based networks. Designed to provide predictable, hard-deadline scheduling in real time within the SSF constraints, the total time to respond to any fault is limited to within 1 sec. This system will automatically detect, isolate, and diagnose network faults based on data passing on the network. The network monitor is important for telerobotics and docking-critical systems. Therefore, it is necessary to identify bottlenecks quickly so that the system can be reconfigured with minimal loss. The AANMS also provides a flexible user interface for semi-automated systems. The primary focus is to prove functionality, so hardware is not currently a prime concern. When the system concept is proven it will be ported to state-of-the-art system technology.

The driving requirement for porting systems to the SSF is to use COTS systems. However, the technology required to successfully accomplish this job is not available. The current laboratory system is based on a Silicon Graphics experiment with NetVisualizer: conventional network monitoring software, running on their 16 MIPS personal IRIS 4D/25 workstation. This software was able to process only 4% of the tested FDDI traffic. Yet this is for data capture and extraction functions only. Processing of this data would require additional capability. This performance is not in the baseline design, nor is it currently identified in the upgrades. The memory requirements for storing and analyzing the data are prohibitive. The FDDI will generate (at most) 12.5 MB/sec of data. For data capture, this implies 40 MB of RAM are required for each 3 sec. Other functions are projected to require around 24 MB of RAM. This implies a need for aminimum of 64 MB. Clearly the AANMS concept is beyond the current state-of-the-art for the DMS.

C.4. Flight Telerobotic Servicer (FTS)

One presentation at the Computational Requirements Assessment Workshop, July 9-11, 1990, was given by Stan White of Martin Marietta on the FTS Data Management Processing System (DMPS). White indicated that the prime computational driver is the critical path "around the loop" flow. This system involves the workstation where the telerobot is located and controlled. The workstation has a hand controller that feeds commands to the 80386/387 clocked at 16 MHz with 256 KB RAM. This is linked via 1553 bus to the telerobot control computer. The original requirement for the FTS was that the throughput be 2.25 MIPS, 2 MB memory, 32 bits floating point, running Ada on a 1553 bus. The baseline of the 80386 fits these requirements and the mission has not realized sacrifices due to this. However, the FTS program is undergoing rescoping/changes and the selected SDP cannot fulfill the requirements. This is due to the growth in the around-the-loop timing, due in part to the increased algorithm complexity, better software implementation definition, and reduced SDP performance (3.1 MIPS).

The current plan for evolution is that a margin of 50% processor usage and 35% RAM is provided; however, these do not address around-the-loop timing. Spare bus nodes are available for additional processors, but parallel processing does not improve around the loop timing—the algorithms used here are serial.

The DMPS architecture study, initiated by Goddard Space Flight Center, indicated future requirements of 200 Hz around-the-loop (versus 50 Hz) with increased vision processing in line with increased autonomy and graphics simulation for path planning preview and training. These capabilities cannot be achieved by the 80386. Without a design with a faster system, these are not evolutions that can be realized quickly without sufficient redesign in space. The FTS would probably remain as it is deployed, with little or no autonomy, initially or evolutionarily.

It is recommended that, in the baseline design, a faster computational platform, with increased performance by a factor of six, multiple CPUs (at least four), and spare slots for growth be used. As previously stated, no sacrifices have yet been realized to fit to the chosen platform, but control algorithms, software, and hardware reviews are conducted to squeeze out performance. So far in this project, the primary lesson learned is that the performance margin, initially set at 25% for around-the-loop control algorithm growth, is too small. This is being compensated for by targeting Ada RTE to bare machine, thus eliminating the operating system overhead. Greater emphasis must be placed on performance margins.

D. Integrated Design Constraints

The latest trend in technology and business philosophy points the way toward fully extensible, mission-adaptive computer platforms. Such "platform" designs offer advanced inter-component, inter-board, and inter-system connectivity. They allow program-specific processor modularity, while remaining open to a continuous stream of increasingly advanced common components and modules with minimal mission disruption and risk. Due to its long mission life, the SSF program demands component design decisions based on the ability to enhance minimum cost for maintenance and upgrade in the long run. Its DMS can serve as a test case for using open system "platform" design.

When considering the ability of hardware/software technology to evolve in performance/function over the long term, many sub-component factors must be considered. It is only at this subsurface level, that these ingrained "evolvability and compatible scalability" traits are visible. Evaluation at this sub-component/sub-issue level is both critical and difficult.

Figure D-1 introduces a discussion of eleven major "architectural dimensions" critical in determining the long-range "scalability" of any proposed system design. Constraint in any category affects the whole. The first two categories relate to issues of connectivity, the next six describe critical processor and component technology issues. The ninth deals with "open systems" business and cost factors; the tenth relates to software considerations and the eleventh, full-system reliability characteristics. Each system component must be able to evolve through the full spectrum of each column in order to survive. The difficulty in designing a long-range scalable system can be better appreciated when one considers the complexity in trying to satisfy the constraints represented in these eleven issues.

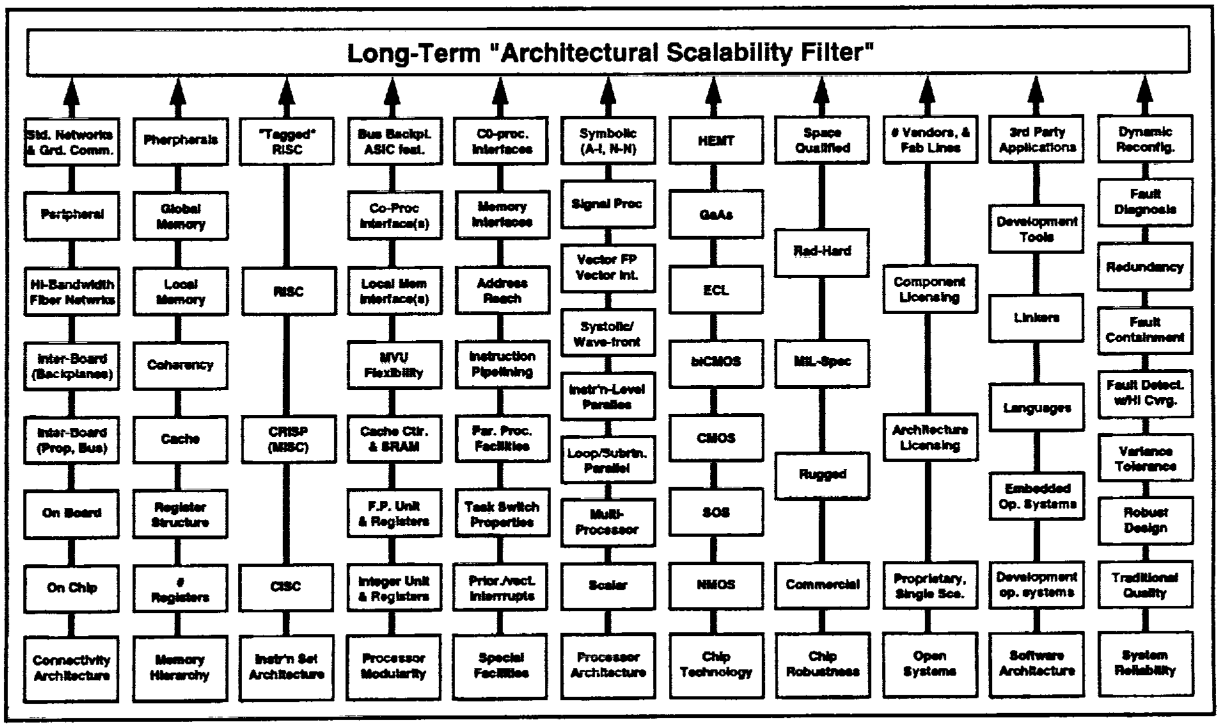


Figure D-1. Integrated design constraints for computer systems.

D.1. Connectivity Architecture

The connectivity architecture is the foundation of a computer system. Many levels of connectivity must be addressed (fig. D-1). Without sufficient/balanced connectivity from top to bottom, it is difficult to attain either optimal present performance or long-range performance expansion.

D.1.1– Recent innovations in the chip- and board-level connectivity domains are too numerous for detailed description in this report. These innovations are closely related to the underlying chip technology being considered in section D.7. Because each technology (CMOS, GaAs, etc.) exhibits different performance and gate/transistor densities, different methods of board- and chip-level connectivity are required. Complicating these density/connectivity equations still further are the additional baseline architectural design factors outlined in sections D.2-D.8.

D.1.2– Standards committees generally focus on the bus, backplane, network, and peripheral levels. Major innovations and significant trends are occurring in these areas.

D.1.2.1. The terms "bus" and "backplane" are often thought of as synonymous. They connote specific inter-board connectivity wiring designs. For the purpose of this report, "bus" will be used to denote the high bandwidth, proprietary board interconnect designs found throughout the industry. "Backplane" is used to describe board interconnect designs specifically defined and generally accepted within the standards community. Major cost savings have been realized by both government and industry from the advent of standard backplanes such as VME and Multibus II. Buses, on the other hand, are still quite common and tend to be motivated by two factors: (1) a realistic need to connect boards (such as processor and memory boards) at bandwidth levels above those possible on standard backplanes, and (2) the widespread desire of many computer manufacturers and integrators to maintain proprietary performance advantage. Between these philosophies are found widely accepted, but still proprietary, standards. These are discussed in section F.9.

The standards community have establish a new class of scalable, high bandwidth backplane standards." These efforts have primarily been motivated by a universal realization that present backplane standards are reaching the limits of their ability to sustain higher levels of performance. The most visibly successful of the new, scalable standards efforts has been the IEEE P896.1 FutureBus+. IEEE, U.S. Navy NGCR and SPAWAR programs, and Pentagon standards bodies have already accepted it. Both the VME and Multibus II international trade associations have said that this standard will be their next generation. Intel, DEC, Motorola, and HP/Apollo have also adopted it, and many others are reportedly nearing announcement. The U.S. Navy has already contracted with three different defense contractors to build six different processor boards for FutureBus+. From sheer market acceptance criteria alone, NASA should seriously consider FutureBus+ as a future backplane standard throughout its programs. Technically, the FutureBus+ specification is impressive:

- The physics of the backplane has been thoroughly assessed (power requirements, microwave properties, etc.).
- It is asynchronous, allowing boards to run at different speeds, including new generations of compatible upgrades over time.
- It is performance scalable, from 100 MB/sec to 3.2 Gbytes/sec.
- It is data width scalable, from 32, to 64, to 128, to 256 bits wide.
- Multiple application specific integrated circuit (ASIC) vendors are already committed to building interface ASICs.
- Both VME and Multibus II are designed to interface with existing boards without redesign.
- It has full parallel processing and multiprocessing protocol, cache coherency, and processor synchronization mechanisms in its design.
- It supports both real time (priority based) and fairness (equal opportunity) contention arbitration schema.
- It adopts two intercompatible transmission methods to allow full system performance optimization.

D.1.2.2. One of the most advanced innovations in connectivity architectures is the combination of gigabit fiber optics transmission media with gigabit circuit switches. While these are typically viewed as inter-system network developments, their reach extends far beyond. The first level of extension is the ability of this gigabit media to connect multiple types of backplanes together into one inter-operative system. Initial plans of companies in regard to backplane interface are in the following order: (1) Nubus (Macintosh and NeXT), (2) VME bus (Sun, MIPS, embedded real time processors, etc.), (3) ATbus (IBM PCs), Microchannel (IBM PS/2), (4) Multibus II (embedded processors), and (5) FutureBus+. The second phase of development leads to a domain that might be called "bus-less" or "backplane-less" system design. Bus-less designs could offer benefits in heterogeneous and homogeneous multiprocessing, long-term evolvability, mission adaptive modularity, and reliability. These designs should be evaluated in regard to minimization of cost, power, and weight while maximizing performance, functionality, reliability and "openness."

AT&T, Network Systems, and Ancor Gigabit switch offerings should be studied as well as American National Standards Institute (ANSI) Fiberchannel 0, 1, 2 and scalable coherent interface fiber-optic connectivity standards. Advanced protocols, such as XTP (Protocol Engines, Inc.) and optimized TCP/IP, should also be studied. From an ASIC hardware standpoint, protocol transparency, processor/coprocessor appealability, power requirements, and space qualifiability should also be studied.

Linear buses, whether backplanes such as Multibus II or networks such as Ethernet, must constantly deal with collision detection and arbitration overheads. Each node must delay a round-trip propagation time (after it sees the last bit of the preceding packet) to ensure that a new packet does not interfere with a preceding packet. Ring structures, such as the IBM token ring or FDDI, also suffer from similar delay of at least one round trip propagation delay before any node can receive a token to transmit another packet. For circuit switches (such as crossbars or omega-net-like space switches) the round trip propagation delay is required to perform the initial point-to-point connection, but this is performed only once. Subsequent connections can be established while a current message is being transmitted. This property could be viewed as a connectivity analog to computational pipelining. It offers the opportunity of board interconnectivity at rates far exceeding today's standard backplanes, in any combination, at physical locations far beyond that which is possible on buses and backplanes.

D.1.2.3. From a peripheral connectivity standpoint, many other new, advanced, scalable standards are forming. The general progression path is from storage module device to small computer system interface (SCSI) to intelligent peripheral interface 1, 2, 3 to high performance peripheral interface (HPPI). All are now in production in the commercial market and should be studied for their space applicability. Cooperative research efforts could be easily invoked involving such projects as GSFC's configurable high rate processing system, which utilizes the ANSI IEEE HPPI interface, and advanced laser disk and laser communications projects such as the LaRC information sciences experiment system project.

D.1.2.4. Lower speed connectivity (such as 1553B and IEEE 802.4) and ground link communication criteria are another layer of connectivity that must be advanced for future mission applications. These areas are critical to decisions such as the distribution of computation between ground and space. Similar developments of advanced connectivity are being studied in the Defense Advanced Research Project Agency, Department of Defense, Department of Energy, Strategic Defense Initiative Office, U.S. Navy, U.S. Air Force, and intelligence communities.

D.2. Memory Hierarchy

Items in the memory hierarchy column in figure D-1 deal with the many levels of consideration that must be given in regard to keeping data operands and instructions close to functional units. As stated before, it does not matter how fast a processor is if it cannot send and receive data at a rate equivalent to its processing capabilities. The connectivity architecture described in section D.1. is the media on which the data is conveyed. The memory hierarchy, however, serves a critical function in the conservation of over-all, system-wide bandwidth by minimizing the need for electrically distant data transmission. Insufficiency in either local memory capacity or performance will be felt throughout the system and must be studied for each candidate architecture in great detail. Additional memory features are also critical to long-range evolvability. Vector processors require a global memory that is organized into "banks." If operands are to be fetched from random locations, memory access facilities such as "gather" and "scatter" could be critical. If parallel processing techniques are used, coherency mechanisms must be included in the design to ensure that no processor is working with stale data that it thinks is current. Register structure, especially in RISC processors that are typically equipped with many local registers, can be quite critical in performing high-speed task switching in a real time environment. The memory hierarchies below the surface of any processor/system candidate are critical to its ability to evolve over time and must be studied before selection is made.

D.3. Instruction Set Architecture

This dimension of scalability includes (fig. D-1): CISC, Medium Instruction Set Computers (MISC)—also called complexity-reduced instruction set processors (CRISP), RISC, and Tagged-RISC (RISC processors with additional data-tagging facilities for use in artificial intelligence applications). Recent trends have been in the direction of the arrow in figure D-1. IBM is generally credited with invention of the RISC architectures with the development of its early 801 processor chip. RISC processors have been developed because hardware implementation of complex instructions slows down the clock rate of the entire processor and consumes precious chip area. Circuit paths on present day chip surfaces are reaching the point where they are literally molecules wide. Reduction of feature sizes has plateaued. Similarly, molecular purity of large-scale chip surfaces is extremely difficult to maintain. Consequently, chip area is scarce. In CMOS, for example, there is only about 1 cm^2 to work with. In ECL and GaAs, there is even less surface area. Processor designs with a large number of, or highly complex instructions are an inefficient use of scarce chip area. RISC, reduces both the number and complexity of instructions. As a result, more of the working chip area is dedicated to the most frequently used functions and operands can be kept close to functional units (by using a large registers set). RISC instructions are typically simple, symmetrical, and easily decodable because of their fixed length. Instructions are thereby streamlined to achieve an instruction execution rate at or below one clock cycle per instruction. Extensive use of pipelining can also be made. Another performance enhancement comes from the large local register file, which may reduce access to global memory. Reliability desires are also served by the RISC architectures on two fronts:

1. Microcode is generally eliminated in RISC architectures, moving complex functionality to the compilers and libraries. This allows the verification and validation functions to occur in a more visible way.

2. Reliability causes are also served by the simpler, more efficient hardware.

CISC processors, because of their high gate/transistor density are hard to scale in performance. Design simulations and fabrications technologies are very complex. Further, high gate/transistor densities also prevent easy transition to faster (but lower density) technologies, such as emitter-coupled logic (ECL), GaAs, and high electron mobility transfer (HEMT).

In summary, the instruction set architecture is one of the major contributors to chip complexity. More gates or transistors on a given chip means less power per gate, reduced driving strength per gate, and, therefore, lowered reliability. Also, large circuit sizes often result in longer and more complex signal delays. These types of problems inevitably reduce operating speed, yield, and reliability while increasing design complexity, cost, and development time. MISC is simply between CISC and RISC. Some RISC processors, however, are aimed at the commercial markets where high levels of integration can be translated into simple economies of mass market scale. In addition, for space applications, high levels of integration may actually be counterproductive. Processors may include an on-chip cache, which has no EDAC circuitry and therefore should be disabled for space or life-critical applications (the i486 is a good example). The result is wasted space and power for the additional external cache chips. On-chip memory management already included may also not be well suited for highly robust real time applications. Memory access characteristics of the memory management unit (MMU) might have multiuser "fairness doctrine" rules built in that may conflict with a real time program desiring full, uninterrupted control.

D.4. Processor Modularity

The processor is composed of many functional units (fig. D-1). The nominal list includes:

1. The integer unit that typically decodes all instructions and processes the non-floating-point instructions.

2. The floating-point controller, floating-point functional unit, and floating-point registers.

3. The cache controller and cache static RAM, and the MMU.

Additional processor units and facilities such as local memory interfaces, coprocessor interface pinouts, and bus/backplane interfaces may all be integrated onto a single, high density chip. High density integration will, if designed specifically for the application, yield high potential levels of performance. It will also yield economies of scale if the markets are sufficient in size to pay for the heavy cost of development. On the other hand, in military specifications (MILSPEC) or space-qualified applications, high-levels of integration may be counterproductive. Discrete components allow the designer to more easily tailor the specific processor characteristic to one or more specific applications. With individual units, per-chip gate counts are lower, allowing far easier transition to faster, but less dense technologies. Individual units may also be replaced more easily as unit innovations (or corrections) occur.

D.5. Special Facilities

Additional design features such as real time facilities, parallel processing, pipelining adaptations, memory access facilities, coprocessor interfaces (for such things as signal processing, image processing, etc.), can be critical to functional flexibility and performance evolvability in NASA applications (fig. D-1). Both the capabilities and trends of special facilities should be assessed by NASA for its candidate processors.

D.6. Processor Architecture

This dimension describes the character of the specific machine instructions and the interplay between individual instructions and processor units (fig. D-1). In scalar processing, a single instruction performs a single operation generating a single result. In vector processing, a single instruction performs its operation on a series of similar input operands, producing a series of pipelined results much in the same fashion as the individual steps of a factory production line. Vector processing is computationally efficient because after an instruction is decoded once, volumes of data can be read in from main memory and processed at clock-rate. Multi-processing is the use of multiple processors for many different programs. Parallel processing combines multiple processors to speedup an individual computation. Parallel operations can occur at the instruction level (often called superscalar, wide instruction word or dataflow), the loop, or subroutine level. Systolic techniques tend to pump instruction, loop, or subroutine result sets through a globally networked scheme (like a n-dimensional production line) until final results are created for all operations. In the parallel cases, instructions can be either scalar or vector operations. Instructions can be tailored to work with integer, floating-point, or logical/symbolic information formats.

Multiprocessors require high bandwidths, low latency CPUs, buses, memory, and I/O connectivity. Parallel processing requires facilities that prevent race-conditions, memory-locking, as well as cache coherency. For example, prevention of two processors storing to the same cell in memory at the same time. Test-and-set instructions help implement semaphores. Memory locking instructions can force momentary sequentiality of instructions so that operations do not occur in the wrong order. Cache coherency measures prevent data from going "stale" in separate caches—one processor being unaware of actions that occurred in another processor on its data items. Coprocessors, such as vector floating-point, vector integer (imaging), and signal processors (processors tailored to perform fast fourier transform (FFT) and convolutions at high rates), are highly dependent on the interfaces to the processors that invoke them. These interfaces can be at either chip, board, or system network level and will produce efficiency levels in direct relation to the nature of the connectivity paths. Some processor chip sets provide coprocessor interfaces (pinouts) at the chip level and are, therefore, most efficient and flexible in regard to the type of coprocessor.

D.7. Chip Technology

The chip technology column in figure D-1 is organized according to raw technology speed of operation. Transition from CMOS gate arrays to GaAs produces at least a ten-fold transparent increase in performance. The less dense and complex the instruction set architecture, the easier the transition through the full spectrum of technologies. Digital GaAs technologies are just now reaching 20 thousand gate densities. The highest device density that has been reported for advanced radiation-hardened (VHSIC Phase II) technologies to date, developed by IBM, TRW, and Honeywell, is around 3.5 million transistors. A recent projection of the gate densities for 386 successors showed the i486 processor with 1.2 million transistors, the i586 with 4 million, the 686 with 22 million, and the 786 at 100 million. At least two RISC processors, on the other hand, exhibit transistor densities in the range of 50-75 thousand. NASA should carry out an in-depth study regarding the radiation and SEU resistance of the many oncoming technologies. From there, an analysis of candidate CISC, MISC and RISC processor architectures should be performed in regard to the gate/transistor densities required for all processor-related components. This will give an indication of the optimum combinations of chip technology and processor architecture in regard to long-term scalability.

D.8. Chip Robustness

As stated in the previous section, different chip technologies exhibit varying levels of robustness in regard to being able to transition the full spectrum of this column—from commercial grade, to "rugged" to MILSPEC to radiation-hardened to space (SEU) qualified. The next paragraph also describes the many benefits of open architectures in multi-sourcing and multi-source fabrication line availability. Some of the chip foundries building open-systems processors, for example, already have at least radiation-hardened fabrication facilities.

D.9. Open Systems

Industry is quickly moving from the old, competitive, "proprietary" business model to the cooperative "open systems" business model. At the surface there is still much confusion and hyperbole—everyone is claiming to be "open." NASA should do all it can to foster, shape, and take advantage of the open systems model. The key difference between a true open systems company and one that only claims to be open is found in the licensing documents that should be signed by those who wish to use the open technology design rights. Essentially, there are two classes: (1) an "architecture" license, and (2) a "component" license.

The false open systems company will only offer an architecture license, which allows those who sign it the right to build systems using the (single-source) chipset. In this manner, the company who owns the proprietary rights to the technology can fully control both the behavior of the licensee and the price of the components. An analog also exists in the software world. Control in this case is maintained by source-right licensing. Many companies will port programs to a machine, but will not license the source rights. Others are open and will license the source rights—but in the software world they are few and far between.

A true open systems company offers component license along with the architecture license. This allows multiple chip vendors to build instruction-compatible versions of a processor, for example, that creates two distinct benefits for the user community. The first relates to the cost benefits that result from multiple sources for a given part and from a healthy level of competition. The second benefit of multiple sourcing is more technical. One licensee might have centered its expertise in building extremely low-cost, high-volume parts using CMOS standard cells, while another can bring its expertise in building MILSPEC, radiation-hardened, and space qualified chipsets to the government market. A third vendor may have the ability to add high-performance features such as "super-scalar" (instruction-level parallel) to the chipset. The owner of the rights benefits by expanding his market into domains they could never address alone. On the other side of the coin, the chip foundries and vendors benefit by being able to use a processor that has already gained popularity in the commercial market and does not have to reinvent the entire set of OS, languages, and third party applications.

MIPS and Sun are examples of open systems companies. There are five different vendors licensed to build compatible chipsets for MIPS, and eight vendors for Sun's scalable processor architecture. Motorola and Intel, on the other hand, are only single sources with some exceptions.

D.10. Software Architecture

From the software standpoint, developers in the real time arena (such as aircraft avionics) have found that there are two different sets of real time requirements: (1) the development environment, and (2) the embedded environment. They concluded that UNIX offers many positive facilities for the development environment, especially in its ability to combine design, development, diagnosis, debug, and documentation tool sets. The embedded environment, however, often requires highly optimized kernel OS functions which a UNIX-based system cannot offer due to its innate code mass at the kernel level. If the kernel OS will fit in ROM, instead of 2 MB of memory, both power minimization and performance/reliability maximization causes are served. Smaller OS kernels offer higher performance in priority control, interrupt handling, task-switch overheads, and dispatch latency facilities, as well as raw program performance. Additionally, these real time kernel OSs have demonstrated that they can more easily and efficiently invoke the newer and more innovative language constructs (such as Ada Tasking) in performing real time task management than is possible by retailoring the far more massive UNIX kernel structure. A recent trend among these real time kernel OSs is to bridge the gap between the development (UNIX) and embedded (RTOS) environments. This path allows the direct linkage of the two, via standard UNIX socket software interfaces, remote procedure call communications tools, I/O calls, and even the same memory management routines.

These UNIX facilities, and the third party applications that run on them, along with standard UNIX computer aided software engineering (CASE)-like tools and debugging facilities (using UNIX connectivity tools) are available for on-line, real time diagnostics and debugging purposes to ease verification and validation. Full evaluation of the more prevalent of these real time kernel OSs should be performed for both present and evolutionary purposes.

E. Fault Tolerant/Redundancy Management Analysis

This section describes the analysis process and modeling techniques used in support of the DMS evolution report. The effort was based on a system modeling technique called Digraph Matrix Analysis (DMA). The process, two system modeling techniques, and an example of the use of each technique is presented. The results of these investigations are useful in determining single- and double-points of failure in the DMS, as well as predicting the performance of various system

E.1. Evaluation Approaches and Criteria

E.1.1. Evaluation criteria– The general failure tolerance requirements for SSF systems (ref. 17, paragraph 3.1.10.1.1) are:

Systems functions which are essential for crew safety or Freedom Station survival shall be designed to be two-failure tolerant as a minimum (except primary structure and pressure vessels in rupture mode) and shall be restorable on orbit to the original failure tolerance level. During initial assembly and periods of maintenance, these functions shall be single-failure tolerant as a minimum. Systems functions which are essential for critical mission support shall be designed to be single-failure tolerant as a minimum and on-orbit restorable. Safety detection, monitoring, or control functions shall also be designed to be single-failure tolerant as a minimum and on-orbit restorable. Noncritical functions shall be designed to fail in a safe mode and be on-orbit restorable. Functional redundancy required to provide this failure tolerance may be common or uncommon and may be achieved by considering the space station manned base as an integrated system.

E.2. Failure Tolerance/Redundancy Management Process

An FT/RM evaluation process is described in the FT/RM Design Guide (ref. 18). Since SSF is a long-duration facility, the FT/RM analysis is focused on the functional requirements of integrated systems. The FT/RM process is initiated with the definition of integrated system functions and assignment of criticality (ref. 17, table 3-6). Once the functions are defined and their criticality established, the design is analyzed to verify its consistency and completeness. The redundancy requirements corresponding to each criticality category are also given in table 3-6 (ref. 17). These requirements, along with a function's concurrency and period of criticality, are examined in the functional criticality analysis. The next step in the FT/RM process is determination of the design requirements for each system and element in order to implement FT/RM. Before the design is placed under configuration control, its fault tolerance is verified.

The directed graph (digraph) models for several critical functions are currently being constructed and reviewed. A more detailed description of the digraph approach and several modeling examples are given in the next section.

E.3. Modeling Techniques

One widely used technique for modeling complex systems is combinatorial models. DMA is the combinatorial modeling approach used by WP-2. The approach is based on the representation of the flow of effects between components of a system as a digraph. Digraphs include logical connectives such as and-gates and or-gates that allow the representation of redundant components and multiple dependencies. Once the digraphs are constructed, they can be processed to provide a designer with information including the components whose failure will cause failure of the entire system, called singletons, and pairs of components whose combined failure will cause failure of the system, called doubletons. With a similar procedure, the analyst can also study tripletons (triples of components whose failure will cause system failure). When available, probabilistic information can be added to the model and reliability estimates calculated.

ARC had developed a translation program to convert digraphs into another widely used reliability graph model called fault trees. Since other work packages are using fault tree models, this is one way to make the models as versatile as possible. The results produced by reducing a fault tree, called cut sets, are equivalent to the singletons, doubletons, and tripletons provided by DMA. Fault trees may also be used in reliability studies if probabilistic data are available.

These combinatorial models are quite useful for the redundancy analysis of the DMS, and a great deal of insight can be gained from a very simple, top level model. This process is illustrated in the next section. When a system is designed to dynamically reconfigure as is the case with the DMS, the combinatorial models are not appropriate for the reliability analysis. Markov models are a popular modeling approach used in the fault tolerant computing community for systems utilizing hot and cold sparing and dynamic reconfiguration when failures occur in the system. An example of this approach is also given in the next section.

E.3.1. Model limitations– The current model is limited to analysis of the optical network, RCs, and SDPs with successful operation being defined by the ability of the SDPs to communicate across the network. This level of detail is adequate for the current phase of DMS design when redundancy analysis is the key focus. More detail can be added to the model later.

E.3.2. Redundancy analysis of the token ring network– A top level model of the DMS can be developed from the simplified schematic shown in figure E-1. The dual-fiber optic ring, dual RCs, and critical SDPs are indicated in the diagram. Each RC is connected to one ring. To better illustrate details of the communication path at a site representative of every site on the network, an expanded view of one site on the ring is shown in figure E-2 . The naming convention used in the diagram,

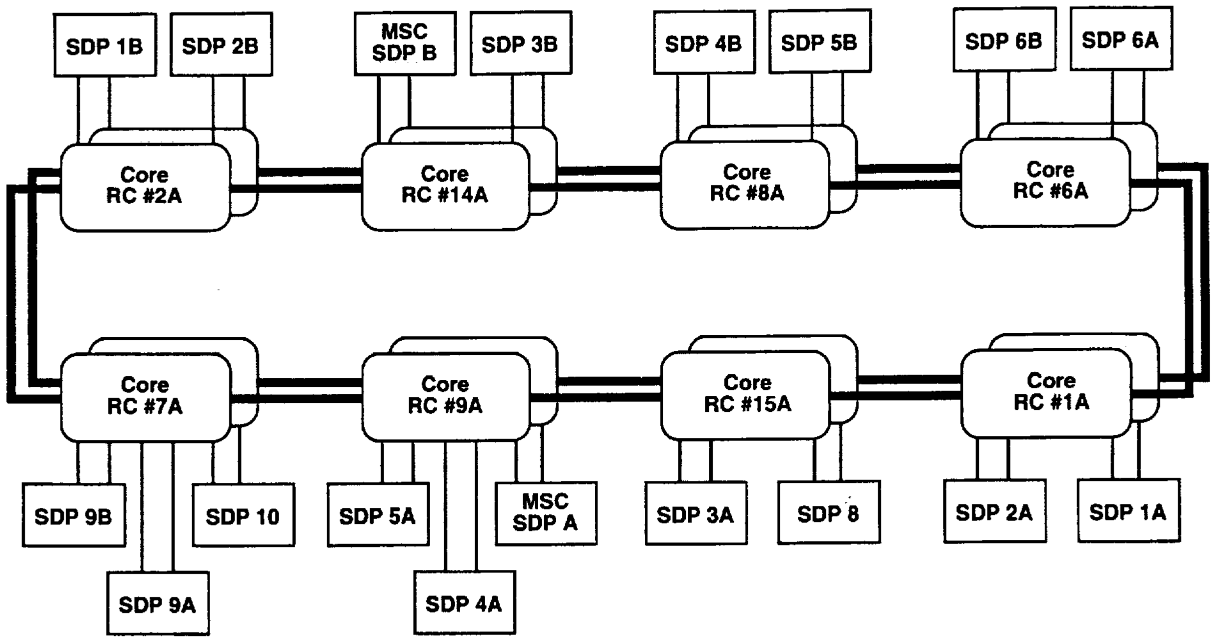


Figure E-1. Simplified schematic diagram of the DMS at PMC.

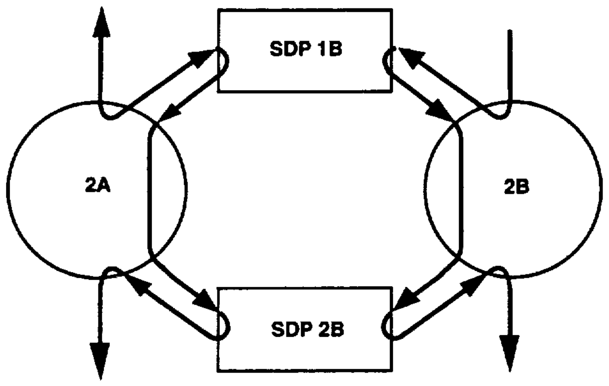


Figure E-2. Expanded view of site 2.

SDP XA and SDP XB, and also RC (indicated by a circle) XA and RC XB, represent a primary, A, and a backup (hot or cold), B, component.

The failure modes and effects analysis (ref. 21) focuses on two main modes of failure: failure of the RC due to power supply failure, and network media failures due to breaks in the fiber and/or connectors. If it is possible for an RC to fail as a unit, then FDDI standard reconfiguration procedures after such a failure would result in isolation of all nodes connected to it. This situation is illustrated in figure E-3, where failure of RC 8A isolates SDP 4B and SDP 5B from the rest of the network. SDP 6A and 9B were interchanged to simplify the model by making it consistent with respect to primary and backup placement. SDP 10 and 8 were also removed under the assumption that their function is not critical to this analysis. We have insufficient detail regarding the architecture of DMS RC to understand how possible failures might impact the network. This does not mean that the DMS is failure tolerant to this mode of failure, however, there will always exist a second processor on the network that can take over the tasks if the processors isolated are primary processors. In figure E-3, the processors that are isolated as a result of the failure of RC 8 are both backup processors. When primary processors are isolated, a smooth transition to the backup processors is essential for uninterrupted service. Scenarios such as this should be run on test beds to determine the sensitivity of the system to various types of both transient and permanent failures. Most safety critical systems in the past have relied upon triple-modular-redundancy, further emphasizing the need for test-bedding this unique configuration as early as possible.

This example illustrates the straightforward process of redundancy analysis. Of course, the simplicity of this model is not indicative of the models that will result when more components (such as local buses and MDMs) are included in the analysis. Larger models are not solvable by inspection as illustrated here. Instead, the DMA tool should be employed to find all single and double points of failure.

E.3.3. Reliability analysis of the token ring network– A reliability analysis was performed on the token ring configuration shown in figure E-1. At this point in the design of the DMS, reliability analyses serve only as a

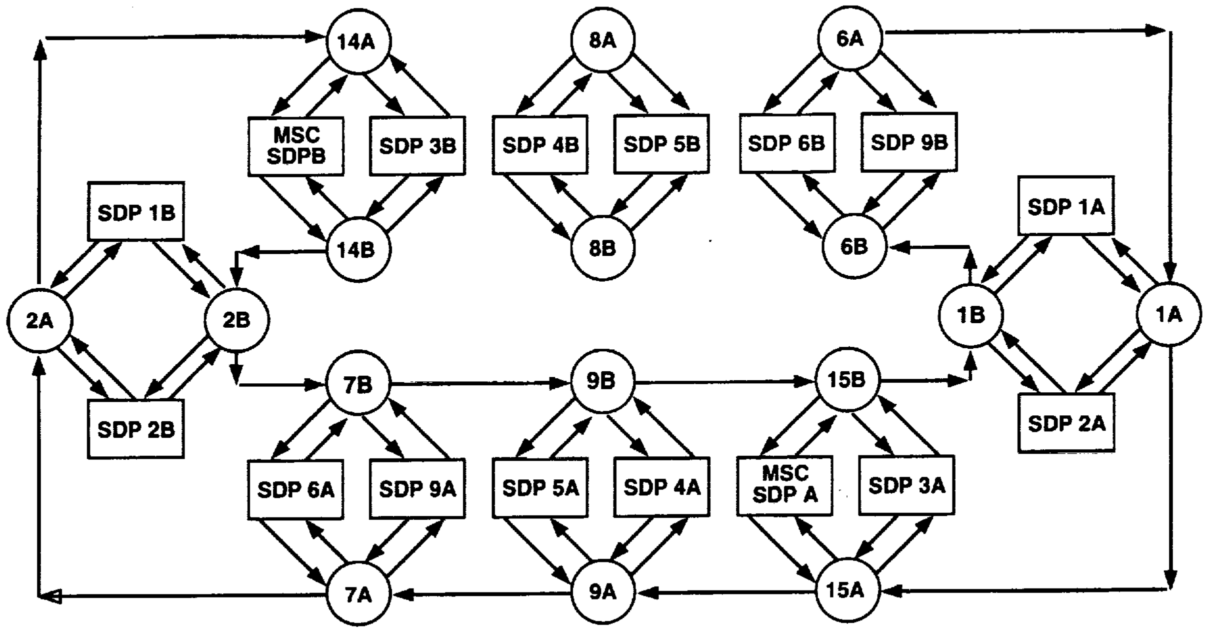


Figure E-3. Reconfiguration example for site 8 failure.

means of comparing different designs. The absolute results are meaningless since quite a few assumptions should be made, however, the relative values resulting from varying the initial assumptions are informative. The assumptions used in the analysis are summarized below:

1. Failure rates:

   • Processor failure rates are 114.16 per million hours (MTBF = 1 year).

   • RC failure rates are 57.08 per million hours (MTBF = 2 years).

2. Recovery model for processors:

   • 90% of all failures are transient, with 96% of the transients detected, and recovery time is uniformly distributed between 0 and 1 sec.

   • 10% of failures are permanent, with 96% of the permanents detected, and a constant recovery time (reboot time) of 1 min.

   • 4% of all failures are undetected and cause immediate unit failure.

3. Recovery model for RCs:

   • All failures are permanent and take 5 sec for recovery, and recovery is always successful.

4. If node types 3 and 4 have nearly simultaneous failures (i.e., if a second node of type 3 or 4 fails during recovery from a first fault), the system fails (near-coincident failures). This is because nodes 3 and 4 provide electrical power.

5. The mission length is 100 hours (this shortens the computation time required to solve the models). This causes the system unreliability to be higher than what will be expected for mission durations of years, however, the relative differences are comparable.

For the network configuration shown in figure E-1 (excluding SDP 8 and SDP 10), with RCs at each station that connect to one direction of the fiber, and assuming hot backups, the system unreliability is 0.031. If the two RCs at each station are replaced with one that is connected to both directions of the fiber, the unreliability is reduced to 0.019. Notice that in figure E-1, the primary and backup processors for nodes 6 and 9 are each on the same RC (RC 6 for nodes 6A and 6B, and RC 7 for nodes 9A and 9B). If nodes 6A and 9B are interchanged, assuming hot backups and one RC at each station, the system unreliability is 0.008016. Taking this last configuration and making the backups cold spares instead of hot, the system unreliability is 0.004338. Summarizing these results: for the configuration in figure E-1, the system is 50% less reliable if the RCs are connected to one direction of the fiber; for the configuration shown in figure E-3, with RCs connected to both directions of the fiber, the system is roughly twice as reliable if the backups are powered off (cold) than if they are kept powered on (hot).

These results illustrate the types of trade-off studies that can be performed early in the design of a system to analyze the performance of different system configurations and parameter values under failure conditions. It is important that these results be compared to actual test-bed experiments in order to validate the models and revise assumptions to more accurately predict the actual system behavior.

F. Acronyms

A&R	automation and robotics
AANMS	Advanced Automation Network Monitoring System
AC	Assembly Complete
ACD	Architecture Control Document
AMD	Advanced Micro Devices
ANSI	American National Standards Institute
APAE	attached payload accommodation equipment
APP	application
ARC	Ames Research Center
ASIC	application specific integrated circuit
AT&T	American Telephone & Telegraph
BCU	Bus Control Unit
BIA	Bus Interface Adapter
biCMOS	bipolar complementary metal oxide semiconductor
BIU	bus interface unit
BNIU	bus network interface unit
BSP	baseband signal processor
CDR	Critical Design Review
CEI	Contract End Item
CHeCS	Crew Health Care Systems
CHRPS	configurable high rate processing system
CISC	complex instruction set computer
CMOS	Complementary Metal Oxide Semiconductor
COTS	commercial, off-the-shelf
CPU	central processing unit
CR	Change Request
CRISP	complexity-reduced instruction set processors
CSA	Canadian Space Agency
CSCI	computer software configuration items
CTS	Communications and Tracking System
C&T	communications and tracking
DARPA	Defense Advanced Research Project Agency
DFRF	Ames/Dryden Flight Research Facility
DMA	Digraph Matrix Analysis
DMS	Data Management System
DoD	Department of Defense
DoE	Department of Energy
DPNS	Distributed Processing Network Simulator
DSAR	data storage and retrieval
ECL	emitter-coupled logic
ECLSS	Environmental Control and Life Support System
EDAC	error detection and correction
EDP	embedded data processor
EEPROM	electronically erasable programmable read only memory
EISA	Extended Industry Standard Architecture
EMADS	Emergency Monitor and Distribution System
EMTT	external maintenance task team
EPS	Electrical Power System
ESA	European Space Agency
EVA	extravehicular activities
F/C	Firmware Controller
FDDI	fiber distributed data interface
FDIR	fault detection, isolation, and recovery
FEU	Functional Equivalent Unit
FMEA	failure modes and effects analysis
FMS	Fluid Management System
FT/RM	failure tolerance and redundancy management
FTS	Flight Telerobotic Servicer

GaAs	Gallium Arsenide
GN&C	Guidance, Navigation, and Control System
GPC	General Purpose Computer
GSFC	Goddard Space Flight Center
GW	gateway
HAB	habitat
HEMT	high electron mobility transfer
HPPI	high performance peripheral interface
HRL	high-rate links
HSTA	Harris System Testability Analyzer
IBM	International Business Machines
IEEE	Institute of Electrical and Electronics Engineers
I/O	inputs and outputs
IOCU	I/O control unit
IPI	Intelligent Peripheral Interface
ISA	instruction set architecture
ISES	Information Sciences Experiment System
ISO	International Standards Organization
ITA	Integrated Truss Assembly
ITCS	internal thermal control system
IT&V	Integrated Test and Verification
IVA	intravehicular activity
JEM	Japanese experiment module
JSC	Johnson Space Center
kbps	kilobit per second = 10^3 bits per second
LAN	local area network
LANES	Local Area Network Extensible Simulator
LaRC	Langley Research Center
LynxOS	Lynx real-time operating system
Mbps	megabit per second = 10^6 bits per second
MB	megabyte
MDM	multiplexer-demultiplexer
MDSSC	McDonnell Douglas Space Systems Company
MILSPEC	military specification
MIPS	million instructions per second
MIPS*	Microprocessor without Interlocked Pipeline Stages
MISC	medium instruction set computers
MMU	memory management unit
MODB	master object data base
MPAC	multi-purpose application console
MPAC-C	multi-purpose application console—cupola
MPAC-F	multi-purpose application console—fixed
MPAC-O	multi-purpose application console—orbiter
MSD	mass storage disk
MSU	mass storage unit
MT	mobile transporter or man tended
NGCR	Next Generation Computer Resource
NIA	network interface adapter
NIU	network interface unit
NMOS	N-Metal Oxide Semiconductor
NOS	network operating system
OMA	Operations Management Application
OMS	Operations Management System
ORU	orbital replaceable units
OS	operating system
OSI	Open Systems Interconnection
PDR	Preliminary Design Review
PDU	Protocol Data Unit
P/L	payload
PMAD	Power Management and Distribution
PMC	Permanent Manned Configuration
POSIX	Portable Operating System Interface for Computer Environments
PRD	Program Requirements Document
PWR	power
RAM	random access memory
RC	ring concentrator
RF	radio frequency
RID	review item disposition
RISC	reduced instruction set computer
RODB	runtime object database

ROM	read-only memory
RPC	remote procedure call
RTE	runtime environment
SATWG	Strategic Avionics Technology Working Group
SCSI	small computer system interface
SDIO	Strategic Defense Initiative Office
SDP	standard data processor
SEI	Space Exploration Initiative
SEU	single event upset
SM	System Management
SMD	storage module device
SOS	silicon on sapphire
SPARC	Scalable Processor Architecture
SPAWAR	Space and Naval Warfare Program
SRAM	static random-access memory
SRS	Software Requirements Specification
SSF	Space Station Freedom
SSIS	Space Station Information System
SSMB	space station manned base
STSV	Standard Services
TCP/IP	Transmission Control Protocol/Internet Protocol
TCS	Thermal Control System
TDB	Time Distribution Bus
TDRSS	Tracking and Data Relay Satellite System
TGU	Time Generation Unit
UAS	user application software
USE	User Support Environment
V&V	Verification and Validation
WAN	wide-area network
WP-2	Work Package-2
XTP	eXpress Transfer Protocol
ZOE	zone of exclusion

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE September 1992	3. REPORT TYPE AND DATES COVERED Technical Memorandum

4. TITLE AND SUBTITLE Space Station Freedom Data Management System Growth and Evolution Report	5. FUNDING NUMBERS 488-51-01
6. AUTHOR(S) R. Bartlett, G. Davis, T. L. Grant, J. Gibson, R. Hedges, M. J. Johnson, Y. K. Liu, A. Patterson-Hine, N. Sliwa, H. Sowizral, and J. Yan	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Ames Research Center Moffett Field, CA 94035-1000	8. PERFORMING ORGANIZATION REPORT NUMBER A-91163
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) National Aeronautics and Space Administration Washington, DC 20546-0001	10. SPONSORING/MONITORING AGENCY REPORT NUMBER NASA TM-103869

11. SUPPLEMENTARY NOTES
Point of Contact: T. L. Grant, Ames Research Center, MS 269-4, Moffett Field, CA 94035-1000;
(415) 604-4200

12a. DISTRIBUTION/AVAILABILITY STATEMENT Unclassified — Unlimited Subject Category 17	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The Information Sciences Division at the NASA Ames Research Center has completed a 6-month study of portions of the Space Station Freedom Data Management System (DMS). This study looked at the present capabilities and future growth potential of the DMS, and the results are documented in this report. Issues have been raised that were discussed with the appropriate Johnson Space Center (JSC) management and Work Package-2 contractor organizations. Areas requiring additional study have been identified and suggestions for long-term upgrades have been proposed. This activity has allowed the Ames personnel to develop a rapport with the JSC civil service and contractor teams that does permit an independent check and balance technique for the DMS.

14. SUBJECT TERMS FDDI LAN, 1553B bus, RODB, FDIR, Ada, Real-time, Operating system, LAN (local area network) OSI Protocol, 386 Processor	15. NUMBER OF PAGES 72
	16. PRICE CODE A04

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT